Chapter 1
Introduction
# Chapter 1: Introduction

## Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1 Scope</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Intended Audience</td>
<td>1</td>
</tr>
<tr>
<td>1.3 Organization of the Manual</td>
<td>1</td>
</tr>
</tbody>
</table>
The *AWIPS System Manager’s Manual (SMM)* provides information for use in managing the Advanced Weather Interactive Processing System (AWIPS) at a site level. System management includes controlling user access, supporting user functions, and some file and database maintenance. Each area of responsibility requires knowledge of and the ability to use AWIPS components.

The operations and maintenance of AWIPS is being performed by Raytheon under U.S. Department of Commerce, National Oceanic and Atmospheric Administration (NOAA) Contract DG133W-05-CQ-1067.

If you find errors in the manual, please write to the AWIPS Documentation Team at nws.hq.awips.doc.team@noaa.gov.

1.1 **Scope**

This manual is designed to serve as a guide for the management of Operational AWIPS. As new AWIPS II releases are implemented, the SMM will continue to be updated to capture additional or changed functionality.

1.2 **Intended Audience**

This manual is intended for use by AWIPS system managers. Users are assumed to have a basic understanding of, and some experience using, UNIX, Linux, Postgres, SQL, Java, and Python.

1.3 **Organization of the Manual**

The *AWIPS System Manager’s Manual* provides a detailed description of the system and its associated environments tailored for AWIPS system managers. It is organized into the following chapters and appendices:

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chapter 1</td>
<td>Introduction</td>
</tr>
<tr>
<td>Chapter 2</td>
<td>AWIPS System Architecture</td>
</tr>
<tr>
<td>Chapter 3</td>
<td>Individual User Accounts</td>
</tr>
<tr>
<td>Chapter 4</td>
<td>Data Flow Overview</td>
</tr>
<tr>
<td>Chapter 5</td>
<td>Ingest of Satellite Imagery</td>
</tr>
<tr>
<td>Chapter 6</td>
<td>Ingest of NWSTG Data</td>
</tr>
<tr>
<td>Chapter 7</td>
<td>Ingest of Radar Data</td>
</tr>
<tr>
<td>Chapter 8</td>
<td>Local Data Acquisition and Dissemination (LDAD) System</td>
</tr>
<tr>
<td>Chapter 9</td>
<td>Background Applications</td>
</tr>
<tr>
<td>Chapter 10</td>
<td>Asynchronous Product Scheduler</td>
</tr>
<tr>
<td>Chapter 11</td>
<td>Crons and Purging</td>
</tr>
<tr>
<td>Chapter 12</td>
<td>Database Management</td>
</tr>
<tr>
<td>Chapter 13</td>
<td>Event Notification</td>
</tr>
<tr>
<td>Chapter 14</td>
<td>Message Handling System</td>
</tr>
</tbody>
</table>
Chapter 2
AWIPS System Architecture
## Chapter 2. AWIPS System Architecture

### Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1 AWIPS Communications Network</td>
<td>1</td>
</tr>
<tr>
<td>2.1.1 AWIPS Satellite Broadcast Network</td>
<td>3</td>
</tr>
<tr>
<td>2.1.2 AWIPS WAN</td>
<td>4</td>
</tr>
<tr>
<td>2.1.2.1 Very Small Aperture Terminal (VSAT) WAN Backup</td>
<td>10</td>
</tr>
<tr>
<td>2.1.3 LAN</td>
<td>10</td>
</tr>
<tr>
<td>2.1.4 Routers</td>
<td>10</td>
</tr>
<tr>
<td>2.1.5 Circuits</td>
<td>16</td>
</tr>
<tr>
<td>2.1.6 NESDIS Interface Router</td>
<td>16</td>
</tr>
<tr>
<td>2.2 Site Hardware Architecture</td>
<td>16</td>
</tr>
<tr>
<td>2.2.1 Rack Components</td>
<td>17</td>
</tr>
<tr>
<td>2.2.1.1 Satellite Broadcast Network (SBN)</td>
<td>23</td>
</tr>
<tr>
<td>2.2.1.2 Synchronous Communications (Radar)</td>
<td>25</td>
</tr>
<tr>
<td>2.2.1.3 Application Server</td>
<td>26</td>
</tr>
<tr>
<td>2.2.1.4 Data Servers</td>
<td>26</td>
</tr>
<tr>
<td>2.2.1.4.1 Data Server File Systems and Raw Partitions</td>
<td>29</td>
</tr>
<tr>
<td>2.2.1.4.1.1 File Systems on the Linux Data Server (DX)</td>
<td>29</td>
</tr>
<tr>
<td>2.2.1.4.1.2 Raw Partitions for PostgreSQL on the Linux Data Server</td>
<td>33</td>
</tr>
<tr>
<td>2.2.1.4.2 AX File Systems</td>
<td>34</td>
</tr>
<tr>
<td>2.2.1.5 LDAD Hardware</td>
<td>36</td>
</tr>
<tr>
<td>2.2.1.5.1 LDAD Server File Systems</td>
<td>38</td>
</tr>
<tr>
<td>2.2.1.6 Workstations</td>
<td>40</td>
</tr>
<tr>
<td>2.2.1.6.1 Reserved</td>
<td>42</td>
</tr>
<tr>
<td>2.2.1.6.2 Linux Workstation File Systems</td>
<td>42</td>
</tr>
<tr>
<td>2.2.1.7 Linux Preprocessor</td>
<td>44</td>
</tr>
<tr>
<td>2.2.1.7.1 Preprocessor Server File Systems</td>
<td>47</td>
</tr>
<tr>
<td>2.2.1.7.2 Preprocessor Server File Systems</td>
<td>47</td>
</tr>
<tr>
<td>2.3 NCF Time Source</td>
<td>50</td>
</tr>
<tr>
<td>2.4 Printers</td>
<td>50</td>
</tr>
<tr>
<td>2.5 AWIPS Software</td>
<td>50</td>
</tr>
<tr>
<td>2.5.1 AWIPS Contractor-Developed Software</td>
<td>50</td>
</tr>
<tr>
<td>2.5.2 Government-Developed Software</td>
<td>51</td>
</tr>
<tr>
<td>2.5.3 Commercial Off-the-Shelf (COTS) Software and Freeware</td>
<td>51</td>
</tr>
<tr>
<td>2.5.3.1 AWIPS II: COTS Software and Freeware</td>
<td>51</td>
</tr>
</tbody>
</table>
2.5.3.2 Mobile Code Technologies ................................................................. 54
2.6 AWIPS I and AWIPS II ........................................................................ 54
  2.6.1 What’s Changed in AWIPS .............................................................. 54
  2.6.2 What’s Not Changed in AWIPS ..................................................... 56
2.7 Basic AWIPS II Software Deployment ............................................... 57
2.8 AWIPS II Server Clustering ................................................................ 61
2.9 Basic AWIPS II Communication Architecture .................................... 62
2.10 Basic AWIPS II Logging Architecture .............................................. 63
2.11 Basic AWIPS II Data Storage Architecture ....................................... 64
2.12 Basic AWIPS II Visualization Architecture ....................................... 65
2.13 Basic AWIPS II Data Processing Architecture ................................... 66
  2.13.1 Basic AWIPS II Data Receipt Architecture ................................... 66
  2.13.2 Basic AWIPS II Data Decoding Architecture ............................... 67
  2.13.3 Basic AWIPS II Data Storage Architecture .................................... 68
    2.13.3.1 AWIPS II Raw Data Storage .................................................. 68
    2.13.3.2 AWIPS II Processed Data Storage ......................................... 69
    2.13.3.3 PyPIES (Python Process Isolated Exchange Storage) ........... 72
  2.13.4 Basic AWIPS II Data Retrieval Architecture ............................... 72
2.15 Basic AWIPS II Data Purge Architecture ......................................... 74
  2.15.1 AWIPS II Processed Data Storage Purge Architecture ................ 74
  2.15.2 AWIPS II Raw Data Storage Purge Architecture ....................... 75
2.16 AWIPS II Localization Store .............................................................. 76
2.17 EDEX Distribution Files ................................................................. 76
  2.17.1 Editing an EDEX Distribution File ............................................... 77

List of Exhibits

Exhibit 2-1. AWIPS System Boundaries and Subsystems .......................... 2
Exhibit 2.1.1-1. AWIPS Communications Network .................................. 3
Exhibit 2.1.2-1. WFOs Connected to Adjacent RFCs ................................. 6
Exhibit 2.1.2-2. WFOs Connected to Alternate RFCs ............................... 7
Exhibit 2.1.2-3. RFCs to RFCs and RFCs to NCF ..................................... 8
Exhibit 2.1.2-4. RFCs to BNCF ............................................................. 9
Exhibit 2.1.2.1-1. Block Diagram of VSAT System Setup at a WFO .......... 11
Exhibit 2.1.4-1. Interconnection of Wide Area Network Devices ............ 12
Exhibit 2.1.4-2. WFO Router Connection Diagram .............................. 13
Exhibit 2.1.4-3. IP Router Connection Diagram for NC ......................... 14
Exhibit 2.1.4-4. RFC Router Connection Diagram ............................... 15
Exhibit 2.2-1. Stand-Alone WFO Site Hardware Architecture ................ 18
Exhibit 2.2-2. RFC Site Hardware Architecture ................................................................. 19
Exhibit 2.2-3. LDAD Hardware Architecture ................................................................. 20
Exhibit 2.2-4. AWIPS WFO Network Diagram ............................................................. 21
Exhibit 2.2.1-1. Standard Stand-Alone WFO/RFC Rack Configuration ....................... 22
Exhibit 2.2.1.6-1. AWIPS Workstation ....................................................................... 41
Exhibit 2.2.1.7-1. Linux Preprocessors (PX) Architecture ........................................... 45
Exhibit 2.2.1.7-2. Linux Preprocessor (PX) Hardware .................................................. 45
Exhibit 2.2.1.7-3. Linux Preprocessor (FX) Interfaces .................................................... 46
Exhibit 2.7-1. Overall AWIPS II Data Flow ................................................................. 58
Exhibit 2.7-2. Overall AWIPS II Servers Processes ...................................................... 59
Exhibit 2.7-3. AWIPS II Servers Overview ................................................................. 60
Exhibit 2.9-1. AWIPS II Inter-Process Communication .................................................. 63
Exhibit 2.12-1. AWIPS II Visualization ...................................................................... 65
Exhibit 2.13.1-1. AWIPS II Data Receipt ..................................................................... 66
Exhibit 2.13.2-1. AWIPS II Data Decoding ................................................................. 67
Exhibit 2.13.3.1-1. AWIPS II Raw Data Storage ............................................................. 69
Exhibit 2.13.3.2-1 AWIPS II Processed Data Storage .................................................... 70
Exhibit 2.13.3.2-2. CAVE/Processed Data Storage Interaction .................................... 71
Exhibit 2.13.4-1. AWIPS II Data Retrieval ................................................................. 73
Exhibit 2.14-1. AWIPS II Message Retransmission ....................................................... 74
Exhibit 2.15.1-1. AWIPS II Processed Data Storage Purge .......................................... 75
Exhibit 2.16-1. CAVE/Localization Store Interaction .................................................... 76

List of Tables

Table 2.2.1.4-1. Data Server Configurations at AWIPS Sites ........................................... 27
Table 2.2.1.4-2. WFO Archive Server Configuration at AWIPS Sites ......................... 28
Table 2.2.1.4-3. RFC Archive Server Configuration at AWIPS Sites .......................... 28
Table 2.2.1.5-1. Port Assignments ................................................................................. 37
Table 2.2.1.6-1. Workstation Processor Configurations at AWIPS Sites ..................... 42
Table 2.2.1.7-1. Linux Preprocessor Configurations at AWIPS Sites ............................. 46
Table 2.5.3.1-1. COTS Software and Freeware Used in AWIPS II ................................. 51
Table 2.7-1. AWIPS II Software .................................................................................. 60
Table 2.8-1. AWIPS II Server Failover Strategies ......................................................... 61
Table 2.8-2. AWIPS II Server Aliases ......................................................................... 62
Table 2.10-1. Log Locations for AWIPS II Components ............................................... 64
Table 2.11-1. AWIPS II Data Storage ......................................................................... 64
Table 2.13.1-1. AWIPS II Interfaces to Data Sources .................................................... 67
The design of the AWIPS system architecture is driven by the requirements for expandability, flexibility, availability, and portability. This system has been designed for easy expandability to allow for the introduction of new functionality and the augmentation of network and processing capacities. AWIPS can accommodate the evolving state of operational forecasting to enable the NWS to meet the objectives of the modernization effort. AWIPS is designed so that software and data can be migrated to new platforms as technology evolves.

The AWIPS system architecture illustrated in Exhibit 2-1 gives a macro-level view of the AWIPS system with the various types of AWIPS sites and the interfaces to external entities, including the following:

- Network Control Facility (NCF)
- Satellite Broadcast Network (SBN)
- Wide Area Network (WAN)
- Weather Forecast Office (WFO)
- River Forecast Center (RFC)
- National Center (NC)
- NOAAPORT Receive System (NRS)

Information flows into the AWIPS NCF (ANCF) in Silver Spring, Maryland, or the Backup NCF (BNCF) in Fairmont, West Virginia, from the National Weather Service Telecommunications Gateway (NWSTG) and the National Environmental Satellite, Data, and Information Service (NESDIS). These data are sent to all AWIPS and NRS sites via the SBN.

The WAN provides additional communications infrastructure utilizing TCP/IP (point-to-point (PTP) and point-to-multipoint (PTM)) communications, which allows AWIPS sites to communicate with each other. In addition to the data received from the NCF, AWIPS sites have access to data from local sources, such as the Automated Surface Observing System (ASOS), the Weather Surveillance Radar-1988 Doppler (WSR-88D) radar, and Terminal Doppler Weather Radars. [Note: All channels except GOES support automatic retransmission. GOES does not because the NCF software does not support it.]

2.1 **AWIPS Communications Network**

The AWIPS Communications Network (ACN) is a nationwide communications system consisting of an SBN and a WAN.

The SBN provides PTM communications between the NCF and all sites. It is used to distribute satellite, radar, grid, text, graphic, synoptic, BUFR (Binary Universal Form for data Representation), and point data to the sites from the National Centers for Environmental Prediction (NCEP), NESDIS, the NWSTG, AWIPS sites, and other data sources.
Exhibit 2-1. AWIPS System Boundaries and Subsystems
The WAN provides TCP/IP (PTP, PTM) communications between all AWIPS sites. It is used to send data from one site to any other site. There is also a backup system that consists of a dial-up modem that the NCF can use to dial into a site.

### 2.1.1 AWIPS Satellite Broadcast Network

The SBN is a PTM network that distributes weather data from the NCF or the BNCF to receiving sites throughout the United States (see Exhibit 2.1.1-1). The network consists of the Satellite Broadcast Processor (SBP); the uplink Master Ground Station (MGS) antenna in Hauppauge, New York, and the backup MGS (BMGS) antenna at the NASA Independent Verification and Validation (IV&V) Facility in Fairmont, West Virginia; the SES-1 satellite; the downlink antenna at the locations receiving data; and the satellite Digital Video Broadcast (DVB) receivers at the sites. Primary data transmitted are Geostationary Operational Environmental Satellite (GOES)-EAST and GOES-WEST satellite imagery, model data, observational data, and text products. [Note: All channels except GOES support automatic retransmission. GOES does not because the NCF software does not support it.]

---

**Exhibit 2.1.1-1. AWIPS Communications Network**

---

AWP.MAN.SMM.A2

– Hard copies uncontrolled. Verify effective date prior to use. –
The NCF and the BNCF receive the satellite and model data from NESDIS and the NCEP, respectively. The data is processed at the NCF or the BNCF, stored on disk, and sent to the MGS in Hauppauge or the BMGS in Fairmont. The primary MGS is a commercial satellite communications facility owned by Globecomm Systems, Inc. (Globecomm or GSI); the BMGS is a Government-owned facility. Both the primary and the backup MGS transmit the data to a commercial communications satellite (SES-1) that transmits the data to the sites.

The MGS uses the frequency division multiple access (FDMA) transmission technique, which requires an independent modulator for each data stream. Four different substreams are required to transmit SBN data. Each site receiving data can be customized to receive only those streams that are appropriate for its mission. Ninety percent of the sites need only two of the four data streams. A channel failure does not result in total loss of all data streams, only the affected channel.

### 2.1.2 AWIPS WAN

The AWIPS WAN is a terrestrial network designed to support distribution of PTP and PTM data among all AWIPS sites. The WAN gives field sites, NCs, and headquarters sites the ability to communicate with each other and send data to the central facility for rebroadcast. The data includes products and messages generated by the local AWIPS office and observation data generated or received by the local AWIPS office from systems external to AWIPS. Messages or information intended for transmission from one AWIPS site to another (PTP) will be sent via the WAN. The WAN also supports PTM distribution of data so that all AWIPS offices can send data to the NCF. It also supports other types of traffic at sites as follows:

- Interactive capabilities
- Request/reply data
- Retransmission requests for lost products
- Error and fault notification from sites to the NCF
- Remote log-in from site to site
- Distribution of software releases from the NCF
- Remote diagnostic action by the NCF
- Network management.

The WAN provides an infrastructure for field sites to provide backup support for other sites if an operations failure occurs.

The WAN is also used to synchronize all AWIPS system clocks using the Network Time Protocol (NTP). The AWIPS time source is located at the NCF/BNCF; it receives and decodes time from the National Institute of Standards and Technology (NIST) time-signal broadcast call letters (WWV) external time source and provides that information to AWIPS. This time information is obtained through a dial-up connection to the timing source. After the NCF/BNCF communication server is synchronized with the NIST
WWV, it synchronizes all other local area network (LAN) clocks at the site. The NCF/BNCF broadcasts the time to all the other AWIPS sites over the WAN. All AWIPS system clocks are synchronized to within 1 second of the NIST WWV time.

The colocated WFO/RFC sites and the NCF/BNCF are major nodes of the network. They currently use Generic Routing Encapsulation (GRE) tunnels through OPSnet to act as hubs, identical to the Hub and Spoke topology used in the previous AWIPS dedicated circuit network. [Note: The transition of the GRE tunnels is on hold until OPSnet completes the transition from Sprint to Verizon. When the final site is moved to Verizon, a new schedule for GRE tunnel elimination will be developed.]

WFOs, Regional Headquarters (RHQ), and NCs are the spur nodes of the network. The WAN consists of circuits linking the following site types:

- WFOs to adjacent RFCs (Exhibit 2.1.2-1)
- WFOs to alternate RFCs (Exhibit 2.1.2-2)
- RFCs to RFCs and RFCs to NCF (Exhibit 2.1.2-3)
- RFCs to BNCF (Exhibit 2.1.2-4).

Verizon and Sprint are the FTS2001 service providers. The FTS2001 contract provides the circuits used in AWIPS to interconnect the sites. In Alaska, the frame relay Permanent Virtual Circuits (PVC) are provided by General Communication, Inc. (GCI). The circuits provide site interconnection at up to 1.5 Mbps. The WAN’s high availability is achieved by incorporating real-time network management, supplying capacious circuits, providing multiple paths between nodes, and implementing backup circuits for dedicated links.

The types of circuits used in the WAN are as follows:

- GRE is used to send IP packets from one AWIPS network to another, without being parsed or treated like IP packets by any intervening routers. GRE tunnels provide WAN node-to-node connectivity to conterminous United States (CONUS) AWIPS sites. For CONUS AWIPS sites, the terrestrial network uses the NOAAnet. It is based on fast packet technology and aimed at high-speed, bursty data applications such as LAN connections. The NOAAnet utilizes Multi-Protocol Label Switching (MPLS) for WAN connectivity. See Section 2.1.4, Routers, which diagrams the GRE tunnel from the WFO to the RFC and from the RFC to the NCF.

- Dedicated Transmission Service (DTS) for Hawaii, Guam, and Puerto Rico provides intersite communications and connections to CONUS AWIPS sites.

- Switched Data Service (SDS) circuits provide backup for AWIPS sites. Dial backup circuits are provided at all locations. Each non development site has an FTS2001 SDS circuit connected to a router port for use as a dial-up backup to the dedicated link. Colocated WFO/RFC sites have four SDS circuits.

- Monitor and control functions use Switched Voice Service (SVS) circuits for backup.

- Integrated Services Digital Network (ISDN) circuits provide backup for AWIPS sites along with the SDS circuits.
Exhibit 2.1.2-1. WFOs Connected to Adjacent RFCs
Exhibit 2.1.2-2. WFOs Connected to Alternate RFCs
Exhibit 2.1.2-3. RFCs to RFCs and RFCs to NCF
Exhibit 2.1.2-4. RFCs to BNCF
2.1.2.1 Very Small Aperture Terminal (VSAT) WAN Backup

When a site loses its AWIPS WAN connection for what is expected to be an extended period of time due to some kind of disaster, or when a test of the system is desired, the NWS will notify Raytheon that the backup system (one or more of the VSAT systems) is required. At the same time, the NWS will work with the affected region and site to ensure access to the affected site for the equipment and a Field Site Representative (FSR). The Raytheon AWIPS Team will immediately deploy a Flyaway Satellite Terminal and an FSR to the affected site(s). The FSR will set up the terminal and, working with the local Electronic Systems Analyst (ESA), connect the satellite terminal to the site LAN in place of the non-functional WAN. Once the terminal becomes operational, the deployed technician will return home. The AWIPS NCF will monitor the communications link via existing T1 lines to the Network Operations Center (NOC) of the Long Island International Teleport (LIIT) at Hauppauge, NY, as it does for the AWIPS WAN. LIIT is operated by Raytheon AWIPS teammate Globecomm.

The NWS Radar Operations Center (ROC) will be kept in the loop on all VSAT deployment activities in order to be available for possible backup support.

When the AWIPS WAN has been returned to operational status, the Raytheon AWIPS Team FSR will return to the site to disconnect and take down the satellite terminal and return it to its storage location.

Detailed deployment procedures are outlined in AWP.PLN.AWB-01.02, AWIPS VSAT WAN Backup: Deployment Plan and Concept of Operations, dated May 5, 2011. A block diagram of the VSAT system setup at a WFO is provided in Exhibit 2.1.2.1-1.

2.1.3 LAN

The AWIPS site LAN comprises industry-standard 100/1000Base-T Ethernet network technologies to support the distribution of data throughout the AWIPS sites. The LAN provides concurrent transmission of data at speeds of 1000 Mbps, and a 1 Gbps wire speed is provided to each Ethernet interface. The LAN is implemented via CISCO 2960 switches for all AWIPS sites.

2.1.4 Routers

Cisco routers are used to pass data from the site LAN to the WAN and to the site LAN from the WAN. WFO routers are sized to connect to one LAN and employ two Ethernet connections to the Site CE Router. RFC and NC routers are sized to connect to two LANS and employ between two Ethernet connections to the Site CE Router. Exhibits 2.1.4-1 through 2.1.4-4 show how the routers interface with the LAN and the WAN.

NOTE: At a colocated site, the RFC provides the routers for both the RFC and the WFO.
Exhibit 2.1.2.1-1. Block Diagram of VSAT System Setup at a WFO
Exhibit 2.1.4-1. Interconnection of Wide Area Network Devices
Exhibit 2.1.4-2. WFO Router Connection Diagram
Exhibit 2.1.4-3. IP Router Connection Diagram for NC
Exhibit 2.1.4-4. RFC Router Connection Diagram

Legend

- Primary
- Alternate
- LAN

SMM 9/30/10
2.1.5 *Circuits*

The communications processor (CP) circuits provide connectivity between AWIPS and external systems. Four terrestrial NESDIS circuits provide connectivity between the NESDIS site and the NCF, and two terrestrial circuits provide connectivity between NESDIS and the BNCF. These circuits are external to AWIPS and outside the ACN.

2.1.6 *NESDIS Interface Router*

Ten routers provide the AWIPS-NESDIS routing function. Five of these routers are located at NESDIS, two at the ANCF, two at the BNCF and one at TNCF.

2.2 *Site Hardware Architecture*

The AWIPS site hardware architecture has been standardized across all AWIPS site types (WFOs, RFCs, and NCs) so that the only differences are in quantity (i.e., number of workstations and amount of disk space) and, at the NCs, some additional peripheral items.

Each AWIPS site functions independently with its own hardware. Because the RFC sites and the colocated WFO system share the WAN routers, the colocated WFO systems do not have WAN routers.

Each site has a site ground station (SGS) that receives the SBN signal. The SGS is provided and supported by Globecomm, under subcontract to Raytheon. The Pacific Tsunami Warning Center (PTWC) has the antenna that serves the WFO in Guam, Honolulu WFO, and Pacific Region Headquarters.

The SGS consists of the following:

- A receive-only antenna that provides the physical interface to the SBN.
- A low-noise-block (LNB) down-converter that converts the C-band satellite signal to lower L-band frequencies.
- The interfacility link (IFL), which is the cable from the antenna to the DVB Receivers.
- Redundant DC power supplies that provide power to the LNB.
- DVB receivers that receive the digital data stream from the SBN and transmit it to the CPs.

Colocated AWIPS sites take advantage of an IFL splitter to share the SGS antenna feed. Locations with more than two AWIPS installations also use an IFL splitter with multiple output ports (up to 8 per splitter).

All AWIPS sites use client/server technology to achieve function reuse, failure recovery, and functional partitioning. The servers are as follows:
• **WFO Archive Server (WAX).** WAX, a Linux-based device, is currently only used for tape backup. It runs the weekly nas backup script because the tape device is connected to it.

• **RFC Archive Server (RAX).** RAX, a Linux-based device, provides additional disk storage capacity for hydrologic data via Postgres database and provides the processing power and associated software to process this hydrologic data at RFCs.

• **Linux Data Servers (DX).** The DXs provide site-level decoding, message handling and data acquisition processing and hosts the DNS, NTP, SNMP, and LDAD TerminalServer. It also provides site-level data repository and database resources.

• **Workstations.** The AWIPS workstations are used to access all site-level system functions; they are also used to process and display textual, graphical, and image data.

• **Communications Processors.** The CPs provide site-level interface and processing to external communications.

• **Local Data Acquisition and Dissemination (LDAD) server.** The LDAD server provides local data acquisition, data quality control, and dissemination interfaces for local NWS offices.

• **Linux Preprocessors (PX).** The PXs provide additional data processing and storage ability for hydrometeorological applications.

The following exhibits illustrate the site hardware architecture:

- WFO (Exhibit 2.2-1)
- RFC (Exhibit 2.2-2)
- LDAD (Exhibit 2.2-3).

The WFO network diagram (Exhibit 2.2-4) represents the interface between the AWIPS LAN and the NOAAnet. The diagram illustrates the physical connections between the AWIPS routers and the NOAAnet routers.

### 2.2.1 Rack Components

Hardware is usually installed at each site in standard WFO and RFC rack configurations, as shown in Exhibit 2.2.1-1. There are stand-alone WFO rack configurations and collocated WFO rack configurations. Collocated WFO racks include all stand-alone equipment except for routers, which are provided by the colocated RFC.
Exhibit 2.2-1. Stand-Alone WFO Site Hardware Architecture
Chapter 2 AWIPS System Architecture

Exhibit 2.2-2. RFC Site Hardware Architecture

Legend
- Interfaces
- Site Boundary

PowerVault is being replaced by DX-DAS. (DX-DAS is a requirement for AWIPS II)

AWIPS MAN.SMM.A2 2-19
Chapter 2

AWIPS System Architecture

Exhibit 2.2-3. LDAD Hardware Architecture

- Hard copies uncontrolled. Verify effective date prior to use. -
Chapter 2

AWIPS System Architecture

Exhibit 2.2-4. AWIPS WFO Network Diagram
Exhibit 2.2.1-1. Standard Stand-Alone WFO/RFC Rack Configuration
2.2.1.1 Satellite Broadcast Network (SBN)

The Communications Processors for the SBN (Communications Processors - Satellite Broadcast Network [CPSBN]) accept data from the DVB receivers, separate the individual products from the data stream, and transfer the products to the Linux data servers for additional processing.

- **DVB Receivers**

  Each AWIPS site has two DVB receivers that receive SBN data. Each DVB receiver receives all of the SBN data transmitted to the site and each receiver acts as a backup for the other. The DVB receivers replaced the demodulators, and the demodulators have been removed from the AWIPS hardware rack. The DVB receiver output is a dedicated LAN segment directly connected to the associated CPSBN.

- **Satellite Broadcast Processors**

  At each AWIPS site, there are two CPSBNs responsible for receiving GOES products and a combination of NCEP products from the NCF.

  **NOTE:** No site has more than two CPSBNs unless it has an NRS.

  CPSBN1 and CPSBN2 function as a heartbeat cluster; failover is managed automatically or manually. Note that only a single high-availability package is currently used on this cluster. The package name is a2cp1apps. In failover mode, package execution shifts to the backup server. LDM is controlled by the a2cp1apps package.

  The a2cp1apps package manages the cp1f floating server name. Note that floating names are available only when the high-availability packages are running; they cannot be used to start or stop the high-availability packages.

  Each CPSBN is an HP DL380 G6 server with a dual 2.26 GHz E5520 processor, 6 GB of RAM, and 4 Ethernet LAN ports.

  IP Virtual Server is installed on the CPSBNs. IPVS acts as a load balancer at the front of the DX3/DX4 EDEX cluster real servers; it directs requests from the workstations for TCP/UDP-based services to the real servers, and makes services of the real servers appear to the workstations as a virtual service on a single IP address. IPVS is aliased as ec. Clients such as CAVE (Common AWIPS Visualization Environment) connect to the EDEX server via the ec alias; this provides for continuation of service in the event of system failover.

  Queue Processor Interface Daemon (QPID) is a JVM (Java Virtual Machine) message broker configured to run on the CPSBN machines. It is responsible for keeping a queue of messages sent by ingest processes alerting an EDEX cluster of new or available data. Each EDEX process in an EDEX cluster is responsible for retrieving the next product in the queue, deleting that product from the queue, and acting upon
the data product based on the information inside the JVM message. In this respect, it
works on a first-come-first-serve basis. So the EDEX process that has the most
system resources available and is processing most quickly will, in theory, get to more
products before the others. A monitoring script has been added and a cron has been
set up in a2cp1cron, which will collect qpid stats and keep a week’s worth of data in
/data/fxa/qpid.

The major file systems on the CPSBNs are:

- **root ( / ), /tmp, /usr, /var.** Linux mandates that these file systems exist. They are
  maintained as separate logical volumes to reduce the risk of any of these file systems
  filling up and affecting system operations. All of the file systems are ext3. The /tmp
  file system will be re-created with each boot sequence and performs no journaling.

- **/boot.** This file system is used to store the Linux kernel and boot-up instructions.

- **/dev/shm.** This is the designated shared memory.

- **/home.** This file system contains the entire user working areas.

- **/awips.** This file system is used to store baselined software and any associated log or
  configuration files. It may contain binary applications, scripts, logs, or data as
  required to support operations.

- **/data.** This file system is used to store the data received over the SBN and also the
  acquisition and SBN process logs. Subdirectories will further isolate application- or
  product-specific logs.

- **/awips2.** This file system is used to store baselined AWIPS II software.

- **/data/fxa.** This file system stores AWIPS data products that are not maintained within
  the RDBMS. This would include the LDAD data.

- **/data_store.** Folders are usually laid out exactly like the sbn folders on the EDEX
  server, with each plug-in having a folder on the data store. However, some of them do
  not follow the same convention; for example, data sent to the 'metar' endpoint will be
  found in the /data_store/text folder. See Chapter 6 for more details on other data
  types. Additionally, if new format ingest is being worked, you will find new data
  types not yet found on the development or integration systems. Those files will be
  located in /data_store/experimental.

[root@cpsbn2-tbw3 ~]# df –H

The following example illustrates CPSBN mount points for the WFOs.

<table>
<thead>
<tr>
<th>Filesystem</th>
<th>Size</th>
<th>Used</th>
<th>Avail</th>
<th>Use%</th>
<th>Mounted on</th>
</tr>
</thead>
<tbody>
<tr>
<td>/dev/mapper/vg01-lvol101</td>
<td>1.1G</td>
<td>416M</td>
<td>565M</td>
<td>43%</td>
<td>/</td>
</tr>
<tr>
<td>tmpfs</td>
<td>3.1G</td>
<td>0</td>
<td>3.1G</td>
<td>0%</td>
<td>/dev/shm</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_awips</td>
<td>529M</td>
<td>141M</td>
<td>362M</td>
<td>28%</td>
<td>/awips</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_awips2</td>
<td>5.2G</td>
<td>1.3G</td>
<td>3.7G</td>
<td>27%</td>
<td>/awips2</td>
</tr>
<tr>
<td>/dev/sda3</td>
<td>204M</td>
<td>65M</td>
<td>129M</td>
<td>34%</td>
<td>/boot</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_home</td>
<td>260M</td>
<td>12M</td>
<td>236M</td>
<td>5%</td>
<td>/home</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_tmp</td>
<td>529M</td>
<td>18M</td>
<td>485M</td>
<td>4%</td>
<td>/tmp</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_usr</td>
<td>16G</td>
<td>6.2G</td>
<td>8.6G</td>
<td>43%</td>
<td>/usr</td>
</tr>
</tbody>
</table>
2.2.1.2 Synchronous Communications (Radar)

- **Switch Panel**

The switch panel provides redundant, remotely controllable communications interfaces for the PX asynchronous multiplexers (mux). Specifically, the switch panel switches incoming and outgoing signals from the primary to the spare PX mux or ports when directed. The switch panel is controlled locally by the AWIPS Terminal Server and remotely from the NCF through the AWIPS Terminal Server. The switch panel provides a passive patch capability for each communications line. This allows communications problems to be monitored and diagnosed without disconnecting any communications equipment.

The VIR switches provide the ability to switch between primary and backup equipment. The switches may be operated remotely (by the NCF) or manually at the site (by the system managers). There are redundant connections and pathways to all...
AWIPS hardware except between the antenna and the DVB receivers (there is only one IFL).

- **Modem Nest**

  AWIPS accommodates a variety of site-specific communications requirements through a high-reliability rack-mounted Modem Nest with redundant power supplies. The Modem Nest supports up to 16 single modem cards. Every site has either a 10-modem-card modem nest or a 16-modem-card modem nest.

### 2.2.1.3 Application Server

Application servers have been decommissioned at all sites except the NCF.

### 2.2.1.4 Data Servers

The data server systems support the functions for processing, archiving, and retrieval. The processing and storage functions receive and store data that were acquired and distributed via the multiple CPs and service data requests to all workstations. The archiving functions provide general capabilities for retrieving site archive data, storing data in the site archive (both automatic and requested storage), and retrieving data from the site archive. The data server hardware components are the DX (1-4) servers, the archive servers (WAX and RAX), and the network attached storage (NAS).

- **Data Server Processors**

  Four Linux-based DX processors (two primary and two secondary) are deployed at every AWIPS site. The DXs provide the centralized computing and centralized database functions. They support the processing required to store and process many types of data and products that are acquired, derived, or manually entered. They also store software, map backgrounds, and site-specific data.

  The PostgreSQL object-relational database management system is supported by Network Appliance (NetApp) FAS2020C rack-mounted mass storage. These devices are direct-connected to DL380G6 Linux Data Servers (DX1 and DX2). Refer to Appendix E, Mass Storage Design, for a description of the disk allocations.

  DX1 and DX2 function as a heartbeat cluster; failover is managed automatically or manually. Note that both DX1 and DX2 normally run separate high-availability packages: a2dx1apps on DX1 and a2dx2apps on DX2. In failover mode, both packages shift to the same server. Access to DX1 and DX2 is via server name alias: dx1f points to the server running the a2dx1apps package; dx2f points to the server running the a2dx2apps package. Note that the floating names are available only when the high-availability packages are running; they cannot be used to start or stop the high-availability packages.

  DX3 and DX4 are independent servers, both of which run the EDEX server software. There is no failover between DX3 and DX4. Both servers are load balanced and share
data processing responsibilities; if one server is stopped, the other will pick up the entire data processing load. Processing load balancing between DX3 and DX4 is provided by QPID as data messages are removed from QPID queues and processed by the EDEX instance having available processing capacity. Client request balancing for DX3 and DX4 is managed by Internet Protocol Virtual Server (IPVS) (the pulse service). Both QPID and IPVS run on CPSBN1 and CPSBN2.

Additional information on IPVS, including monitoring tools, is provided in Chapter 25, AWIPS II/EDEX Administration Guide. The DX configurations are summarized in Table 2.2.1.4-1. Each Linux DX is connected to the Gigabit LAN.

### Table 2.2.1.4-1. Data Server Configurations at AWIPS Sites

<table>
<thead>
<tr>
<th>Type</th>
<th>Site</th>
<th>Processor</th>
<th>RAM</th>
<th>CACHE</th>
<th>Internal Storage</th>
</tr>
</thead>
<tbody>
<tr>
<td>HP DL380 G6</td>
<td>All Sites (DX server)</td>
<td>2x2.26 GHz Xeon CPU Quad Core</td>
<td>12 GB</td>
<td>1 MB L2 cache</td>
<td>2x146 GB</td>
</tr>
<tr>
<td>Dell PowerEdge 2950</td>
<td>All AWIPS Sites</td>
<td>Dual Quad Core 2.33 GHz</td>
<td>8 GB SDRAM</td>
<td>6 MB</td>
<td>2X 146 GB</td>
</tr>
</tbody>
</table>

- **River Ensemble Processors (REP)**

The River Ensemble Processor (REP) suite, part of the NWS’ Advanced Hydrologic Prediction Service (AHPS) program, was designed as a high-performance “blank slate” on which the River Forecast Centers can load new or site-modified applications in an environment that is very similar to AWIPS. It includes an availability infrastructure that allows failing-over of processes and crons if desired. This hardware hosts “baseline” Office of Hydrology (OH) applications implemented via AWIPS or AHPS, as appropriate. The REP provides the RFCs with a platform that was designed for easy expansion based on mission requirements. The REP suite is a self-contained unit that includes a rack, a NAS device, two commodity servers, and a GbE LAN. The REP was preconfigured, including IP addresses, to facilitate installation and allow the RFCs to take advantage of the additional processing power as soon as desired. The two Linux Operating System commodity servers have a file system structure that is consistent with current operations.

The REP NAS provides storage at the RFCs for the hydrologic data for the prediction models run on the REP processors. The new RFC NAS is a clustered Sun STK5320 with two shelves (where WFOs only have one shelf) of 16x500GB, per shelf (8TB), for a total raw capacity of 16TB. It is connected to the Gigabit Ethernet and has two FibreChannel adapters. The file system structure of the NAS is consistent with current operations and is ready to support installation of hydrology software.

The hardware specifications for the REP are as follows.

- Dell PowerEdge 2950 (RFC) (REP)
- Dual Quad Core CPU
- 8 GB DDR, 6 MB L2 cache
- Hard disk drive, 146 GB [2 x 146 GB] RAID 1
- **WFO Archive Server**

The WFO AX (WAX) (DS1E) is a Linux-based device, currently used only for tape backup. It runs the weekly nas backup script because the tape device is connected to it.

The WAX configurations are summarized in Table 2.2.1.4-2.

**Table 2.2.1.4-2. WFO Archive Server Configuration at AWIPS Sites**

<table>
<thead>
<tr>
<th>Type</th>
<th>Site</th>
<th>Processor</th>
<th>RAM</th>
<th>CACHE</th>
<th>Internal Storage</th>
</tr>
</thead>
<tbody>
<tr>
<td>HP ML350</td>
<td>Stand-alone WFOs and Collocated WFOs</td>
<td>1X Dual Core 2.66 GHz CPU</td>
<td>1 GB</td>
<td>512 KB</td>
<td>3X 146 GB</td>
</tr>
</tbody>
</table>

An RS-232-C interface connects the WAX processor console port to the site CP/monitor and control interface for monitor control.

- **RFC Archive Server**

The RFC AX (RAX) (DS1E) is a Linux-based device dedicated to providing additional disk storage capacity and processing power for hydrologic data. The server hosts Postgres and has the processing power and associated software to process large collections of hydrologic data. It is not a redundant server; however, it does have an SCSI RAID Level 5 storage array. It also has a DVD/CDR, and data can be selected and moved to more permanent storage or written to a CD or DVD.

The RAX configurations are summarized in Table 2.2.1.4-3.

**Table 2.2.1.4-3. RFC Archive Server Configuration at AWIPS Sites**

<table>
<thead>
<tr>
<th>Type</th>
<th>Site</th>
<th>Processor</th>
<th>RAM</th>
<th>CACHE</th>
<th>Internal Storage</th>
</tr>
</thead>
<tbody>
<tr>
<td>HP ML350</td>
<td>RFCs</td>
<td>1X Dual Core 2.66 GHz CPU</td>
<td>4 GB</td>
<td>512 KB</td>
<td>6X 146 GB</td>
</tr>
</tbody>
</table>

An RS-232-C interface connects the RAX processor console port to the site CP/monitor and control interface for monitor control.

- **Network Attached Storage**

The NAS (DS1F) provides storage at AWIPS sites and at the RFCs, storage for REP hydrologic data as well. The NAS is a Sun ST5320 with 8TB (WFO) or 16TB (RFC) of storage. It is connected to the Gigabit Ethernet.

- **Archive Storage**

A portable 1 TB Hard Disk Drive (HDD) with eSATA interface to the Archive Server has been added to the Archive Storage system, a prerequisite for AWIPS II. The DX and REP Tape Drive’s FibreBridge connection has been removed; the tape drive is now directly connected to the Archive Server via the LVD SCSI interface.
2.2.1.4.1 Data Server File Systems and Raw Partitions

The DX file systems support the following:

- Operating system
- COTS applications
- Developed applications.

2.2.1.4.1.1 File Systems on the Linux Data Server (DX)

The major file systems on the Linux-OS DXs are:

- `/root`, `/tmp`, `/usr`, `/var`. Linux mandates that these file systems exist.
- `/boot`. This file system contains the Linux kernel and boot-up instructions.
- `/dev/shm`. This file system is the Linux shared memory.
- `/awips/fxa`. This file system stores baselined Forecast Systems Laboratory (FSL)-developed (presently Earth System Research Laboratory/Global Systems Division (ESRL/GSD)) WFO-Advanced software and any associated log or configuration files. It may contain binary applications, scripts, logs, or data that are required to support operations.
- `/awips/ifps`. This file system is for baselined IFPS software and data.
- `/awips/ldad`. This file system contains scripts and data for LDAD server.
- `/awips/ops`. This file system stores baselined contractor-developed software and any associated log or configuration files. It may contain binary applications, scripts, logs, or data as required to support the operations.
- `/data/logs`. This file system contains the operational logs for software executing on the platform. Subdirectories will further isolate application- or product-specific logs.
- `/awips2`. This file system is used to store baseline AWIPS II software.
- `/awips2/data`. Contains database files (only on DX1/2)

The following directory is mounted on the DXs from the NAS:

- `/data_store`. Folders are usually laid out exactly like the sbn folders on the EDEX server with each plug-in having a folder on the data store. But some of them do not follow the same convention, for e.g., data sent to the 'metar' endpoint will be found in the `/data_store/text` folder. See Chapter 6 for more details on other data types. Additionally, if new format ingest is being worked, you will also find new data types not yet found on the development or integration systems. Those files will be located in `/data_store/experimental`.

The following directories are also mounted on the DX:
- /awips/dev. This file system contains the source code and libraries required for local software development at field sites. The /awips/dev file system is NFS-exported to support development from the workstations.

- /awips/adapt. This file system contains the AWIPS Decision Assistance Production Preparation Tools (ADAPPT) software for the most recent software installation at WFO sites. This includes the Hourly Weather Roundup (HWR), Climate, the Interactive Forecast Preparation System (IFPS), the Local AWIPS Model Output Statistics (MOS) Program (LAMP), and Pre-LAMP software. This file system may contain binary applications, scripts, logs, or data.

- /awips/GFESuite. This file system contains the GFESuite software. This file system may contain binary applications, scripts, logs, or data.

- /awips/hydroapps. This file system contains the NWS WFO Hydrologic Forecast System (HFS) and NWS River Forecast System (NWSRFS) baseline software for the most recent software installation.

- /awips/ops. This file system stores baselined contractor-developed software and any associated log or configuration files. It may contain binary applications, scripts, logs, or data as required to support the operations.

- /awips/rep. This file system (RFCs only) is mounted from the secondary NAS(NS1) and is used for REP.

- /data/fxa. This file system stores AWIPS data products that are not maintained within the RDBMS. This would include the LDAD data.

- /data/GFE. This file system (WFOs only) stores the GFESuite data products.

- /awips/db. This file system stores database software.

- /data/local. This file system contains locally acquired or site-specific data and is the file system that sites should use to store locally developed software. Each site specifies what is included in this file system, so it may contain binary applications, scripts, logs, or data.

- /data/x400. This directory contains the Message Handling System (MHS) files.

- /data/adapt. This file system (WFOs only) contains the IFPS, GFESuite, and LAMP data, including GIF and text file products, topographic maps, shapefiles, IFPS database grids and maps, and GFESuite databases, products, and topographic maps at WFO sites.

- /home. This file system contains all the user working areas.

- /DS_shared. This directory contains the GNU compiler collection, an obsolete logs directory, an MHS directory that is still in use and an obsolete HP service Guard Directory.

- /awips2. This file system is used to store baselined AWIPS II software.

- /awips/storage. A location for the storage of miscellaneous AWIPS related files.
- `/awips2/GFESuite`. Contains scripts and data relating to inter site coordination (ISC) and service backup.
- `/awips2/edex/data/utility`. Contains localization store and EDEX configuration files.

An example of mount points for a dx3/4 at the WFOs follows.

```
[root@dx3-tbdw ~]# df -H

Filesystem               Size  Used  Avail % Mounted on
/dev/mapper/vg01-lvol01  1.2G   529M  564M  50% /
tmpfs                   4.2G      0   4.2G  0% /dev/shm
/dev/mapper/vg01-lvol_fxa 5.5G   1.3G   4.2G  24% /awips/fxa
/dev/mapper/vg01-lvol_ifps 1.4G   82M  1.2G   7% /awips/ifps
/dev/mapper/vg01-lvol_ldad 1.1G   144M  837M  15% /awips/ldad
/dev/mapper/vg01-lvol_ops 285M  136M  135M  51% /awips/ops
/dev/mapper/vg01-lvol_awips2 41G  18G  21G  46% /awips2
/dev/sda3              102M   65M   36M  66% /boot
/dev/mapper/vg01-lvol_logs 2.2G   71M   2.1G  4% /data/logs
/dev/mapper/vg01-lvol_tmp 1.1G  19M   961M  2% /tmp
/dev/mapper/vg01-lvol_usr 11G   7.6G   2.3G  77% /usr
/dev/mapper/vg01-lvol_var 3.1G  329M   908M  12% /var
nas1:/awipsADAPT        1.1G  701M  374M  66% /awips/adapt
nas1:/awipsDEV          4.3G  132k  4.2G  1% /awips/dev
nas1:/awipsGFESuit      108G  108G  14G   17% /awips/GFESuite
nas1:/awipsHYDRO        108G  809M  26G   1% /awips/hydroapps
nas1:/dataLOCAL         162G  162G    0   0% /data/local
nas1:/dataX400          630M  630M    2M   0% /data/x400
nas1:/DSshared         1.1G  410M  830M   3% /DS_shared
nas1:/dataADAPT         3.3G   3.3G    0% /data/adapt
nas1:/dataGFES           21M    0   21M   0% /data/GFES
nas1:/awipsHOME         119G  119G  206M  79% /home
nas1:/dataFXA           537G  537G  148M  73% /data/fxa
nas1:/aiidata           537G  537G  148M  73% /awips2/edex/data
nas1:/GFESuite2         11G  11G   1G   1% /awips2/GFESuite
nas1:/localapps         54G  54G   0% /localapps
nas2:/dataSTORE        2.2T  2.2T   0% /data_store
nas2:/awips2ARCHIVE    3.3T  3.3T   0% /archive
```

An example of mount points for a dx3/4 at the RFCs follows.

```
[root@dx3-tbdr ~]# df -H

Filesystem               Size  Used  Avail % Mounted on
/dev/mapper/vg01-lvol01  1.2G   615M  540M  50% /
tmpfs                   4.2G      0   4.2G  0% /dev/shm
/dev/mapper/vg01-lvol_fxa 5.5G   911M  4.3G  18% /awips/fxa
/dev/mapper/vg01-lvol_ifps 1.4G   36M  1.3G   3% /awips/ifps
/dev/mapper/vg01-lvol_ldad 1.1G   91M  890M  10% /awips/ldad
/dev/mapper/vg01-lvol_ops 285M  204M  81M  25% /awips/ops
/dev/mapper/vg01-lvol_awips2 41G  39G  4.1k 100% /awips2
/dev/sda3              102M   35M   67M  37% /boot
```
```bash
[root@dx1-oax ~]# df -H

Filesystem                      Size  Used Avail Use% Mounted on
/dev/mapper/vg01-lvol01         1.1G  447M  534M  46% /
/tmpfs                          6.3G  5G  0  6.3G  0% /dev/shm
/dev/mapper/vg01-lvol_fxa       5.2G  1.9G  3.1G  38% /awips/orda
/dev/mapper/vg01-lvol_ifps      1.4G  82M  1.2G  7% /awips/ifps
/dev/mapper/vg01-lvol_ldad      1.1G  84M  927M  9% /awips/ldad
/dev/mapper/vg01-lvol_ops       305M  200M  90M  70% /awips/ops
/dev/mapper/vg01-lvol_awips2    34G  2.4G  30G  8% /awips2
/dev/mapper/mpathap3            102M  67M  30M  70% /boot
/dev/mapper/vg01-lvol_logs       11G  159M  9.9G  2% /awips/logs
/dev/mapper/vg01-lvol_tmp       1.1G  20M  961M  2% /tmp
/dev/mapper/vg01-lvol_usr        3.3G  8.3G  1.6G  85% /usr
/dev/mapper/vg01-lvol_var        3.1G  749M  1.6G  26% /var
/nas1:/awipsADAPT                1.1G  701M  374M  66% /awips/adapt
/nas1:/awipsDEV                  4.3G  132k  4.3G  1% /awips/dev
/nas1:/awipsGFESuit              108G  36G  72G  34% /awips/GFESuite
/nas1:/awipsHYDRO                108G  53G  55G  52% /awips/hydroapps
/nas1:/dataLOCAL                162G  22G  140G  14% /data/local
/nas1:/dataX400                   630M  13M  617M  2% /data/x400
/nas1:/DSshared                   1.6G  217M  1.4G  14% /DS_shared
/nas1:/dataADAPT                  3.3G  0  3.3G  0% /data/adapt
/nas1:/dataGFE                    21M  0  21M  0% /data/GFE
/nas1:/awipsHOME                  119G  58G  62G  49% /home
/nas1:/dataFXA                    537G  16G  522G  3% /awips2/edex/data
/nas1:/localapps                  54G  21M  54G  1% /localapps
/nas1:/dataSTORE                  537G  49G  489G  9% /data_store
/nas1:/GFESuite2                  11G  121M  11G  2% /awips2/GFESuite
```

```bash
 AWIPS System Architecture

Chapter 2

AWP.MAN.SMM.A2

2-32

– Hard copies uncontrolled. Verify effective date prior to use. –
```
An illustration of the mount points for a RFC DX1/2 follows.

```
[root@dx2-tbdr ~]# df -H
```

<table>
<thead>
<tr>
<th>Filesystem</th>
<th>Size</th>
<th>Used</th>
<th>Avail</th>
<th>Use%</th>
<th>Mounted on</th>
</tr>
</thead>
<tbody>
<tr>
<td>/dev/mapper/vg01-lvol01</td>
<td>1.1G</td>
<td>523M</td>
<td>458M</td>
<td>54%</td>
<td>/</td>
</tr>
<tr>
<td>tmpfs</td>
<td>6.3G</td>
<td>0</td>
<td>6.3G</td>
<td>0%</td>
<td>/dev/shm</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_fxa</td>
<td>5.2G</td>
<td>4.2G</td>
<td>737M</td>
<td>85%</td>
<td>/awips/fxa</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_ifps</td>
<td>1.4G</td>
<td>83M</td>
<td>1.2G</td>
<td>7%</td>
<td>/awips/ifps</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_ldad</td>
<td>1.1G</td>
<td>91M</td>
<td>890M</td>
<td>10%</td>
<td>/awips/ldad</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_ops</td>
<td>305M</td>
<td>71M</td>
<td>219M</td>
<td>25%</td>
<td>/awips/ops</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_awips2</td>
<td>34G</td>
<td>3.9G</td>
<td>29G</td>
<td>13%</td>
<td>/awips2</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_logs</td>
<td>11G</td>
<td>161M</td>
<td>9.9G</td>
<td>2%</td>
<td>/data/logs</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_tmp</td>
<td>11G</td>
<td>19M</td>
<td>962M</td>
<td>2%</td>
<td>/tmp</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_var</td>
<td>11G</td>
<td>8.9G</td>
<td>995M</td>
<td>90%</td>
<td>/usr</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_adapt</td>
<td>1.1G</td>
<td>172M</td>
<td>902M</td>
<td>17%</td>
<td>/var</td>
</tr>
<tr>
<td>nas1:/awips/DEV</td>
<td>4.3G</td>
<td>1.2M</td>
<td>4.3G</td>
<td>1%</td>
<td>/awips/dev</td>
</tr>
<tr>
<td>nas1:/awips/GFESuite</td>
<td>108G</td>
<td>26G</td>
<td>82G</td>
<td>24%</td>
<td>/awips/GFESuite</td>
</tr>
<tr>
<td>nas1:/awips/HYDRO</td>
<td>108G</td>
<td>53G</td>
<td>55G</td>
<td>50%</td>
<td>/awips/hydroapps</td>
</tr>
<tr>
<td>nas1:/data/LOCAL</td>
<td>162G</td>
<td>22G</td>
<td>140G</td>
<td>14%</td>
<td>/data/local</td>
</tr>
<tr>
<td>nas1:/data/X400</td>
<td>630M</td>
<td>13M</td>
<td>617M</td>
<td>2%</td>
<td>/data/X400</td>
</tr>
<tr>
<td>nas1:/DSshared</td>
<td>1.6G</td>
<td>217M</td>
<td>1.4G</td>
<td>14%</td>
<td>/DS_shared</td>
</tr>
<tr>
<td>nas1:/data/ADAPT</td>
<td>3.3G</td>
<td>0</td>
<td>3.3G</td>
<td>0%</td>
<td>/data/ADAPT</td>
</tr>
<tr>
<td>nas1:/data/GFESuite</td>
<td>21M</td>
<td>0</td>
<td>21M</td>
<td>0%</td>
<td>/data/GFESuite</td>
</tr>
<tr>
<td>nas1:/awips/HOME</td>
<td>119G</td>
<td>58G</td>
<td>62G</td>
<td>49%</td>
<td>/home</td>
</tr>
<tr>
<td>nas1:/data/FXA</td>
<td>537G</td>
<td>362G</td>
<td>175G</td>
<td>68%</td>
<td>/data/FXA</td>
</tr>
<tr>
<td>nas1:/aiidata/utility</td>
<td>537G</td>
<td>16G</td>
<td>522G</td>
<td>3%</td>
<td>/awips2/edex/data/utility</td>
</tr>
<tr>
<td>nas1:/localapps</td>
<td>54G</td>
<td>21M</td>
<td>54G</td>
<td>1%</td>
<td>/localapps</td>
</tr>
<tr>
<td>nas1:/data/STORE</td>
<td>537G</td>
<td>49G</td>
<td>489G</td>
<td>10%</td>
<td>/data/STORE</td>
</tr>
<tr>
<td>nas1:/storage</td>
<td>537G</td>
<td>86G</td>
<td>452G</td>
<td>16%</td>
<td>/awips/storage</td>
</tr>
<tr>
<td>/dev/sda3</td>
<td>102M</td>
<td>70M</td>
<td>27M</td>
<td>73%</td>
<td>/boot</td>
</tr>
<tr>
<td>/dev/mapper/vg_aiidb-awipsiidb</td>
<td>159G</td>
<td>20G</td>
<td>131G</td>
<td>14%</td>
<td>/awips2/data</td>
</tr>
<tr>
<td>nas1:/GFESuite2</td>
<td>11G</td>
<td>121M</td>
<td>11G</td>
<td>2%</td>
<td>/awips2/GFESuite</td>
</tr>
<tr>
<td>/dev/mapper/vg_aiihdf5-awipsiihdf5</td>
<td>422G</td>
<td>48G</td>
<td>354G</td>
<td>12%</td>
<td>/awips2/edex/data/hdf5</td>
</tr>
</tbody>
</table>

### 2.2.1.4.1.2 Raw Partitions for PostgreSQL on the Linux Data Server

The raw partition for PostgreSQL on the DX is mounted to the dx1f device from the FAS2020 Direct Attached Storage Device.

The following is an example of the PostgreSQL mount point on the dx1f.

```
[root@dx1-oax ~]# df -H /awips2/data
```
2.2.1.4.2 **AX File Systems**

The AX file systems support the following:

- Operating system
- COTS applications
- Developed applications
- Archived Data.

The major file systems on the AX for WFO and RFC are as follows:

- **/root**, **/tmp**, **/usr**, **/var**. Linux mandates that these file systems exist.
- **/boot**. This file system contains the Linux kernel and boot-up instructions.
- **/awips/archiver**. This file system stores the archiver data. It contains scripts, logs, configuration files, and documentation.
- **/local**. This file system contains scripts and logs related to the installation of AWIPS software.
- **/data**. This file system contains the archiver (5-day rollover and compressed) and local radar data.
- **/dev/shm**. This file system is the Linux shared memory input queue.
- **/awips/ops**. This file system stores baselined software and any associated log or configuration files. It may contain binary applications, scripts, logs, or data as required to support operations.
- **/awips/flow**. This file system stores baselined FSL-developed (presently ESRL/GSD) WFO-Advanced software and any associated log or configuration files. It may contain binary applications, scripts, logs, or data, as required to support operations.
- **RFC Mount Points**
  - **/rfc_arc**. This file system is at RFCs only and may contain binary applications, scripts, logs or data for RFC hydrological operations.
  - **/rfc_arc_data**. This file system is at RFCs only and contains flat files of archived hydrological data and products for RFC hydrological operations.
- **NFS Mount Points**
  - The following directory is mounted by the AX server from the NAS in support of AWIPS functions.
- /data_store
  - The following directories are mounted by the AX server from the NAS in support of AWIPS functions.
  - /data/fxa
  - /data/local
  - /home

- RFCs only
  - /awips/rep
  - /awips/hydroapps
  - /home

Refer to Chapter 24, Data Archive Server, for additional information.

An example of mount points for a WFO AX follows.

```
[root@ax-tbdw ~]# df -H
Filesystem     Size   Used  Avail  Use% Mounted on
/dev/mapper/vg00-lvol01  1.1G   578M   410M  59% /
/dev/mapper/vg00-lvol08  1.1G    23M   949M   3% /awips/archiver
/dev/mapper/vg00-lvol06  1.1G   693M   279M  72% /awips/fxa
/dev/mapper/vg00-lvol10 309M    65M   229M  23% /awips/ops
/dev/cciss/c0d0p1      104M    45M    54M  46% /boot
/dev/mapper/vg00-lvol11 265G   101G   151G  40% /data
none              530M      0   530M   0% /dev/shm
/dev/mapper/vg00-lvol12 2.1G    37M   1.9G  2% /local
/dev/mapper/vg00-lvol02 520M    22M   472M   5% /tmp
/dev/mapper/vg00-lvol03  9.2G    6.6G    2.6G  44% /usr
/dev/mapper/vg00-lvol04  3.1G    1.4G    1.6G  46% /var
nas1:/dataFXA  537G    377G    161G  71% /data/fxa
nas1:/dataLOCAL 162G    69G     93G  43% /data/local
nas1:/awipsHOME 119G    99G     20G  84% /home
nas1:/localapps  54G    6.4M   54G   1% /localapps
nas1:/aiidata/utility 537G    25G    513G   5% /awips/edex/data/utility
nas1:/dataSTORE  1.1T    48G    1.1T   5% /data_store
```

An example of mount points for a RFC AX follows.

```
[root@ax-tbdr ~]# df -H
Filesystem     Size   Used  Avail  Use% Mounted on
/dev/mapper/vg01-lvol01  1.1G   308M   696M  31% /
```
### 2.2.1.5 **LDAD Hardware**

The LDAD System consists of two LDAD servers (LS2/3), a LAN switch (SMC 8024), a Terminal Server (Cyclades ACS32), Modems (MultiTech MT5600BR), and a LAN DMZ (HP ProCurve 2824). The DMZ consists of two Juniper Netscreen 25 Firewalls, a Netgear 16 switch, and two Netgear 5 port hubs. The LDAD baseline processes run on the LDAD Cluster (DMZ) and the AWIPS PX Cluster (Internal). Other local applications may run on other internal clusters, such as DX cluster in the case of the LDAD Dissemination Server. Data is transmitted through the DMZ either to the Trusted (internal) AWIPS system or to the Untrusted (External) Users/Systems.

- **LDAD Terminal Server**

  The LDAD Terminal Server is a Cyclades Alterpath ACS32, which provides an Ethernet interface for connection to the LDAD LAN Switch and 32 serial ports for connections to communication devices, such as the modems in the Modem Nest, and Console Management ports such as the LS2/3 Servers. Each port can be configured to a maximum speed of 56 K. The terminal server in an average configuration is used to connect 10 dial-in/dial-out modems (including MicroART, RRS, and ASOS), four dedicated modems, a fax modem, and various console connections. Port assignments are shown in Table 2.2.1.5-1 as an example.

<table>
<thead>
<tr>
<th>Directory</th>
<th>Size</th>
<th>Used</th>
<th>Available</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>/tmp</td>
<td>2.1G</td>
<td>70k</td>
<td>2.1G</td>
<td>1%</td>
</tr>
<tr>
<td>/dev/shm</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_ops</td>
<td>309M</td>
<td>39M</td>
<td>255M</td>
<td>14%</td>
</tr>
<tr>
<td>/dev/cciss/c0d0p3</td>
<td>102M</td>
<td>71M</td>
<td>26M</td>
<td>74%</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_local</td>
<td>2.1G</td>
<td>37M</td>
<td>2.0G</td>
<td>2%</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_rfc_arc</td>
<td>2.1G</td>
<td>451M</td>
<td>1.6G</td>
<td>24%</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_rfc_arc_data</td>
<td>5.2G</td>
<td>160M</td>
<td>4.8G</td>
<td>4%</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_tmp</td>
<td>529M</td>
<td>18M</td>
<td>485M</td>
<td>4%</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_usr</td>
<td>16G</td>
<td>7.9G</td>
<td>6.9G</td>
<td>54%</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_var</td>
<td>3.2G</td>
<td>728M</td>
<td>2.3G</td>
<td>25%</td>
</tr>
<tr>
<td>/dev/mapper/vg00-lvol_data</td>
<td>658G</td>
<td>300G</td>
<td>325G</td>
<td>49%</td>
</tr>
<tr>
<td>nasl:/dataFXA</td>
<td>537G</td>
<td>362G</td>
<td>175G</td>
<td>68%</td>
</tr>
<tr>
<td>nasl:/dataLOCAL</td>
<td>162G</td>
<td>22G</td>
<td>140G</td>
<td>14%</td>
</tr>
<tr>
<td>nasl:/awipsHOME</td>
<td>119G</td>
<td>58G</td>
<td>62G</td>
<td>49%</td>
</tr>
<tr>
<td>nasl:/awipsHYDRO</td>
<td>108G</td>
<td>53G</td>
<td>55G</td>
<td>50%</td>
</tr>
<tr>
<td>nasl:/aiidata/utility</td>
<td>537G</td>
<td>16G</td>
<td>521G</td>
<td>3%</td>
</tr>
<tr>
<td>/dev/cciss/c0d0p1</td>
<td>102M</td>
<td>69M</td>
<td>28M</td>
<td>72%</td>
</tr>
<tr>
<td>/dev/mta</td>
<td>985G</td>
<td>22G</td>
<td>914G</td>
<td>3%</td>
</tr>
</tbody>
</table>

---

---
Table 2.2.1.5-1. Port Assignments

<table>
<thead>
<tr>
<th>Port</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Used to dial-out to devices configured for 7 bits, even parity</td>
</tr>
<tr>
<td>2</td>
<td>Used to dial-out to devices configured for 8 bits, no parity</td>
</tr>
<tr>
<td>3</td>
<td>First dial-in line, 8 bits, no parity</td>
</tr>
<tr>
<td>4</td>
<td>Second dial-in line, 8 bits, no parity</td>
</tr>
<tr>
<td>5</td>
<td>Third dial-in line, 8 bits, no parity</td>
</tr>
<tr>
<td>6</td>
<td>Fourth dial-in line, 8 bits, no parity</td>
</tr>
<tr>
<td>7</td>
<td>Fifth dial-in line, 8 bits, no parity</td>
</tr>
<tr>
<td>8</td>
<td>Sixth dial-in line, 8 bits, no parity</td>
</tr>
<tr>
<td>9</td>
<td>MicroART Modem</td>
</tr>
<tr>
<td>10</td>
<td>ASOS Modem</td>
</tr>
<tr>
<td>21</td>
<td>First Dedicated Modem</td>
</tr>
<tr>
<td>22</td>
<td>Second Dedicated Modem</td>
</tr>
<tr>
<td>23</td>
<td>Third Dedicated Modem</td>
</tr>
<tr>
<td>24</td>
<td>Fourth Dedicated Modem</td>
</tr>
<tr>
<td>30</td>
<td>Fax Modem</td>
</tr>
</tbody>
</table>

The LDAD Server system is configured with a block of IP addresses assigned by the site. These Class C IP Addresses were assigned to the site (non-AWIPS IP addresses) by the NWS, and generally a block of 15 IP addresses should be set aside specifically for LDAD usage. IP addresses are required for the Terminal Server, software services (i.e., Radius) running on the Terminal Server, certain ports on the Terminal Server (dependent upon function), the LDAD firewalls, and the NAT (Network Address Translation) Table stored on the firewalls.

- **LAN Switch**
  The LDAD LAN Switch is an SMC Networks 8024 Ethernet switch with 24 1000BaseT ports. The LAN Switch provides connectivity to external users/systems, the LDAD Terminal Server (Management/Modem RS-232), and the Untrusted side of the LDAD Firewalls (FW1/2). Refer to Appendix L, LDAD Configuration Samples and Firewall Architecture, for more information about the LDAD Firewall.

- **Asynchronous Interface**
  The asynchronous interface provides up to 4 asynchronous serial connections between the LDAD Server and external interfaces.

- **Modem Nest**
  The LDAD modems are located in the MultiTech chassis. A total of 16 slots are available for MultiTech modem cards (MT5600BR) and two power supplies (PS1600). All sites have at least 10 modems for dial-out, dial-in, and fax capability.

- **Firewall**
  Two Juniper SSG320M firewalls are employed to provide network security for the LDAD and AWIPS systems. Each AWIPS site will have two firewalls controlled by
central configuration management servers. The central configuration management servers are located at the NCF, and at the BNCF in Fairmont, West Virginia. Control over the firewall configurations is tiered, with the central configuration management server having ultimate control. The regions have a client Netscreen Security Manager (NSM) that connects to the NCF Central Server, which in turn connects back to the firewall. Individual AWIPS sites (forecast offices) will not have direct control over their own configurations.

The firewalls are stateful inspection firewalls, and use NAT to prevent AWIPS IP addresses from being advertised outside of the AWIPS network. (Refer to Appendix L for more on Firewall Architecture.)

- **LDAD Server**

  The LDAD server cluster is a pair of HP ProLiant DL320 Linux servers that act as a primary and hot spare. The LDAD server machines provide the focal point for all external communications between the AWIPS site and the community, functioning as a pass-through device for all incoming and outgoing data.

  Only one of the two machines (ls2 and ls3) is actively working as the LDAD server at any time, and it will be the one that is reachable through the address 192.168.1.10 ("ls1" or "ls") via the LDAD LAN switch. In this document, “the active LS” refers to whichever machine is actively working as the LDAD server.

  The LS1 LDAD server (LS2/LS3 listening address) interface is configured on both LDAD servers but it will only be up on the active LS. The active LS is running LDAD ingest processes, listening on the address 192.168.1.10, and reachable through the AWIPS LAN via the hostname “ls1” or "ls."

  The ls2/ls3 servers have unique interfaces (ls2-192.168.1.11, ls3-192.168.1.12) that are up at all times. The ls2/3 heartbeat is a private lan pair (ls2-172.16.0.2, ls3-172.16.0.1).

  The ls2/ls3 servers share an external IP address for access to the site local LAN and the Internet for data push/pull collection. This external ldad address is assigned by the site and configured by the region in the site's LDAD firewall. Packets are routed by the firewall using Mapped IP (MIP) to the active LS. There is no external network interface connection on the LDAD server.

  The VIR switch provides the capability to switch asynchronous serial connections between LS2 and LS3.

2.2.1.5.1 **LDAD Server File Systems**

- **root (/), /tmp, /usr, /var.** These file systems are maintained as separate logical volumes to reduce the risk of any of these file systems filling up and affecting system operations. All of the remaining file systems are ext3. The /tmp file system will be re-created with each boot sequence and performs no journaling.
• **swap space.** The swap space is divided into two LVs (Logical Volume) on two separate spindles. The primary swap space on the root drive is also designed as a dump space in the event of a system panic. This is designated as a swap/dump file system.

• **/opt.** This is used to load COTS applications.

• **/data/logs.** This file system contains the operational logs for software executing on this platform. Subdirectories will further isolate application- or product-specific logs. It also stores all non-system LDAD logs (all FSL- and OST-developed software logs).

• **/ldad.** This file system stores baselined FSL-developed (presently ESRL/GSD) software and any configuration files. It may contain binary applications, scripts, or data as required to support LDAD functions.

• **/data/ldad.** This file system contains scripts and data for LDAD server.

• **/data/Incoming.** This file system is used for temporary storage of raw data acquired from external observation systems (e.g., local automatic remote collector [LARC] gauges, mesonet sensors, and precipitation gauges).

• **/home.** This file system contains all of the user working areas.

An example of mount points on the LDAD servers follows.

```bash
[root@ls1-tbw3 ~]# df -H
```

<table>
<thead>
<tr>
<th>Filesystem</th>
<th>Size</th>
<th>Used</th>
<th>Avail</th>
<th>Use%</th>
<th>Mounted on</th>
</tr>
</thead>
<tbody>
<tr>
<td>/dev/mapper/vg01-lvol01</td>
<td>1.1G</td>
<td>428M</td>
<td>553M</td>
<td>44%</td>
<td>/</td>
</tr>
<tr>
<td>tmpfs</td>
<td>6.3G</td>
<td>463k</td>
<td>6.3G</td>
<td>1%</td>
<td>/dev/shm</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_fxa</td>
<td>4.7G</td>
<td>2.0G</td>
<td>2.5G</td>
<td>45%</td>
<td>/awips/fxa</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_ifps</td>
<td>1.6G</td>
<td>83M</td>
<td>1.4G</td>
<td>6%</td>
<td>/awips/ifps</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_ldad</td>
<td>1.1G</td>
<td>75M</td>
<td>906M</td>
<td>8%</td>
<td>/awips/ldad</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_ops</td>
<td>305M</td>
<td>84M</td>
<td>206M</td>
<td>30%</td>
<td>/awips/ops</td>
</tr>
<tr>
<td>/dev/sda3</td>
<td>102M</td>
<td>63M</td>
<td>34M</td>
<td>65%</td>
<td>/boot</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_data</td>
<td>4.2G</td>
<td>77M</td>
<td>3.9G</td>
<td>2%</td>
<td>/data</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_dhm</td>
<td>1.1G</td>
<td>19M</td>
<td>962M</td>
<td>2%</td>
<td>/data/dhm</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_local</td>
<td>2.1G</td>
<td>37M</td>
<td>2.0G</td>
<td>2%</td>
<td>/local</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_tmp</td>
<td>508M</td>
<td>12M</td>
<td>471M</td>
<td>3%</td>
<td>/tmp</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_usr</td>
<td>16G</td>
<td>11G</td>
<td>4.0G</td>
<td>74%</td>
<td>/usr</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_var</td>
<td>3.1G</td>
<td>307M</td>
<td>2.7G</td>
<td>11%</td>
<td>/var</td>
</tr>
<tr>
<td>nas1:/awipsADAPT</td>
<td>1.1G</td>
<td>701M</td>
<td>374M</td>
<td>66%</td>
<td>/awips/adapt</td>
</tr>
<tr>
<td>nas1:/awipsDEV</td>
<td>4.3G</td>
<td>173k</td>
<td>4.3G</td>
<td>1%</td>
<td>/awips/dev</td>
</tr>
<tr>
<td>nas1:/awipsHOME</td>
<td>119G</td>
<td>94G</td>
<td>25G</td>
<td>79%</td>
<td>/home</td>
</tr>
<tr>
<td>nas1:/awipsHYDRO</td>
<td>108G</td>
<td>809M</td>
<td>107G</td>
<td>1%</td>
<td>/awips/hydroapps</td>
</tr>
<tr>
<td>nas1:/dataADAPT</td>
<td>3.3G</td>
<td>0</td>
<td>3.3G</td>
<td>0%</td>
<td>/data/adapt</td>
</tr>
<tr>
<td>nas1:/dataFXA</td>
<td>537G</td>
<td>390G</td>
<td>148G</td>
<td>73%</td>
<td>/data/fxa</td>
</tr>
<tr>
<td>nas1:/dataLOCAL</td>
<td>162G</td>
<td>46G</td>
<td>116G</td>
<td>29%</td>
<td>/data/local</td>
</tr>
<tr>
<td>nas1:/awipsGFESuite/ws</td>
<td>108G</td>
<td>18G</td>
<td>90G</td>
<td>17%</td>
<td>/awips/GFESuite</td>
</tr>
<tr>
<td>nas1:/verify</td>
<td>2.2T</td>
<td>3.1G</td>
<td>2.2T</td>
<td>1%</td>
<td>/data/verify</td>
</tr>
<tr>
<td>nas1:/localapps</td>
<td>54G</td>
<td>39M</td>
<td>54G</td>
<td>1%</td>
<td>/localapps</td>
</tr>
<tr>
<td>nas1:/alidata/share</td>
<td>537G</td>
<td>12G</td>
<td>526G</td>
<td>3%</td>
<td>/awips2/edex/data/share</td>
</tr>
<tr>
<td>nas2:/dataSTORE</td>
<td>2.2T</td>
<td>49G</td>
<td>2.2T</td>
<td>3%</td>
<td>/data_store</td>
</tr>
<tr>
<td>nas2:/awips2ARCHIVE</td>
<td>3.3T</td>
<td>11G</td>
<td>3.3T</td>
<td>1%</td>
<td>/archive</td>
</tr>
<tr>
<td>nas1:/dataARCHIVER</td>
<td>1.1T</td>
<td>4.8G</td>
<td>1.1T</td>
<td>1%</td>
<td>/data/archiver</td>
</tr>
</tbody>
</table>

---

*AWP.MAN.SMM.A2* 2-39

– Hard copies uncontrolled. Verify effective date prior to use. –
2.2.1.6 Workstations

Workstations (see Exhibit 2.2.1.6-1) provide the primary human/machine interface in AWIPS. They perform a variety of mission-related and system management functions, including the display of alphanumeric, image, and graphic data; animation displays; and toggling, zooming, and panning displays. It has four HWCs: WK1B; WK1C, Color X-Terminal; WK1F, Type IV Workstation (NCF only); and WK1G, Linux Workstation.

AWIPS forecasters use the Linux workstations on mission-critical functions. Upgraded Linux OS X-terminals have replaced the HP Color X-Terminals. New Workstations + X-terminals have three 19-inch LCD monitors and an internal CD-ROM, which support all AWIPS workstations.

```
[root@ls1-tbdr ~]# df -H

Filesystem           Size  Used  Avail %Mounted on
/dev/mapper/vg01-lvol01 22G   1.3G   19G   7%  /
/tmpfs               6.3G    0   6.3G   0%  /dev/shm
/dev/mapper/vg01-lvol_awips2 11G  158M   9.9G   2%  /awips2
/dev/sda3            102M   35M   62M  37%  /boot
/dev/mapper/vg01-lvol_Incoming 22G  181M   20G   1%  /data/Incoming
/dev/mapper/vg01-lvol_logs 53G  476M   50G   1%  /data/logs
/dev/mapper/vg01-lvol_home 22G  986M   20G   5%  /home
/dev/mapper/vg01-lvol_ldad 22G   1.1G   20G   6%  /ldad
/dev/mapper/vg01-lvol_opt 22G  332M   20G   2%  /opt
/dev/mapper/vg01-lvol_tmp  5.3G  151M   4.9G   3%  /tmp
/dev/mapper/vg01-lvol_usr 27G   7.2G   18G  29%  /usr
/dev/mapper/vg01-lvol_var 16G  390M   15G   3%  /var
/dev/mapper/vg02-lvol_dataLdad 492G  18G  449G   4%  /data/lda
```
All data are centrally stored and managed at the site. Data in use at the workstation, such as image or graphic products, are retrieved from the Data Server or the Linux Preprocessor and presented at the workstation. The product may be manipulated for display and retained at the workstation for as long as it is being used. Upon completion, it is removed from the local workstation unless the user or the application retains it.

The color X-terminal (WK1C) is a Linux-based workstation platform with one 19-inch monitor. It is connected to the GBLAN and is used primarily for text messaging functions. The Linux workstations (WK1G HWC) support a processing capacity of at least 48 MIPS and include online storage for applications software. They provide high-speed 100 Mbps connectivity to the site LAN and support other interfaces.

The AWIPS Linux workstations are equipped with three 19-inch LCD monitors and support all AWIPS workstation applications.

Workstations are sized to handle the WK HWCI load and are expandable to meet functional and performance requirements as they evolve throughout the AWIPS life cycle. Multiple workstations at each site provide for redundancy. If a workstation fails, a forecaster can move to another workstation and resume work.

The 100 Mbps interface connects to the site LAN (HWCI) via Ethernet for the Linux workstation. The RS-232-C interface connects to the site Monitor and Control Interface HWC.
The file systems on the workstations support the operating system, COTS applications, and developed applications. The workstation configuration has no database requirement; therefore, it does not need to support any raw partitions.

Table 2.2.1.6-1 summarizes the workstation configurations.

<table>
<thead>
<tr>
<th>Type</th>
<th>Site</th>
<th>Processor</th>
<th>RAM</th>
<th>CACHE</th>
<th>Internal Storage</th>
</tr>
</thead>
<tbody>
<tr>
<td>HP Z600</td>
<td>All WFO/RFCs (LX Replacement)</td>
<td>1 CPU @ 2.26 GHZ</td>
<td>12 GB</td>
<td>8 MB</td>
<td>146 GB</td>
</tr>
<tr>
<td>HP XW4400</td>
<td>All WFO/RFCs (X-term replacement)</td>
<td>1 CPU @ 2.4 GHZ</td>
<td>6 GB</td>
<td>2 MB</td>
<td>160 GB</td>
</tr>
</tbody>
</table>

2.2.1.6.1 Reserved

2.2.1.6.2 Linux Workstation File Systems

- **root** ( / ), **/tmp**, **/usr**, and **/var**. Linux mandates that these file systems exist.
- **/boot**. This file system stores the Linux kernel and boot-up instructions.
- **swap space**. Swap space is a dedicated partition defined in /etc/fstab and activated at boot time. Swap space is used to store inactive pages of memory when system RAM is full.
- **/local**. This file system contains the output files for the Linux configuration and setup and the Red Hat, GFESuite, and WFOA software installations.
- **/awips/chps_local**. Reserved for CHPS.
- **/chps**. Reserved for CHPS.
- **/dev/shm**. This file system is the Linux shared memory input queue.
- **/awips/ifps**. This file system is for baselined IFPS software and data.
- **/awips/ldad**. This file system will store baselined FSL-developed WFO-Advanced software and any associated configuration files. It may contain binary applications, scripts, or data as required to support operations.
- **/awips/fxa**. This file system stores baselined FSL-developed (presently ERL/GSD) WFO-Advanced software and any associated logs or configuration files. It may contain binary applications, scripts, logs, or data as required to support operations.
- **/awips/ops**. This file system stores baselined Contractor-developed software and any associated log or configuration files. It may contain binary applications, scripts, logs, or data as required to support operations.
- **/data**. This file system stores data products needed on the workstation.
- **data/dhm**. Data Hydrological Modeling on LX.
- **NFS Mount Points.** The following directory is mounted from the NAS1 in support of AWIPS functions:
  - `/data_store`

- The following directories are mounted from the NAS in support of AWIPS functions.
  - `/awips/adapt`
  - `/awips/dev`
  - `/data/adapt` (WFOs only)
  - `/data/fxa`
  - `/awips/hydroapps`
  - `/awips/GFESuite`
  - `/data/local`
  - `/data/verify`
  - `/home`
  - `/usr/local/viz/edex/data/hdf5`
  - `/awips/rep` (RFCs only for REP)
  - `/awips/chps_share` (RFCs only)

The following is an example of WFO Linux workstation mounts.

```
[root@lx1-tbw3 ~]# df -H
```

**Note:** Workstation running in 64 bit OS.

```bash
Filesystem            Size  Used Avail Use% Mounted on
/dev/mapper/vg01-lvol01       1.1G  428M  553M  44% /
tmpfs                    6.3G   463k  6.3G   1% /dev/shm
/dev/mapper/vg01-lvol_fxa     4.7G  2.0G  2.5G  45% /awips/fxa
/dev/mapper/vg01-lvol_ifps    1.6G  83M  1.4G   6% /awips/ifps
/dev/mapper/vg01-lvol_ldad    1.1G  75M  906M   8% /awips/ldad
/dev/mapper/vg01-lvol_ops     305M  84M  206M  30% /awips/ops
/dev/sda3                 102M   63M   34M  65% /boot
/dev/mapper/vg01-lvol_data   4.2G  77M  3.9G  2% /data
/dev/mapper/vg01-lvol_dhm    1.1G  19M  962M   2% /data/dhm
/dev/mapper/vg01-lvol_local  2.1G  37M  2.0G  2% /local
/dev/mapper/vg01-lvol_tmp    508M  12M  416M  3% /tmp
/dev/mapper/vg01-lvol_usr    16G  11G  4.0G 74% /usr
/dev/mapper/vg01-lvol_var   3.1G  307M  2.7G 11% /var
nas1:/awipsADAPT           1.1G  701M  374M  66% /awips/adapt
nas1:/awipsDEV             4.3G  173k  4.3G  1% /awips/dev
nas1:/awipsHOME            119G  94G  25G  79% /home
nas1:/awipsHYDRO            108G 809M 107G  1% /awips/hydroapps
nas1:/dataADAPT            3.3G  0  3.3G  0% /data/adapt
nas1:/dataFXA               537G 390G 148G 73% /data/fxa
nas1:/dataLOCAL             162G 46G 116G  29% /data/local
nas1:/awipsGFESuit/ws       108G  18G  90G 17% /awips/GFESuite
nas1:/verify               2.2T  3.1G  2.2T 1% /data/verify
nas1:/localapps            54G  39M  54G  1% /localapps
nas1:/aiidata/share        537G 12G 526G  3% /awips2/edex/data/share
nas2:/dataSTORE            2.2T  42G  2.2T  2% /data_store
nas2:/awips2ARCHIVE        3.3T 309G 3.3T  1% /archive
nas1:/dataARCHIVER         1.1T  4.8G  1.1T 1% /data/archiver
```

---

_AWP.MAN.SMM.A2_ 2-43

-- Hard copies uncontrolled. Verify effective date prior to use. --
2.2.1.7  **Linux Preprocessor**

Two PX preprocessors (PX1 and PX2) are at every AWIPS site to run applications. The PX architecture is shown in Exhibit 2.2.1.7-1.

The PXs at the site are each sized to handle the full system load in the event one of them fails. Each contains 2X 146 GB of internal storage. Refer to Exhibit 2.2.1.7-2, Linux Preprocessor (PX) Hardware.
Exhibit 2.2.1.7-1. Linux Preprocessors (PX) Architecture

Exhibit 2.2.1.7-2. Linux Preprocessor (PX) Hardware
PX configuration is summarized in Table 2.2.1.7-1.

<table>
<thead>
<tr>
<th>Type</th>
<th>Site</th>
<th>Processor</th>
<th>RAM</th>
<th>CACHE</th>
<th>Internal Storage</th>
</tr>
</thead>
<tbody>
<tr>
<td>HP DL380 G7</td>
<td>All AWIPS Sites</td>
<td>2x2 4 GHZ Quad Core</td>
<td>12 GB SDRAM</td>
<td>12 MB</td>
<td>2X 146 GB</td>
</tr>
</tbody>
</table>

The PX preprocessors maintain a continuous heartbeat between the two processors and accomplish an automatic processing switch to the other processor should a processor fail. Each PX executes a predefined software package, and each PX can take over the responsibility for all PX functions on the failure of one of the PXs.

The PX has LAN and asynchronous serial interfaces. The serial interface is Electronics Industry Association (EIA) RS-232 and is used to provide a system console connection.

Exhibit 2.2.1.7-3 shows the PX processor interfaces. The RS-232-C interfaces represent the site monitor and control interface, which provides console port interfaces to the CP, workstation, SBN, LAN, and WAN. The 1 Gbps interface connects to the site LAN.

- **Linux Preprocessor Processors (PX1A)**

  Two PX preprocessors (PX1A), a primary and a secondary server, are deployed to every AWIPS site. The PX preprocessors are HP DL380 G7 servers.
2.2.1.7.1 Preprocessor Server File Systems

The PX file systems support the following:

- Operating system
- COTS applications
- Developed applications.

The file systems on the PX are:

- **root** (/). This file system is superuser’s (root) home directory.
- **/boot**. This file system contains boot-related files (kernel).
- **/tmp**. This file system is used to store temporary files that users and system programs create.
- **/usr**. This file system holds most user-related documents, command-related files, and installation files.
- **/var**. This file system contains system log files for most applications (mail, print spoolers, and system information).

2.2.1.7.2 Preprocessor Server File Systems

The PX file systems support the following:

- Operating system
- COTS applications
- Developed applications.

The file systems on the PX are:

- **root** (/). This file system is superuser’s (root) home directory.
- **/boot**. This file system contains boot-related files (kernel).
- **/tmp**. This file system is used to store temporary files that users and system programs create.
- **/usr**. This file system holds most user-related documents, command-related files, and installation files.
- **/var**. This file system contains system log files for most applications (mail, print spoolers, and system information).
- **swap space**. The swap space is divided into one logical volume on the root disk. The primary swap space on the root drive is also designed as a dump space in the event of a system panic. This is designated as a swap/dump file system.
• /awips/ops. This file system is used to store baselined contractor-developed software and any associated log or configuration files. It contains binary applications, scripts, logs, or data as required to support operations.

• /awips/fxa. This file system stores baselined WFO-Advanced software developed by the FSL (presently ESRL/GSD) and any associated configuration files. It may contain binary applications, scripts, or data as required to support operations.

• /awips/ifps. This file system is for baselined IFPS software and data.

• /awips/laps. This file system will store baselined FSL-developed Local Analysis and Prediction System (LAPS) software and any associated configuration files. It may contain binary applications, scripts, or data as required to support operations.

• /awips/ldad. This file system will store baselined FSL-developed WFO-Advanced software and any associated configuration files. It may contain binary applications, scripts, or data as required to support operations.

• /data/logs. This file system contains the operational logs for software executing on the platform. Subdirectories will further isolate application- or product-specific logs.

• /dev/shm. This file system contains the shared memory.

• NFS Mount Points. The following directories are mounted from the NAS:
  - /home
  - /data/GFE
  - /data/txa
  - /awips/adapt
  - /awips/GFESuite
  - /awips/hydroapps
  - /data/adapter
  - /data/local
  - /awips/dev

The following is an example of the PX mount points at the WFOs.

```
[root@px1-tbdw ~]# df -H

Filesystem            Size  Used  Avail  Use%  Mounted on
/dev/mapper/vg01-lvol01  1.1G  444M   537M  46%  /
/dev/mapper/vg01-lvol_fxa  5.2G  1.6G   3.4G  33%  /awips/txa
/dev/mapper/vg01-lvol_ifps  1.6G  36M   1.5G   3%  /awips/ifps
/dev/mapper/vg01-lvol_laps  1.1G  144M  837M  15%  /awips/laps
/dev/mapper/vg01-lvol_ldad  1.1G  179M  802M  19%  /awips/ldad
/dev/mapper/vg01-lvol_ops  305M  85M  205M  30%  /awips/ops
/dev/mapper/vg01-lvol_awips2  11G  1.9G   8.2G  19%  /awips2
/dev/sda3               102M   65M   32M  67%  /boot
/dev/mapper/vg01-lvol_logs  2.1G  653M   1.4G  34%  /data/logs
/dev/mapper/vg01-lvol_tmp  508M   11M  471M   3%  /tmp
/dev/mapper/vg01-lvol_usr  16G   7.6G   7.2G  52%  /usr
/dev/mapper/vg01-lvol_var  3.1G  361M   2.6G  13%  /var
nas1:/awipsADAPT       1.1G  701M  374M  66%  /awips/adapt
nas1:/awipsDEV         4.3G  173k  4.3G   1%  /awips/dev
```
The following is an example of the PX mount points at the RFCs.

```
[root@px1-tbdr ~]# df -H

Filesystem            Size  Used  Avail Use% Mounted on
/dev/mapper/vg01-lvol1  1.1G   507M   474M  52% /
/dev/shm               6.3G      0    6.3G   0% /dev/shm
/dev/mapper/vg01-lvol_fxa  5.2G   983M   4.0G  21% /awips/fxa
/dev/mapper/vg01-lvol_ifps  1.6G   36M    1.5G   3% /awips/ifps
/dev/mapper/vg01-lvol_laps  1.1G   97M    1.0G  10% /awips/laps
/dev/mapper/vg01-lvol_ldad  1.1G  107M   874M  11% /awips/ldad
/dev/mapper/vg01-lvol_ops  305M   61M   229M  21% /awips/ops
/dev/mapper/vg01-lvol_awips2  11G   1.7G    8.4G  17% /awips2
/dev/sga3             102M    35M    67M  37% /boot
/dev/mapper/vg01-lvol_logs  2.1G   692M   1.3G  36% /data/logs
/dev/mapper/vg01-lvol_tmp  508M   11M   471M   3% /tmp
/dev/mapper/vg01-lvol_usr  16G   8.0G   6.8G  54% /usr
/dev/mapper/vg01-lvol_var  3.1G   535M   2.5G  19% /var
/nas1:/awipsADAPT  1.1G  172M   902M  17% /awips/adapt
/nas1:/awipsDEV  4.3G  1.2M   4.3G   1% /awips/dev
/nas1:/awipsGFESuit/ws  108G  26G   82G  24% /awips/GFESuite
/nas1:/awipsHOME  119G  58G   62G  49% /home
/nas1:/awipsHYDRO  106G  53G   53G  50% /awips/hydroapps
/nas1:/dataADAPT  3.3G    0  3.3G   0% /data/adapt
/nas1:/dataFXA  537G  362G  175G  68% /data/fxa
/nas1:/dataGFE  21M    0  21M   0% /data/GFE
/nas1:/dataLOCAL  162G  46G  116G  29% /data/local
/nas1:/localapps  54G  54G    0  1% /localapps
/nas1:/dataSTORE  537G  495G   42G   8% /data_store
/nas1:/verify  2.2T  1.1G   2.2T   1% /data/verify
/nas1:/aiidata  537G  12G   526G   3% /awips2/edex/data
/nas1:/localapps  54G  150M    11G  2% /awips2/GFESuite
/nas1:/dataADAPT  3.3G    0  3.3G   0% /data/adapt
/nas1:/awipsDEV  4.3G  1.2M   4.3G   1% /awips/dev
/nas1:/awipsGFESuit/ws  108G  26G   82G  24% /awips/GFESuite
/nas1:/awipsHOME  119G  58G   62G  49% /home
/nas1:/awipsHYDRO  106G  53G   53G  50% /awips/hydroapps
/nas1:/dataADAPT  3.3G    0  3.3G   0% /data/adapt
/nas1:/dataFXA  537G  362G  175G  68% /data/fxa
/nas1:/dataGFE  21M    0  21M   0% /data/GFE
/nas1:/dataLOCAL  162G  46G  116G  29% /data/local
/nas1:/localapps  54G  54G    0  1% /localapps
/nas1:/dataSTORE  537G  495G   42G   8% /data_store
/nas1:/verify  2.2T  1.1G   2.2T   1% /data/verify
/nas1:/aiidata  537G  12G   526G   3% /awips2/edex/data
/nas1:/localapps  54G  150M    11G  2% /awips2/GFESuite
```
2.3 **NCF Time Source**

The NCF time source is located at the ANCF and the BNCF. It receives, decodes, and provides AWIPS with time from the NIST WWV external time source. After the ANCF/BNCF Comm Server is synchronized with the NIST WWV, it synchronizes all other LAN clocks at the site. All AWIPS system clocks are synchronized to within 1 second of the NIST WWV time using the network time protocol over the WAN. NRSs are synchronized over the SBN.

2.4 **Printers**

All AWIPS sites have two color graphics printers.

Both color graphics printers are Dell 5130cdn. The color graphics printers generate publication-quality color output for use in publications and journals and print high-quality black and white products. The color graphics printers can generate laser-quality output of 300 dpi on paper up to 8.5 inches by 14 inches (legal size). Each color graphics printer has its own IP address and is connected to the GB LAN (Ethernet).

2.5 **AWIPS Software**

AWIPS software is divided into three general classes:

1. AWIPS contractor-developed software
2. Government-developed software
3. COTS software and Freeware.

2.5.1 **AWIPS Contractor-Developed Software**

The AWIPS contractor-developed software categories are:

- **Communications Software.** Receives and processes the satellite signal. This software processes the signal from the demodulators and stores the decoded products in the AWIPS database.

- **Monitor and Control (M&C) Software.** Tracks functioning of the AWIPS system at the site and transmits the collected information to the NCF. This software allows the NCF to control site hardware and software, if needed.

- **EDEX System Deployment.** A fully functional AWIPS II EDEX installation requires deployment and operation of several pieces of software. These include a PostgreSQL database, a Unidata Local Data Manager (LDM) as a data source, an AWIPS II Radar Server (RCM), an Apache Qpid messaging service, and the EDEX software itself.
2.5.2 Government-Developed Software

The Government has developed most of the AWIPS weather applications.

- NOAA’s Earth System Research Laboratory/Global Systems Division (GSD, formerly FSL) developed the AWIPS D2D user interface, IFPS (jointly, with MDL), LAPS, and LDAD applications.

- The NWS Meteorological Development Laboratory (MDL) developed the background hydrometeorological applications, including Climate, AvnFPS, SCAN, FFMP, IFPS (jointly, with FSL), HWR, LAMP, Pre-LAMP, Guardian, FSI, SNOW, GFE, and ADAPPT Foundation.

- Other Government-developed software appear as stand-alone applications, for example, the Office of Hydrologic Development (OHD) and the WFO Hydrologic Forecast System (HFS) applications (HydroView, RiverPro, and HydroBase). These applications are documented in their own manuals.

NOTE: Most of the Government-developed software has been transitioned to Raytheon Software Maintenance and Support for maintenance.

2.5.3 Commercial Off-the-Shelf (COTS) Software and Freeware

2.5.3.1 AWIPS II: COTS Software and Freeware

AWIPS relies on COTS software when possible. Often the COTS software is invisible to users. For example, all of the individual hardware items that compose the AWIPS system are commercially available, and all of them use the software that their manufacturers provide. COTS software is documented via the commercial documentation provided with the COTS software package.

Table 2.5.3.1-1 lists the COTS software and freeware used in AWIPS II.

<table>
<thead>
<tr>
<th>Component</th>
<th>Version</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActiveMQ</td>
<td>5.3.0</td>
<td>JMS (still used by AlertViz and internally in parts of Camel)</td>
</tr>
<tr>
<td>Apache Batik</td>
<td>1.6</td>
<td>A Java-based toolkit for applications or applets that want to use images in the Scalable Vector Graphics (SVG) format for various purposes, such as display, generation or manipulation</td>
</tr>
<tr>
<td>Apache MINA</td>
<td>1.1.7</td>
<td>Network application framework</td>
</tr>
<tr>
<td>Apache WSS4J</td>
<td>1.6.5</td>
<td>Web Services Security</td>
</tr>
<tr>
<td>Ant</td>
<td>1.7.1</td>
<td>Java Build Tool</td>
</tr>
<tr>
<td>Ant-Contrib</td>
<td>1.0b3</td>
<td>Additional useful tasks and types for Ant</td>
</tr>
<tr>
<td>Antlr</td>
<td>2.7.6</td>
<td>Parser generator</td>
</tr>
<tr>
<td>Atomikos TransactionEssentials</td>
<td>3.6.2</td>
<td>Transaction management system</td>
</tr>
<tr>
<td>Bitstream Vera Fonts</td>
<td>1.10</td>
<td>Font library from Gnome</td>
</tr>
<tr>
<td>bzip2</td>
<td>none</td>
<td>Stream compression algorithm</td>
</tr>
<tr>
<td>Component</td>
<td>Version</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>C3p0</td>
<td>0.9.1</td>
<td>An easy-to-use library for making traditional JDBC drivers “enterprise-ready” by augmenting them with functionality defined by the jdbc3 spec and the optional extensions to jdbc2</td>
</tr>
<tr>
<td>Camel</td>
<td>2.4</td>
<td>Enterprise Service Bus</td>
</tr>
<tr>
<td>Cglib (Byte Code Generation Library)</td>
<td>2.2</td>
<td>A high-level API; used to generate and transform JAVA byte code</td>
</tr>
<tr>
<td>CherryPy</td>
<td>3.1.2</td>
<td>Object-oriented HTTP framework</td>
</tr>
<tr>
<td>commons-beanutils</td>
<td>1.8.3</td>
<td>Apache Common Libraries</td>
</tr>
<tr>
<td>commons-codec</td>
<td>1.4.1</td>
<td>Apache Common Libraries</td>
</tr>
<tr>
<td>commons-collection</td>
<td>3.2</td>
<td>Apache Common Libraries</td>
</tr>
<tr>
<td>commons-configuration</td>
<td>1.6</td>
<td>Apache Common Libraries</td>
</tr>
<tr>
<td>Commons-cli</td>
<td>1.2</td>
<td>Apache Common Libraries</td>
</tr>
<tr>
<td>commons-digester</td>
<td>1.8.1</td>
<td>Apache Common Libraries</td>
</tr>
<tr>
<td>commons-cxf</td>
<td>2.5</td>
<td>Apache Common Libraries</td>
</tr>
<tr>
<td>commons-lifecycle</td>
<td>3.1</td>
<td>Apache Common Libraries</td>
</tr>
<tr>
<td>commons-lang</td>
<td>2.3</td>
<td>Apache Common Libraries</td>
</tr>
<tr>
<td>commons-logging</td>
<td>1.1.1</td>
<td>Apache Common Libraries</td>
</tr>
<tr>
<td>commons-management</td>
<td>1.0</td>
<td>Apache Common Libraries</td>
</tr>
<tr>
<td>commons-pools</td>
<td>1.3</td>
<td>Apache Common Libraries</td>
</tr>
<tr>
<td>commons-validator</td>
<td>1.2</td>
<td>Apache Common Libraries</td>
</tr>
<tr>
<td>dom4j</td>
<td>1.6.1</td>
<td>An open source library for working with XML, XPath, and XSLT on the Java platform using the Java Collections Framework</td>
</tr>
<tr>
<td>dwr (direct web remoting) Getahead</td>
<td>1.1.3</td>
<td>Java open source library</td>
</tr>
<tr>
<td>Eclipse</td>
<td>3.8.2</td>
<td>Java IDE</td>
</tr>
<tr>
<td>Eclipse CDT</td>
<td>5.0.2</td>
<td>C/C++ IDE for Eclipse</td>
</tr>
<tr>
<td>ehcache</td>
<td>1.3.0</td>
<td>Caching Support</td>
</tr>
<tr>
<td>GEOS</td>
<td>3.0.2</td>
<td>Geometry Engine, Required for PostGIS</td>
</tr>
<tr>
<td>GeoTools</td>
<td>2.6.4</td>
<td>Java API for Manipulation of Geospatial Data</td>
</tr>
<tr>
<td>Geronimo-jms</td>
<td>1.1 spec 1.1.1</td>
<td>Server runtime framework</td>
</tr>
<tr>
<td>GRIJava</td>
<td>8.0</td>
<td>Grib Java Decoder</td>
</tr>
<tr>
<td>h5py</td>
<td>1.3.0-3</td>
<td>HDF5 for Python</td>
</tr>
<tr>
<td>hdf5</td>
<td>1.8.4-patch1</td>
<td>Core HDF5 APIs</td>
</tr>
<tr>
<td>hdf5</td>
<td>2.5</td>
<td>Core HDF5 APIs</td>
</tr>
<tr>
<td>Hibernate</td>
<td>3.5.0</td>
<td>Data Access Layer</td>
</tr>
<tr>
<td>IzPack</td>
<td>4.2.0</td>
<td>Installer creator for EDEX</td>
</tr>
<tr>
<td>JAI</td>
<td>1.1.3</td>
<td>Java API for Image Manipulation</td>
</tr>
<tr>
<td>JAI – Image I/O</td>
<td>1.1</td>
<td>Plug-ins for JAI</td>
</tr>
<tr>
<td>Jasper</td>
<td>1.900.1</td>
<td>JPEG-2000 codec</td>
</tr>
<tr>
<td>Java</td>
<td>1.6.0.43-1</td>
<td>Kit for both 32-bit and 64-bit</td>
</tr>
<tr>
<td>javax.mail</td>
<td>1.4.3</td>
<td>mail modeling classes</td>
</tr>
<tr>
<td>javax.measure</td>
<td>1.0-b gauche-2</td>
<td>Strong types for measurements</td>
</tr>
<tr>
<td>javax.persistence</td>
<td>1.0.0</td>
<td>persistence classes and interfaces</td>
</tr>
<tr>
<td>javax.vecmath</td>
<td>1.3.1</td>
<td>Coordinates and vectors</td>
</tr>
<tr>
<td>Jep</td>
<td>2.3+</td>
<td>Java Python interface</td>
</tr>
<tr>
<td>Jetty</td>
<td>7.2.2</td>
<td>Jetty provides an HTTP server, HTTP client, and javax.servlet container</td>
</tr>
<tr>
<td>Component</td>
<td>Version</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>jGrapht</td>
<td>0.6.0</td>
<td>A free Java graph library that provides mathematical graph-theory objects and algorithms</td>
</tr>
<tr>
<td>JMock</td>
<td>2.0.0</td>
<td>Java Mock Object Framework</td>
</tr>
<tr>
<td>jna (java native access)</td>
<td>3.09</td>
<td>Provides Java programs easy access to native shared libraries (DLLs on Windows) without writing anything but Java code—no JNI or native code is required. This functionality is comparable to Windows’ Platform/Invoke and Python’s ctypes. Access is dynamic at runtime without code generation.</td>
</tr>
<tr>
<td>Jogl</td>
<td>1.1.1-rc8</td>
<td>Provides hardware-supported 3D graphics</td>
</tr>
<tr>
<td>Jscience</td>
<td>4.3.1</td>
<td>Library for Scientific Calculations and Visualizations</td>
</tr>
<tr>
<td>JTS Topology Suite</td>
<td>1.10</td>
<td>Java API for 2D spatial data</td>
</tr>
<tr>
<td>JUnit</td>
<td>4.10</td>
<td>Java Unit Test Framework</td>
</tr>
<tr>
<td>lapack</td>
<td>3.0.0</td>
<td>Linear Algebra Package for python</td>
</tr>
<tr>
<td>Ldm</td>
<td>6.11.5-9</td>
<td>Local Data Manager</td>
</tr>
<tr>
<td>Log4J</td>
<td>1.2.16</td>
<td>Logging Component used by Commons Logging</td>
</tr>
<tr>
<td>libgfortran</td>
<td>4.1.2</td>
<td>Fortran Library</td>
</tr>
<tr>
<td>matplotlib</td>
<td>0.99.1.1-3</td>
<td>Python 2D Plotting Library</td>
</tr>
<tr>
<td>Mozilla Rhino</td>
<td>1.6R7</td>
<td>Implementation of JavaScript embedded in Java</td>
</tr>
<tr>
<td>NCEP Grib2 Libraries</td>
<td></td>
<td>Libraries for decoding &amp; encoding data in GRIB2 format</td>
</tr>
<tr>
<td>cvngrib</td>
<td>1.1.8 and 11.9</td>
<td>Fortran GRIB1 &lt;-&gt; GRIB2 conversion utility</td>
</tr>
<tr>
<td>g2clib</td>
<td>1.1.8</td>
<td>“C” grib2 encoder/decoder</td>
</tr>
<tr>
<td>g2lib</td>
<td>1.1.8 and 11.9</td>
<td>Fortran grib2 encoder/decoder and search/indexing routines</td>
</tr>
<tr>
<td>w3lib</td>
<td>1.6 and 1.7.1</td>
<td>Fortran grib1 encoder/decoder and utilities</td>
</tr>
<tr>
<td>nose</td>
<td>0.11.1-3</td>
<td>Python unit test extension</td>
</tr>
<tr>
<td>NumPy</td>
<td>1.5.0-3</td>
<td>Numerical Python Scientific package for Python</td>
</tr>
<tr>
<td>objectweb asm</td>
<td>2.1</td>
<td>An all-purpose Java bytecode manipulation and analysis framework. It can be used to modify existing classes or dynamically generate classes, directly in binary form</td>
</tr>
<tr>
<td>Openfire</td>
<td>3.7.1</td>
<td>Collaboration Server – Not used but eventually will replace Wildfire. Only 3.7 approved</td>
</tr>
<tr>
<td>Pil</td>
<td>1.1.6-3</td>
<td>Python Imaging Library</td>
</tr>
<tr>
<td>Pgadmin3</td>
<td>1.16.1-1</td>
<td>A database design and management application designed for use with PostgreSQL. It is written in C++ using the wxWidgets (<a href="http://www.wxwidgets.org">http://www.wxwidgets.org</a>) cross-platform framework to allow it to run on multiple platforms.</td>
</tr>
<tr>
<td>pmw</td>
<td>1.3.2-3</td>
<td>A toolkit for building high-level compound widgets in Python using the Tkinter module</td>
</tr>
<tr>
<td>PostGIS</td>
<td>1.3.5</td>
<td>Geographic Object Support for PostgreSQL</td>
</tr>
<tr>
<td>PostgreSQL</td>
<td>9.2.3, 9.2.4-1</td>
<td>Database</td>
</tr>
<tr>
<td>Proj</td>
<td>4.6.1</td>
<td>Cartographic Projections library</td>
</tr>
<tr>
<td>pupynere</td>
<td>1.0.13-3</td>
<td>Python module for reading and writing NetCDF files</td>
</tr>
<tr>
<td>pycairo</td>
<td>1.2.2-3</td>
<td>A 2D graphics library with support for multiple output devices.</td>
</tr>
<tr>
<td>Pydev</td>
<td>1.5</td>
<td>Python Development Environment</td>
</tr>
<tr>
<td>pygtk</td>
<td>2.8.6-3</td>
<td>Used to easily create programs with a graphical user interface using the Python programming language</td>
</tr>
<tr>
<td>PyTables</td>
<td>2.1.2</td>
<td>Python package for managing hierarchical datasets</td>
</tr>
<tr>
<td>Python</td>
<td>2.7.1-10</td>
<td>Dynamic programming language</td>
</tr>
<tr>
<td>Python megawidgets</td>
<td>1.3.2</td>
<td>Toolkit for building high-level compound widgets in Python using the Tkinter module.</td>
</tr>
<tr>
<td>Qpid</td>
<td>0.18-4</td>
<td>Open Source AMQP (Advanced Message Queuing Protocol) Messaging</td>
</tr>
</tbody>
</table>
2.5.3.2 Mobile Code Technologies

NIST SP 800-53, Information Security, Appendix B, provides the following definitions regarding mobile code:

- **Mobile Code Software**: Programs or parts of programs obtained from remote information systems, transmitted across a network, and executed on a local information system without explicit installation or execution by the recipient.

- **Mobile Code Technologies Software**: Technologies that provide the mechanisms for the production and use of mobile code (e.g., Java, JavaScript, ActiveX, VBScript).

- AWIPS II mobile code consists of micro engine scripts that are written in Python. These micro engine scripts execute only within EDEX.

2.6 AWIPS I and AWIPS II

2.6.1 What’s Changed in AWIPS

AWIPS II aims to be a blackbox rewrite of AWIPS I. It is written mostly in java, but python is also extensively used, especially in GFE. AWIPS II is built upon the following Open Source projects:

- eclipse: http://www.eclipse.org
• camel: http://camel.apache.org/
• spring: http://www.springsource.org/
• hibernate: http://www.hibernate.org/
• qpid: http://qpid.apache.org
• Others (postgres, HDF5, log4j, thrift).

CAVE, the “Common AWIPS Visualization Environment” for AWIPS II, runs on the workstations, like D2D, GFE, and HYDRO. CAVE is an RCP application (“RCP” stands for Eclipse’s “Rich Client Platform”). The RCP is a framework of a core set of approximately 30 plug-ins, although plug-ins are the smallest unit of RCP functionality. With plug-ins, code can be loosely coupled; the code is also maintainable and extensible. The core plug-ins provide extension points, which applications can use to extend the functionality of the framework. One such extension point is org.eclipse.ui.perspectives. In AWIPS II, D2D, GFE, and HYDRO are CAVE perspectives.

CAVE variations include the following:

• Map pans and zooms.

• Localization perspective. A localization perspective exists. It is from the localization perspective that smartTools and Utilities are edited; they are not edited by right-clicking in the EA window as it is done in AWIPS I.

• GFE perspective. There is no GM realignment button in AWIPS II. You can realign the GM by dragging.

The AWIPS II server is EDEX (Environmental Data Exchange). EDEX consists of:

• An Enterprise Service Bus (ESB)
  – JVMs (Java Virtual Machines). There are currently four JVMs per server, clustered across two servers. Spring and Camel form the backbone of the ESB. It is in this environment that AWIPS II server code runs.
    ▪ Spring manages software objects. It creates them (initiates them), and injects them with initial values using directives retrieved from xml. This makes for a loosely coupled, flexible system. The runtime behavior of the system can be changed by modifying the xml, not the java source code. See http://static.springsource.org/spring/docs/2.5.x/reference/beans.html
    ▪ Camel routes messages using the Enterprise Integration Patterns described at http://www.enterpriseintegrationpatterns.com/eapatterns.html
  – A JMS broker – messaging middleware.
    ▪ AWIPS II’s JMS broker is QPID.
    ▪ QPID is an important part of data ingest. The SBN LDM client, radar, and LDAD ingest processes all send messages to a broker queue, which is read by the ESB. Also, MHS (just like LDAD and LDM) writes a message to
/data_store and posts a message to qpid. That is how MHS products get into AWIPS II.

- Once the message is read, the ESB looks in /awips2/edex/data/utility/edex_static/base/distribution to determine which data plug-in to use to ingest and decode the raw product (yes, EDEX has plug-ins too, although they are not RCP plug-ins). Each data type has a plug-in (there are about 50), and each plug-in has a Data Access Object (DAO) or a set of DAOs that is used to store and retrieve data from the database. The DAOs interact with postgres’ metadata db and the hdf5 filesystem.

The postgres-metadata database is used in conjunction with the hdf5 filesystem to form the AWIPS II database, with HYDRO being the notable exception. HYDRO databases will be left as they are in AWIPS I.

Data is stored in the HDF5 filesystem. It is not meant to be accessed directly. See http://www.hdfgroup.org/HDF5/.

AWIPS II has a development environment. Called the “AWIPS Development Environment,” or “the ADE,” it consists of Eclipse's IDE (also an RCP app) and the software for the entire system, minus the legacy rehosted code. Clients like CAVE can be run from the ADE, so it is possible to make changes to CAVE code. You can make changes to EDEX, then compile and deploy with the ADE and run your modified EDEX.

AWIPS II continues to use heartbeat software to manage application packages on the PX1/2 and DX1/2 clusters. Although heartbeat is no longer used on the DX3/4 server pair, it has been added to the CPSBN1/2 server pair. AWIPS II clustering strategy and heartbeat packages are discussed in section 2.8, “AWIPS II Server Clustering.” Because AWIPS II does not use the heartbeat cluster on DX3/4, any locally defined crons formerly managed by the dx3apps and dx4apps packages have been moved to another server.

The Direct Attached Storage (DAS). File system mounts are covered in section 2.11, “Basic AWIPS II Data Storage Architecture.”

**AWIPS II software performs its own logging.** Log locations vary with the different AWIPS II software components. AWIPS II logging is covered in section 2.10, “Basic AWIPS II Logging Architecture.”

### 2.6.2 What’s Not Changed in AWIPS

AWIPS II reuses a number of the existing AWIPS I applications. These include:

- MHS (Message Handling System)
- Climate software
- HWR
- NWRWAVES/NWREditor/CRS
- FSI (server path has changed; the GUI has been wrapped)
• LDAD (Local Data Acquisition and Dissemination)
• NWWS (NOAA Weather Wire Service)
• ASYNC Scheduler
• LSR (Local Storm Report)
• FloodEvent Archiver
• HydroGen
• RiverPro
• WFO/RFC Archiver (Weather Forecast Office/River Forecast Center Archiver)
• LAPS/MSAS (Local Analysis and Prediction System/MAPS Surface Assimilation System)
• Chatserver
• LegalArchiver
• Certain Command Line Interfaces (textdb, fxaAnnounce, sendMsgToGuardian).

For the retained AWIPS I software, basic management techniques are unchanged; high-availability packages and server cron configuration files are still used (the package filenames start with a2, for example, a2px2apps). Log names and locations are unchanged.

2.7 *Basic AWIPS II Software Deployment*

AWIPS II replaces a large portion of the existing AWIPS data ingest, processing, and storage capabilities, and it consolidates many forecaster applications into a single visualization tool. The primary AWIPS II application for data ingest, processing, and storage is the Environmental Data EXchange (EDEX); the primary AWIPS II application for forecaster visualization/data manipulation is the Common AWIPS Visualization Environment (CAVE).

AWIPS II takes a unified approach to data ingest. Most data types follow a standard path through the system. Variations on this basic data flow include local radar (Open Radar Product Generator (ORPG) / Supplemental Product Generator (SPG)) products, and Local Data Acquisition and Dissemination (LDAD) delivered products. AWIPS II provides interfaces for communicating with the ORPG/SPG and LDAD servers.

At a high level, data flow describes the path a piece of data follows through the system. The path starts with the data source and includes storing the raw data, decoding the data, and storing the decoded data in a form that makes it available for display and/or manipulation by the forecaster. For more on this process, see Chapters 4 – 7.

AWIPS II supports automated processing using a cron-like capability built into the EDEX process. EDEX also provides database and Processed Data Storage management; Raw data storage is managed by the Local Data Manager (LDM) process. Script running in response to data arrival is also supported within EDEX. These topics are covered in Chapters 11 and 12 of this manual.
In addition to programs developed specifically for AWIPS, AWIPS II uses several commercial off-the-shelf (COTS) and Free or Open Source software (FOSS) products to assist in its operation. These applications include the Unidata LDM, the Apache web server, the Queue Processor Interface Daemon (QPID) Apache AMQP Message Broker, and the Java and Python runtime systems.

At a high level, data flow describes the path taken by a piece of data from its source to its display by a client system. This basic data flow concept is shown in Exhibit 2.7-1. The overall AWIPS II Server Processes and AWIPS II Servers Overview are shown in Exhibits 2.7-2 and 2.7-3 respectively.

Exhibit 2.7-1. Overall AWIPS II Data Flow
Exhibit 2.7-2. Overall AWIPS II Servers Processes
AWIPS II installs new AWIPS II-specific software on several of the existing AWIPS systems, both servers and workstations.

Table 2.7-1 lists the specific AWIPS II-provided software installed on each system.

Table 2.7-1. AWIPS II Software

<table>
<thead>
<tr>
<th>System</th>
<th>Key Software Packages</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPSBN1 and CPSBN2</td>
<td>LDM, DVB, QPID, EDEX Bridge, LDM Writer, IPVS</td>
</tr>
<tr>
<td>PX1 and PX2</td>
<td>Rehosted Applications</td>
</tr>
<tr>
<td>DX1 and DX2</td>
<td>PostgreSQL Data Base Management System (DBMS), PyPIES (dx2f), Radar-Coded Message (RCM),</td>
</tr>
<tr>
<td>DX3 and DX4</td>
<td>EDEX (ingest, ingestGrib, ingestDat, request)</td>
</tr>
<tr>
<td>LX Workstations</td>
<td>CAVE, AlertViz, CLI, PSQL, Java, Python</td>
</tr>
<tr>
<td>XT Workstations</td>
<td>CAVE, AlertViz, CLI, PSQL, Java, Python</td>
</tr>
</tbody>
</table>

On the servers (CPSBN1 and CPSBN2, PX1 and PX2, and DX1–4) AWIPS II software is installed in /awips2. On the workstations, it is installed in /usr/local/viz.
2.8 **AWIPS II Server Clustering**

AWIPS II server software is generally deployed on pairs of servers in order to maximize processing availability. Various failover strategies are used, as detailed in Table 2.8-1.

**Table 2.8-1. AWIPS II Server Failover Strategies**

<table>
<thead>
<tr>
<th>Server Pair</th>
<th>Failover Strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPSBN1 and CPSBN2</td>
<td>CPSBN1 and CPSBN2 function as a heartbeat cluster; failover is managed automatically or manually. Note that only a single high-availability package is currently used on this cluster. The package name is a2cp1apps. In failover mode, package execution shifts to the backup server. The a2cp1apps package manages the cp1f floating server name. Note that floating names are available only when the high-availability packages are running; they cannot be used to start or stop the high-availability packages.</td>
</tr>
<tr>
<td>PX1 and PX2</td>
<td>PX1 and PX2 function as a heartbeat cluster; failover is managed automatically or manually. PX1 and PX2 normally run separate high-availability packages (a2px1apps on PX1 and a2px2apps on PX2). In failover mode, they shift to the same server. Access to PX1 and PX2 is via floating server names: px1f points to the server running the a2px1apps package; px2f points to the server running the a2px2apps package. The floating server names are available only when the high-availability packages are running; they cannot be used to start or stop the high-availability packages.</td>
</tr>
<tr>
<td>DX1 and DX2</td>
<td>DX1 and DX2 function as a heartbeat cluster; failover is managed automatically or manually. DX1 and DX2 normally run separate high-availability packages (a2dx1apps on DX1 and a2dx2apps on DX2). In failover mode, they shift to the same server. Access to DX1 and DX2 is via server name alias: dx1f points to the server running the a2dx1apps package; dx2f points to the server running the a2dx2apps package. The floating names are available only when the high-availability packages are running; they cannot be used to start or stop the high-availability packages.</td>
</tr>
<tr>
<td>DX3 and DX4</td>
<td>DX3 and DX4 are independent servers, both of which run the EDEX server software. <strong>There is no failover between DX3 and DX4.</strong> Both servers are load balanced and share data processing responsibilities; if one server is stopped, the other will pick up the entire data processing load. Processing load balancing between DX3 and DX4 is provided by QPID as data messages are removed from QPID queues and processed by the EDEX instance having available processing capacity. Client request balancing for DX3 and DX4 is managed by Internet Protocol Virtual Server (IPVS) (the pulse service). Both QPID and IPVS run on CPSBN1 and CPSBN2.</td>
</tr>
</tbody>
</table>

As indicated in Table 2.8-1, server name aliasing is used to direct clients to the appropriate member of each cluster. “Appropriate member” generally refers to the server running a specific high-availability package; in the case of DX3 and DX4, however, it refers to an available EDEX process.

AWIPS II server aliases and their target servers are listed in Table 2.8-2. Note that the floating names, e.g., dx1f and px1f, are managed by the high-availability packages; as a result, the floating server names are only available when the packages are running and cannot be used when starting or stopping the high-availability packages.
Table 2.8-2. AWIPS II Server Aliases

<table>
<thead>
<tr>
<th>Server Alias</th>
<th>Target System</th>
</tr>
</thead>
<tbody>
<tr>
<td>dx1f</td>
<td>The server, DX1 or DX2, which is currently running the a2dx1apps high-availability package.</td>
</tr>
<tr>
<td>dx2f</td>
<td>The server, DX1 or DX2, which is currently running the a2dx2apps high-availability package. LDM was removed from a2dx2apps and shut down on dx2f.</td>
</tr>
<tr>
<td>px1f</td>
<td>The server, PX1 or PX2, which is currently running the a2px1apps high-availability package.</td>
</tr>
<tr>
<td>px2f</td>
<td>The server, PX1 or PX2, which is currently running the a2px2apps high-availability package.</td>
</tr>
<tr>
<td>cp1f</td>
<td>The server, CPSBN1 or CPSBN2, which is currently running the a2cp1apps high-availability package. Note: A new awips2-ldm rpm installed on CPs will wrap both upstream and downstream functionality. /data_store is mounted on CPs, pqact files will now be maintained on CPs, LDM was added to a2cp1apps.</td>
</tr>
<tr>
<td>ec, edexcluster</td>
<td>Connections are redirected to DX3 or DX4, based on a least connections strategy. In a least connections strategy, a new connection request is forwarded to the server, DX3 or DX4, currently having the lower number of connections. This alias only applies to connections on port 9581. These connections are managed by IPVS service, which is managed by the a2cp1apps high-availability package, which runs on CPSBN1 or CPSBN2.</td>
</tr>
</tbody>
</table>

2.9 Basic AWIPS II Communication Architecture

The AWIPS II ingest and request processes are a highly distributed system; messaging is used for Inter-Process Communication (IPC). Exhibit 2.9-1 shows the basic IPC architecture.

Exhibit 2.9-1 shows the basic lines of communication that are used for IPC in AWIPS II. There are three primary communication channels:

1. Advanced Message Queuing Protocol (AMQP) messages are routed between the various AWIPS II processes via the QPID message broker.
2. Transport Control Protocol (TCP) messages are routed between EDEX and other AWIPS II components.
3. TCP messages from CAVE to EDEX are routed through the IPVS to request load balancing between DX3 and DX4.

Combined with the clustering strategies discussed in section 2.8, “AWIPS II Server Clustering,” the Communication Architecture provides improved system availability and processing load balancing.
2.10  Basic AWIPS II Logging Architecture

AWIPS II does not implement a standardized logging strategy; rather, each AWIPS II component provides process logs that may be examined to determine process status and assist in system troubleshooting. Table 2.10-1 lists log locations for these components.

For more information on AWIPS II process logging, including log naming patterns and configuration, see the applicable sections of Chapter 25 of this manual.
Table 2.10-1. Log Locations for AWIPS II Components

<table>
<thead>
<tr>
<th>Server</th>
<th>Process</th>
<th>Log Directory</th>
</tr>
</thead>
<tbody>
<tr>
<td>dx1f</td>
<td>Radar Server</td>
<td>/awips2/rcm/data/logs</td>
</tr>
<tr>
<td></td>
<td>PostgreSQL DBMS</td>
<td>/awips2/data/pg_log</td>
</tr>
<tr>
<td></td>
<td>PyPIES</td>
<td>/awips2/pypies/logs</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/awips2/http_pypies/var/log/httpd</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/tmp</td>
</tr>
<tr>
<td>cron</td>
<td></td>
<td>/var/logs</td>
</tr>
<tr>
<td>dx2f</td>
<td></td>
<td>/awips2/edex/logs</td>
</tr>
<tr>
<td>dx3/dx4</td>
<td>EDEX</td>
<td>/awips2/edex/logs</td>
</tr>
<tr>
<td>cpsbn1/cpsbn2</td>
<td>QPID</td>
<td>/awips2/qpid/var/log</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/var/logs</td>
</tr>
<tr>
<td></td>
<td>LDM</td>
<td>/usr/local/ldm/logs</td>
</tr>
<tr>
<td>LX/XT</td>
<td>AlertViz</td>
<td>~/caveData/logs</td>
</tr>
<tr>
<td></td>
<td>CAVE</td>
<td>~/caveData/etc/user/${USER}/logs</td>
</tr>
</tbody>
</table>

2.11 Basic AWIPS II Data Storage Architecture

AWIPS II stores on two shared data servers – the Direct Attached Storage (DAS) server and the Network Attached Storage (NAS) server. Both of these servers provide for shared data access; this provides improved system resiliency in the event of server failure. DAS access is limited to the DX1/2 cluster; NAS access is available from any server or workstation on the Local Area Network (LAN). All systems are mounted to access the data on the NAS. The AWIPS II Raw Data Storage is directly accessible only from NAS; however, other systems – DX1, DX3, and DX4 – also access this data by mounting the NAS. They do so using a file system map to nas1:/data_store. The primary AWIPS II data storage locations are listed in Table 2.11-1

Table 2.11-1. AWIPS II Data Storage

<table>
<thead>
<tr>
<th>Data Component</th>
<th>Location</th>
<th>Access From</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWIPS II Raw Data Storage</td>
<td>nas1:/data_store</td>
<td>dx1, dx2, dx3, dx4,ax, px, workstations (lx, xt)</td>
</tr>
<tr>
<td>AWIPS II HDF5 Data Storage</td>
<td>das1:/aiihdf5</td>
<td>dx1f</td>
</tr>
<tr>
<td>AWIPS II Database</td>
<td>das1:/aiidb</td>
<td>dx1f</td>
</tr>
<tr>
<td>AWIPS II QPID Shared Data</td>
<td>nas1:/qpid</td>
<td>cp1f</td>
</tr>
<tr>
<td>AWIPS II EDEX Shared Data</td>
<td>nas1:/aiidata</td>
<td>dx3, dx4, px1, px2</td>
</tr>
<tr>
<td>AWIPS II Service Backup</td>
<td>nas1:/GFESuite2</td>
<td>dx1-4, px1-2</td>
</tr>
<tr>
<td>AWIPS II Service Backup</td>
<td>nas1:/aiidata/utility</td>
<td>dx1, dx2</td>
</tr>
<tr>
<td>AWIPS II Hydroapps and Avnfps Applications</td>
<td>nas1:/aiidata/share</td>
<td>lx, xt</td>
</tr>
</tbody>
</table>

Note that for mounts that are accessed from a floating server name, e.g., dx2f, the mounts are managed by the high-availability package that provides the floating name. For example, the mount to DAS1:/aiildm, which is accessed using the dx2f floater, is managed by the a2dx1apps high-availability package.
2.12 Basic AWIPS II Visualization Architecture

AWIPS II uses the CAVE application as its main data visualization/manipulation tool. CAVE incorporates a number of the legacy AWIPS I applications such as Display Two-Dimensional (D2D), Graphical Forecast Editor (GFE), and Text Workstation. AWIPS II has also reengineered GUARDIAN into AlertViz. CAVE and AlertViz are installed on both the LX and the XT workstations.

Exhibit 2.12-1 shows the basic visualization architecture.

![Exhibit 2.12-1. AWIPS II Visualization](image)

Exhibit 2.12-1 shows the basic information/data flow between CAVE and EDEX. AlertViz will display CAVE errors as well as messages from other parts of the system. The information/data flow is as follows:

1. Data requests are sent from CAVE to a virtual server name (ec). The virtual server name is resolved by IPVS.
2. IPVS forwards the data request to the available EDEX Request process on either DX3 or DX4.
3. The EDEX Request process returns the requested data to CAVE.
4. The EDEX Ingest process broadcasts data-ingested messages that are picked up by CAVE.

CAVE can be started in various modes by passing its switches through the command line (or via application launchers). For example, CAVE can be launched to emulate the Text Workstation or only to display the D2D perspective. See SMM Chapter 25 for more information on the CAVE command line switches.
2.13  **Basic AWIPS II Data Processing Architecture**

AWIPS II separates data processing into four logically separate components: 1) data receipt; 2) data decoding; 3) data storage; and 4) data request. These components are distributed over the various systems in AWIPS II. This distribution is designed to provide improved data throughput coupled with improved system resiliency. AWIPS II reuses a number of existing components in data processing.

2.13.1  **Basic AWIPS II Data Receipt Architecture**

AWIPS II receives data from three main sources: the Satellite Broadcast Network (SBN); the LDAD network; and the Open Radar Product Generator (ORPG)/Supplemental Product Generator (SPG) network. Regardless of the data source, the received data is fed to data ingest, which is performed by the EDEX software on DX3 and DX4. This basic data receipt concept is shown in Exhibit 2.13.1-1.

![Exhibit 2.13.1-1. AWIPS II Data Receipt](image)

As shown in Exhibit 2.13.1-1:

1. The AWIPS II interface process obtains a data product from the data source.
2. The interface process writes the data to a file in Raw Data Storage.
3. The interface process posts a “data available” message to the QPID message broker.
4. The EDEX Ingest process obtains the “data available” message from QPID and removes the message from the message queue.
5. The EDEX Ingest process obtains the data file from Raw Data Storage.

AWIPS II either modifies or provides interfaces between the existing AWIPS components and AWIPS II data ingest. The AWIPS II interfaces are shown in Table 2.13.1-1.
Table 2.13.1-1. AWIPS II Interfaces to Data Sources

<table>
<thead>
<tr>
<th>Source</th>
<th>AWIPS II Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>SBN</td>
<td>LDM obtains data from the SBN on CPSBN1/2 cluster. LDM writes the data to a file in Raw Data Storage and posts a “data available” message to the QPID message broker on the CPSBN1/2 cluster.</td>
</tr>
<tr>
<td>LDAD</td>
<td>LDAD has been modified to feed data into the AWIPS II ingest data flow. Specifically, three new router processes have been added to the LDAD suite on PX1/2. These router processes are routerStoreEDEX, routerShefEncoderEDEX, and routerStoreTextEDEX. These processes allow baseline LDAD data flows to be processed by EDEX.</td>
</tr>
<tr>
<td>ORPG/SPG</td>
<td>The Radar Server on the DX1/2 cluster interacts with ORPG/SPG to obtain radar data. When the data is obtained, the server writes the data to a file in Raw Data Storage and posts a “data available” message to the QPID message broker on the CPSBN1/2 cluster.</td>
</tr>
</tbody>
</table>

Note that this architecture provides separation between data sources and ingest processing. Any data source that follows this architecture will be able to provide data for EDEX to process.

2.13.2 Basic AWIPS II Data Decoding Architecture

Data decoding is defined as the process of converting data from a raw format into a decoded format that is usable by the AWIPS II visualization software. In AWIPS II, data decoding is performed by the EDEX Ingest processes, which run on both DX3 and DX4. The basic data decoding concept is shown in Exhibit 2.13.2-1.

Exhibit 2.13.2-1. AWIPS II Data Decoding
As shown in Exhibit 2.13.2-1:

1. The EDEX Ingest process obtains the “data available” message from the QPID message broker. The EDEX Ingest process determines the appropriate data decoder based on the message contents and forwards the message to the decoder. Finally, the message is removed from the message queue.
2. The EDEX Ingest process reads the data from Raw Data Storage.
3. The EDEX Ingest Process decodes the data.
4. The EDEX Ingest process forwards the decoded data to Data Storage.
5. The EDEX Ingest process sends a message to a client queue that decoded data is available.

The current architecture has two EDEX Ingest processors, DX3 and DX4. Load balancing of Ingest processing among multiple Ingest processes is provided by the QPID message broker. Although multiple EDEX Ingest processes are possible, each “data available” message is serviced by whichever EDEX Ingest processor gets the message first. Either EDEX Ingest processor may be taken offline without stopping data ingest.

It is also important to note that in AWIPS II all data types are processed on both Ingest Processors. Although a specific data file is processed by a single EDEX Ingest process, data files for a data type are processed on both Ingest processors.

Once this process is complete, clients may obtain and perform additional processing on the data. This includes display of the data in CAVE, additional processing such as SmartInit scripts that are run on Gridded Binary (GRIB) data, text watch/warn triggered scripts, etc.

2.13.3 Basic AWIPS II Data Storage Architecture

AWIPS II provides data storage in two areas – Raw Data Storage and Processed Data Storage. These two data storage areas use a combination of PostgreSQL database tables, HDF5 files, and raw data files to store data.

Raw Data Storage contains the raw data that has been received from various data sources. The Raw Data Storage area is physically located on the AWIPS II NAS.

Processed Data Storage contains the decoded data. Processed Data Storage also hosts a collection of static data such as topological and map data. Most of this data is physically stored on either the AWIPS II Network Attached Storage (NAS) or the DAS device. The Processed Data Storage area is maintained by EDEX.

2.13.3.1 AWIPS II Raw Data Storage

AWIPS II Raw Data Storage is contained in a file system on the AWIPS II NAS. The EDEX Data Servers (DX1, DX2, DX3, and DX4), CPSBN, the PX servers and AX servers, and the LX and XT workstations mount Raw Data Storage off NAS. Raw Data
Storage is mounted on /data_store on all these systems. The accessibility of the Raw Data Storage is shown in Exhibit 2.13.3.1-1.

Exhibit 2.13.3.1-1. AWIPS II Raw Data Storage

As shown in Exhibit 2.13.3.1-1:

1. AWIPS II Raw Data Storage is located in NAS [NAS2].
2. All the hosts (DX1, DX2, DX3, DX4, AX, CPSBN, and Workstations) mount volume off NAS.

The basic structure of Raw Data Storage is

/data_store/<data type>

The directory structure under each data type as well as the file names used for the raw data are determined by the configuration of LDM(D).

Procedures for verifying the mount points are presented in Chapter 25 of this manual.

2.13.3.2 AWIPS II Processed Data Storage

The AWIPS II Processed Data Storage is used to store decoded data. The decoded data is stored in a PostgreSQL database and/or HDF5 files, depending on the type of data. (Processed data storage is covered in Chapters 4 – 7; techniques for checking data storage are covered in Chapter 25.) The basic structure of the Processed Data Storage is shown in Exhibit 2.13.3.2-1.
As shown in Exhibit 2.13.3.2-1, Processed Data Storage is located on the DAS and NAS devices. Note that file systems on the DAS are accessible only from the DX1/2 cluster.

1. On the NAS, two file systems contain portions of Processed Data Storage. The two file systems are /aiidata and /GFESuite2.
   a. Processes on DX3/4 and PX1/2 have access to the entire NAS:/aiidata file system via NFS mount on /awips2/edex/data. This provides access to the Localization Store, hydro applications, and static data.
   b. Processes on the LX and XT workstations have access to the share portion of the NAS:/aiidata file system via NFS mount on /awips2/edex/data/share. This provides access to hydro applications and aviation climo data.
   c. Processes on PX1/2 have access to the manual portion of the NAS:/aiidata file system. This provides access to the manual inbox for data insertion.
d. Processes on DX servers and PX servers have access to the NAS:/GFESuite2. This provides access to the Service Backup and Inter-Site Coordination (ISC) functionality.

e. Raw Data Storage (NAS:/data_store): All the hosts (DX1, DX2, DX3, DX4, CPSBN, AX and Workstations) mount volume off NAS.

f. QPID shared data (NAS /qpid) is accessed from CP1f.

g. /aiidata/utility (NAS) is accessed from DX1/2

2. On the DAS, two file systems contain portions of Processed Data Storage. The two file systems are /aiihdf5 and /aiidb.

a. Processes on DX1/2 have access to the DAS:/aiidb file system via local mount on /awips2/data. This provides access to the PostgreSQL database tables.

b. Processes on DX1/2 have access to the DAS:/aiihdf5 via local mount on /awips2/edex/data/hdf5. This provides access to HDF5-based data storage for decoded imagery, forecast grids and point data.

Note that client applications such as CAVE do not directly access the HDF5 files in Processed Data Storage. Rather, CAVE interacts through the EDEX Request process and PYPIES, as shown in Exhibit 2.13.3.2-2.

As shown in Exhibit 2.13.3.2-2, CAVE data retrieval is based on data type and consists of the following:

1. CAVE sends a data request to an EDEX Request process on DX3 or DX4. [Note: When loading satellite and radar data, CAVE will work with the request (which works with postgres) to get a list of the datauri it wants and then CAVE will go through pypies to retrieve that data. For point data, like obs, CAVE will ask request for that data, and it will get the requested data from postgres and then pypies, and
then send it back to the requesting CAVE. GFE data interaction also works this way, simply because of the strict transaction control needed for that data.

a. On the CP cluster, IPVS simply forwards the connection request to either DX3 or DX4. Note that EDEX is not aware of the IP aliasing.

b. Once the connection is made, EDEX and CAVE communicate directly.

2. The EDEX Request process obtains metadata and data from Processed Data Storage.
   a. Metadata is obtained from the PostgreSQL database (DBMS).
   b. Depending on the data type, data is retrieved from the PyPIES server (a2dx2) or the PostgreSQL database.
      i. Imagery, point data, and gridded data are retrieved from the PyPIES server.
      ii. Other decoded data, such as text products, are retrieved from the PostgreSQL database.

3. The data and metadata are returned to CAVE for display. Note that the response is directly from EDEX Request to CAVE; it does not pass through IPVS. CAVE is also capable of inserting data into Processed Data Storage. HDF5 data is delivered directly from PyPIES to the requesting CAVE client.

Procedures for verifying access to Processed Data Storage are presented in Chapter 25.

2.13.3.3 PyPIES (Python Process Isolated Exchange Storage)

PyPIES was created for AWIPS II to isolate the management of HDF5 Processed Data Storage from the EDEX processes. PyPIES manages access, i.e., reads and writes, of data in the HDF5 files. In a sense, PyPIES provides functionality similar to a DBMS; all data being written to an HDF5 file is sent to PyPIES, and requests for data from an HDF5 file are processed by PyPIES.

PyPIES is implemented in two parts:

1. The PyPIES manager is a Python application that runs as part of an Apache HTTP server on dx1f. It is the PyPIES manager that handles requests to store and retrieve data. The Apache HTTP server provides request management and isolates the HDF5 read/write to a separate process.

2. The PyPIES logger is a Python process that coordinates logging.

PyPIES is controlled by the httpd-pypies System V script; it is part of the a2dx2apps high-availability package.

2.13.4 Basic AWIPS II Data Retrieval Architecture

Data retrieval is the process used by an AWIPS II client to obtain data with the EDEX Servers. In making the data retrieval, the client interacts with an EDEX Request process; the Request process obtains the requested data from the Processed Data Store and returns it to the client. The basic data retrieval process is shown in Exhibit 2.13.4-1.
As shown in Exhibit 2.13.4-1:

1. The client sends a request via TCP protocol to a virtual host name managed by IPVS.
2. IPVS forwards the request, which includes the return address of the client, to an available EDEX Request process.
3. The EDEX Request process obtains the requested metadata from the AWIPS II database.
4. The EDEX Request process forwards the requested metadata directly to the client.
5. If the client has requested data which resides in hdf5 store, then it communicates directly with PyPIES on dx2f to retrieve the necessary information to visualize the data.

This architecture allows the client access via any available EDEX Request process, resulting in improved system reliability and accessibility.

2.14 Basic AWIPS II Data Retransmission Design

AWIPS II uses the sequence numbers provided by the SBN to implement an automated retransmission capability for missed products delivered by the SBN. The retransmission capability has been implemented via a daemon process, sbn_retransmit, which is installed as part of the LDM installation on the CPSBN1/2 cluster. The AWIPS II Message Retransmission Design is shown in Exhibit 2.14-1. Notice that AWIPS II reuses most of the existing retransmission infrastructure. [Note: All channels except GOES support automatic retransmission. GOES does not because the NCF software does not support it.]
To determine if a product retransmission is required, sbn_retransmit monitors the LDM product queue. When a gap is detected, sbn_retransmit forwards a request to the Network Control Facility (NCF) for a retransmission of the missing products.

2.15 Basic AWIPS II Data Purge Architecture

AWIPS II stores data in two separate collections: 1) the AWIPS II Raw Data Storage, which contains unprocessed data; and 2) the AWIPS II Processed Data Storage, which contains decoded and other processed data. These two data collections use different purge mechanisms.

2.15.1 AWIPS II Processed Data Storage Purge Architecture

AWIPS II has implemented a data plug-in based purge strategy. The default purge strategy is version-based, similar to AWIPS I’s version-based purge strategy. Each data plug-in is able either to use the default strategy or to implement its own strategy. The GFE and text data plug-ins implement their own purge strategies based on the purge strategies used in AWIPS I; all other data plug-ins currently use the default, version-based purge strategy.
Rules for the version-based purge are defined in XML files located in the AWIPS II Localization Store, which is in /awips2/edex/data/utility on DX3/4. See Chapter 11 for more information on the purging and purge rules.

The basic Processed Data Storage purge strategy is shown in Exhibit 2.15.1-1.

As shown in Exhibit 2.15.1-1:

1. A quartz event is triggered in the EDEX Ingest process causing the Purge Service to execute.
   a. The EDEX Purge Service obtains a purge lock; if the lock is already taken, the Purge Service exits. This strategy ensures that only a single EDEX Ingest process actually performs the purge.
   b. The EDEX Purge Service obtains the purge rules for each data plug-in.
2. The EDEX Purge Service sends a delete message to the PostgreSQL DBMS.
3. The PostgreSQL DBMS deletes the data from the database table(s).
4. If HDF5 data is to be purged, the EDEX Purge Service sends a purge message to the PyPIES service.
5. The PyPIES service deletes the data from the HDF5 file(s).

2.15.2 AWIPS II Raw Data Storage Purge Architecture

AWIPS II Raw Data Storage is purged by EDEX based on entries in RAW_DATA.xml. 

Note: Starting 13.5.3 release, a brand-new and customizable archiving capability is provided. It populates an archiving staging area (/archive) on an hourly basis with extracts from the database (.bin files) and compressed versions of the HDF files. It also includes an archive purge function so that both raw and processed data can remain on the real-time system for up to 7 days. This function replaces the LDM scour. From the 7-day storage (also known as the rollover), a new case generation GUI saves data cases so they can be transferred to other media for long-term storage or to the WES-2 Bridge for later viewing or playback.
2.16 **AWIPS II Localization Store**

AWIPS II uses a Base/Site/User model for localizing/customizing software. The Localization Store consists of a directory structure (/aiidata/utility) that is visible at /awips2/edex/data/utility on the EDEX Data Servers (DX 3 and DX4). When CAVE is started on an LX workstation (or as the Text Workstation application on an XT workstation), CAVE obtains the latest localization information from the EDEX Request process and updates the local localization cache, found in <user-home>/caveData. This caching mechanism allows CAVE to: 1) use the latest available localization data for operation; and 2) utilize the same user data on any workstation. CAVE’s localization interaction is shown in Exhibit 2.16-1.

![Exhibit 2.16-1. CAVE/Localization Store Interaction](image)

As shown in Exhibit 2.16-1, the centralized Localization Store is accessed via the EDEX Request process. This access occurs when CAVE is started and as files for the site localization are modified. Prior to requesting a localization file, the cached copy in caveData is checked for currency; the file is requested from EDEX only if the cached copy is out of date. In that case, the file from the Localization Store overwrites the local copy.

Modification of a site’s localization is performed using the Localization Perspective in CAVE. Any changes made using the Localization Perspective are propagated back to the Localization Store.

[Note: Do not manually modify files located under caveData; manual changes are not propagated to the Localization Store and will be overwritten when CAVE is restarted.]

2.17 **EDEX Distribution Files**

Distribution files tell EDEX how to invoke a decoder plugin on a raw data file. The base files are in /awips2/edex/data/utility/edex_static/base/distribution/. Site-level files are in /awips2/edex/data/utility/edex_static/site/\<your site\>/distribution. Each plugin has a distribution file that contains the files that the plugin can process.
Raw files are written to /data_store, and a message is sent via QPID to the EDEX distribution service from either the LDM (a2cp1apps), the RadarServer, or LDAD. The distribution service compares the file header that is sent in the message against the regular expressions in every distribution file. When it finds a match, the raw file is placed in a queue for the matching plugin to decode and process. The distribution files are used to match file headers as well as filenames, which is how files dropped into EDEX’s manual endpoint (/awips2/edex/data/manual) are processed. Because the RadarServer actually writes its raw files according to the AWIPS I radar directory structure (/data_store/radar/<radarid>/<product mnemonic>/....) those files would have to be renamed before being dropped into the manual endpoint for manual reprocessing.

For example, a particular 0.5 base reflectivity product from the KOAX radar written by the RadarServer would be located in /data_store/radar/koax/Z/elev0_5/res0_25/az0_5/level256/koax.153.20120106_1625. (In this filename, the “153” designates the radar product number; in this case, 153 stands for a super-resolution 8-bit base reflectivity product.) By looking at the radar.xml file, which is included in the examples provided in section 2.17.1, you can see that a koax.* filename would not be matched. In real-time processing, the radar server adds “RadarServer” to the beginning of the file header sent in the message via QPID (for example, “RadarServer.koax.153.20120106_1625”). So, if these radar files needed to be manually reprocessed using the manual endpoint for any reason, either a site-level override of the distribution file needs to include a pattern that starts with “koax,” or the koax.* files need to be renamed to RadarServer.koax.* prior to copying them into the manual endpoint.

2.17.1 Editing an EDEX Distribution File

Because these files are in edex_static, they have to be manually edited using a text editor. You should not edit the base files; rather, you should copy the base version to your site and then edit the site version. The regular expressions in the distribution files need to correspond with the regular expressions in the LDM pqact.conf file. If patterns exist in pqact.conf but are not in the distribution files, then raw data files will be written to /data_store but would never be ingested and processed by EDEX. Entries for these non-ingested files would be written to the unrecognized files log in /awips/edex/logs.

Examples follow.

**obs.xml:** Processes any file header that starts with “SA” or “SP”, which should match any WMO header that contains METAR data (e.g., SAUS, SPUS, SACN, SAMX).

```xml
<requestPatterns xmlns:ns2="group">   <regex>^[A-AP].*</regex>
</requestPatterns>
```

**text.xml:** Processes lots of WMO patterns. The second pattern `^[A-CEG-Z].*` matches any header that starts with “S” except for “SD” or “SF,” so it also matches the “SA” and “SP” files that the obs.xml plugin matches. This means that METARs are processed by both plugins simultaneously.

```xml
<requestPatterns>   <regex>^[ACFNRUW][A-Z].*</regex>   <regex>^[A-CEG-Z].*</regex>   <regex>^[T][BCX].*</regex>   <regex>^[S][A-OQ-TV-
```
radar.xml: Matches files from the LDM and the RadarServer. Files that match the pattern ^SDUS4[1-6].* in obs.xml also match ^SDUS[234578]..* in radar.xml. This means that certain radar products that have text data are processed by both plugins. The RadarServer’s file header messages always begin with “RadarServer”; they do not have WMO headers.

<dhr.xml: Matches a subset of RadarServer files and some of the radar files that arrive over the SBN. The numbers at the end of the RadarServer regular expressions refer to the radar product numbers (32 = DHR, 80 = STP, 138 = digital storm total precip; these numbers are defined in interface control documents published by the Radar Operations Center at [http://www.roc.noaa.gov](http://www.roc.noaa.gov)).
Chapter 3
Individual User Accounts
Chapter 3. Individual User Accounts

Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1 General Guidance on Individual User Accounts</td>
<td>1</td>
</tr>
<tr>
<td>3.2 Managing Individual User Accounts Using setupAwipsUser.sh</td>
<td>1</td>
</tr>
<tr>
<td>3.2.1 Creating a New User Account</td>
<td>2</td>
</tr>
<tr>
<td>3.2.2 Re-Creating Desktop and ssh Keys for an Existing User Account</td>
<td>4</td>
</tr>
<tr>
<td>3.2.3 Removing an Existing User Account</td>
<td>6</td>
</tr>
<tr>
<td>3.3 High-Level View of setupAwipsUser.sh</td>
<td>6</td>
</tr>
<tr>
<td>3.4 Changing User or Device Password Using the Command Line</td>
<td>7</td>
</tr>
<tr>
<td>3.4.1 Changing Non-Root Accounts</td>
<td>7</td>
</tr>
<tr>
<td>3.4.1.1 User Change to Password</td>
<td>7</td>
</tr>
<tr>
<td>3.4.1.2 Root Change to User Account Password</td>
<td>8</td>
</tr>
<tr>
<td>3.4.2 Changing Root Password on Linux Devices</td>
<td>8</td>
</tr>
<tr>
<td>3.4.3 Changing NAS CLI and Console Root Passwords</td>
<td>9</td>
</tr>
<tr>
<td>3.4.3.1 Changing NAS CLI Root Account</td>
<td>9</td>
</tr>
<tr>
<td>3.4.3.2 Changing NAS Console Root Account</td>
<td>10</td>
</tr>
<tr>
<td>3.4.3.3 Testing CLI Root Password</td>
<td>11</td>
</tr>
<tr>
<td>3.5 Synchronizing SSH Keys with VerifySSHkeys.sh</td>
<td>12</td>
</tr>
<tr>
<td>3.6 Synchronizing CRS SSH Keys with installCRSssh.sh</td>
<td>12</td>
</tr>
<tr>
<td>3.7 Recover from a Screensaver Lockout</td>
<td>15</td>
</tr>
<tr>
<td>3.8 User Name Procedures</td>
<td>15</td>
</tr>
<tr>
<td>3.9 Localization Perspective User Roles (userRoles.xml)</td>
<td>15</td>
</tr>
</tbody>
</table>
Individual user accounts are used to control access to AWIPS. They provide authorized users with a way to log in to AWIPS and use its resources, while maintaining system security.

The AWIPS System Manager sets up individual user accounts. For each user, the same account will be used on both the graphics and the text workstations.

### 3.1 General Guidance on Individual User Accounts

User names should follow the convention `<first initial><last name>`, all lower case. If necessary, the last name should be truncated so that the user name is no more than eight characters long. Use an integer at the end of the user name if necessary to prevent duplication. Examples: `gbush`, `gbush2`, `gwashing`, `gwashin2`.

Account names must:
- Consist of eight or fewer characters
- Start with a letter
- Contain only lower case letters and numbers
- Be unique.

User passwords must be consistent with the following criteria:
- Passwords for user accounts must have at least twelve (12) non-blank characters.
- Passwords must contain characters from at least three (3) of the following four (4) categories:
  - English upper case characters (A … Z);
  - English lower case characters (a … z);
  - Base 10 digits (0 … 9); and
  - Non-alphanumeric characters (e.g., $#%).
- Passwords must not contain common words, nouns, pronouns, acronyms, contractions, and geographic locations (i.e., dictionary words).

### 3.2 Managing Individual User Accounts Using `setupAwipsUser.sh`

You must use the script `/home/awipsadm/install/setupAwipsUser.sh`, rather than any system GUI or direct calls to command line utilities, to manage individual user accounts. This script can be used to add a new user, remove an existing user, and re-synch ssh for an existing user. This will ensure that the AWIPS workstation software is running in an environment that has been tested and which is supported by the NCF.

The general format of the `setupAwipsUser.sh` command is:

```
setupAwipsUser.sh <username> [SSH] [PROMPT] [REMOVE] <user real name>
```

---

Script options are displayed by executing the script with no options.

```
[root@dx1-ntcb install]# ./setupAwipsUser.sh
ERROR : please include a username as a paramater!
USAGE : /home/awipsadm/install/setupAwipsUser.sh <username> <SSH|RSH> [PROMPT|REMOVE]<real name>
<username> can be up to 8 characters
<SSH|RSH> must be either SSH to use Secure shell or RSH to use remote
shell
use PROMPT as the third paramater to make the script interactive
use REMOVE as the third paramater to disable a user created by this
script
<real name> is the real name field for /etc/passwd [EXAMPLE: John E. Doe,x8605]
  it can be up to 9 space delimited strings
```

### 3.2.1 Creating a New User Account

To create a new individual account for “George Washington,” log on to DX1 as user root and execute the following commands:

```
TYPE: cd /home/awipsadm/install
TYPE: ./setupAwipsUser.sh gwashing SSH "George Washington"
```

*Note:* The new user will be added to /awips/fxa/data/fxa-users, but the user will not be removed from this file when the account is removed from the system. Over time, this file will grow and will require manual trimming by the ESA or the ITO.

```
[root@dx1-ntcb install]# ./setupAwipsUser.sh gwashing SSH George Washington
Checking that gwashing is not an existing administrator user name /usr/local/bin/ssh test to lx1 passes!
We will use lx1 to populate desktop for new account...
Account name entered ----> gwashing
This is a new user account.
Real name entered ----> George Washington
User id entered ----> 127
Setting up account for gwashing...
******************************************************************************
NOTE that you must create a password for gwashing!
Then execute /var/yp/ypmake on dx1.
As root on dx1:
  prompt> passwd gwashing
  prompt> /var/yp/ypmake
(gwashing is locked until a password is created.)
```
pushing out passwd changes via NIS...
gmake[1]: Entering directory `/var/yp/ntcb.awips1'
gmake[1]: `ypservers' is up to date.
gmake[1]: Entering directory `/var/yp/ntcb.awips1'
Updating passwd.byname...
Updating passwd.byuid...
Updating group.byname...
Updating group.bygid...
Updating shadow.byname...
Updating netid.byname...
gmake[1]: Entering directory `/var/yp/ntcb.awips1'
creating .cshrc for gwashing...
creating .login for gwashing...
creating .sawfishrc for gwashing...
/home/awipsadm/install/GnomeLXdesktop.tar exists not recreating it!
  .gnome files
  .kde files
fixing desktop files...
Running install-gnome-kde.sh gwashing

Creating gwashing desktop files...
mv: cannot stat `./.gnome2/nautilus-scripts/D2D (lx) - TextWS (xt)': No such
  file or directory
mv: cannot stat `./.gnome2/nautilus-scripts/AWIPS start-up menu': No such file
  or directory
mv: cannot stat `./.gnome-desktop/zzzztemp\'s Home': No such file or directory
mv: cannot stat `./.gnome2/nautilus-scripts/D2DlxTextWSxt': No such file or
directory
mv: cannot stat `./.gnome2/nautilus-scripts/AWIPSstartupmenu': No such file or
directory
mv: cannot stat `./.gnome-desktop/zzzztempsHome': No such file or directory
mv: cannot stat `./.kde/share/apps/konqsidebartng/entries/zzzztemp.desktop':
No such file or directory
tarring up new desktop and installing into /home/gwashing...

Done creating gwashing desktop...

Updating User's AWIPS Desktop Complete...
setting up /var/local/bin/ssh keys for gwashing...
Making /home/gwashing/.ssh
Generating public/private dsa key pair.
Your identification has been saved in /home/gwashing/.ssh/id_dsa.
Your public key has been saved in /home/gwashing/.ssh/id_dsa.pub.
The key fingerprint is:
chown: cannot access `/home/gwashing/.gnome/magicdev': No such file or
directory
Copying gwashing's .ssh dir to dx1:/awips/.ssh/gwashing
Copying gwashing's .ssh dir to dx2:/awips/.ssh/gwashing
Copying gwashing's .ssh dir to dx3:/awips/.ssh/gwashing
Copying gwashing's .ssh dir to dx4:/awips/.ssh/gwashing
Copying gwashing's .ssh dir to px1:/awips/.ssh/gwashing
Copying gwashing's .ssh dir to px2:/awips/.ssh/gwashing
Copying gwashing's .ssh dir to ax:/awips/.ssh/gwashing
Copying gwashing's .ssh dir to lx1:/awips/.ssh/gwashing
Copying gwashing's .ssh dir to lx2:/awips/.ssh/gwashing
Copying gwashing's .ssh dir to lx3:/awips/.ssh/gwashing
Chapter 3 Individual User Accounts

Re-Creating Desktop and ssh Keys for an Existing User Account

The script can be used to regenerate the desktop and ssh keys for an individual user. You will be prompted to continue.

This user account already exists.
Ensuring gid, shell, and home dir are standard for gwashing...

This is an existing user - do you want to rebuild the desktop (y/n)?

```
[root@dx1-ntcb install]# ./setupAwipsUser.sh gwashing SSH George Washington
[...]
```

```
Creating .cshrc for gwashing...
Creating .login for gwashing...
```

```
Creating .sawfishrc for gwashing...
/home/awipsadm/install/GnomeLXdesktop.tar exists not recreating it!
```

```
Running install-gnome-kde.sh gwashing
```

```
done running setupAwipsUser.sh...
[root@dx1-ntcb install]#```
Creating gwashing desktop files...
mv: cannot stat `./.gnome2/nautilus-scripts/D2D (lx) - TextWS (xt)'
  : No such file or directory
mv: cannot stat `./.gnome2/nautilus-scripts/AWIPS start-up menu'
  : No such file or directory
mv: cannot stat `./.gnome-desktop/zzzztemp's Home'
  : No such file or directory
mv: cannot stat `./.gnome2/nautilus-scripts/D2DlxTextWSxt'
  : No such file or directory
mv: cannot stat `./.gnome2/nautilus-scripts/AWIPSstartupmenu'
  : No such file or directory
mv: cannot stat `./.gnome-desktop/zzzztempsHome'
  : No such file or directory
mv: cannot stat `.kde/share/apps/konqsidebartng/entries/zzzztemp.desktop'
  : No such file or directory
tarring up new desktop and installing into /home/gwashing...

Done creating gwashing desktop...

Updating User's AWIPS Desktop Complete...
setting up /usr/local/bin/ssh keys for gwashing...
Recreating /home/gwashing/.ssh
Generating public/private dsa key pair.
/home/gwashing/.ssh/id_dsa already exists.
Overwrite (y/n)? Your Identification has been saved in
/home/gwashing/.ssh/id_dsa.
Your public key has been saved in /home/gwashing/.ssh/id_dsa.pub.
The key fingerprint is:
chown: cannot access `/home/gwashing/.gnome/magicdev'
  : No such file or directory
Copying gwashing's .ssh dir to dx1:/awips/.ssh/gwashing
Copying gwashing's .ssh dir to dx2:/awips/.ssh/gwashing
Copying gwashing's .ssh dir to dx3:/awips/.ssh/gwashing
Copying gwashing's .ssh dir to dx4:/awips/.ssh/gwashing
Copying gwashing's .ssh dir to px1:/awips/.ssh/gwashing
Copying gwashing's .ssh dir to px2:/awips/.ssh/gwashing
Copying gwashing's .ssh dir to ax:/awips/.ssh/gwashing
Copying gwashing's .ssh dir to lx1:/awips/.ssh/gwashing
Copying gwashing's .ssh dir to lx2:/awips/.ssh/gwashing
Copying gwashing's .ssh dir to lx3:/awips/.ssh/gwashing
Copying gwashing's .ssh dir to lx4:/awips/.ssh/gwashing
Copying gwashing's .ssh dir to lx5:/awips/.ssh/gwashing
Copying gwashing's .ssh dir to xt1:/awips/.ssh/gwashing
Copying gwashing's .ssh dir to xt2:/awips/.ssh/gwashing
Copying gwashing's .ssh dir to xt3:/awips/.ssh/gwashing
Copying gwashing's .ssh dir to xt4:/awips/.ssh/gwashing
Copying gwashing's .ssh dir to xt5:/awips/.ssh/gwashing
Adding gwashing to /awips/fxa/data/fxa-users on workstations...
Copying /awips/fxa/data/fxa-users to lx1-ntcb...
Copying /awips/fxa/data/fxa-users to lx3-ntcb...
Copying /awips/fxa/data/fxa-users to lx4-ntcb...
Copying /awips/fxa/data/fxa-users to lx5-ntcb...
Copying /awips/fxa/data/fxa-users to xt1-ntcb...
Copying /awips/fxa/data/fxa-users to xt2-ntcb...
Copying /awips/fxa/data/fxa-users to xt3-ntcb...
Copying /awips/fxa/data/fxa-users to xt4-ntcb...
Copying /awips/fxa/data/fxa-users to xt5-ntcb...

done running setupAwipsUser.sh...
3.2.3 Removing an Existing User Account

You will be prompted to enter REMOVE to continue. Edit /awips/fxa/data/fxa-users manually and remove the account on every LX and XT workstation.

```
[root@dx1-ntcb install]# ./setupAwipsUser.sh gwashing SSH REMOVE
Checking that gwashing is not an existing administrator user name

WARNING : YOU ARE ABOUT TO DISABLE gwashing

ARE YOU SURE YOU WANT TO DISABLE gwashing?
enter REMOVE to continue: REMOVE

gwashing BEING DISABLED!

Using userdel on gwashing and then running ypmake

gmake[1]: Entering directory `/var/yp/ntcb.awips1'
gmake[1]: `ypservers' is up to date.
gmake[1]: Leaving directory `/var/yp/ntcb.awips1'
gmake[1]: Entering directory `/var/yp/ntcb.awips1'
Updating passwd.byname...
Updating passwd.byuid...
Updating group.byname...
Updating group.bygid...
Updating shadow.byname...
Updating netid.byname...
gmake[1]: Leaving directory `/var/yp/ntcb.awips1'
Removing gwashing from ax archiver authorized_keys2 file
Removing gwashing from ls ldad authorized_keys2 file

/home/gwashing is being backed up to /home/gwashing.SAVEOLDUSER

Removing gwashing's .ssh dir from dx1:/awips/.ssh/gwashing
Removing gwashing's .ssh dir from dx2:/awips/.ssh/gwashing
Removing gwashing's .ssh dir from dx3:/awips/.ssh/gwashing
Removing gwashing's .ssh dir from dx4:/awips/.ssh/gwashing
Removing gwashing's .ssh dir from px1:/awips/.ssh/gwashing
Removing gwashing's .ssh dir from px2:/awips/.ssh/gwashing
Removing gwashing's .ssh dir from ax: /awips/.ssh/gwashing
Removing gwashing's .ssh dir from lx1:/awips/.ssh/gwashing
Removing gwashing's .ssh dir from lx2:/awips/.ssh/gwashing
Removing gwashing's .ssh dir from lx3:/awips/.ssh/gwashing
Removing gwashing's .ssh dir from lx4:/awips/.ssh/gwashing
Removing gwashing's .ssh dir from lx5:/awips/.ssh/gwashing
Removing gwashing's .ssh dir from xt1:/awips/.ssh/gwashing
Removing gwashing's .ssh dir from xt2:/awips/.ssh/gwashing
Removing gwashing's .ssh dir from xt3:/awips/.ssh/gwashing
Removing gwashing's .ssh dir from xt4:/awips/.ssh/gwashing
Removing gwashing's .ssh dir from xt5:/awips/.ssh/gwashing

gwashing DISABLED!
```

3.3 High-Level View of setupAwipsUser.sh

A high-level view of what the setupAwipsUser.sh script does follows. For further details, please review the script itself.
Chapter 3 Individual User Accounts

- Prompts for certain information unless that information is provided on the command line (new user’s real name [e.g., George J. Washington], account name [e.g., gwashing], etc.).
- Creates a new account with shell /bin/csh (you can specify ksh if your site is an RFC), group fxalpha (group ID 200), and home directory in /home. It also creates the new home directory.
- Creates the new user’s .cshrc file. The .cshrc will source SFXA_HOME/.cshrc.
- Creates the new user’s .login file. The .login file will call xhost to allow processes running on the server to pop up windows on the user’s display.
- Creates a .sawfishrc file for the new user. The .sawfishrc file will cause F11 key presses or left-clicks on the background (root) window to post the AWIPS menu.
- Installs and configures various Gnome, Sawfish, .KDE, and other setup files and directories to ensure that the user has a standard environment that will work on the AWIPS three-headed graphics workstation and on the text workstation.
- Generates and installs SSH keys so that scripts running under the user’s account can transfer files and execute commands remotely.
- Adds the new user name to the D-2D user list maintained in /awips/ fxa/data/ fxa-users.

NOTE: You can also use setupAwipsUser.sh to reconfigure an existing account for use with AWIPS software. The account’s environment files will be overwritten, its home directory may be changed, and its group ID may be changed.

The setupAwipsUser.sh script has a number of other options. Please see the header documentation in the script for a description of them.

3.4 Changing User or Device Password Using the Command Line

User names and device passwords can be changed or updated using a Terminal window and Linux commands. The /etc/passwd file should be the same on all devices.

This section provides guidance on changing user or device passwords using the Command line. For the rules on choosing an acceptable password, see section 3.1.

3.4.1 Changing Non-Root Accounts

User accounts are controlled through NIS and can be updated by the user or by root.

3.4.1.1 User Change to Password

From a workstation, the user can update his or her NIS-controlled password with the yppassword command.
Changing NIS account information for gwashing on dx1-ntcb.
Please enter old password:
Changing NIS password for gwashing on dx1-ntcb.
Please enter new password:
Please retype new password:

The NIS password has been changed on dx1-ntcb.

The user can test the change by performing an su to his or her account.

The superuser can change a user account password by updating /etc/shadow on the NIS master server (dx1) with the passwd command and then updating the NIS shadow file with /var/yp/ypmake.

The root account is not under NIS control so the root password must be changed on every AWIPS device. To change root password on the Linux devices, log in to each (LX, AX, DX, and PX) separately as root. A simple loop at the shell prompt from the dx1 is a fast way to change the root password. Tip: You can speed up the process further by typing the root password at the command line prompt, highlighting it with your mouse, and then pressing the middle mouse button when prompted for the root password.
3.4.3 Changing NAS CLI and Console Root Passwords

The NAS root account must be changed for both the console and CLI mode.

3.4.3.1 Changing NAS CLI Root Account

From a workstation, ssh to the ATS as root and start the ts_menu. The ATS root password is “access”.

[Note: The output that follows is ats1 from dx1-tbdw.]

```
[root@dx1-tbdw ~]# ssh root@ats1
Password: <access>
[root@ats1 root]# ts_menu
```

Select option 8 from the ATS menu to connect to the NAS.

```
Serial Console Server Connection Menu for your Master Terminal Server

1 DX1            2 DX2            3 DX3            4 DX4
5 PX1            6 PX2            7 AX             8 NAS
9 02-b6-65P9    10 02-b6-65P10   11 P5W            12 DMZ
13 ROUTER1       14 ROUTER2       15 HS1            16 HSW2
17 FW1           18 FW2           19 VIR            20 ADTRAN1
21 LX1           22 LX2           23 LX3            24 LX4
25 02-b6-65P25   26 02-b6-65P26   27 02-b6-65P27   28 02-b6-65P28
29 02-b6-65P29   30 02-b6-65P30   31 XT1            32 XT2
33 XT3           34 XT4           35 XT5            36 02-b6-65P36
37 CPSBN         38 CPSBN2        39 M&C_modem     40 DX1-DAS
41 DX2-DAS       42 02-b6-65P42   43 02-b6-65P43   44 02-b6-65P44
45 02-b6-65P45   46 02-b6-65P46   47 OPEN          48 OPEN

Type 'q' to quit, a valid option[1-48], or anything else to refresh : 8
```

Entering character/text Mode
Escape character is ^]
If you see “->” or “SUNSP00144F793780 login:” then you are already at the CLU prompt. If you see the NAS Console prompt nas1-<site> then type <Shift><Esc><9> to stop the NAS Console.

```
nas1-ntca > <Shift><Esc><9>
Serial console stopped.
->
```

Change the CLI root password by typing `set /SP/users/root password`. You will be prompted twice to enter the new password.

```
-> set /SP/users/root password
Changing password for user /SP/users/root...
Enter new password: ********
Enter new password again: ********
New password was successfully set for user /SP/users/root
```

After you have successfully changed the NAS root password, restart the NAS console by typing `start /SP/console` and then press Enter twice.

```
-> start /SP/console
Are you sure you want to start /SP/console (y/n)? y
Serial console started. To stop, type ESC ( <Enter><Enter>
```

### 3.4.3.2 Changing NAS Console Root Account

Change the NAS root password at the Console prompt by typing “password”. You will be prompted twice to enter the new password.

```
nas1-ntca > password
Changing password for admin
<CR> to abort, * to clear password
enter new password for admin : ********
enter again to confirm : ********
password changed
nas1-ntca > 02/07/11 19:43:53 I Environment saved.
```

When you have successfully changed the NAS root password, exit the NAS telnet connection by typing <Ctrl>[<] and then “e” to exit.

```
nas1-ntca > ^] Console escape. Commands are:
l  go to line mode
c  go to character mode
z  suspend telnet
b  send break
t  toggle binary
e  exit telnet
e
```
Exit the ATS menu by typing “q”. Then type “exit” to close the ats1 connection. [Note: The output that follows is ats1 from dx1-tbdw.]

<table>
<thead>
<tr>
<th>Serial Console Server Connection Menu for your Master Terminal Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 DX1</td>
</tr>
<tr>
<td>5 PX1</td>
</tr>
<tr>
<td>9 02-b6-65P9</td>
</tr>
<tr>
<td>13 ROUTER1</td>
</tr>
<tr>
<td>17 FW1</td>
</tr>
<tr>
<td>21 LX1</td>
</tr>
<tr>
<td>25 LX5</td>
</tr>
<tr>
<td>29 02-b1-33P29</td>
</tr>
<tr>
<td>33 XT3</td>
</tr>
<tr>
<td>37 CPSBN1</td>
</tr>
<tr>
<td>41 DX2-DAS</td>
</tr>
<tr>
<td>45 02-b6-65P45</td>
</tr>
</tbody>
</table>

Type 'q' to quit, a valid option[1-48], or anything else to refresh: q

[root@ats1 root]# exit
logout
Connection to ats1 closed.

3.4.3.3 Testing CLI Root Password

Connect to the NAS and stop the Console. At the CLI prompt, type “exit” to close the CLI session. You should see the “SUNSP00144F793780 login:” prompt.

`-> exit`

SUNSP00144F793780 login:

Log in to the NAS using the new root password. If you are successful you will see the CLI prompt.

SUNSP00144F793780 login: root
Password:
Sun(TM) Integrated Lights Out Manager
Version 1.1.1
Copyright 2006 Sun Microsystems, Inc. All rights reserved.
Use is subject to license terms.

Restart the NAS Console by typing “start /SP/console”. Perform step 2 to disconnect from the NAS.

`-> start /SP/console`
Are you sure you want to start /SP/console (y/n)? y
Serial console started. To stop, type ESC (nas1-ntcb >
3.5 **Synchronizing SSH Keys with VerifySSHkeys.sh**

The VerifySSHkeys.sh script sets up and synchronizes every account’s keys across every machine on the platform so that when you log in to any machine, it recognizes your keys. Because it recognizes your keys, you do not need a password between machines. **Please Note:** After running verifySSHkeys.sh the fxa user keys must be re-synchronized with CRS using installCRSssh.sh.

**On PX1 as root,**

```
TYPE: cd /home/awipsadm/ssh
```

```
TYPE: ./VerifySSHkeys.sh
```

**NOTE:** Make sure that no users are logged on to other servers. Note that it will take up to 10 minutes to execute this script. LDM user has been added to VerifySshKeys.sh so that passwordless ssh works as user ldm between dx1/2 (and other devices) to help maintain local pqact.conf.lll file between devices.

3.6 **Synchronizing CRS SSH Keys with installCRSssh.sh**

This installCRSssh script is used only at sites equipped with the Console Replacement System (CRS). AWIPS products configured to be sent to CRS are transferred via ssh. In order to facilitate this communication, AWIPS ssh keys for the fxa user must be installed on the CRS system. Please use caution! Without the correct configuration, NWR messages in AWIPS will not be sent to CRS.

The installCRSssh script will prompt for the root password on the CRS twice. The script assumes that the root password is the same on 0mp and 5mp (configurations). If the passwords are different, then either change the root password on the CRS to match, or contact the NCF for assistance.

The script will then copy over the fxa key to 0mp and 5mp. It will also copy back the rsa key for 0mp and 5mp, and install it in the known_hosts on each AWIPS device.

**Note:** If any keys are ever regenerated, by running VerifySshKeys.sh, or VIP keys on CRS, then the installCRSssh.sh script must be run again.

For future maintenance of the AWIPS-to-CRS connection, provide the root password of 0mp and 5mp to the NCF.

As user root on **DX1**, follow this procedure:

```
TYPE: script -a -f /local/install/installCRSkeys.out
```

```
TYPE: cd /home/awipsadm/ssh
```

```
TYPE: ./installCRSssh.sh (takes about 2 minutes)
```

```
TYPE: exit
```
Verify that the script worked correctly. As user fxa on DX1:

**TYPE:** `ssh crs@0mp uname -a`

**TYPE:** `ssh crs@5mp uname -a`

The commands should succeed without being prompted to enter either ‘yes’ or a password.

The commands should work on any device as user fxa. If you encounter problems, contact the NCF for assistance.

```
[root@dx1-ntcb ssh]# ./installCRSssh.sh
Starting CRS SSH Key setup, Wed Mar  9 14:17:30 GMT 2011
Compiling CRS authorized keys file...
Spawning process for CRS host and user keys...
Please type in root password for 0MP and 5MP and press enter:
Verify password and press enter:

Attempting to contact 0MP...
spawn scp crs@0MP:/usr/local/etc/ssh_host_rsa_key.pub /tmp/crs_0MPkey

crs@0mp's password:
ssh_host_rsa_key.pub 100%  218
 0.2KB/s  00:00

OMP completed!

Attempting to contact 5MP...
spawn scp crs@5MP:/usr/local/etc/ssh_host_rsa_key.pub /tmp/crs_5MPkey

crs@5mp's password:
ssh_host_rsa_key.pub 100%  218
 0.2KB/s  00:00

5MP completed!

Attempting to contact 0MP...
spawn scp /tmp/crs_userkey crs@0MP:

crs@0mp's password:
crs_userkey 100% 1804
 1.8KB/s  00:00

spawn telnet 0MP
Trying 165.92.107.241...
Connected to 0MP.
Escape character is "^]".
SCC UnixWare 7.1.1 (OMP) (pts/5)

login: crs
Password:
UnixWare 7.1.1
OMP
OMP{crs} su - root -c 'cp /crs/.ssh/authorized_keys /crs/.ssh/authorized_keys.bak'
Password:
OMP{crs} su - root -c 'cat /crs/crs_userkey >> /crs/.ssh/authorized_keys'
Password:
OMP{crs} exit
Connection closed by foreign host.
```
OMP completed!

Attempting to contact 5MP...
spawn scp /tmp/crs_userkey crs@5MP:
crs@5mp's password:
crs_userkey
   1.8KB/s  00:00
spawn telnet 5MP
Trying 165.92.107.242...
Connected to 5MP.
Escape character is '^['.
login: crs
Password:
UnixWare 7.1.1
5MP
5MP{crs} su - root -c 'cp /crs/.ssh/authorized_keys
/crs/.ssh/authorized_keys.bak'
Password:
5MP{crs} su - root -c 'cat /crs/crs_userkey >> /crs/.ssh/authorized_keys'
Password:
5MP{crs} exit
Connection closed by foreign host.

5MP completed!
Editing AWIPS hosts ssh_known_hosts files...
ax done
cpsbn1 done
cpsbn2 done
dx1-ntcb done
dx2-ntcb done
dx3-ntcb done
dx4-ntcb done
lx1-ntcb done
lx2-ntcb done
lx3-ntcb done
lx4-ntcb done
lx5-ntcb done
px1-ntcb done
px2-ntcb done
xt1-ntcb done
xt2-ntcb done
xt3-ntcb done
xt4-ntcb done
xt5-ntcb done
ls2 done
ls3 done
Finished with CRS SSH Key setup, Wed Mar  9 14:18:36 GMT 2011
[root@dx1-ntcb ssh]#

dx1-ntcb{fxa}9: ssh crs@0mp uname -a
UnixWare OMP 5 7.1.1 i386 x86at SCO UNIX_SVR5
dx1-ntcb{fxa}10:
dx1-ntcb{fxa}10: ssh crs@5mp uname -a
UnixWare 5MP 5 7.1.1 i386 x86at SCO UNIX_SVR5
dx1-ntcb{fxa}11:
3.7 Recover from a Screensaver Lockout

If a locking screen saver is running on a workstation and the password holder is not available, there are two ways to bring the workstation back into service:

- Kill the X server and thereby force a logout.
  
  **TYPE:** [Ctrl][Alt][Backspace]
  
  - The X server and all its clients (D2D, for example) will be killed. In a few seconds you will have a new login screen.

  **NOTE:** This method is strongly discouraged for routine logouts because it may cause problems with the AWIPS application software (undeleted temporary files, open socket connections, etc.). All currently known cases where an abnormal exit causes future problems have been fixed, but it is impossible to test for every situation.

  OR

- Kill the screen saver.

If a locking screen saver is running on an AWIPS workstation, there will be one or more processes called `xscreensaver` running. If you log into the locked workstation from another workstation and kill all of those processes, the screen savers will disappear and you will be able to continue with the login session that was previously locked.

The `xscreensaver` processes are owned by the account that logged in to the workstation. You can kill them only if you are logged in on that account or on the `root` account. Effectively, that means that only `root` can use this method, because, if you could log in to the user account with the password, you could unlock the screen saver in the normal way.

3.8 User Name Procedures

The user name procedures are stored in two places:

```
/home/<username>/caveData/etc/user/<username>/procedures and
/awips2/edex/data/utility/cave_static/user/<username>/procedures
```

The Practice Module in Section 3.3.9 of the *AWIPS CAVE-D2D User’s Manual* also includes information on how to use bundles; how to create and save bundles; and how to copy bundles from one procedure to another.

3.9 Localization Perspective User Roles (userRoles.xml)

This section provides an overview of the following topics as they pertain to managing user roles with respect to the ability to localize CAVE: implementation of user roles and overrides of user roles; base and site locations of the userRoles.xml file, which controls
user access to files in the localization perspective; how to edit the userRoles.xml file; and the structure and syntax of the userRoles.xml file.

- **Implementation and Overrides of User Roles**
  
  **Implementation**: User File.
  
  **Site Override**: Offices are able to create a site override of this file.

- **Base File Location**
  
  The BASE userRoles.xml file is located in the following directory:
  
  `/awips2/edex/data/utility/common_static/base/roles/`

- **Site File Location**
  
  To override the userRoles.xml file as site, the base file should be placed in the following directory and modified appropriately:
  
  `/awips2/edex/data/utility/common_static/site/XXX/roles/`

- **Editing the File**
  
  This file is not accessible in the localization perspective, and must be edited via a text editor on the system.
  
  **Note**: The User Admin GUI helps to manage this file.

- **Structure and Syntax**
  
  A description of the different sections within the file (userRoles.xml) and how they should be used follows.

  ➢ **Defining Permissions**
  
  Any and all file locations represented in the localization perspective can be defined in userRoles.xml. Each location is defined as a permission (`<permission id>`). The base file defines all the different file locations for those files accessible from the localization perspective and should not have to be overridden at the local office. For example, the following entry defines permission for the config directory used by AvnFPS.

  ```xml
  <permission id="com.raytheon.localization.site/cave_static/aviation/config"></permission>
  ```

  ➢ **Creating Users and Assigning Permission to Users**
  
  Once the desired permissions have been identified, the next section defines the users and those permissions assigned to those respective users. By default in the userRoles.xml file, all users have access to certain files in the Localization perspective via this entry:

  ```xml
  <user userId="ALL">
      <userPermission>com.raytheon.localization.site/common_static/purge</userPermission>
  </user>
  ```
The file access control provided by this file can be generic or granular. Offices can leave the default providing all users with access to all files, or it can limit user access based on office roles, etc. For example, the following entry gives the GFE Focal Point, jzeltwan, access to the four permissions controlling SITE-level GFE files:

```xml
<user userId="jzeltwan">
  <userPermission>com.raytheon.localization.site/common_static/gfe</userPermission>
  <userPermission>com.raytheon.localization.site/cave_static/gfe/combodata</userPermission>
  <userPermission>com.raytheon.localization.site/cave_static/gfe</userPermission>
  <userPermission>com.raytheon.localization.site/common_static/isc</userPermission>
  etc.
</user>
```

Creating Roles and Assigning Users to Roles

If you have multiple people performing the same role (i.e., GFE Focal Point), you can set up roles to which you can assign users, rather than creating the same permissions entry for multiple users. So, taking the GFE Focal Point example, first create a role (see the following for an example).

```xml
<role roleId="GFEFocal">
  <roleDescription>
    This is the GFE Focal Point Role
  </roleDescription>
  <rolePermission>com.raytheon.localization.site/common_static/gfe</rolePermission>
  <rolePermission>com.raytheon.localization.site/cave_static/gfe/combodata</rolePermission>
  <rolePermission>com.raytheon.localization.site/cave_static/gfe</rolePermission>
  <rolePermission>com.raytheon.localization.site/cave_static/gfe</rolePermission>
</role>
```
<rolePermission>com.raytheon.localization.site/common_static/isc</rolePermission>
</role>

And then assign users to the role. In the following example, jzeltwan and dcokely have been assigned to the role of GFE Focal Point:

<user userId="jzeltwan">
  <rolePermission>GFEFocal</rolePermission>
</user>

<user userId="dcokely">
  <rolePermission>GFEFocal</rolePermission>
</user>
Chapter 4
Data Flow Overview
Chapter 4: Data Flow Overview

Table of Contents

4.1 AWIPS Satellite Broadcast Network and Local Data Grids ................................................. 1
  4.1.1 AWIPS Satellite Broadcast Network ................................................................. 1
  4.1.2 Local Data Grids ......................................................................................... 3
4.2 WSR-88D/Terminal Doppler Weather Radar (TDWR) Network ........................................ 3
4.3 Local Data Processes .............................................................................................. 3
4.4 Data Routing Overview ......................................................................................... 5
4.5 Configuration Files ................................................................................................. 9
  4.5.1 LDM Configuration ..................................................................................... 9
  4.5.2 EDEX Data Distribution Files ....................................................................... 9
    4.5.2.1 Potential Problems ........................................................................ 11
4.6 Monitoring Data Ingest .......................................................................................... 12
  4.6.1 EDEX Log Types ....................................................................................... 12
  4.6.2 EDEX Log Format ...................................................................................... 16
4.7 General Monitoring of the EDEX System ............................................................... 19
  4.7.1 EDEX Log Event Levels ........................................................................... 20
4.8 Monitoring the LDM Data Flow .............................................................................. 21
4.9 Product Identifier .................................................................................................. 28
4.10 Qpid .................................................................................................................. 29
4.11 LDM Ingest Checklist ......................................................................................... 31

List of Exhibits

Exhibit 4.3-1. Code Sharing in AWIPS II ............................................................................. 4
Exhibit 4.4-1. LDM Data Flow ....................................................................................... 5
Exhibit 4.4-2. Radar Data Flow ...................................................................................... 8

List of Tables

Table 4.3-1. AWIPS II Software Functional Groups ......................................................... 4
The AWIPS data management system is responsible for the storage, retrieval, and maintenance of meteorological data sets. The various hardware and software components of this system handle data from two primary sources: the AWIPS Satellite Broadcast Network (SBN) and the WSR-1988 Doppler Weather Radar (WSR-88D) network.

4.1 **AWIPS Satellite Broadcast Network and Local Data Grids**

4.1.1 **AWIPS Satellite Broadcast Network**

The main source of nationally distributed data for AWIPS is the SBN. Hardware components unique to the SBN include the demodulators, the communications processors (CPSBN1 and CPSBN2), and the visible infrared (VIR) switches, which connect the digital video broadcast (DVB) receivers to the communications processor SBNs (CPSBN). Refer to Exhibits 2.2-1, 2.2-2, and 2.2.1-1 in Chapter 2.

Descriptions of the four channels follow.

1. **GOES.** The GOES Channel contains information from the GOES East and the GOES West satellites. The GOES East satellite is a data stream consisting of the following imagery products: visible infrared; and water vapor for the Eastern Conterminous United States (CONUS), Puerto Rico, supernational composites, and Northern Hemisphere (NH) composites. The GOES West satellite is a data stream consisting of the following imagery products: visible infrared and water vapor for CONUS, Alaska, and Hawaii as well as supernational composites, and NH composites.

2. **NMC.** From the NWS Telecommunications Gateway (NWSTG), a data stream consisting of model output from the National Centers for Environmental Prediction (NCEP); the observations, forecasts, watches and warnings produced by NWS Forecast Offices; WSR-88D radar products; and most observational data over North America.

3. **NMC2.** The NWSTG2 channel supplements the NWSTG channel. It contains all Model Grid data.

4. **NOAAPORT_OPT Channel.** This channel's data stream includes GOES DCP data, GMS/GOES-West/GOES-East/METEOSAT-5/METOSAT-7 composites for visible infrared, and water vapor products (every 3 hours), and OCONUS grids.

Each channel requires its own demodulator.

Under normal conditions, both NWSTG and NMC2 products and GOES satellite images are obtained from the SBN on CPSBN1. The CP ingest is controlled by the cp1f heartbeat package.
A more detailed discussion of the basic data flow follows in section 4-4. In the case of failover, initial download of these products shifts to CPSBN2; refer to Chapter 18, Failover Management Procedures, for details.

The vast majority of products on the SBN can be used without restriction. However, there are three exceptions: binary lightning; ACARS (Aircraft Communications Addressing and Reporting System); and one-degree ECMWF (European Centre for Medium-Range Weather Forecasts) products. Use of these three products is restricted to official duties, which include data utilization for the purposes of analysis/warning/forecast product preparation, research projects, and training. In general, these three products may not be redistributed to parties outside of NWS, especially on a routine or ongoing basis.

Concerning the one-degree resolution ECMWF grids, the following should also be noted:

- First, redistribution of one-degree (high resolution) ECMWF grids from AWIPS is prohibited. This limitation is formalized in a 2007 NOAA/ECMWF inter-agency agreement. The one-degree (i.e., "Hi Res") ECMWF grids in AWIPS are intended for use only in the conduct of official duties within NWS offices and at designated AWIPS test and support sites (such as AWIPS support contractor test beds and the OAR/Global Systems Division AWIPS node). ECMWF products – and products that are directly and exclusively derived from them – may not be accessed from AWIPS for provision to third parties.

- Second, the ECMWF grids should be considered "data of opportunity." By that it is meant that the availability and timeliness of the ECMWF grids could, at times, be
somewhat inconsistent or inferior (i.e., relative to other scheduled NOAA operational products). The potential for somewhat-degraded availability and timeliness is related to the external nature of the data source (i.e., the European Center) and the characteristics of some of the intermediate product dissemination legs.

4.1.2 Local Data Grids

Several gridded data sets are produced on the local AWIPS system rather than being received across the AWIPS SBN:

- GFE: Graphical Forecast Editor
- MPE: Multisensor Precipitation Estimator
- HPE: High-resolution Precipitation Estimator.

4.2 WSR-88D/Terminal Doppler Weather Radar (TDWR) Network

The WSR-88D and Terminal Doppler Weather Radio (TDWR) radar networks plus a few additional Federal Aviation Administration (FAA) radar sites provide additional data sources for AWIPS. Most sites receive their radar data via TCP/IP; the others have dedicated modems. The hardware components involved in radar acquisitions include the Open Radar Product Generator (ORPG), the Supplemental Product Generator (SPG), the radar modems, and the VIR switches that connect the modems to the Synchronous Communication Processors (CPSYNC).

For sites that receive radar data via TCIP/IP, the data enters AWIPS from the ORPG and the SPG via the AWIPS LAN.

For sites that acquire radar data via dedicated modems, CPSYNC1 handles the radar data stream, while CPSYNC2 sits idle. The VIR switches allow the radar data stream to be switched from one CP to the other in the event of a hardware failure. Refer to Chapter 18, Failover Management Procedures, for detailed information.

Note: The AWIPS 2 RadarServer currently only supports "dialing" in the sense that it can make Class 2 connections over TCP/IP (i.e., the AWIPS WAN). Actual X.25-over-modem dial-up is not supported.

4.3 Local Data Processes

AWIPS II software is broadly separated into two components: the Common AWIPS Visualization Environment (CAVE) and the Environmental Data EXchange (EDEX).

- CAVE provides an integrated set of tools for data visualization and manipulation.
- EDEX supports data acquisition, decoding, storage, and preprocessing. Both CAVE and EDEX utilize a plug-in architecture to support flexible deployments. All AWIPS II components are Eclipse Rich Client Platform (RCP) plug-ins; thus all AWIPS II components utilize a standard organizational structure. This enables code sharing between CAVE and EDEX.
Most data processing in AWIPS II requires both an EDEX component for decoding, processing, and storing the data and a CAVE component for retrieving, rendering, and manipulating the data. As a result, processing for each data type is generally separated into three parts: 1) a data decoder bundled with EDEX; 2) a rendering component bundled with CAVE; and 3) a common library bundled with both CAVE and EDEX. This three-part approach to data processing is illustrated in Exhibit 4.3-1.

![Exhibit 4.3-1. Code Sharing in AWIPS II](image_url)

As a consequence of the architectures chosen, a breakdown of AWIPS II into traditional Computer Software Configuration Items (CSCI) is problematical. A more productive approach is to look at functional groupings of software. Table 4.3-1 contains the mapping of the AWIPS II software into functional groups. This table illustrates how CAVE and EDEX fit into the general Software Functional Group.

### Table 4.3-1. AWIPS II Software Functional Groups

<table>
<thead>
<tr>
<th>Functional Group</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Installer Projects</td>
<td>Provide support for build component distribution installers for AWIPS II.</td>
</tr>
<tr>
<td>Software Build Projects</td>
<td>Provide support for building major software systems.</td>
</tr>
<tr>
<td>Static Data Projects</td>
<td>Include static data files, base configuration files, and standard Python scripts.</td>
</tr>
<tr>
<td>EDEX Infrastructure Projects</td>
<td>Provide basic Service Oriented Architecture (SOA) infrastructure for EDEX. Also provide some shared (library) projects.</td>
</tr>
<tr>
<td>EDEX Data Decoder Projects</td>
<td>Provide EDEX data decoding functionality.</td>
</tr>
<tr>
<td>EDEX/AWIPS Interface Projects</td>
<td>Provide runtime interfaces between EDEX and AWIPS software.</td>
</tr>
<tr>
<td>EDEX/CAVE Common Projects</td>
<td>Provide a library of functionality and interfaces common to both CAVE and EDEX.</td>
</tr>
<tr>
<td>CAVE Core Projects</td>
<td>Provide basic CAVE infrastructure.</td>
</tr>
<tr>
<td>CAVE Plug-in Projects</td>
<td>Provide CAVE data rendering functionality.</td>
</tr>
<tr>
<td>FOSS Projects</td>
<td>Provide standard versions of FOSS software used to support AWIPS II.</td>
</tr>
</tbody>
</table>
4.4 Data Routing Overview

The LDM data flow, illustrated in Exhibit 4.4-1, is a 12-step process. Descriptions of each step follow.
Chapter 4 Data Flow Overview

1. Data is obtained by the LDM noaaportingester from the SBN.
   a. The LDM noaaportingester forwards the product to the LDM Writer instance.
   b. The LDM Writer instance writes the product to the Data Archive. [Data Archive (/data_store) directory is mounted on the hosts off the NAS [NAS2].
   c. The LDM writer instance uses the EDEX Bridge to post a message containing product information to the QPID Message Broker. The product information is the WMO header and the path to the product file.

2. The EDEX Distribution endpoint obtains the WMO (World Meteorological Organization) header for the product file identified by the message. The WMO header is matched against data distribution mappings contained in the data distribution directory to determine product routing. (The WMO header is usually included in the message; if it is not, the Distribution service uses the product file name from the message to determine routing.)

   Note: The files in the data distribution directory are XML (eXtensible Markup Language) files that contain regular expressions which match WMO headers with the appropriate data decoder plug-in.

   a. At EDEX startup, the EDEX Distribution endpoint reads the Data Distribution files from the Data Distribution directory in the Localization Store. It uses the contents of these files to generate the routing table used to determine data routing. The Data Distribution directory is located at /awips2/edex/data/utility/edex_static/base/distribution.
   b. The EDEX Distribution endpoint monitors the Data Distribution directory; when a file in that directory is modified, it rereads that file and rebuilds its routing table.

3. The EDEX Distribution Service determines the Data Decoder Plug-In registered to handle the data in the product file.

4. The EDEX Distribution Service forwards the message to the appropriate Data Decoder Plug-In via the QPID Message Broker.

5. The EDEX Data Decoder Service on the EDEX (DX) cluster pulls the message from the QPID Message broker. The Data Decoder Service reads the file pointed to by the message, decodes the data, and determines the appropriate metadata.

6. The decoded data and metadata are sent to the EDEX Persist endpoint.

7. The EDEX Persist endpoint sends the data to PyPIES and PyPIES writes the data to the EDEX data store in HDF5 format.

8. The EDEX Persist endpoint sends the metadata obtained to the EDEX Index endpoint.

9. The EDEX Index endpoint sends the metadata to the PostgreSQL DBMS.
   a. The PostgreSQL DBMS writes the metadata to the database.
10. The index endpoint does not merely get the dataURI; it also persists the data object to the database. Persistence to the database is executed via a plug-in defined data access object. If the data access object does not override the persistence behavior, then a default persistence strategy is used. The default strategy for persisting data to the database is to first check to see if the data received already exists. If it already exists, then persistence to the database is not attempted for that data. If the data is unique, then it is persisted to the database.

11. (Client notification) The Data URI is forwarded to the EDEX URI Aggregator.
   a. Every 5 seconds the aggregated Data URIs are sent to the JMS Alert Topic.
   b. CAVE monitors the JMS Alert Topic to obtain a list of available data. CAVE determines if it should render the available data.
   c. CAVE retrieves metadata from the EDEX server via a product query sent to the EDEX thrift endpoint.
   d. CAVE retrieves product data from the EDEX server by accessing the EDEX data store’s HDF5 files and CAVE renders the data, updating its display as appropriate.

12. (Subscription) The Data URI is forwarded to the EDEX Subscription endpoint. (This does not apply to Satellite Data Flow.)
   a. The EDEX Subscription endpoint checks the active subscription list to determine if a subscription has been registered for the product.
      1) If no subscription has been registered for the product, no further action is taken.
   b. The EDEX Subscription endpoint obtains the script information from the database.
   c. The EDEX Subscription endpoint obtains a micro-engine to execute the script. The micro-engine executes the subscription script for the product.

It is important to note that some, but not all, Radar Products follow a slightly different ingest path. This path is shown in Exhibit 4.4-2. The main differences are as follows:

- The Radar Server on the Database (DX) cluster obtains a Radar product from the ORPG.
- The Radar Server writes the product to the Data Archive.
- The Radar Server posts a message containing product information to the QPID Message Broker.

**NOTE:** Refer to Chapter 7, Ingest of Radar Data, for a more detailed explanation of the Radar data flow.
Exhibit 4.4-2. Radar Data Flow
4.5 **Configuration Files**

Data flow in AWIPS II is controlled by two sets of configuration files: 1) LDM configuration files; and 2) EDEX Data Distribution files. All of these files are normally set at system installation; the files are editable, but extreme care must be taken during editing because malformed files can cause system failure.

The LDM configuration files control the data that is delivered from the SBN to the EDEX Ingest process; the EDEX Data Distribution files control the internal flow of data to the appropriate data decoders.

4.5.1 **LDM Configuration**

LDM server runs only on the SBN CP. It ingests everything from the SBN.

LDM behavior is controlled by the pqact.conf file, and the patterns are matched against `pqact.conf`. LDM writes the product to `data_store` and posts a QPID message.

- `pqact.conf` controls how data sent to an instance of LDM is handled. This includes the data types to be handled and the location of the raw data archive.

**NOTE:** Modifying `pqact.conf` will have a severe impact on the data flow.

4.5.2 **EDEX Data Distribution Files**

EDEX Data Distribution files are XML files that are bundled with the EDEX Data Decoders. When EDEX is installed on DX3 and DX4, the Data Distribution files are installed into `/awips2/edex/data/utility/edex_static/base/distribution`, referred to as the “base distribution directory.” In the standard AWIPS II installation, the base distribution directory is physically located on the Network Attached Storage (NAS); each EDEX installation accesses it via a Network File System (NFS) mount in `/awips2/edex/data`. As a result, changes made to a data distribution file on one of the EDEX servers (DX3 or DX4) will be available on the other EDEX server.

When EDEX is first started, the Distribution Service reads the Data Distribution files and generates a distribution table matching regular expressions with Data Decoders. The regular expressions are defined in the Data Distribution files and are matched against the WMO header of a data file. If the match is met, the file is routed to the data decoder.

**NOTE:** There is one XML distribution file per data decoder. Using regular expressions, each XML file specifies which WMO headers are appropriate for that decoder.

Each Data Decoder has a Data Distribution file in the base distribution directory. The Data Distribution files are named `<data type>.xml`. For example, the Data Distribution file for satellite data is named `satellite.xml`. Each Data Distribution file is a properly formatted XML document with the following format:

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<requestPatterns xmlns:ns2="group">
```
The “pattern” in each regex tag is a regular expression that is used to identify a data file. Normally, the regular expression matches the WMO header of the data in the file; it can also match the name of the file containing the data.

For example, satellite.xml contains a single regular expression: TI….. …. (TI, 4 dots, space, 4 dots). Because a dot in a regular expression matches any character, this pattern matches a WMO header starting with the characters TI.

For example: TIGE04 KNES

The Distribution Service monitors the base distribution directory. When one of the data distribution files changes, the Distribution Service reads the file and updates its routing table. When the re-read occurs, a message similar to the following message will appear in the EDEX Ingest log file.

INFO <date> <time> [<thread>] DISTRIBUTION: Change to distribution file detected. satellite.xml has been modified.

When distribution patterns are reloaded on the D2D, an entry is written to the EDEX Ingest process logs in each of the EDEX servers.

Once the file has been reloaded, the modified patterns are used for file routing.

To understand why and when to edit these files, consider the following: The LDM uses the expressions defined in pqact.conf to determine which data to write to files in the data archive. The LDM generates a message that includes the WMO header information and forwards it to the EDEX Distribution Service via QPID. When the EDEX Distribution Service retrieves the message from QPID, it uses the Data Distribution files in the data distribution directory to determine two things: 1) the data files to process; and 2) the data decoder to process the files.

As a consequence of this action:

- Any time a new entry is placed in the pqact.conf file on the LDM server on the a2pc1apps, a corresponding entry must be added to the appropriate Data Distribution file in the data distribution directory.
• Any time an entry is removed from the pqact.conf file, the corresponding entry should be removed from the appropriate Data Distribution file in the data distribution directory.

NOTE: When making the changes, remember that the files being edited need to be correctly formatted and make sure that there are no typos. Both LDM and EDEX will fail to start if these files are not properly formatted.

The structure of the message sent to QPID external.dropbox via the EDEX bridge is as follows.

• The WMO header of the data contained in the file, if it is available
• The location of the data file in the data archive
• The start time for the computation of ingest latency
• The destination queue for the message.

The difference between using the manual endpoint and “external.dropbox” is who produces the message. “external.dropbox” is a message queue within QPID. In order to access “external.dropbox” directly, you need to create an application that can generate the appropriate message. If you do so, you must copy the file into the data archive (or another accessible location – it must be accessible to both DX3 and DX4.) On the other hand, when you use the manual endpoint, all you do is copy the data file into /awips2/edex/data/manual on DX3/4 and EDEX does the rest.

NOTE: You should only copy the file you want to be ingested into the manual inbox. Do not try to interface directly with QPID.

4.5.2.1 Potential Problems

Problem 1: What if the manual endpoint does not clean up after itself well? That is, it copies its data to /tmp/sbn/manual; this directory is not purged so it could fill up and cause a system crash.

Discussion: This is only a problem if the installation is not performed correctly. EDEX should not be moving files from /awips2/edex/data/manual to /tmp/sbn/manual; it should be moving manually ingested files from /awips2/edex/data/manual to /data_store/manual. The LDM scour utility should be managing this directory. (Note that this is more of a software issue than a documentation issue. If the reviewer really believes this is a problem he or she should write a ticket (Trac or TTR) to address the issue.)

Problem 2: Obviously, using the manual endpoint for data ingest is easier (at the current time) than interfacing with QPID, although a script could easily be written to facilitate using QPID to ingest data (as done in WTDB). What if, because of its ease of use, the manual endpoint becomes the choice of use for local app developers (and in fact for some regular AWIPS developers for HPE/MPE, LAPS, MSAS, etc.)? The problem is that a regular process that uses the manual endpoint for data ingest compromises the integrity of the data archive.
Discussion: If EDEX is configured correctly when it is installed, this is not an issue.

4.6 Monitoring Data Ingest

All data ingest operations are logged in the EDEX logs on the EDEX (DX3/4) server. In a standard installation, the EDEX logs are located in /awips2/edex/logs. This directory contains several log files; the files starting with “edex–” are the EDEX logs. The files starting with “start–” are the start-up logs. The EDEX logs roll over daily and are date stamped. The naming convention for EDEX logs is edex-\{instance name\}-yyyymmdd.log.

```
edex-ingest-20130212.log
edex-ingestDat-20130212.log
edex-ingest-gen_areal_ffg-20130212.log
edex-ingestGrib-20130212.log
edex-ingest-purge-20130212.log
edex-ingest-radar-20130212.log
edex-ingest-satellite-20130212.log
edex-ingest-shef-20130213.log
edex-ingest-shef-performance-20130212.log
edex-ingest-smartInit-20130212.log
edex-ingest-text-20130212.log
edex-ingest-unrecognized-files-20130212.log
edex-request-20130212.log
edex-request-productSrvRequest-20130212.log
edex-request-thriftSrv-20130212.log
edex-ingest-archive-20130212.log
edex-ingest-activeTableChange-20130212.log
edex-ingestGrib-activeTableChange-20130212.log
edex-ingestGrib-performance-20130212.log
edex-ingest-performance-20130212.log
edex-request-performance-20130212.log
edex-ingest-gen_areal_qpe-20130212.log
start-edex-ingest-20130212.log
start-edex-ingestDat-20130212.log
start-edex-ingestGrib-20130212.log
start-edex-request-20130212.log
edex-request-hibernate-20130212.log
edex-ingest-hibernate-20130212.log
edex-ingestGrib-hibernate-20130212.log
```

4.6.1 EDEX Log Types

The logs break down into seven different types:

1. **EDEX process logs.** Each EDEX process writes to a separate log. The log file names are edex-ingest-yyyymmdd.log, edex-ingestDat-yyyymmdd.log, edex-ingestGrib-
yyyymmdd.log, and edex-request-yyyymmdd.log. These logs contain startup messages and most other messages, including errors, for the EDEX processes. The specific logs that are actually present depend on how EDEX is started.

- edex-ingestGrib-<yyyymmdd>.log
  edex-ingestGrib handles the Grid Products.

- edex-ingest-satellite-<yyyymmdd>.log
  edex-ingest-satellite handles the Satellite products.

- edex-ingest-radar-<yyyymmdd>.log
  edex-ingest-radar handles the Radar products.

- edex-ingest-text-<yyyymmdd>.log
  edex-ingest-text handles other products (metar, sfccobs, etc.).

- edex-ingest-<yyyymmdd>.log
  (edex-ingest handles other products (lightning, MOS, etc.).

  edex-ingest-activeTableChange-<yyyymmdd>.log (activeTableChange)
  edex-ingest-gen_areal_ffg-<yyyymmdd>.log (Flash Flood Guidance)
  edex-ingest-gen_areal_qpe-<yyyymmdd>.log (QPE)

- edex-ingestDAT-<yyyymmdd>.log (edex-ingestDAT handles backend data services for the Decision Aid Tools, such as FFMP and SCAN)

**Note:** Requests to the thrift service no longer go to the edex-request log; they go to the log edex-request-thriftSrv. This consists of most of the requests from CAVE and a few requests from python command line utilities, such as some GFE items. The productSrv (aka uengine) service is unchanged and therefore will be still logging to the old place (a few spots in CAVE and localapps still use productSrv). The thriftSrv log contains a log entry for every request received. In the past these were only logged if it took longer than 200 ms; now, every request is logged. Requests from CAVE include the workstation id, which indicates the user (<username>), network address (<lx3-tbw3>), process name (<CAVE>), process id (<17176>), etc., and a unique identifier string. On the CAVE side, the console log has an entry for each request that takes one second or more to complete. This log entry also contains the unique identifier, so when CAVE reports a slow request, it can be matched to an edex entry on dx3 or dx4 by finding the matching unique id in their logs.

2. **Data type-specific logs.** Logs with names similar to edex-ingest-<data-type>-yyyymmdd.log contain messages relating to a specific data type. These include both status messages and error messages. For a data type having its own log file, this is the log to examine to find and diagnose problems. The ingest-shef log file describes the shef decoder, although it is fed by the NWSTG feed.
3. **edex-ingest-unrecognized-files log.** This log is probably among the most important of them all. The edex-ingest-unrecognized-files-yyyymmd.log can be checked to determine if data ingest is failing due to a problem with distribution files. Unfortunately, the unrecognized file log only identifies the file that was unrecognized; it does not provide a path to the file.

4. **EDEX startup logs.** Files with names starting with “start” capture messages generated by the edex_camel service script. They are start-edex-ingest, start-edex-ingestGrib, start-edex-ingestDat, and start-edex-request logs. In normal operation, these log files will be zero size.

5. ***-activeTableChange-*log files.** These log files are written to when the active table is updated. This will normally only happen in ingest and request JVMs. These logs were added to see what changes are (or are not) being made to the active table.

6. ***-performance-*log files.** These files are used to log performance-related information to allow the users to determine how long operations are taking in order to find performance issues or to verify if they are meeting performance metrics.

7. ***-gen_areal_qpe-* logs.** These logs are written to during the generation of qpe mosaics. When the qpe grids from the RFCs are combined into a mosaic image, the process is logged in the gen_areal_qpe logs.

8. **hibernate files.** The hibernate log file will now be produced for each EDEX instance. Any errors produced by org.hibernate will be logged to this file. This log file should primarily be referenced whenever there are persistence errors in the EDEX instance logs.

   The log file is a plain ASCII text file and can be viewed using any text viewing utility such as `more`, `less`, `view`, or `tail`.

To get a general view of processing, open a terminal session to the server and enter the following commands:

**On DX3/DX4:**

```
TYPE:  cd /awips2/edex/logs
TYPE:  date +%F
       2013-02-12
TYPE:  ls edex-*20130212*
edex-ingest-20130212.log
edex-ingestDat-20130212.log
edex-ingest-gen_areal_qpe-20130212.log
edex-ingestGrib-20130212.log
edex-ingest-purge-20130212.log
edex-ingest-radar-20130212.log
edex-ingest-satellite-20130212.log
edex-ingest-shef-20130213.log
```
For ingestGrib,
    TYPE: tail -f edex-ingestGrib-20130212.log

For Satellite,
    TYPE: tail -f edex-ingest-satellite-20130212.log

For Radar,
    TYPE: tail -f edex-ingest-radar-20130212.log

For Archive,
    TYPE: tail -f edex-ingest-archive-20130212.log

For GFEPerformance,
    TYPE: tail -f edex-ingest-GFEPerformance-20130212.log

For hibernate,
    TYPE: tail -f edex-ingest-hibernate-20130212.log

For activeTableChange,
    TYPE: tail -f edex-ingest-activeTableChange-20130212.log

For areal_qpe,
    TYPE: tail -f edex-ingest-gen_areal_qpe-20130212.log
For other products (refer to Chapter 6),

**TYPE:** tail -f edex-ingest-20130212.log

**TYPE:** tail -f edex-ingest-text-20130212.log

### 4.6.2 EDEX Log Format

To monitor a specific ingest stream, a little understanding of the logging strategy used by EDEX is required. After ingest of each data file is complete, the ingest endpoint prints a single line to the EDEX system log. The format of the log entry is

```
INFO <<date-time>> [thread] Ingest: <<type>>: <<file name>> <<statistics>>
```

Where,

- **<<date-time>>** is the date-time stamp (format yyyy-mm-dd hh:mm:ss,ttt)
- [thread] identifies the Camel thread that ran the decoder.
- **<<type>>** identifies the data type, e.g., grib, radar.
- **<<file name>>** is the path to the file that was processed, including the unique product sequence number.
- **<<statistics>>** lists the ingest statistics for the file. There are two statistics provided: *process time* and *latency*. (See note at the end of this section for more information on latency.)

A typical log entry for a grib product in edex-ingestGrib-YYYYMMDD.log is

```
INFO  2013-02-12 00:00:18,153 [gribThreadPool-1] Ingest: EDEX: Ingest – grib2::
/data_store/grib2/20130211/23/RUC2/GRID130/2300Z_F010_HGHT-LHDRK90_KWBG_112300_38585284.grib2.2013021200 processed in:
0.2800 (sec) Latency: 0.2860 (sec)
```

The information from this entry is formatted as:

- **Date:** 2013-02-12
- **Time:** 00:00:18
- **Thread:** [gribThreadPool-1]
- **Type:** grib2
- **Path to File:**
  
  /data_store/grib2/20130211/23/RUC2/GRID130/2300Z_F010_HGHT-LHDRK90_KWBG_112300_38585284.grib2.2013021200

- **Sequence Number:** 38585284
- **Process Time:** 0.2800 (sec)
- **Latency:** 0.2860 (sec)

A typical log entry for Satellite in edex-ingest-satellite-YYMMDD.log is
INFO 2013-02-12 00:12:18,095 [satelliteThreadPool-1] Ingest: EDEX: Ingest - satellite:: /data_store/sat/20130212/00/GOES-15/0000Z_VIS_2km_AK-_REGIONAL_TIGA1_KNES_14685231.satz.2013021200 processed in: 0.4890 (sec) Latency: 0.5510 (sec)

A typical log entry for Radar in edex-ingest-radar-YYMMDD.log is


A typical log entry for products like metar, MOS in edex-ingest-text-YYMMDD.log is

INFO 2013-02-12 00:00:03,775 [textThreadPool-2] Ingest: EDEX: Ingest - text:: /data_store/shef/20130211/23/SRUS55_KMSO_112359_61698592.2013021200 processed in: 0.0050 (sec) Latency: 0.0110 (sec)

INFO 2013-05-20 17:35:29,380 [textThreadPool-2] Ingest: EDEX: Ingest - text:: /data_store/forecast/20130520/17/FCNO35_ENMI_201700_128724123.2013052017 processed in: 0.0050 (sec) Latency: 0.0310 (sec)

INFO 2013-02-12 00:00:02,556 [textThreadPool-1] Ingest: EDEX: Ingest - text:: /data_store/metar/20130715/23/SAUS44_KMOB_152359_17312349.2013071600 processed in: 0.0080 (sec) Latency: 0.0190 (sec)

A typical log entry for products in activeTableChange is

INFO 2013-10-29 23:53:33,177 [genericThreadPool-91] ActiveTableChange: Updated OPERATIONAL Active Table: decoded

Vtec: /O.NEW.KEAX.FG.Y.0009.131030T0000Z-131030T1400Z/

Hdln: ???

Start: Wed Oct 30 00:00:00 2013 1383091200L Action: NEW

Pil: NPW
A typical log entry for products like metar, MOS in edex-ingest-text-YYMMDD.log is

```
********************
siteId = host
mosaicDir = /awips2/edex/data/share/hydroapps/whfs/local/data/grid/misc
lookbackLimit = 60
minArea = 0.8
RFC Names
  MARFC
  SERFC
Durations
  1
  3
  6
Last run time = 1382659546000
HSA = AKQ
DEBUG 2013-10-25 00:14:14,427 [genericThreadPool-2] GenArealFFG: Only run every 12 minutes. 8.0 minutes since last run.
DEBUG 2013-10-25 00:16:25,863 [genericThreadPool-2]
```
Chapter 4  Data Flow Overview

A typical log entry for all other products (e.g.: binlightning ) in edex-ingest-YYMMDD.log is

INFO  2013-05-20 00:00:00,067 [genericThreadPool-21]

This information is logged into the EDEX process logs for every file that is ingested. The general format of the message is

INFO  <date> <time> [thread] Ingest: <type>:: <file>
processed in: <time> Latency: <time>.

(Note that there are two spaces between INFO and the start of the date stamp.) This pattern can be used to monitor the EDEX Ingest process logs for ingest latency, using the pattern to filter out undesirable messages.

NOTE: Latency is the elapsed time between when the AWIPS II system first contacts a data file and when the data has been ingested, decoded, and is ready for retrieval by either CAVE or another client application. The “first contact” depends on the data type. For most data types, latency measurement starts when the product has been downloaded by the LDM Writer on the CP1f. This latency start time is part of the message generated by the LDM’s EDEX Bridge and sent to QPID (on the CP1/2 cluster) for processing by EDEX. For Radar products obtained from the ORPG via the AWIPS II Radar Server, latency measurement starts when the product is downloaded by the Radar Server on the DX1/2 cluster. In the case of ORPG products, the latency start time is part of the message sent by the Radar Server to QPID. Note that this latency measure is strictly internal for the AWIPS II data ingest subsystem.

4.7  General Monitoring of the EDEX System

To monitor the EDEX logs for other specific events,

- Open a terminal session to the dx3 or dx4 EDEX server.
- Change directory to the edex log directory /awips2/edex/logs.
- Tail the current log file and pipe the output into grep with the appropriate search string. To determine the search string, first examine the appropriate EDEX log to find the event you want to monitor, and then determine the search pattern.

Here are some specific events captured from the EDEX log.

```
edex-ingest.log
LVR VA0001 2247S 16522E 232059 F360 MS470 271/075 TB0 S031
333 F360 VG000=
=================>toPointData
```
4.7.1 EDEX Log Event Levels

Note that each “event” starts with either FATAL, ERROR, WARN, INFO, or DEBUG and continues to the start of the next “event.”

- The ERROR level designates error events that might still allow the application to continue running.
- The FATAL level designates very severe error events that will presumably lead the application to abort.
- The INFO level designates informational messages that highlight the progress of the application at coarse-grained level.
- The WARN level designates potentially harmful situations.
- The DEBUG Level designates fine-grained informational events that are most useful to debug an application.

A “significant event” is anything a developer has decided should be logged. The AWIPS II system is built on top of a number of Open Source projects (Camel, Hibernate, etc.), which is referred to as platform code. Some of the logging is from the platform code. An example of logging generated by the platform code is a database error. Database errors are usually generated by the platform code. An example of a database error (somewhat truncated) is

```
ERROR 2013-02-12 18:15:59,933 [genericThreadPool-5]
JDBCExceptionReporter: ERROR: update or delete on table "taf" violates foreign key constraint "fk5e120f1d80f77fb" on table "taf_change_groups"
```
Caused by: org.postgresql.util.PSQLException: ERROR: update or delete on table "taf" violates foreign key constraint "fk5e120f1d80f77fb" on table "taf_change_groups"

Other ERROR messages are generated by AWIPS II-specific code. An example of an error generated by AWIPS II code is

```
ERROR 2013-02-12 18:16:08,426 [genericThreadPool-72]
TextDecoder: Message failed parsing for WMO message header: [SAAP32 FAPR 251800 RRA], products stored [0/1]
```

As a general rule, all “ERROR” messages and most “WARN” messages are important. (Unfortunately, some messages logged as ERROR should be WARN.) INFO messages can also be important. The only person who can determine if a message is important is the person analyzing the log. For example, every file that is ingested is logged with an INFO message. The message will be similar to

```
INFO 2013-02-12 00:00:02,962 [genericThreadPool-65]
Ingest: EDEX: Ingest - sfcobs:
/data_store/maritime/20130211/23/SXUS21_KWNB_112300_6169854
4.2013021200 processed in: 0.1960 (sec) Latency: 0.2050 (sec)
```

This message can become important in either of these scenarios:

- If a user is attempting to trace dataflow through the system, this message verifies that the file SXUS21_KWNB_112300_61698544.2013021200 has been ingested.
- If a user is interested in system performance, the actual latency time reported becomes important.

In fact, what is not logged can also be “important. The best example here is trying to trace data through the system. If the LDM reports that a file was written to the data archive but there is no corresponding “processed” log entry in the EDEX logs, this is significant/important information and it needs to be checked out.

### 4.8 Monitoring the LDM Data Flow

The data flow from the LDM may be monitored using the `ldmadmin` tool. The `watch` option on this tool provides a running list of the LDM’s activity. This will show the data passing through the LDM. To use this tool, first log onto the LDM server (CPSBN) as the ldm user, and then enter:

```
ldmadmin watch
(Type ^D when finished)
```

```
Jan  6 15:48:31 pqutil INFO: 19843 20140106154830.803 NEXRAD3
32776549 SDUS21 KBOX 061546 /pN1QBOX !nids/
Jan  6 15:48:31 pqutil INFO: 4411 20140106154830.804 NEXRAD3
32776550 SDUS84 KEPZ 061541 /pN2HHDX !nids/
```
Use the -f option and specify a FEED TYPE to see only products from that LDM feed. Examples follow.

**ldmadmin watch –f NNEXRAD**

**(Type ^D when finished)**

```
Feb 13 14:45:30 pqutil INFO: 16586 20130213144529.997 NEXRAD3
65743231  SDUS52 KCHS 131444 /pN0QCLX !nids/
Feb 13 14:45:30 pqutil INFO: 165 20130213144529.997 NEXRAD3
65743232  SDUS33 KSGF 131439 /pNMDSGF !nids/
Feb 13 14:45:30 pqutil INFO: 2298 20130213144529.997 NEXRAD3
65743233  SDUS53 KLBF 131440 /pNTPLNX
Feb 13 14:45:30 pqutil INFO: 1296 20130213144530.011 NEXRAD3
65743234  SDUS53 KSGF 131439 /pNCRSGF
Feb 13 14:45:30 pqutil INFO: 11509 20130213144530.082 NEXRAD3
65743242  SDUS53 KABR 131444 /pN0RABR
Feb 13 14:45:30 pqutil INFO: 8287 20130213144530.099 NEXRAD3
65743243  SDUS73 KABR 131444 /pN0ZABR
```

**ldmadmin watch –f GRID**

**(Type ^D when finished)**

```
Jan 06 15:51:34 pqutil INFO: 83790 20140106155134.021 NGRID
23276444  LVOF97 KWBC 061200 !grib2/ncep/GFS/#254/201401061200F030/VREL/0 K*m**2*kg-1*s-1 POTV
Jan 06 15:51:34 pqutil INFO: 78746 20140106155134.029 NGRID
23276445  LVOF97 KWBC 061200 !grib2/ncep/GFS/#254/201401061200F030/VREL/1 K*m**2*kg-1*s-1 POTV
Jan 06 15:51:34 pqutil INFO: 61034 20140106155134.037 NGRID
23276446  LVOF96 KWBC 061200 !grib2/ncep/GFS/#254/201401061200F030/VREL/0 - MWSL
```
Jan 06 15:51:34 pqutil INFO: 1264 20140106155134.037 NGRID
23276447 MHGL98 KWBT 061200 !grib2/ncep/GFS/#255/201401061200F027/HGHT/0 - NONE

Jan 06 15:51:34 pqutil INFO: 7227 20140106155134.037 NGRID
23276448 ELCM88 KWBJ 061200 !grib2/ncep/GMGWM/#255/201401061200F051/WVHGT/0 - NONE

Jan 06 15:51:34 pqutil INFO: 146981 20140106155134.052 NGRID
23276449 LPOF97 KWBC 061200 !grib2/ncep/GFS/#254/201401061200F030/PRES/0 K*m**2*kg-1*s-1 POTV

Jan 06 15:51:34 pqutil INFO: 78358 20140106155134.061 NGRID
23276450 LROF98 KWBC 061200 !grib2/ncep/GFS/#254/201401061200F030/UREL/2 K*m**2*kg-1*s-1 POTV

Jan 06 15:51:35 pqutil INFO: 24587 20140106155135.285 NGRID
23276451 EYDM88 KWBJ 061200 !grib2/ncep/GMGWM/#255/201401061200F051/SWPER/1 - OSEQ

Jan 06 15:51:35 pqutil INFO: 76360 20140106155135.018 NGRID
23276452 LROF98 KWBC 061200 !grib2/ncep/GFS/#254/201401061200F030/UREL/2 K*m**2*kg-1*s-1 POTV

ldmadmin watch –f HDS
(Type ^D when finished)

Jul 17 15:33:09 pqutil INFO: 14214 20130717153308.853 NEXRAD3
137781 SDUS84 KHGX 171531 /pN2HHGX !nids/

Jul 17 15:33:09 pqutil INFO: 9145 20130717153308.856 NEXRAD3
137782 SDUS84 KHGX 171531 /pOH1AHGX !nids/

Jul 17 15:33:09 pqutil INFO: 11937 20130717153308.859 NEXRAD3
137783 SDUS84 KHGX 171531 /pDODHGX !nids/

Jul 17 15:33:09 pqutil INFO: 37327 20130717153308.975 NEXRAD3
137784 SDUS84 KHGX 171531 /pDAAHGX !nids/

Jul 17 15:33:09 pqutil INFO: 45002 20130717153308.988 NEXRAD3
137785 SDUS84 KHGX 171531 /pN2CHGX !nids/

Jul 17 15:33:09 pqutil INFO: 21377 20130717153308.995 NEXRAD3
137786 SDUS84 KHGX 171531 /pN2CHGX !nids/

Jul 17 15:33:09 pqutil INFO: 55640 20130717153309.011 NEXRAD3
137787 SDUS84 KOUN 171531 /pN1XVNX !nids/

Jul 17 15:33:09 pqutil INFO: 15291 20130717153309.015 NEXRAD3
137788 SDUS85 KBYZ 171530 /pN3HBLX !nids/
Jul 17 15:33:09 pqutil INFO: 345 20130717153309.015 NEXRAD3
137789 SDUS81 KOKX 171526 /pDSOKX !nids/

Jul 17 15:33:09 pqutil INFO: 40361 20130717153309.027 NEXRAD3
137790 SDUS84 KOUN 171531 /pN2CFDR !nids/

ldmadmin watch –f NIMAGE
(Type ^D when finished)

Nov 06 13:28:03 pqutil INFO: 47600 20121106132802.869 NIMAGE
6851 satz/ch2/GOES-15/SOUND-11.03/20121106 1301/WEST-CONUS/10km/
TIGW48 KNES 061301

Nov 06 13:28:07 pqutil INFO: 33639 20121106132807.005 NIMAGE
6852 satz/ch2/GOES-15/SOUND-7.43/20121106 1301/WEST-CONUS/10km/
TIGW50 KNES 061301

Nov 06 13:28:09 pqutil INFO: 31610 20121106132809.217 NIMAGE
6853 satz/ch2/GOES-15/SOUND-7.02/20121106 1301/WEST-CONUS/10km/
TIGW51 KNES 061301

6854 satz/ch2/GOES-15/SOUND-6.51/20121106 1301/WEST-CONUS/10km/
TIGW52 KNES 061301

Nov 06 13:28:15 pqutil INFO: 31514 20121106132814.754 NIMAGE
6855 satz/ch2/GOES-15/SOUND-4.45/20121106 1301/WEST-CONUS/10km/
TIGW55 KNES 061301

ldmadmin watch –f IDS
(Type ^D when finished)

Jan 06 15:52:19 pqutil INFO: 315 20140106155219.378 IDS|DDPLUS
10998153 SXBO40 KWAL 061551

Jan 06 15:52:19 pqutil INFO: 76 20140106155219.378 IDS|DDPLUS
10998154 SRK520 KWAL 061551

Jan 06 15:52:19 pqutil INFO: 177 20140106155219.379 IDS|DDPLUS
10998155 SRMN70 KWAL 061551

Jan 06 15:52:19 pqutil INFO: 96 20140106155219.379 IDS|DDPLUS
10998156 SRWY20 KWAL 061551

Jan 06 15:52:19 pqutil INFO: 143 20140106155219.380 IDS|DDPLUS
10998157 SRNJ20 KWAL 061551

Jan 06 15:52:19 pqutil INFO: 144 20140106155219.380 IDS|DDPLUS
10998158 SRPA20 KWAL 061551

Jan 06 15:52:19 pqutil INFO: 186 20140106155219.380 IDS|DDPLUS
10998159 SROK20 KWAL 061551

Jan 06 15:52:19 pqutil INFO: 144 20140106155219.381 IDS|DDPLUS
10998160 SRPA30 KWAL 061551

Jan 06 15:52:19 pqutil INFO: 144 20140106155219.387 IDS|DDPLUS
10998161 SRNY20 KWAL 061551

Jan 06 15:52:19 pqutil INFO: 224 20140106155219.387 IDS|DDPLUS
10998162 SRAL75 KWAL 061551
Jan 06 15:52:19 pqutil INFO:  113 20140106155219.387 IDS|DDPLUS
  10998163  SRLA20 KWAL 061551
Jan 06 15:52:19 pqutil INFO:  129 20140106155219.388 IDS|DDPLUS
  10998164  SXCO50 KWAL 061551
Jan 06 15:52:19 pqutil INFO:  160 20140106155219.388 IDS|DDPLUS
  10998165  SROH30 KWAL 061551
Jan 06 15:52:19 pqutil INFO:  123 20140106155219.388 IDS|DDPLUS
  10998166  SXUT50 KWAL 061551
Jan 06 15:52:19 pqutil INFO:  130 20140106155219.389 IDS|DDPLUS
  10998167  SRWV30 KWAL 061551
Jan 06 15:52:19 pqutil INFO:  167 20140106155219.389 IDS|DDPLUS
  10998168  SXCA50 KWAL 061551
Jan 06 15:52:19 pqutil INFO:  436 20140106155219.389 IDS|DDPLUS
  10998169  SXCN40 KWAL 061551
Jan 06 15:52:19 pqutil INFO:  118 20140106155219.389 IDS|DDPLUS
  10998170  SRMO20 KWAL 061551
Jan 06 15:52:19 pqutil INFO:  136 20140106155219.389 IDS|DDPLUS
  10998171  SXID50 KWAL 061551
Jan 06 15:52:19 pqutil INFO:  155 20140106155219.389 IDS|DDPLUS
  10998172  SXXX03 KWAL 061551
Jan 06 15:52:19 pqutil INFO:  128 20140106155219.397 IDS|DDPLUS
  10998173  SRLA30 KWAL 061551
Jan 06 15:52:19 pqutil INFO:  134 20140106155219.397 IDS|DDPLUS
  10998174  SRSN40 KWAL 061551
Jan 06 15:52:19 pqutil INFO:  192 20140106155219.397 IDS|DDPLUS
  10998175  SRMD20 KWAL 061551
Jan 06 15:52:19 pqutil INFO:  144 20140106155219.398 IDS|DDPLUS
  10998176  SXCA50 KWAL 061551
Jan 06 15:52:19 pqutil INFO:  143 20140106155219.398 IDS|DDPLUS
  10998177  SRNQ20 KWAL 061551
Jan 06 15:52:19 pqutil INFO:  167 20140106155219.398 IDS|DDPLUS
  10998178  SXCH40 KWAL 061551
Jan 06 15:52:19 pqutil INFO:  191 20140106155219.399 IDS|DDPLUS
  10998179  SRNY20 KWAL 061551
Jan 06 15:52:20 pqutil INFO:  163 20140106155220.366 IDS|DDPLUS
  10998180  SXMX40 KWAL 061551

To see connections, open with ldm.

```
netstat -ta | grep ldm

[ldm@cpsbn2-box ~]$ netstat -ta | grep ldm
tcp        0      0 cpsbn1-box.er.a:unidata-ldm *:*  
            LISTEN
            getnameinfo failed
```
tcp 0 0 cpsbn1-box.er.a:unidata-ldm as2-box.er.awips.noaa:58330 ESTABLISHED

To monitor the product queue v.

pqmon

Aug 09 15:24:55 pqmon NOTE: Starting Up (15747)
Aug 09 15:24:55 pqmon NOTE: nprods nfree nempty nbytes maxprods maxfree minempty maxext age
Aug 09 15:24:55 pqmon NOTE: 75133 1 169006 999891712 81526 3 162613 109824 1652
Aug 09 15:24:55 pqmon NOTE: Exiting

[Note: The pqmon command is useful for looking at the state of the products queue, and whether or not the existing product queue size is sufficient for the amount of data coming in.]

To check for the running of the LDM

ps -U ldm -u ldm u

<table>
<thead>
<tr>
<th>USER</th>
<th>PID</th>
<th>%CPU</th>
<th>%MEM</th>
<th>VSZ</th>
<th>RSS</th>
<th>TTY</th>
<th>STAT</th>
<th>START</th>
<th>TIME</th>
<th>COMMAND</th>
</tr>
</thead>
<tbody>
<tr>
<td>ldm</td>
<td>11734</td>
<td>0.0</td>
<td>0.0</td>
<td>4548</td>
<td>952</td>
<td>?</td>
<td>Ss</td>
<td>2013</td>
<td>0:00</td>
<td>ldmd -I</td>
</tr>
<tr>
<td></td>
<td>165.92.29.180</td>
<td>-P 388 -M 256 -m 3600</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ldm</td>
<td>11736</td>
<td>0.4</td>
<td>15.8</td>
<td>1000952</td>
<td>987460</td>
<td>?</td>
<td>S</td>
<td>2013</td>
<td>113:22</td>
<td>pqact -e</td>
</tr>
<tr>
<td>ldm</td>
<td>11737</td>
<td>0.0</td>
<td>0.4</td>
<td>55224</td>
<td>24988</td>
<td>?</td>
<td>S</td>
<td>2013</td>
<td>9:08</td>
<td>edexBridge -vxl /usr/local/ldm/logs/edexBri</td>
</tr>
<tr>
<td>root</td>
<td>11738</td>
<td>0.5</td>
<td>19.2</td>
<td>1238064</td>
<td>1196952</td>
<td>?</td>
<td>S</td>
<td>2013</td>
<td>144:39</td>
<td></td>
</tr>
<tr>
<td>noaaportIngester -b 50000 -m 224.0.1.1 -n -</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>root</td>
<td>11739</td>
<td>0.0</td>
<td>19.2</td>
<td>1253936</td>
<td>1199688</td>
<td>?</td>
<td>S</td>
<td>2013</td>
<td>8:48</td>
<td></td>
</tr>
<tr>
<td>noaaportIngester -b 50000 -m 224.0.1.2 -n -</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>root</td>
<td>11740</td>
<td>0.6</td>
<td>21.7</td>
<td>1454292</td>
<td>1355356</td>
<td>?</td>
<td>S</td>
<td>2013</td>
<td>148:20</td>
<td></td>
</tr>
<tr>
<td>noaaportIngester -b 50000 -m 224.0.1.3 -n -</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>root</td>
<td>11741</td>
<td>0.0</td>
<td>19.1</td>
<td>1233948</td>
<td>1190448</td>
<td>?</td>
<td>S</td>
<td>2013</td>
<td>14:32</td>
<td></td>
</tr>
<tr>
<td>noaaportIngester -b 50000 -m 224.0.1.4 -n -</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>root</td>
<td>11743</td>
<td>0.0</td>
<td>19.3</td>
<td>1246528</td>
<td>1204156</td>
<td>?</td>
<td>S</td>
<td>2013</td>
<td>19:33</td>
<td></td>
</tr>
<tr>
<td>noaaportIngester -b 50000 -m 224.0.1.5 -n -</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ldm</td>
<td>20590</td>
<td>0.0</td>
<td>0.0</td>
<td>4656</td>
<td>1532</td>
<td>pts/D</td>
<td>S</td>
<td>15:48</td>
<td>0:00</td>
<td>-bash</td>
</tr>
<tr>
<td>ldm</td>
<td>21440</td>
<td>0.0</td>
<td>0.0</td>
<td>4376</td>
<td>988</td>
<td>pts/D</td>
<td>R</td>
<td>15:53</td>
<td>0:00</td>
<td>ps -U ldm -u</td>
</tr>
</tbody>
</table>

NOTE: At each AWIPS site, there are two CPSBNs responsible for receiving GOES products and a combination of NCEP products from the NCF. No site has more than two CPSBNs unless it has an NRS.

Although each of the two CPSBNs (CPSBN1 and CPSBN2) receive the entire stream of data, for AWIPS II only one CP is configured to forward the entire data to AWIPS data acquisition server. The data are split by channels, and at any point in time only one CPSBN forwards channel data so duplicate data are not being ingested. Because each CPSBN can process the entire data stream, they back each other up. When one CPSBN is actively processing all 4 channels, the other CPSBN will be the hot spare. To check channels received, examine the ldmd.conf file to look at running ldm processes; the config_dvb command is not implemented in AWIPS II.
[Note: The CP ingest is tied to the cp1f package. The hb_stat output will show the following.]

Heartbeat Status Monitor

Heartbeat Status Monitor Jul 17 15:36:08

----------------------------------- Member Status -----------------------------------

<table>
<thead>
<tr>
<th>Member</th>
<th>Status</th>
<th>IP address</th>
</tr>
</thead>
<tbody>
<tr>
<td>cpsbn1-oax</td>
<td>Up</td>
<td>165.92.109.50</td>
</tr>
<tr>
<td>cpsbn2-oax</td>
<td>Up</td>
<td>165.92.109.51</td>
</tr>
</tbody>
</table>

----------------------------------- Service Status -----------------------------------

<table>
<thead>
<tr>
<th>Service</th>
<th>IPaddr</th>
<th>Cronfile</th>
<th>Owner</th>
<th>Start Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>a2cp1apps</td>
<td>165.92.109.60</td>
<td>a2cp1cron,a2SIT cpsbn1-oax</td>
<td>2013-06-06 14:12:42</td>
<td></td>
</tr>
<tr>
<td>a2cp2apps</td>
<td>165.92.109.62</td>
<td>a2cp2cron,a2SIT none</td>
<td>down</td>
<td></td>
</tr>
</tbody>
</table>

[root@cpsbn1-oax]# grep EXEC /usr/local/ldm/etc/ldmd.conf | grep -v "#"
EXEC "pqact -e"
EXEC "edexBridge -s cp1f"
EXEC "noaaportIngester -m 224.0.1.1 -n -u 3 -t mhs -r 1 -s NMC"
EXEC "noaaportIngester -m 224.0.1.2 -n -u 4 -t mhs -r 1 -s GOES"
EXEC "noaaportIngester -m 224.0.1.3 -n -u 5 -t mhs -r 1 -s NMC2"
EXEC "noaaportIngester -m 224.0.1.4 -n -u 6 -t mhs -r 1 -s NOAAPORT_OPT"
EXEC "noaaportIngester -m 224.0.1.5 -n -u 7 -t mhs -r 1 -s NMC3"

[Note: These numbers in bold in the example above correspond to the different channels of SBN downlink. There is one dvbs_multicast process that listens to each multicast address and processes that particular channel.]

[root@cpsbn1-tbdw]# ps -fu ldm
UID    PID  PPID  C   STIME TTY   TIME CMD
ldm    11734  1     0 2013 ?   00:00:00 ldmd -I 165.92.29.180 -P 388 -M 256 -m 3600 -o 3600 -q /usr/
ldm    11736  1    01:53:23 pqact -e
ldm    11737  11734  0  2013 ?   00:09:08 edexBridge -vxl /usr/local/ldm/logs/edexBridge.log -s cp1f
ldm    20590 20589  0 15:48 pts/0 00:00:00 -bash
ldm    21519 20590  0 15:55 pts/0 00:00:00 ps -fu ldm
4.9 Product Identifier

The LDM writer uses patterns in pqact.conf to determine where to write data files, including the path and file name.

For example, take an entry that matches an LDM product with these fields:

Use the following reference to help decipher a pqact.conf entry.

```
NNEXRAD         ^(SDUS[2-8].|NXUS6.)(K|P|T)(LWX|BGM|CHS|RLX|ILN|CLE|AKQ|JKL|CTP|MHX|MRX|OKX|PHI) (.)(.)(.)
/p(...)(...)
FILE   -overwrite -log -close -edex
/data_store/radar/\2\8/\7/\5\6_\2\8_\7_(seq).rad
```

The first part, NNEXRAD, is the FEEDTYPE.

The Regular Expression describes any product starting with SDUS and follows with any number between 2 and 8 OR starting with NXUS6.

The () store that strings into the variable \1, which can be used when storing to physical disk.

(K|P|T) matches either a K, a P, or a T, and stores it to the variable \2.

(LWX|BGM|CHS|RLX|ILN|CLE|AKQ|JKL|CTP|MHX|MRX|OKX|PHI) matches any of those site IDs and stores it to variable \3.

(.)(.)(.) matches the next 6 characters of any type and stores them in pairs to \4, \5 and \6 respectively.

/p matches the exact string /p

(...)(...) matches the next 6 characters of any type and stores them in trios to \7 and \8.

So the following line would match this entry in pqact and store in the following location:

**Product:**

```
NEXRAD3 56424268 SDUS51 KLWX 211529 /pTZLDCA !nids/
```

**Storage Location:**

```
/data_store/radar/KDCA/TZL/1529_KDCA_TZL_56424268.rad
```

For DPA products, the pattern is nearly identical:

In this case, only DPA products are matched, so the NNN entry is hard coded as DPA.

**Product:**

```
NEXRAD3 56424268 SDUS51 KLWX 211529 /pDPADCA !nids/
```

**Storage Location:**

```
/data_store/radar/KDCA/TZL/1529_KDCA_TZL_56424268.rad
```
The number before the ".rad" is the sequence number (56424268 in this case) that is set at the Gateway when transmitted over the SBN and can be traced to the ldm logs on the active cpsbn.

[Note: Sequence numbers are NOT reset at all. The only exception occurs when the data channels are moved from one DSUP to another. Also, it will be different when the operations are moved from ANCF to BNCF servers.]

To determine which cpsbn is active, perform the following steps and look for running ldm processes.

**CPSBN1 Example:**
```
[root@cpsbn1-tbdw ~]# ps -ef | grep ldm
ldm  11734  1  0  2013 ?    00:00:00 ldmd -I 165.92.29.180 -P 388 -M 256 -m 3600 -o 3600 -q /usr/local/ldm/data/ldm.pq /usr/local/ldm/etc/ldmd.conf
ldm  11736 11734  0  2013 ?    01:53:26 pqact -e
ldm  11737 11734  0  2013 ?    00:09:08 edexBridge -vxl /usr/local/ldm/logs/edexBridge.log -s cp1f
root 22616 20316  0 16:00 pts/0    00:00:00 grep ldm
```

**CPSBN2 Example:**
```
[root@cpsbn2-tbdw ~]# ps -ef | grep ldm
root 21927  1  0 13:12 pts/0    00:00:00 grep ldm
```

So in this case, CPSBN1 is active and look for the sequence number 56424268 (as above) in the ldm logs.

### 4.10 Qpid

In the qpid commands that follow, you are looking for queues in which the msg column has a large number. This means that messages are coming in, but nothing is taking them out.

You could also look at the msgIn and msgOut columns to get the same type of picture on the health of the system. The NCF has set up monitors based on this data in order to keep track of the health of the system, and restart things if necessary.

```
[root@cpsbn1-oax ~]# ps -ef | grep qpid
root  22948  1  0 16:02 pts/0    00:00:00 grep qpid
root  32392  1  0 2013 ?    00:00:00 su awips -c /bin/bash /awips2/qpid/bin/qpid-wrapper
awips 32394 32392  0  2013 ?    00:00:00 /bin/bash /awips2/qpid/bin/qpid-wrapper
awips 32397 32394  0  2013 ?    00:00:08 /awips2/java/bin/java -Xmx32m -XX:MaxPermSize=12m -XX:ReservedCodeCacheSize=4m -jar /awips2/qpid/bin/yajsw/wrapper.jar -c /awips2/qpid/conf/wrapper.conf
```
awips 32424 32397 9 2013 ? 1-09:08:41 /awips2/java/bin/java
-classpath
org.rzo.yajsw.app.WrapperJVMMain

[root@cpsbn1-oax ~]# qpid-queue-count -S msg -L10

Queues
queue msg bytes cons
===============================================================================
Ingest.Grib 31 3.19k 8
metaDataPurgeWork 1 0 1
_pluginPurged@amq.topic_b98ca05d-fd4c-4259-b2d1-f4b9ef5e4c0e 0 0 1
Ingest.lar 0 0 2
edex.alerts.utility@amq.topic_6576915d-b5b9-4bc8-97f322a0b575b11 0 0 1
Ingest.bufrssmi 0 0 2
Ingest.nucaps 0 0 2
Ingest.regionalsat 0 0 2
mpeLightSrvScheduledWork 0 0 2
_edex.alerts.utility@amq.topic_31ccc933-82d0-420c-af6b7186001e7082 0 0 1

[root@cpsbn1-oax ~]# qpid-stat -q -S msg -L10

Queues
queue msgIn msgOut bytes bytesIn bytesOut cons bind
dur excl msg
======================================================================================
edex.alerts.gfe@amq.topic_1f0d6518-7564-469e-b192-a5bd3e7f78d1 Y 0 30.6k 30.6k 0 111m 111m 1 2
Ingest.ssha Y 0 0 0 0 2 2
Ingest.idadprofiler Y 0 0 0 0 2 2
_purgeGridInfoCache@amq.topic_b09f7f29-fc1e-4da2-9254-7c09fe463e1f Y 0 3 3 0 440 440 1 2
Ingest.Mesowest Y 0 0 0 0 2 2
Ingest.acars Y 0 39.1k 39.1k 0 3.19m 3.19m 1 2
handleoup.dropbox Y 0 391 391 0 19.8k 19.8k 2 2
_edex.alerts@amq.topic_4c2c0a0b-013e-4d4e-82e2-4fe5239f9227 Y 0 8.28k 8.28k 0 94.1m 94.1m 1 2
mpeLightSrvScheduledWork 0 0 328 328 0 0 0 2 2
4.11 **LDM Ingest Checklist**

In order to add a line to the pqact.conf or pqact.local file for LDM ingest, certain information is necessary. The following is a general checklist of information that can be used when adding new data from the SBN into the system.

- Note the WMO header or pattern for desired product: ________________
- Note the LDM FEEDTYPE for the desired product: ________________
- Note the raw archive location for the desired product: ________________
- Create entry in the pqact.conf or pqact.local file: ________________
- Restart or send HUP to ldmd process.

The following steps may be useful in finding the above information.

1. **Find the WMO header or pattern for the desired product.**
   
   If the product is being ingested into an AWIPS I system, look at the /data/fxa/customFiles/acqPatternAddOns.txt file. It is possible that you will discover the WMO pattern needed for ingesting. A listing of WMO header bulletins can also be found starting at the following URL: http://www.nws.noaa.gov/tg/table.html

2. **Find the LDM FEEDTYPE for the desired product.**
   
   In all cases, the FEEDTYPE “ALL” will match all the FEEDTYPES listed. However, tools are available that may help identify the proper FEEDTYPE, if that is of interest. As user `ldm` the following commands might be useful:

   ```bash
   # ldmadmin watch
   ``
   
   This command will show all data coming from the LDM server. This should be every product that the ldm is ingesting. Watching can identify specific feedtypes of the products being ingested. A sample output of the command follows.

   ```bash
   # notifyme -l -h $LDM_HOST -v -p '<WMO ID Pattern>'
   ```
This command will print to the screen every product available from <LDM IP/Hostname> that matches <WMO ID Pattern>. For example, the following would print all products starting T available from CPSBN1:

```
notifyme -l- -h cpsbn1 -v -p '^T.*'
```

**Note:** The pattern passed to notifyme, and put into the pqact.conf file, must be a valid regular expression.

A sample output of the command follows.

```
-bash-3.2$ notifyme -l- -v -h adam1 -p '^sat.*TI.*'
Jan 21 15:17:37 notifyme[6887] NOTE: Starting Up: adam1:
20110121151737.886 TS_ENDT {{ANY, "^sat.*TI.*"}}
Jan 21 15:17:37 notifyme[6887] NOTE: LDM-5 desired product-class:
20110121151737.886 TS_ENDT {{ANY, "^sat.*TI.*"}}
Jan 21 15:17:37 notifyme[6887] INFO: Resolving adam1 to 165.92.24.36 took 0.000398 seconds
```

Both of these commands will give other information that can be used in writing the data to disk in the raw archive in a more useful format.

3. **Note the raw archive location for the product.**

   In general, the raw archive used is /data_store/<product type>. For example, radar products are put into a subtree of /data_store/radar based on the information. It is good practice to continue to follow this paradigm. However, as long as the –edex argument is used in the pqact.conf entry, the product should be decoded no matter where the physical storage location ends up as long as the EDEX server has access to that location.

4. **Create the pqact.conf entry.**

   Create an entry in the pqact.conf file.

5. **Restart or send HUP signal to LDM server.**

   Restart or send a HUP signal to the LDM server.
Chapter 5
Ingest of Satellite Imagery
Chapter 5. Ingest of Satellite Imagery

Table of Contents

5.1 Overview................................................................................................................................ 1
5.2 Satellite Ingest Data Flow....................................................................................................... 1
   5.2.1 Data Archive and File Naming Conventions for Satellite Data................................. 4
      5.2.1.1 LDM Configuration .................................................................................... 6
   5.2.2 Decoding – Satellite Data .......................................................................................... 6
   5.2.3 Data Ingest and Logging of Satellite Products ......................................................... 6
5.3 Monitoring the LDM Data Flow and GOES data................................................................. 8
   5.3.1 Verifying LDM Operation on the CP cluster............................................................. 9
   5.3.2 Monitoring LDM Data Flow on the CP or DX Cluster ........................................... 10
5.4 GOES Troubleshooting and Tips......................................................................................... 13
   5.4.1 Troubleshooting ....................................................................................................... 13
5.5 Data Archive Directory........................................................................................................ 14

List of Exhibits

Exhibit 5.2-1. Satellite Ingest Data Flow...................................................................................... 2
Exhibit 5.2.1-1. Acquire and Ingest of Satellite Products .............................................................. 5

List of Tables

Table 5.3.2-1. Selected LDM Feed Types.................................................................................... 12
5.1 Overview

All GOES imagery products received by AWIPS sites are distributed frame by frame over two of the four NOAAPORT/SBN Channels. The four channels are:

1. **GOES.** The GOES Channel contains information from the GOES East satellite and the GOES West satellite. The GOES East satellite is a data stream consisting of these imagery products: visible infrared and water vapor for the Eastern Continious United States (CONUS), Puerto Rico, supernational composites, and Northern Hemisphere (NH) composites. The GOES West satellite is a data stream consisting of the following imagery products: visible infrared and water vapor for CONUS, Alaska, and Hawaii; supernational composites, and NH composites. As of this build, AWIPS II incorporates the capability to ingest and display National Polar Program (NPP) satellite imagery. NPP polar orbiter imagery is critical for WFOs located at high latitudes, such as those in Alaska.

2. **NMC.** From the NWS Telecommunications Gateway (NWSTG), a data stream consisting of model output from the National Centers for Environmental Prediction (NCEP); the observations, forecasts, watches, and warnings produced by NWS Forecast Offices; WSR-88D radar products; and most observational data over North America.

3. **NMC2.** The NWSTG2 channel supplements the NWSTG channel. It contains all Model Grid data.

4. **NOAAPORT_OPT Channel.** This channel's data stream includes GOES DCP data, GMS/GOES-West/GOES-East/METEOSAT-5//METEOSAT-7 composites for visible infrared and water vapor products (every 3 hours), and OCONUS grids.

5.2 Satellite Ingest Data Flow

The Satellite Ingest data flow, illustrated in Exhibit 5.2-1, is a 12-step process. Descriptions of each step follow.

1. Satellite data is obtained by the LDM noaaportIngester from the SBN.
   a. The LDM noaaportIngester forwards the product to the LDM Writer instance.
   b. The LDM Writer instance writes the product to the Data Archive. [Data Archive (/data_store) directory is mounted on the hosts off the NAS [NAS2].]
   c. The LDM writer instance uses the EDEX Bridge to post a message containing product information to the QPID Message Broker. The product information is the WMO header and the path to the product file.
Exhibit 5.2-1. Satellite Ingest Data Flow

2. The EDEX Distribution endpoint on the EDEX (DX) cluster pulls the message from the QPID Message Broker. The WMO header is matched against data distribution mappings contained in the data distribution directory to determine product routing. (The WMO header is included in the message; if it is not, the Distribution service uses the product file name from the message to determine routing.) *Note:* The files in
the data distribution directory are XML files that contain regular expressions that match WMO headers with the appropriate data decoder plug-in.

a. At EDEX startup, the EDEX Distribution endpoint reads the Data Distribution files from the Data Distribution directory in the Localization Store. It uses the contents of these files to generate the routing table used to determine data routing. The Data Distribution directory is located at /awips2/edex/data/utility/edex_static/base/distribution.

b. The EDEX Distribution endpoint monitors the Data Distribution directory; when a file in that directory is modified, it re-reads that file and rebuilds its routing table.

3. The EDEX Distribution Service determines the Data Decoder Plug-In registered to handle the data in the product file.

4. The EDEX Distribution Service forwards the message to the appropriate Data Decoder Plug-In via the QPID Message Broker.

5. The EDEX Data Decoder Service on the EDEX (DX) cluster pulls the message from the QPID Message broker. The Data Decoder Service reads the file pointed to by the message, decodes the data, and determines the appropriate metadata.

6. The decoded data and metadata are sent to the EDEX Persist endpoint.

7. The EDEX Persist endpoint sends the data to PyPIES, and PyPIES writes the data to the EDEX data store in HDF5 format.

8. The EDEX Persist endpoint sends the metadata obtained to the EDEX Index endpoint.

9. The EDEX Index endpoint sends the metadata to the PostgreSQL DBMS.
   a. The PostgreSQL DBMS writes the metadata to the database.

10. The index endpoint persists the data object to the database. It does not just get the dataURI. Persistence to the database is executed via a plugin defined data access object. If the data access object does not override the persistence behavior, then a default persistence strategy is used. The default strategy for persisting data to the database is to first check to see if the data received already exists. If it already exists, then persistence to the database is not attempted for that data. If the data is unique, then it is persisted to the database.

11. (Client notification) The Data URI is forwarded to the EDEX URI Aggregator.
   a. Every 5 seconds the aggregated Data URIs are sent to the JMS Alert Topic.
   b. CAVE monitors the JMS Alert Topic to obtain a list of available data. CAVE determines if it should render the available data.
   c. CAVE retrieves metadata from the EDEX server via a product query sent to the EDEX thrift endpoint.
d. CAVE retrieves product data from the EDEX server by accessing the EDEX data store’s HDF5 files and CAVE renders the data, updating its display as appropriate.

12. (Subscription) The Data URI is forwarded to the EDEX Subscription endpoint. (Note: This does not apply to Satellite Data)
   a. The EDEX Subscription endpoint checks the active subscription list to determine if a subscription has been registered for the product.
      1) If no subscription has been registered for the product, no further action is taken.
   b. The EDEX Subscription endpoint obtains the script information from the database.
   c. The EDEX Subscription endpoint obtains a micro-engine to execute the script.
      1) The micro-engine executes the subscription script for the product.

5.2.1 Data Archive and File Naming Conventions for Satellite Data

As illustrated in Exhibit 5.2.1-1, the satellite data received over the SBN are stored in the following directory structure.

/data_store/sat/YYYYMMDD/<HH>/<SECTOR ID>/<PHYSICAL ELEMENT>/

The <SECTOR ID> is the satellite the image was derived from. GOES_WEST and GOES_EAST are not always GOES-15 and GOES-14 respectively. This depends on how NESDIS moves the satellites in and out of storage orbits. This is important because GOES-15 and GOES-14 have blackout periods during which they do not send products. For example, east CONUS is predominately covered by GOES-14, but for a short period in the early morning, its source of data is GOES-15. In the case of supernational products, GOES-15 is the source for the top and bottom of the hour, and GOES-14 is the source at 15 and 45 minutes past the hour.

Note: AWIPS II is still receiving products from NESDIS with the GOES-11 and GOES-12 identifiers. This is intentional, and is done in order to maintain compatibility with the AWIPS decoder.

- The <PHYSICAL ELEMENT> indicates if it is an element in the visible infrared, water vapor, etc. ranges.
Exhibit 5.2.1-1. Acquire and Ingest of Satellite Products
5.2.1.1 LDM Configuration

LDM server runs only on the SBN CP. It ingests everything from the SBN.

LDM behavior is controlled by the pqact.conf file, and the patterns are matched against ***pqact.conf***. LDM writes the product to data_store and posts a QPID message.

- **pqact.conf** controls how data sent to an instance of LDM is handled. This includes the data types to be handled and the location of the raw data archive.

**NOTE:** Modifying pqact.conf will have a severe impact on the data flow.

5.2.2 Decoding – Satellite Data

Data decoding operations are performed on the EDEX (DX3/4) servers. Both EDEX servers share the load in the ingest process, which includes decoding the raw data, writing the decoded data into the Data Store, writing metadata to the database, and broadcasting a notification that the newly ingested data is available.

The raw satellite files are stored in the Data Archive (:/data_store). The physical directory is on the NAS2. All the hosts, including the CPSBN, mount the volume off NAS2.

The processed data is stored in /awips2/edex/data/hdf5 in the hdf5 format on the dx2f server. The current satellite data in the HDF5 store is organized according to the pattern:

/awips2/edex/data/hdf5/satellite/satellite/satellite-YYYY-MM-DD-<HH>.h5,

5.2.3 Data Ingest and Logging of Satellite Products

All satellite data ingest operations are logged in the EDEX logs on the EDEX (DX3/4) server. In a standard installation, the EDEX logs are located in /awips2/edex/logs.

The edex-ingest-satellite log file handles the satellite products.

The EDEX logs roll over daily and are date stamped.

For the satellite decoder, the instance name is “ingest,” so the satellite ingest log for Feb. 13, 2013, is named **edex-ingest-satellite-20130213.log**.

**NOTE:** Each EDEX Server (DX3 and DX4) has its own set of separate log files. Both sets of log files have the same names and structure, but they contain different information.

The log file is a plain ASCII text file and can be viewed using any text-viewing utility such as *more*, *less*, *view*, or *tail*. 
To get a general view of processing, open a terminal session to the server and enter the following commands:

**On DX3/DX4,**

**TYPE:** `cd /awips2/edex/logs`

**TYPE:** `date +%F`

2013-02-13

**TYPE:** `ls edex*20130213*`

- edex-ingest-20130213.log
- edex-ingestDat-20130213.log
- edex-ingest-gen_areal_ffg-20130213.log
- edex-ingest-gen_areal_qpe-20130213.log
- edex-ingestGrib-20130213.log
- edex-ingest-purge-20130213.log
- edex-ingest-radar-20130213.log
- edex-ingest-satellite-20130213.log
- edex-ingest-shef-20130213.log
- edex-ingest-shef-performance-20130213.log
- edex-ingest-smartInit-20130213.log
- edex-ingest-text-20130213.log
- edex-ingest-trigger-20130213.log
- edex-ingest-unrecognized-files-20130213.log
- edex-request-20130213.log
- edex-request-productSrvRequest-20130213.log
- edex-request-thriftSrv-20130213.log
- edex-ingest-archive-20130213.log
- edex-ingest-GFEPerformance-20130213.log
- edex-request-GFEPerformance-20130213.log
- edex-ingest-activeTableChange-20130213.log
- edex-ingestDat-activeTableChange-20130213.log
- edex-ingestDat-performance-20130213.log
- edex-ingestGrib-activeTableChange-20130213.log
- edex-ingestGrib-performance-20130213.log
- edex-ingest-performance-20130213.log
- edex-request-activeTableChange-20130213.log
- edex-request-performance-20130213.log
- edex-ingest-gen_areal_qpe-20130213.log
- edex-request-hibernate-20130213.log
- edex-ingest-hibernate-20130213.log
- edex-ingestGrib-hibernate-20130213.log
Chapter 5  
Ingest of Satellite Imagery

Note: edex-ingestDat log file is only available on large memory servers.

**TYPE:**    
`tail -f edex-ingest-satellite-20130213.log`

After ingest of each data file is complete, the ingest endpoint prints a single line to the EDEX system log. The format of the log entry is

```
INFO <<date-time>> [thread] Ingest: <<type>>:: <<file name>> <<statistics>>
```

Where,

- `<<date-time>>` is the date-time stamp (format yyyy-mm-dd hh:mm:ss,ttt)
- `[thread]` identifies the Camel thread that ran the decoder.
- `<<type>>` identifies the data type, e.g. grib, radar, etc.
- `<<file name>>` is the path to the file that was processed.
- `<<statistics>>` lists the ingest statistics for the file. There are two statistics provided; `process time` and `latency` (refer to the note at the end of this section for more information on latency)

The satellite log entry is as follows.

```
INFO 2013-02-13 15:54:30,836 [satelliteThreadPool-1] Ingest: EDEX: Ingest - satellite:: /data_store/sat/20130213/15/GOES-15/1545Z_3.9_8km_AK-REGIONAL_TIGA04_KNES_15636353.satz.2013021315 processed in: 0.0820 (sec) Latency: 0.1280 (sec)
```

**NOTE:** Latency is the elapsed time between when the AWIPS II system first contacts a data file and when the data has been ingested, decoded, and is ready from retrieval by either CAVE or another client application. The “first contact” depends on the data type. For most data types, latency measurement starts when the product has been downloaded by the LDM Writer on CP1f. This latency start time is part of the message generated by the LDM’s EDEX Bridge and sent to QPID (on the CP 1/2 cluster) for processing by EDEX. Note that this latency measure is strictly internal for the AWIPS II data ingest subsystem. This information is logged into the EDEX process logs for every file that is ingested.

5.3 Monitoring the LDM Data Flow and GOES data

LDM is the system used by AWIPS II to transfer data from the SBN to the EDEX ingest processes. In AWIPS II, LDM runs on the CP cluster.

**NOTE:** For data to be made available to the EDEX ingest processors, LDM must be operational.
The LDM system includes a general administration tool, \textit{ldmadmin}. That tool is used to start and stop the LDM; it also provides some of the monitoring capabilities used in this set of examples.

5.3.1 \textbf{Verifying LDM Operation on the CP cluster}

Although \textit{ldmadmin} provides a tool for determining if the LDM is running, it can be easier to use the Linux \texttt{ps} utility to verify operation. This example presents the approach for the CP cluster.

- \textbf{Scenario 1: Verify LDM Operation Using the Linux ps Utility}

$$\texttt{ssh}\ ldm@cpsbn1f$$

The authenticity of host 'cpsbn1f (165.92.109.60)' can't be established.


Are you sure you want to continue connecting (yes/no)? Type <yes>

ldm@cpsbn1f’s password:

$$\texttt{ps} \ -u\ ldm\ -o\ user,pid,args$$

<table>
<thead>
<tr>
<th>USER</th>
<th>PID</th>
<th>COMMAND</th>
</tr>
</thead>
<tbody>
<tr>
<td>ldm</td>
<td>11734</td>
<td>lmdm -I 165.92.29.180 -P 388 -M 256 -m 3600 -o 3600 -q /usr/local/ldm/data/ldm.pq /usr/local/ldm/etc/ldmd.conf</td>
</tr>
<tr>
<td>ldm</td>
<td>11736</td>
<td>pqact -e</td>
</tr>
<tr>
<td>ldm</td>
<td>11737</td>
<td>edexBridge -vxl /usr/local/ldm/logs/edexBridge.log -s cp1f</td>
</tr>
<tr>
<td>ldm</td>
<td>24629</td>
<td>-bash</td>
</tr>
<tr>
<td>ldm</td>
<td>24748</td>
<td>ps -u ldm -o user,pid,args</td>
</tr>
</tbody>
</table>

The exact output will vary. Note, however, that the critical lines are those with arguments \texttt{ldmd}, \texttt{pqact}, and \texttt{edexBridge}. These indicate that the various parts of the LDM are operating.

- \textbf{Scenario 2: Verify LDM Using the ldmadmin Program}

The \textit{ldmadmin} program’s \texttt{isrunning} option checks to see if the LDM is running. Unfortunately, this option simply returns 0 or 1 and exits; 0 is returned if the LDM is running, and a 1 is returned if it is not running. After executing \texttt{ldmadmin isrunning}, you need to check its return value. This sequence should work under most shells:

$$\texttt{ssh}\ ldm@cpsbn1f$$

The authenticity of host 'cpsbn1f (165.92.109.60)' can't be established.

Are you sure you want to continue connecting (yes/no)?
Type <yes>

ldm@cpsbn1f’s password:
$ ldmadmin isrunning  
$ echo $?  
0
$

This output indicates that the LDM is running. If it were not running, the `echo $?` command would print a 1.

If you are using the *bash* shell, you can combine things slightly and produce a more readable result. The modified command sequence:

```
$ ldmadmin isrunning > /dev/null 2>&1 && echo "Running"
```

```
Running
$
```

Once again, this output indicates that the LDM is running; if it were not, no output would be displayed.

### 5.3.2 Monitoring LDM Data Flow on the CP or DX Cluster

The *ldmadmin* program provides for real-time monitoring of the LDM product queue. This can be used to determine quickly if data is passing through the LDM; it can also be filtered by data feed to concentrate on a specific data type. Three scenarios are presented: 1) basic product queue monitoring; 2) identifying data feeds; and 3) monitoring a specific data feed.

**Scenario 1: Basic Product Queue Monitoring**

The *ldmadmin* program’s *watch* option provides a real-time view of the LDM’s message queue. Monitoring this queue provides a quick verification of the status of the LDM. These are the basic steps to follow:

```
$ ssh ldm@cpsbn1f
ldm@cpsbn1f’s password:
$ ldmadmin watch
(type ^D when finished)
```

At this point, you should see output similar to

Feb 13 16:07:24 pqutil INFO: 8442 20130213160723.853 HDS 65903041 YVQN25  
KWBC 131200/mGFS!grib/ncp/GFS/#211/201302131200/F108/VGRD/250 mb/  
Feb 13 16:07:24 pqutil INFO: 167 20130213160723.871 IDS/DDPLUS 15640711  
SXCH40 KWAL 131602
Feb 13 16:07:55 pqutil INFO: 26702 20130213160755.554 NGRID 39425358 MPTV96
KWBC 131200 grib2/ncep/GFS/#161/201302131200F102/PRES/0 - MWSL

Feb 13 16:07:55 pqutil INFO: 6005 20130213160755.570 NEXRAD3 65904852 SDUS84
KSHV 131601 pN2MSHV !nids/

Jul 17 18:23:57 pqutil INFO: 38088 20130717182357.831 NEXRAD3 562839 SDUS83
KIWX 171821 pNBXIWX !nids/

Jul 17 18:24:37 pqutil INFO: 43572 20130717182436.838 IDS/DDPLUS 564194 SRUS53
KLMK 171824 pHMLLMK

Jul 17 18:24:37 pqutil INFO: 484514 20130717182436.984 NIMAGE 448
satz/ch1/GOES-13/WV/20130717 1815/EAST-CONUS/4km/ TIGE05 KNES 171815

Jul 17 18:26:33 pqutil INFO: 149 20130717182632.793 HDS 568528 NXUS66 KEKA
171826 pGSMBHX

If data is flowing, this output will scroll past fairly rapidly and the message date
time stamp should be current. If no messages are scrolling, it is likely that there is an issue
with the LDM.

- **Scenario 2: Identifying LDM Data Feeds**

  The *ldmadmin* program’s *watch* facility has an option to filter output for a specific
  feed type. The basic command is *ldmadmin watch -f <feed>*. This scenario presents a
  method for identifying the available feeds.

  The feeds used by LDM are defined in a file named *pqact.conf*, which is located in
  `<ldm-home>/etc`. While this file can be examined to determine the available feed
types, the process can be automated using the known structure of the file and a few
basic Linux utilities. The lines defining LDM feeds have the following structure:

  ```
  FEEDTYPE <tab> pattern <tab> action [<tab> options] [<tab> args]
  ```

  Note that lines defining feed types start with at least one uppercase letter. This allows
  you to filter for the feed definitions and extract the feed names. These are the steps to
  follow:

  ```
  $ ssh ldm@cpsbn1
  ldm@cpsbn1’s password:
  $ cd etc
  $ grep -P "^[A-Z]+" pqact.conf | cut -f1 > feeds.txt
  $ sort feeds.txt > sorted-feeds.txt
  $ uniq sorted-feeds.txt ldm-feeds.txt
  $ cat ldm-feeds.txt
  ANY
  GPSSRC
  GRID
  ```
The final output is a sorted list of feed names. The list may differ somewhat depending on the exact data flow configuration for the server. Some of the feed type names are rather cryptic; common feed types are identified in Table 5.3.2-1.

Table 5.3.2-1. Selected LDM Feed Types

<table>
<thead>
<tr>
<th>Feed Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANY</td>
<td>wildcard – matches any feed type</td>
</tr>
<tr>
<td>GRID</td>
<td>NOAAport high-resolution model output</td>
</tr>
<tr>
<td>NEXRAD2</td>
<td>NEXRAD Level II radar data</td>
</tr>
<tr>
<td>NIMAGE</td>
<td>NOAAport satellite imagery</td>
</tr>
<tr>
<td>NNEXRAD</td>
<td>NEXRAD Level III products</td>
</tr>
</tbody>
</table>

- **Scenario 3: Monitoring a Specific Data Feed**

Log onto the LDM server (CPSBN) as the ldm user. Use the -f option and specify a FEED TYPE to see only products from that LDM feed:

`ldmadmin watch -f NIMAGE`

Press <Crtl>+<D> to terminate the process.

```
Jul 17 18:28:33 pqutil INFO: 1052678 20130717182832.685 NIMAGE
462 satz/ch2/GOES-15/IR/20130717 1800/SUPER-NATIONAL/8km/ TIGN02
KNES 171800
```

```
Jul 17 18:28:37 pqutil INFO: 215727 20130717182837.344 NIMAGE
463 satz/ch1/GOES-13/WV/20130717 1815/NHEM-COMP/24km/ TIGF05
KNES 171815
```

```
Jul 17 18:28:38 pqutil INFO: 313090 20130717182837.839 NIMAGE
464 satz/ch1/GOES-13/IR/20130717 1815/NHEM-COMP/24km/ TIGF02
KNES 171815
```

To see connections open with ldm:

```
netstat -ta | grep ldm
```

```
tcp 0 0 *:unidata-ldm *:*  
LISTEN
```

To monitor the product queue via:
Chapter 5  Ingest of Satellite Imagery

pqmon
Jul 17 18:29:43 pqmon NOTE: Starting Up (5927)
Jul 17 18:29:43 pqmon NOTE: nprods nfree nempty nbytes maxprods
maxfree minempty maxext age
Jul 17 18:29:43 pqmon NOTE:  60048  1 184091 999948616 77061 4 167078 52920 1370
Jul 17 18:29:43 pqmon NOTE: Exiting

ps -U ldm -u ldm u
USER    PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
ldm   11734  0.0  0.0   4548   952 ?        Ss    2013   0:00 ldmd -I 165.92.29.180 -P 388 -M 256 -m 3600 -o 3600 -q /usr/local/ldm/data/ld
ldm   11736  0.4 15.8 1000952 987460 ?      S     2013 113:34 pqact -e
ldm   11737  0.0  0.4   55224  24988 ?        Sl    2013   9:08 edexBridge -vxl /usr/local/ldm/logs/edexBridge.log -s cpif
root  11738  0.5 19.2 1238064 1196952 ?     Sl    2013 144:53 noaaportIngester -b 50000 -m 224.0.1.1 -n -u 3 -t mhs -r 1 -s NMC
root  11739  0.0 19.2 1253936 1199688 ?     Sl    2013   8:49 noaaportIngester -b 50000 -m 224.0.1.2 -n -u 4 -t mhs -r 1 -s GOES
root  11740  0.6 21.7 1454796 1355844 ?     Sl    2013 148:38 noaaportIngester -b 50000 -m 224.0.1.3 -n -u 5 -t mhs -r 1 -s NMC2
root  11741  0.0 19.1 1233948 1190468 ?     Sl    2013   14:34 noaaportIngester -b 50000 -m 224.0.1.4 -n -u 6 -t mhs -r 1 -s NOAAPORT_OPT
root  11743  0.0 19.3 1246528 1204156 ?     Sl    2013 19:33 noaaportIngester -b 50000 -m 224.0.1.5 -n -u 7 -t mhs -r 1 -s NMC3
ldm   24629  0.0  0.0   4652  1468 pts/0  S    16:19   0:00 -bash
ldm   25992  0.0  0.0   4520  1108 pts/0  R+   16:24   0:00 ps -U ldm -u ldm u

5.4 GOES Troubleshooting and Tips

5.4.1 Troubleshooting

Common problems seen with GOES data and their possible remedies follow.

- **Problem A**: The product loop for incoming images is not auto-updating on a workstation, or products are not available at a workstation.
  - **Potential Solution to Problem A**: Check another workstation for similar problems. It may be that only one workstation is having difficulties. If this is the case, exit and restart CAVE on that workstation.

- **Problems B, C, and D**
  - **Problem B**: No GOES product loops are updating at the site.
  - **Problem C**: Data products are arriving late on the workstation.
  - **Problem D**: No GOES products.
• **Potential Solutions to Problems B, C, and D**
  - If the problem is unique to a single workstation, try restarting CAVE. As a last resort, try restarting the workstation.
  - If the problem is more general, try to verify that data is actually available. See section 5.2 for monitoring tools.
  - If data is not available, contact the NCF.

### 5.5 Data Archive Directory

This `/data_store` directory lists the SECTOR ID and PHYSICAL ELEMENT of the satellite data.

#### Determining Satellite SECTOR ID

```
TYPE: cd /data_store/sat
TYPE: ls > ~/sat-sectors.txt
TYPE: cat ~/sat-sectors.txt
```

- COMP
- GOES-13
- GOES-15
- MISC
- POES
- VIIRS

#### Determining Satellite PHYSICAL ELEMENT in the Data Archive

```
TYPE: cd /data_store/sat/20131029/15/GOES-15
TYPE: ls * | grep -v ":" | grep -P "^\w" > ~/sources.txt
TYPE: sort ~/sources.txt > ~/sorted-sources.txt
TYPE: uniq ~/sorted-sources.txt ~/sat-sources.txt
TYPE: cat ~/sat-sources.txt
```

- 2300Z_3.9_4km_HI-REGIONAL-TIGH04_KNES_1285487.satz.2014032323
- 2300Z_IR_14km_HI-NATIONAL-TIGI02_KNES_1286969.satz.2014032323
- 2300Z_IR_24km_NHEM-COMP-TIGF02_KNES_60496.satz.2014032323
- 2300Z_IR_4km_HI-REGIONAL-TIGH02_KNES_1285598.satz.2014032323
- 2300Z_IR_8km_AK-NATIONAL-TIGB02_KNES_1286898.satz.2014032323
Chapter 5 Ingest of Satellite Imagery

2300Z_IR_8km_AK-REGIONAL-TIGA02_KNES_1285462.satz.2014032323
2300Z_VIS_14km_HI-NATIONAL-TIGI01_KNES_1287012.satz.2014032323
2300Z_VIS_1km_HI-REGIONAL-TIGH01_KNES_1285600.satz.2014032323
2300Z_VIS_24km_NHEM-COMP-TIGF01_KNES_60498.satz.2014032323
2300Z_VIS_8km_AK-NATIONAL-TIGB01_KNES_1286757.satz.2014032323
2300Z_VIS_8km_SUPER-NATIONAL-TIGN01_KNES_60497.satz.2014032323
2300Z_WV_14km_HI-NATIONAL-TIGI05_KNES_1287011.satz.2014032323
2300Z_WV_24km_NHEM-COMP-TIGF05_KNES_60495.satz.2014032323
2300Z_WV_4km_HI-REGIONAL-TIGH05_KNES_1285601.satz.2014032323
2300Z_WV_8km_AK-NATIONAL-TIGB05_KNES_1286968.satz.2014032323
2310Z_13.3_4km_WEST-CONUS-TIGW06_KNES_60501.satz.2014032323
2310Z_3.9_4km_WEST-CONUS-TIGW04_KNES_60500.satz.2014032323
2310Z_IR_4km_WEST-CONUS-TIGW02_KNES_60502.satz.2014032323
2310Z_VIS_1km_WEST-CONUS-TIGW01_KNES_60499.satz.2014032323
2324Z_SOUND-11.03_10km_HI-NATIONAL-TIGI48_KNES_60560.satz.2014032323
2324Z_SOUND-14.06_10km_HI-NATIONAL-TIGI43_KNES_60559.satz.2014032323
2324Z_SOUND-3.98_10km_HI-NATIONAL-TIGI57_KNES_60565.satz.2014032323
2324Z_SOUND-4.45_10km_HI-NATIONAL-TIGI55_KNES_60564.satz.2014032323
2324Z_SOUND-6.51_10km_HI-NATIONAL-TIGI52_KNES_60563.satz.2014032323
2324Z_SOUND-7.02_10km_HI-NATIONAL-TIGI51_KNES_60562.satz.2014032323
2324Z_SOUND-7.43_10km_HI-NATIONAL-TIGI50_KNES_60561.satz.2014032323
2324Z_SOUND-VIS_10km_HI-NATIONAL-TIGI59_KNES_60566.satz.2014032323
2330Z_13.3_4km_WEST-CONUS-TIGW06_KNES_60525.satz.2014032323
2330Z_3.9_4km_HI-REGIONAL-TIGH04_KNES_1297688.satz.2014032323
2330Z_3.9_4km_WEST-CONUS-TIGW04_KNES_60523.satz.2014032323
2330Z_3.9_8km_AK-REGIONAL-TIGA04_KNES_1297497.satz.2014032323
2330Z_IR_14km_HI-NATIONAL-TIGI02_KNES_1298251.satz.2014032323
2330Z_IR_24km_NHEM-COMP-TIGF02_KNES_60529.satz.2014032323
2330Z_IR_4km_HI-REGIONAL-TIGH02_KNES_1297849.satz.2014032323
2330Z_IR_4km_WEST-CONUS-TIGW02_KNES_60524.satz.2014032323
2330Z_IR_8km_AK-NATIONAL-TIGB02_KNES_1298116.satz.2014032323
2330Z_IR_8km_AK-REGIONAL-TIGA02_KNES_1297563.satz.2014032323
2330Z_IR_8km_SUPER-NATIONAL-TIGN02_KNES_60527.satz.2014032323
2330Z_VIS_1km_HI-REGIONAL-TIGH01_KNES_1297783.satz.2014032323
2330Z_VIS_1km_WEST-CONUS-TIGW01_KNES_60521.satz.2014032323
2330Z_VIS_24km_NHEM-COMP-TIGF01_KNES_60531.satz.2014032323
2330Z_VIS_2km_AK-REGIONAL-TIGA01_KNES_1297562.satz.2014032323
2330Z_VIS_8km_AK-NATIONAL-TIGB01_KNES_1298034.satz.2014032323
2330Z_VIS_8km_SUPER-NATIONAL-TIGN01_KNES_60530.satz.2014032323
2330Z_WV_14km_HI-NATIONAL-TIGI05_KNES_1298250.satz.2014032323
2330Z_WV_24km_NHEM-COMP-TIGF05_KNES_60528.satz.2014032323
2330Z_WV_4km_HI-REGIONAL-TIGH05_KNES_1297689.satz.2014032323
2330Z_WV_4km_WEST-CONUS-TIGW05_KNES_60522.satz.2014032323
2330Z_WV_8km_AK-NATIONAL-TIGB05_KNES_1298117.satz.2014032323
2330Z_WV_8km_SUPER-NATIONAL-TIGN05_KNES_60526.satz.2014032323
2340Z_13.3_4km_WEST-CONUS-TIGW06_KNES_60543.satz.2014032323
2340Z_3.9_4km_WEST-CONUS-TIGW04_KNES_60541.satz.2014032323
2340Z_IR_4km_WEST-CONUS-TIGW02_KNES_60542.satz.2014032323
2340Z_VIS_1km_WEST-CONUS-TIGW01_KNES_60540.satz.2014032323
2345Z_13.3_4km_WEST-CONUS-TIGW06_KNES_60549.satz.2014032323
2345Z_3.9_4km_WEST-CONUS-TIGW04_KNES_60546.satz.2014032323
2345Z_IR_14km_HI-NATIONAL-TIGI02_KNES_1303550.satz.2014032323
2345Z_IR_4km_HI-REGIONAL-TIGH02_KNES_1303490.satz.2014032323
2345Z_IR_4km_WEST-CONUS-TIGW02_KNES_60550.satz.2014032323
2345Z_VIS_14km_HI-NATIONAL-TIGI01_KNES_1303577.satz.2014032323
2345Z_VIS_1km_HI-REGIONAL-TIGH01_KNES_1303489.satz.2014032323
2345Z_VIS_1km_WEST-CONUS-TIGW01_KNES_60545.satz.2014032323
2345Z_WV_14km_HI-NATIONAL-TIGI05_KNES_1303576.satz.2014032323
2345Z_WV_4km_WEST-CONUS-TIGW05_KNES_60548.satz.2014032323
# Chapter 6: Ingest of NWSTG Data

## Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1</td>
<td>1</td>
</tr>
<tr>
<td>6.2</td>
<td>1</td>
</tr>
<tr>
<td>6.2.1</td>
<td>4</td>
</tr>
<tr>
<td>6.2.2</td>
<td>5</td>
</tr>
<tr>
<td>6.2.3</td>
<td>5</td>
</tr>
<tr>
<td>6.2.4</td>
<td>8</td>
</tr>
<tr>
<td>6.2.5</td>
<td>9</td>
</tr>
<tr>
<td>6.2.6</td>
<td>9</td>
</tr>
<tr>
<td>6.3</td>
<td>11</td>
</tr>
<tr>
<td>6.3.1</td>
<td>11</td>
</tr>
<tr>
<td>6.3.2</td>
<td>13</td>
</tr>
<tr>
<td>6.3.3</td>
<td>13</td>
</tr>
<tr>
<td>6.4</td>
<td>14</td>
</tr>
<tr>
<td>6.4.1</td>
<td>16</td>
</tr>
<tr>
<td>6.4.2</td>
<td>16</td>
</tr>
<tr>
<td>6.4.3</td>
<td>17</td>
</tr>
<tr>
<td>6.5</td>
<td>18</td>
</tr>
<tr>
<td>6.5.1</td>
<td>18</td>
</tr>
<tr>
<td>6.5.2</td>
<td>20</td>
</tr>
<tr>
<td>6.5.3</td>
<td>21</td>
</tr>
<tr>
<td>6.6</td>
<td>23</td>
</tr>
<tr>
<td>6.6.1</td>
<td>24</td>
</tr>
<tr>
<td>6.6.2</td>
<td>31</td>
</tr>
<tr>
<td>6.6.3</td>
<td>32</td>
</tr>
<tr>
<td>6.7</td>
<td>33</td>
</tr>
<tr>
<td>6.7.1</td>
<td>34</td>
</tr>
<tr>
<td>6.7.2</td>
<td>36</td>
</tr>
<tr>
<td>6.7.3</td>
<td>36</td>
</tr>
<tr>
<td>6.8</td>
<td>37</td>
</tr>
<tr>
<td>6.8.1</td>
<td>39</td>
</tr>
<tr>
<td>6.8.2</td>
<td>39</td>
</tr>
<tr>
<td>6.8.3</td>
<td>39</td>
</tr>
</tbody>
</table>
6.9 convSIGMET Data .................................................................................................................. 40
  6.9.1 Data Archive and File Naming Conventions for convSIGMET Data ....................... 40
  6.9.2 Decoding convSIGMET Data ...................................................................................... 41
  6.9.3 Data Ingest and Logging of ConvSIGMET Products .............................................. 41
6.10 Tracing NWSTG (TAF) Products Using the Sequence Number .................................. 42

List of Exhibits

Exhibit 6.2-1. NWSTG (grib and non-grib) Data Flow ................................................................. 2
Exhibit 6.2.3-1. Acquire and Ingest of NWSTG Products .......................................................... 6
Exhibit 6.3-1. Acquire and Ingest of Lightning Products ......................................................... 12
Exhibit 6.4-1. Acquire and Ingest of METAR Products ........................................................... 15
Exhibit 6.5-1. Acquire and Ingest of MOS Products .............................................................. 19
Exhibit 6.6-1. Acquire and Ingest of BUFR Products ............................................................. 25
Exhibit 6.7-1. Acquire and Ingest of Synoptic Products ......................................................... 35
Exhibit 6.8-1. Acquire and Ingest of SSM/I Products .............................................................. 38

List of Tables

Table 6.6.2-1. BUFR Data: Directories and File Names ............................................................. 31
Table 6.6.2-2. Report Code Definitions ..................................................................................... 32
Table 6.7-1. Data URI Codes for Various Synoptic Observation Types ....................................... 34
6.1 Overview

The main source of nationally distributed data for AWIPS is the Satellite Broadcast Network (SBN). The SBN delivers products over four channels; the product sets for the NWSTG and NWSTG2 channels include the following:

- NCEP and other model outputs
  - GFS, GFS Ensembles, NGM, NAM, MesonAM, NOGAPS, RAP13, RAP40, ECMWF, UKMET, GWW, QPE, FFG, SNOW, ENSEMBLE, GLERL, RTGSST, DGEX, and HPCGuide

- BUFR data
  - NAM Model Soundings, GOES Soundings, POES Soundings, ACARS, QuikSCAT ocean winds, GOES High Density Winds, NAM, GFS, NCWF, ASCAT and SSM/I

- Surface observations
  - METAR, lighting, maritime, hydro

- Upper air observations
  - RAOB, profiler, aircraft

- Other text products
  - TAF, CCF, and other coded and plain language text products

- NCEP Redbook graphics
  - Digitized graphics produced by NCEP.

This chapter describes the methods by which AWIPS receives the following data products:

- Gridded data (Section 6.2)
- Lightning data (Section 6.3)
- METAR data (Section 6.4)
- MOS data (Section 6.5)
- BUFR data (Section 6.6)
- Synoptic observation data (Section 6.7)
- SSM/I data (Section 6.8)
- ConvSIGMET data (Section 6.9)

The chapter concludes with guidance on tracing NWSTG (TAF) products using the sequence number (Section 6.10).

6.2 Gridded Data

The data flow for the NWSTG (grib and non-grib) data is illustrated in Exhibit 6.2-1. As the exhibit shows, the data flow is a 12-step process. Descriptions of each step follow.
Exhibit 6.2-1. NWSTG (grib and non-grib) Data Flow
Chapter 6

Ingest of NWSTG Data

1. NWSTG data (grib and non-grib) are obtained by the LDM noaaportIngester from the SBN. Most gridded model data is produced by the NCEP. This data is transmitted over the NWSTG channel of the SBN in GRIB (Gridded Binary) or GRIB2 format.
   a. The LDM noaaportIngester forwards the product to the LDM Writer instance.
   b. The LDM Writer instance writes the product to the Data Archive. The Data Archive (/data_store) directory is mounted on the hosts off the NAS [NAS2].
   c. The LDM Writer instance uses the EDEX Bridge to post a message containing product information to the QPID Message Broker. The product information is the WMO header and the path to the product file.

2. The EDEX Distribution endpoint on the EDEX (DX) cluster pulls the message from the QPID Message Broker. The WMO header is matched against data distribution mappings contained in the data distribution directory to determine product routing. (The WMO header is included in the message; if it is not, the Distribution service uses the product file name from the message to determine routing.) Note: The files in the data distribution directory are XML files that contain regular expressions that match WMO headers with the appropriate data decoder plug-in.
   a. At EDEX startup, the EDEX Distribution endpoint reads the Data Distribution files from the Data Distribution directory in the Localization Store. It uses the contents of these files to generate the routing table used to determine data routing. The Data Distribution directory is located at /awips2/edex/data/utility/edex_static/base/distribution.
   b. The EDEX Distribution endpoint monitors the Data Distribution directory; when a file in that directory is modified, it re-reads that file and rebuilds its routing table.

3. The EDEX Distribution Service determines the Data Decoder Plug-In registered to handle the data in the product file.

4. The EDEX Distribution Service forwards the message to the appropriate Data Decoder Plug-In via the QPID Message Broker.

5. The EDEX Data Decoder Service on the EDEX (DX) cluster pulls the data type specific message from the QPID Message broker. The Data Decoder Service reads the file pointed to by the message, decodes the data, and determines the appropriate metadata.

6. The decoded data and metadata are sent to the EDEX Persist endpoint.

7. The EDEX Persist endpoint sends the data to PyPIES, and PyPIES writes the data to the EDEX data store in HDF5 format.

8. The EDEX Persist endpoint sends the metadata obtained to the EDEX Index endpoint.

9. The EDEX Index endpoint sends the metadata to the PostgreSQL DBMS.
a. The PostgreSQL DBMS writes the metadata to the database.

10. The EDEX Index endpoint extracts the Data URI for the product from the metadata.

11. (Client notification) The Data URI is forwarded to the EDEX URI Aggregator.
   a. Every 5 seconds the aggregated Data URIs are sent to the JMS Alert Topic.
   b. CAVE monitors the JMS Alert Topic to obtain a list of available Data. CAVE determines if it should render the available data.
   c. CAVE retrieves metadata from the EDEX server via a product query sent to the EDEX thrift endpoint.
   d. CAVE retrieves product data from the EDEX server by accessing the EDEX data store’s HDF5 files and CAVE renders the data, updating its display as appropriate.

12. (Subscription) The Data URI is forwarded to the EDEX Subscription endpoint.
   a. The EDEX Subscription endpoint checks the active subscription list to determine if a subscription has been registered for the product.
      1) If no subscription has been registered for the product, no further action is taken.
   b. The EDEX Subscription endpoint obtains the script information from the database.
   c. The EDEX Subscription endpoint obtains a micro-engine to execute the script.
      1) The micro-engine executes the subscription script for the product.

6.2.1 Data Archive and File Naming Conventions

Most gridded model data is produced by the NCEP. This data is transmitted over the NWSTG channel of the SBN in GRIB (Gridded Binary) or GRIB2 format. They are stored in data archive.

The raw grib files are stored in the Data Archive (/data_store/grib or /data_store/grib2). The physical directory (/data_store) is on the NAS. All the hosts mount the volume off NAS.

In addition, when new products are being tested, they are stored under /data_store/experimental/grib.

Data Storage (also known as Data Archive) is not standard across all grids anymore. It is generally stored at /data_store/grib/<YYYYMMDD>/><HH>/GRID_NAME/GRID_TYPE e.g., /data_store/grib/20130211/12/NWS_161/GRID255/2300Z_F001_APCP-ZETA98_KFWR_111214_63882903.grib.2013021220

and
Chapter 6 Ingest of NWSTG Data

/data_store/grib2/<YYYYMMDD>/<HH>/GRID_NAME/GRID_TYPE

e.g., /data_store/grib/20130211/17/RUC2/GRID130/ 1700Z_F018_WXTZ-LMDN98_KWBG_121700_38969009.grib2.2013021218

LDM server runs only on the SBN CP. It ingests everything from the SBN.

LDM behavior is controlled by the pqact.conf file, and the patterns are matched against pqact.conf. LDM writes the product to data_store and posts a QPID message.

- pqact.conf controls how data sent to an instance of LDM is handled. This includes the data types to be handled and the location of the raw data archive.

**NOTE:** Modifying pqact.conf will have a severe impact on the data flow.

6.2.2 Decoding the Gri-based Data

Data decoding operations are performed on the EDEX (DX3/4) servers. Both EDEX servers share the load in the ingest process, which includes decoding the raw data, writing the decoded data into the Data Store, writing metadata to the database, and broadcasting a notification that the newly ingested data is available.

The processed data is stored in /awips2/edex/data/hdf5 in the hdf5 format on the dx2f server. The processed grib data is under /awips2/edex/data/hdf5/grid.

a) The basic directory structure is /awips2/edex/data/hdf5/grid/{GRID_TYPE}/{model_type}.

b) Within each model, a separate HDF5 file is maintained for each forecast hour of a model run. The naming convention for the HDF5 files is {model}-yyyy-mm-dd-hh-FH-{forecasthour}.h5. For example, the GFS212 1200Z 0 forecast hour from Feb. 12, 2013 is in a file named GFS212-2013-02-12-12-FH-000.h5.

c) The internal structure of the HDF5 file reflects the Data URI, which is part of the metadata stored in the PostgreSQL database.

6.2.3 Data Ingest and Logging of Gri-based Data

As illustrated in Exhibit 6.2.3-1, all data ingest operations are logged in the EDEX logs on the EDEX (DX3/4) server. In a standard installation, the EDEX logs are located in /awips2/edex/logs. This directory contains several log files; the files starting with “edex-” are the EDEX logs.

a) EDEX logs roll over daily and are date stamped. The naming convention for EDEX logs is edex-{instance name}-yyyymmdd.log.

b) For the grib decoder, the instance name is “ingestGrib”, so the GRIB ingest logs for May 19, 2013 are named edex-ingestGrib-20130519.log. Note that there should be one log on both of the EDEX servers (DX3/4).
Chapter 6

Ingest of NWSTG Data

Exhibit 6.2.3-1. Acquire and Ingest of NWSTG Products
The log file is a plain ASCII text file and can be viewed using any text viewing utility such as *more*, *less*, *view*, or *tail*.

Open a terminal session to the server and enter the following commands:

**On DX3/DX4**

```bash
TYPE: cd /awips2/edex/logs
TYPE: date +%F
     2013-05-19
TYPE: ls edex*20130519*
edex-ingest-20130519.log
edex-ingestDat-20130519.log
edex-ingest-gen_areal_ffg-20130519.log
edex-ingest-gen_areal_gpe-20130519.log
edex-ingestGrib-20130519.log
edex-ingest-purge-20130519.log
edex-ingest-radar-20130519.log
edex-ingest-satellite-20130519.log
edex-ingest-shef-20130519.log
edex-ingest-shef-performance-20130519.log
edex-ingest-smartInit-20130519.log
edex-ingest-text-20130519.log
edex-ingest-trigger-20130519.log
edex-ingest-unrecognized-files-20130519.log
edex-request-20130519.log
edex-request-productSrvRequest-20130519.log
edex-request-thriftSrv-20130519.log
edex-ingest-archive-20130519.log
edex-ingestGrib-performance-20130519.log
edex-ingest-performance-20130519.log
edex-request-performance-20130519.log
edex-ingest-gen_areal_gpe-20130519.log
edex-request-hibernate-20130519.log
edex-ingest-hibernate-20130519.log
edex-ingestGrib-hibernate-20130519.log

TYPE: tail -f edex-ingestGrib-20130519.log
```

After ingest of each data file is complete, the ingest endpoint prints a single line to the EDEX system log. The format of the log entry is

```
INFO <<date-time>> [thread] Ingest: <<type>>:: <<file name>> <<statistics>>
```
Where,

<<date-time>> is the date-time stamp (format yyyy-mm-dd hh:mm:ss,ttt)
[thread] identifies the Camel thread that ran the decoder
<<type>> identifies the data type, e.g. grib1, grib2 or radar etc.
<<file name>> is the path to the file that was processed
<<statistics>> lists the ingest statistics for the file. There are two statistics provided; process time and latency. (Please refer to the note at the end of this section for more information.)

A log entry for grib is

```
INFO 2013-05-19 00:01:07,128 [gribThreadPool-1]
Ingest: EDEX: Ingest - grib1::
processed In: 0.0390 (sec) Latency: 0.6630 (sec)
```

**NOTE:** Latency is the elapsed time between when the AWIPS II system first contacts a data file and when the data has been ingested, decoded, and is ready for retrieval by CAVE or another client application. The “first contact” depends on the data type. For most data types, latency measurement starts when the product has been downloaded by the LDM Writer on cp1f. This latency start time is part of the message generated by the LDM’s EDEX Bridge and sent to QPID (on the CP1/2 cluster) for processing by EDEX. For Radar products obtained from the ORPG via the AWIPS II Radar Server, latency measurement starts when the product is downloaded by the Radar Server on the DX1/2 cluster. In the case of ORPG products, the latency start time is part of the message sent by the Radar Server to QPID. Note that this latency measure is strictly internal for the AWIPS II data ingest subsystem. This information is logged into the EDEX process logs for every file that is ingested. Additional latency can occur, for example, due to communication issues between the Radar Server and the ORPG. Other sources of latency are possible. For example (this is a hypothetical example), if a client application queries a server each minute for available products, there is an average latency of 30 seconds in that query. To quantify, or even to identify, all possible sources of latency in a system is beyond the scope of this discussion.

### 6.2.4 Data Archive and File Naming Conventions for Other Gridded Data

The remaining gridded data (other than GRIB1 and GRIB2) received over the SBN are stored in the following directory.

```
/data_store/grib/grib/GRID_NAME/GRID_TYPE/YYYYMMDD/
```
6.2.5 Decoding Other Gridded Data

Data decoding operations for the Gridded Data (other than GRIB1 and GRIB2) are performed on the EDEX (DX3/4) servers. Both EDEX servers share the load in the ingest process, which includes decoding the raw data, writing the decoded data into the Data Store, writing metadata to the database, and broadcasting a notification that the newly ingested data is available.

The processed data is stored in /awips2/edex/data/hdf5 in the hdf5 format on the dx2f server. The processed Gridded Data (other than GRIB1 and GRIB2) are under /awips2/edex/data/hdf5/grid.

The basic directory structure is

/awips2/edex/data/hdf5/grid/<GRID_TYPE>/<model_type>

Within each model, a separate HDF5 file is maintained for each model run. The naming convention for the HDF5 files is *yyyy-mm-dd-hhmm*.h5.

/awips2/edex/data/hdf5/grid/AVN211/BL/AVN211-2013-02-12-06-FH-240.h5

/awips2/edex/data/hdf5/grid/AVN211/Dflt/AVN211-2013-02-12-06-FH-000.h5

/awips2/edex/data/hdf5/grid/AVN225/SFC/AVN225-2013-02-12-06-FH-120.h5

/awips2/edex/data/hdf5/grid/AVN211/Dflt/AVN211-2013-02-12-06-FH-00/awips2/edex/data/hdf5/grid/GWW233/SFC0.h5

The internal structure of the HDF5 file reflects the Data URI, which is part of the metadata stored in the PostgreSQL database.

6.2.6 Data Ingest and Logging of Other Gridded Products

All data ingest operations are logged in the EDEX logs on the EDEX (DX3/4) server. In a standard installation, the EDEX logs are located in /awips2/edex/logs. This directory contains several log files; the files starting “edex-” are the EDEX logs.

The log file is a plain ASCII text file and can be viewed using any text viewing utility such as more, less, view, or tail.

Open a terminal session to the server and enter the following commands:

On DX3/DX4:

[root@dx3-tbdw ~]# cd /awips2/edex/logs

[root@dx3-tbdw logs]# date +%F
**TYPE:** `date +%F`  

**2013-05-19**

**TYPE:** `ls edex*20130519*`

```
edex-ingest-20130519.log  
edex-ingestDat-20130519.log  
edex-ingest-gen_areal_ffg-20130519.log  
edex-ingest-gen_areal_qpe-20130519.log  
edex-ingestGrib-20130519.log  
edex-ingest-purge-20130519.log  
edex-ingest-radar-20130519.log  
edex-ingest-satellite-20130519.log  
edex-ingest-shef-20130519.log  
edex-ingest-shef-performance-20130519.log  
edex-ingest-smartInit-20130519.log  
edex-ingest-text-20130519.log  
edex-ingest-trigger-20130519.log  
edex-ingest-unrecognized-files-20130519.log  
edex-request-20130519.log  
edex-request-productSrvRequest-20130519.log  
edex-request-thriftSrv-20130519.log  
edex-ingest-archive-20130519.log  
edex-ingest-activeTableChange-20130519.log  
edex-ingestGrib-performance-20130519.log  
edex-ingest-performance-20130519.log  
edex-request-performance-20130519.log  
edex-ingest-gen_areal_qpe-20130519.log  
edex-request-hibernate-20130519.log  
edex-ingest-hibernate-20130519.log  
edex-ingestGrib-hibernate-20130519.log
```

```
[root@dx3-tbdw logs]# tail -f edex-ingestGrib-20130519.log
```

After ingest of each data file is complete, the ingest endpoint prints a single line to the EDEX system log. The format of the log entry is

**INFO <<date-time>> [thread] Ingest: <<type>>: <<file name>> <<statistics>>**

The product sequence number is to the left of the .grb (_<seq>_.grb).

The log entry:
6.3 Lightning Data

Lightning data is collected by the National Lightning Detection Network (NLDN) and received from the SBN. This data set contains the location that a lightning strike was detected (given by latitude and longitude) and the time that the flash occurred, recorded to the nearest second. The lightning flash’s signal strength is recorded in kiloamperes. Also included is the multiplicity and polarity (either positive or negative) of each flash.

**NOTE:** For more information on the proprietary nature of the lightning data and redistribution restrictions, please refer to: http://www.nco.ncep.noaa.gov/sib/restricted_data/restricted_data_pmb/lightning/

See Exhibit 6.3-1 for the flow of the acquisition and ingest of lightning products.

6.3.1 Data Archive and File Naming Conventions for Lightning Data

The raw lightning files received over the SBN are stored in the Data Archive (/data_store/binlightning). The physical directory (/data_store) is on the NAS. All the hosts mount the volume off NAS.

The basic directory structure is /data_store/binlightning/<YYYYMMDD>/<HH>

e.g.,
/data_store/binlightning/20130212/20/SFUS41_KWBC_122058_288873682.nldn.20130212202
Chapter 6

Ingest of NWSTG Data

Config Files:

Log Files:

Data Files:

Databases:

Exhibit 6.3-1. Acquire and Ingest of Lightning Products
6.3.2 Decoding Lightning Data

Data decoding operations are performed on the EDEX (DX3/4) servers. Both EDEX servers share the load in the ingest process, which includes decoding the raw data, writing the decoded data into the Data Store, writing metadata to the database, and broadcasting a notification that the newly ingested data is available.

The processed data files are stored in /awips2/edex/data/hdf5 in the hdf5 format on the dx2f server. The processed lightning data files are stored under /awips2/edex/data/hdf5/binlightning

The basic directory structure is:

```
/awips2/edex/data/hdf5/binlightning/binlightning-<yyyy>-<mm>-<dd>-<hh>.h5
```

[root@dx2-tbdw binlightning] # ls /awips2/edex/data/hdf5/binlightning/

```
binlightning-2013-02-13-11.h5
binlightning-2013-02-13-12.h5
```

6.3.3 Data Ingest and Logging of Lightning Products

All data ingest operations are logged in the EDEX logs on the EDEX (DX3/4) server. In a standard installation, the EDEX logs are located in /awips2/edex/logs. This directory contains several log files; the files starting “edex-” are the EDEX logs.

The log file is a plain ASCII text file and can be viewed using any text viewing utility such as `more`, `less`, `view`, or `tail`.

To get a general view of processing, open a terminal session to the server and enter the following commands:

**On DX3/DX4:**

```
TYPE: cd /awips2/edex/logs

TYPE: date +%F
2013-02-13

TYPE: ls edex*20130213*
edex-ingest-20130213.log
edex-ingestDat-20130213.log
edex-ingest-gen_areal_ffg-20130213.log
edex-ingest-gen_areal_qpe-20130213.log
edex-ingestGrib-20130213.log
edex-ingest-purge-20130213.log
edex-ingest-radar-20130213.log
```
After ingest of each data file is complete, the ingest endpoint prints a single line to the EDEX system log. The format of the log entry is

```
INFO  <<date-time>> [thread] Ingest: <<type>>:: <<file name>> <<statistics>>
```

A log entry will be as follows.

```
INFO  2013-02-13 01:00:00,036 [genericThreadPool-12] MpeLightningSrv: retrieved datasets for lightning file /binlightning/binlightning-2013-02-13-00.h5
INFO  2013-02-13 00:39:14,049 [Camel (camel) thread #2 - timer://jmsPooledResourceCheck] JmsPooledProducer: EDEX - Closing producer Ingest.binlightning
```

### 6.4 METAR Data

METAR data is received over the NWSTG channel of the SBN. The raw data arrive in text format as one singular report or as a collective report that contains data from several stations.

See Exhibit 6.4-1 for the flow of the acquisition and ingest of METAR products.
Chapter 6

Ingest of NWSTG Data

Config Files:

Log Files:

Data Files:

Databases:

Exhibit 6.4-1. Acquire and Ingest of METAR Products

-- Hard copies uncontrolled. Verify effective date prior to use. --
6.4.1 Data Archive and File Naming Conventions for Metar Data

METAR data received from the SBN is written to the following directory:

/data_store/metar/

Under nearly every plug-in folder is a set of directories representing the various hours in which these products were issued. For metar, under /data_store/text, you will find folders with the day of the month.

[root@dx3-tbdw data_store]# ls /data_store/metar/<YYYYMMDD>

Under the <YYYYMMDD> you will find folders ranging from 00 through 23, which represent the hour.

00 02 04 06 08 10 12 14 16 18 20 22
01 03 05 07 09 11 13 15 17 19 21 23

If you change the directory to /data_store/metar/20130213/23, you will find all observations issued between 23z and 24z.

Typical entries are:

[root@dx3-tbdw data_store]# ls /data_store/metar/20130213/23
SPZZ40_KAWN_202333_129590324.2013021323
SAGD31_TGPY_202300_129515466.2013021323
SAUS44_KCRP_202336_129596616.2013021323
SPZZ40_KAWN_202338_129599847.2013021323

The first two characters denote an observation type (SA = regular observation; SP = special observation). The third and fourth characters denote the country (US = United States; MX = Mexico; CN = Canada).

In the first entry (SPZZ40_KAWN), 301249 denotes the day/hour/minute from the WMO header, and 124162499 represents the unique sequence number assigned by the NWSTG at transmission over the SBN. It is the number used for automatic product reships. 2013021313 represents the date and the hour of the data.

**NOTE:** The KWBC files are often, but not always, collectives.

6.4.2 Decoding Metar Data

METAR data is decoded by the obs decoder, which is part of the EDEX Ingest process running on the DX3/4 cluster. Metadata for the METAR data is stored in the awips schema of the metadata database. The processed metar data files are stored in /awips2/edex/data/hdf5/metar in the hdf5 format on the dx2f server.
6.4.3 Data Ingest and Logging of Metar Products

All data ingest operations are logged in the EDEX logs on the EDEX (DX3/4) server. In a standard installation, the EDEX logs are located in /awips2/edex/logs. This directory contains several log files; the files starting “edex-” are the EDEX logs.

The log file is a plain ASCII text file and can be viewed using any text viewing utility such as more, less, view, or tail.

Open a terminal session to the server and enter the following commands:

On DX3/DX4

```
TYPE: cd /awips2/edex/logs
TYPE: date +%F
2013-02-13
TYPE: ls edex*20130213*
edex-ingest-20130213.log
edex-ingestDat-20130213.log
edex-ingest-gen_areal_ffg-20130213.log
edex-ingest-gen_areal_qpe-20130213.log
edex-ingestGrib-20130213.log
edex-ingest-purge-20130213.log
edex-ingest-radar-20130213.log
edex-ingest-satellite-20130213.log
edex-ingest-shef-20130213.log
edex-ingest-shef-performance-20130213.log
edex-ingest-smartInit-20130213.log
edex-ingest-text-20130213.log
edex-ingest-trigger-20130213.log
edex-ingest-unrecognized-files-20130213.log
edex-request-20130213.log
edex-request-productSrvRequest-20130213.log
edex-request-thriftSrv-20130213.log
edex-ingest-archive-20130213.log
edex-ingestGrib-performance-20130213.log
edex-ingest-performance-20130213.log
edex-request-performance-20130213.log
edex-ingest-gen_areal_qpe-20130213.log
edex-request-hibernate-20130213.log
edex-ingest-hibernate-20130213.log
edex-ingestGrib-hibernate-20130213.log
```
TYPE: \texttt{tail -f edex-ingest-20130213.log}

After ingest of each data file is complete, the ingest endpoint prints a single line to the EDEX system log. The format of the log entry is

\begin{verbatim}
INFO 2013-02-13 00:00:01,220 [genericThreadPool-63] Ingest: EDEX: Ingest - obs::
/data_store/metar/20130213/23/SAAK41_KNKA_192359_126391541.2013021300 processed in: 0.2740 (sec) Latency: 0.2810 (sec)
\end{verbatim}

\begin{verbatim}
INFO 2013-02-13 00:00:09,471 [genericThreadPool-63] Ingest: EDEX: Ingest - obs::
/data_store/metar/20130213/00/SAUS17_KAWN_200000_126391804.2013021300 processed in: 0.5000 (sec) Latency: 0.5060 (sec)
\end{verbatim}

6.5 MOS Data

The MOS data is received over the SBN’s NWSTG channel.

See Exhibit 6.5-1 for the flow of the acquisition and ingest of MOS products.

6.5.1 Data Archive and File Naming Conventions for MOS Data

The MOS data received over the SBN are stored in the following directory.

\texttt{/data\_store/bufrmos/{YYYYMMDD}}

Under \texttt{<YYYYMMDD>} you will find folders ranging from 00 through 23.

\texttt{[root@dx1-tbdw bufrmos]## ls /data\_store/bufrmos/{YYYYMMDD}}

\begin{verbatim}
  00  02  04  06  08  10  12  14  16  18  20  22
  01  03  05  07  09  11  13  15  17  19  21  23
\end{verbatim}

If you change to \texttt{/data\_store/bufrmos/20130213/00}, you will find all the guidance issued between 00z and 01z.

Here is an example with the GFSLAMP, GFS, ETA, HPC, NGM, and MRF files under the MOS data.

\begin{verbatim}
JSMF10_KWNO_300000_143710496.bufr.2013021300 (GFSLAMP)
JSML36_KWNO_300000_144004297.bufr.2013021300 (GFS)
JSML12_KWNO_061200_3526466.bufr.2013021300 (ETA)
JSMT72_KWNH_060000_3461881.bufr.2013021300 (HPC)
JSMT42_KWNH_060000_3875681.bufr.2013021300 (NGM)
JSMT62_KWNO_060000_3784680.bufr.2013021300 (MRF)
\end{verbatim}
Chapter 6

Exhibit 6.5-1. Acquire and Ingest of MOS Products
If you change to /data_store/bufrmos/20130213/00, you will find all the guidance issued between 00z and 01z.

Here is an example with the GFSLAMP, GFS, ETA, HPC, NGM, and MRF files under the MOS data.

JSMF10_KWNO_300000_143710496.bufr.2013021300 (GFSLAMP)
JSML36_KWNO_300000_144004297.bufr.2013021300 (GFS)
JSML12_KWNO_061200_3526466.bufr.2013021300 (ETA)
JSMT72_KWNH_060000_3461881.bufr.2013021300 (HPC)
JSMT42_KWNH_060000_3875681.bufr.2013021300 (NGM)
JSMT62_KWN0_060000_3784680.bufr.2013021300 (MRF)

6.5.2 Decoding BufrMOS Data

Data decoding operations are performed on the EDEX (DX3/4) servers. Both EDEX servers share the load in the ingest process, which includes decoding the raw data, writing the decoded data into the Data Store, writing metadata to the database, and broadcasting a notification that the newly ingested data is available.

The processed data files are stored in /awips2/edex/data/hdf5 in the hdf5 format on the dx2f server. The processed MOS data files are stored under their respective types as follows.

All ingested bufrmosLAMP data is under
/awips2/edex/data/hdf5/bufrmosLAMP/<model>-<yyy>-<mm>-<dd>-<hh>.h5
E.g.: /awips2/edex/data/hdf5/bufrmosLAMP/bufrmos-LAMP-2013-02-13-11.h5

All ingested bufrmosGFS data is under /awips2/edex/data/hdf5/bufrmosGFS/<model>-<yyy>-<mm>-<dd>-<hh>.h5
E.g.: /awips2/edex/data/hdf5/bufrmosGFS/bufrmos-GFS-2013-02-13-11.h5

All ingested bufrmosETA data is under /awips2/edex/data/hdf5/bufrmosETA/<model>-<yyy>-<mm>-<dd>-<hh>.h5
E.g.: /awips2/edex/data/hdf5/bufrmosETA/bufrmos-ETA-2013-02-13-11.h5

All ingested bufrmosHPC data is under /awips2/edex/data/hdf5/bufrmosHPC/<model>-<yyy>-<mm>-<dd>-<hh>.h5
E.g.: /awips2/edex/data/hdf5/bufrmosHPC/bufrmos-HPC-2013-02-13-11.h5

All ingested bufrmosMRF data is under /awips2/edex/data/hdf5/bufrmosMRF/<model>-<yyy>-<mm>-<dd>-<hh>.h5
E.g.: /awips2/edex/data/hdf5/bufrmosMRF/bufrmos-MRF-2013-02-13-00.h5
All ingested bufrmosAVN data is under /awips2/edex/data/hdf5/bufrmosAVN/<model>-<yyyy>-<mm>-<dd>-<hh>.h5

E.g.: /awips2/edex/data/hdf5/bufrmosNGM/bufrmos-AVN-2013-02-13-11.h5

6.5.3 Data Ingest and Logging of BufrMOS Products

All data ingest operations are logged in the EDEX logs on the EDEX (DX3/4) server. In a standard installation, the EDEX logs are located in /awips2/edex/logs. This directory contains several log files; the files starting “edex-” are the EDEX logs.

Open a terminal session to the server and enter the following commands:

On DX3:

[root@dx3-tbdw ~]# cd /awips2/edex/logs
[root@dx3-tbdw logs]# date +%F
2013-02-13
[root@dx3-tbdw logs]# ls edex-*20130213*
edex-ingest-20130213.log
edex-ingestDat-20130213.log
edex-ingest-gen_areal_ffg-20130213.log
edex-ingest-gen_areal_qpe-20130213.log
edex-ingestGrib-20130213.log
edex-ingest-purge-20130213.log
edex-ingest-radar-20130213.log
edex-ingest-satellite-20130213.log
edex-ingest-shef-20130213.log
edex-ingest-shef-performance-20130213.log
edex-ingest-smartInit-20130213.log
edex-ingest-text-20130213.log
edex-ingest-trigger-20130213.log
edex-ingest-unrecognized-files-20130213.log
edex-request-20130213.log
edex-request-productSrvRequest-20130213.log
edex-request-thriftSrv-20130213.log
edex-ingest-archive-20130213.log
edex-ingest-activeTableChange-20130213.log
edex-ingestGrib-performance-20130213.log
edex-ingest-performance-20130213.log
edex-request-performance-20130213.log
edex-ingest-gen_areal_qpe-20130213.log
edex-request-hibernate-20130213.log
edex-ingest-hibernate-20130213.log
Chapter 6

Ingest of NWSTG Data

```
edex-ingestGrib-hibernate-20130213.log

[root@dx3-tbdw logs]
# grep bufrmos edex-ingest-20130213.log | tail -1
```


```
[root@dx3-tbdw logs]
# ssh dx4 "grep bufrmos /awips2/edex/logs/edex-ingest-20130213.log | tail -1"
```


The following are examples of looking for a specific bufrmos types by using the AWIPS1 acq_patterns.txt ingest pattern threads. The grep examples should be performed on both dx3 and dx4.

For GFS extended MRF data:

```
^JSMT6[1-6].*
^JSMT7[1-6].KWNO.*
```

```
[root@dx3-tbdw logs]
# grep "JSMT6[1-6]" edex-ingest-20130213.log
```


For HPC data:

```
^JSMT7[1-6].KWNH.*
```

```
[root@dx3-tbdw logs]
# grep "JSMT7[1-6]_KWNH" edex-ingest-20130213.log
```


For GFS data:

```
^JSML3[1-6].*
```

```
[root@dx3-tbdw logs]# grep "JSML3[1-6]" edex-ingest-20130213.log


```

For ETA data:

```
^JSML1[1-6].*
```

```
[root@dx3-tbdw logs]# grep "JSML1[1-6]" edex-ingest-20130213.log


INFO 2013-02-13 03:16:58,055 [genericThreadPool-81] Ingest: EDEX: Ingest - bufrmos:: /data_store/bufrmos/20130213/00/JSML14_KWNO_130000_289523915.bufr .2013021303 processed in: 40.6400 (sec) Latency: 76.4190 (sec)
```

For GFSLAMP data:

```
^JSMF1[1-6].KWNO.*
```

```
[root@dx3-tbdw logs]# grep "JSMF1[1-6].KWNO" edex-ingest-20130213.log


```

### 6.6 Decoding BUFR Data

Data ingested in BUFR format includes model soundings, GOES soundings, POES soundings, ACARS data, GOES High Density Wind (hdw) data, QuikSCAT ocean wind data, mtsat High Density Wind (mtsathdw), and ncwf.
See Exhibit 6.6-1 for the flow of the acquisition and ingest of BUFR products.

6.6.1 Data Archive and File Naming Conventions for the BUFR Data

The Bufr data (miscellaneous point data) received over the SBN are stored in the following directories.

For model soundings:
/data_store/modelsounding/{YYYYMMDD}

[root@dx3-tbdw modelsounding]# ls /data_store/modelsounding/20130213
02 04 08 10 14 16 20 21 22

[root@dx3-tbdw modelsounding]# ls /data_store/modelsounding/20130213/02
JUSB43_KWNO_130200_289500745.bufr.2013021302
JUSX49_KWNO_130200_289523811.bufr.2013021302
JUSB43_KWNO_130200_289500746.bufr.2013021302
JUSX49_KWNO_130200_289523813.bufr.2013021302
JUSB43_KWNO_130200_289500747.bufr.2013021302
JUSX49_KWNO_130200_289523821.bufr.2013021302
JUSB43_KWNO_130200_289500748.bufr.2013021302
JUSX49_KWNO_130200_289523871.bufr.2013021302

For GOES soundings:
data_store/goessndg/{YYYYMMDD}

[root@dx3-tbdw data_store]# ls /data_store/goessounding/20130213
00 02 04 06 08 10 12 14 16 18 20 22
01 03 05 07 09 11 13 15 17 19 21 23

[root@dx3-tbdw data_store]# ls /data_store/goessounding/20130213/08
JUTX04_KNES_130824_290073061.bufr.2013021308
JUTX07_KNES_130806_290045707.bufr.2013021308
JUTX04_KNES_130824_290073066.bufr.2013021308
JUTX07_KNES_130806_290045711.bufr.2013021308
JUTX04_KNES_130824_290073069.bufr.2013021308
JUTX07_KNES_130806_290045712.bufr.2013021308
Chapter 6

Ingest of NWSTG Data

Exhibit 6.6-1. Acquire and Ingest of BUFR Products
For POES soundings:
/data_store/poessounding/{YYYYMMDD}

[root@dx3-tbdw data_store]# ls /data_store/poessounding/20130213
  00  02  04  06  08  10  12  14  16  18  20  22
  01  03  05  07  09  11  13  15  17  19  21  23

[root@dx3-tbdw data_store]# ls /data_store/poessounding/20130213/06
IUTX05_KNES_130641_289865564.bufr.2013021306
IUTX09_KNES_130628_289865519.bufr.2013021306
IUTX05_KNES_130642_289865633.bufr.2013021306
IUTX09_KNES_130648_289921154.bufr.2013021306
IUTX05_KNES_130643_289865705.bufr.2013021306
IUTX09_KNES_130649_289921063.bufr.2013021306
IUTX05_KNES_130644_289865573.bufr.2013021306
IUTX09_KNES_130650_289921029.bufr.2013021306
IUTX05_KNES_130645_289865527.bufr.2013021306

For ACARS data:
/data_store/acars/acars_encrypted
/data_store/acars/acars_decrypted
/data_store/acars/acars_raw_decrypted

[root@dx3-tbdw data_store]# ls /data_store/acars/acars_encrypted/20140113
  01  02  03  04  05

[root@dx3-tbdw data_store]# ls /data_store/acars/acars_encrypted/20140113/05
IUAX02_KARP_130526_46666697.acars.2014011305
IUAX02_KARP_130526_46666716.acars.2014011305
IUAX02_KARP_130531_46668876.acars.2014011305
IUAX02_KARP_130531_46668933.acars.2014011305
IUAX02_KARP_130536_46670615.acars.2014011305
IUAX02_KARP_130536_46670653.acars.2014011305
IUAX02_KARP_130541_46672243.acars.2014011305
IUAX02_KARP_130541_46672293.acars.2014011305
IUAX02_KARP_130546_46673828.acars.2014011305
IUAX02_KARP_130546_46673850.acars.2014011305
IUAX02_KARP_130551_46677562.acars.2014011305
Ingest of NWSTG Data

[root@dx3-tbdw data_store]# ls
/data_store/acars/acars_decrypted/20140113/
01 01 02 03 04 05

[root@dx3-tbdw data_store]# ls
/data_store/acars/acars_decrypted/20140113/05
IUAX02_KARP_130516_46662957.acars.2014011305.izZPrn
IUAX02_KARP_130521_46664772.acars.2014011305.ML8LOC
IUAX02_KARP_130521_46664829.acars.2014011305.a5Y89D
IUAX02_KARP_130521_46664875.acars.2014011305.C5dSmC
IUAX02_KARP_130526_46666652.acars.2014011305.NxgLQS
IUAX02_KARP_130526_46666697.acars.2014011305.stY2jT
IUAX02_KARP_130526_46666716.acars.2014011305.oFpNMT
IUAX02_KARP_130531_46668933.acars.2014011305.4rrtKA
IUAX02_KARP_130536_46670615.acars.2014011305.kRwZIQ
IUAX02_KARP_130536_46670653.acars.2014011305.1652qS
IUAX02_KARP_130541_46672243.acars.2014011305.h3puje
IUAX02_KARP_130541_46672293.acars.2014011305.BIVWFe
IUAX02_KARP_130546_46673828.acars.2014011305.lHKlfn
IUAX02_KARP_130546_46673850.acars.2014011305.C5PQYl
IUAX02_KARP_130551_46677562.acars.2014011305.zCA8a9

[root@dx3-tbdw data_store]# ls
/data_store/acars/acars_raw_decrypted/20140113/
01 01 02 03 04 05

[root@dx3-tbdw data_store]# ls
/data_store/acars/acars_raw_decrypted/20140113/05
IUAE01_EGRR_130537_46670333.bufr.2014011305
IUAE01_EGRR_130542_46671769.bufr.2014011305
IUAE01_EGRR_130542_46671770.bufr.2014011305
IUAE01_EGRR_130549_46673874.bufr.2014011305
IUAE01_EGRR_130566_46663561.bufr.2014011305
IUAH01_EGRR_130526_46665572.bufr.2014011305
IUAH01_EGRR_130526_46665589.bufr.2014011305
IUAH01_EGRR_130532_46668012.bufr.2014011305
IUAH01_EGRR_130539_46670788.bufr.2014011305
IUAH01_EGRR_130540_46671477.bufr.2014011305
IUAH01_EGRR_130545_46672729.bufr.2014011305
IUAS01_NZKL_130500_46658815.bufr.2014011305
IUAS01_NZKL_130500_46658817.bufr.2014011305
IUAS01_NZKL_130500_46658819.bufr.2014011305
IUAS01_NZKL_130500_46658823.bufr.2014011305
IUAS01_NZKL_130500_46658825.bufr.2014011305
IUAS01_NZKL_130500_46658827.bufr.2014011305
IUAS01_NZKL_130500_46658829.bufr.2014011305
IUAS02_NZKL_130500_46660085.bufr.2014011305
IUAS02_NZKL_130500_46660087.bufr.2014011305

NOTE: In ACARS, EGRR refers to UK, British Met Office, and PANC refers to NWS/NCEP Anchorage Alaska.

For GOES High Density Wind (HDW) data:
data_store/bufrhdw/{YYYYMMDD}

[root@dx3-tbdw data_store]# ls /data_store/bufrhdw/20131030
00  02  05  07  09  12  14  17  19  21
01  03  06  08  11  13  15  18  20  23

[root@dx3-tbdw data_store]# ls /data_store/bufrhdw/20131013/15
JACX11_KNES_301500_57916828.bufr.2013103015
JHCX11_KNES_301521_57931609.bufr.2013103015
JCCX91_KNES_301550_57967612.bufr.2013103015
JECX11_KNES_301500_57916999.bufr.2013103015
JRCX71_KNES_130611_289863561.bufr.2013021306

For ncwf (national convective weather forecast) data:
data_store/bufrncwf/{YYYYMMDD}

[root@dx3-tbdw data_store]# ls /data_store/bufrncwf/20130213
00  02  04  06  08  10  12  14  16  18  20  22
01  03  05  07  09  11  13  15  17  19  21  23

[root@dx3-tbdw data_store]# ls /data_store/bufrncwf/20130213/02
JSAT98_KKCI_130202_289400328.bufr.2013021302
JSAT98_KKCI_130207_289400328.bufr.2013021302
JSAT98_KKCI_130212_289416494.bufr.2013021302
JSAT98_KKCI_130217_289424875.bufr.2013021302
JSAT98_KKCI_130222_289433193.bufr.2013021302
JSAT98_KKCI_130227_289443476.bufr.2013021302

For bufrsigwx data:

data_store/bufrsigwx/{YYYYMMDD}

[root@dx3-tbdw data_store]# ls /data_store/bufrsigwx/20130213

00 06 12 18

[root@dx3-tbdw data_store]# ls /data_store/bufrsigwx/20130213/12

JUBE99_KKCI_130600_140957852.bufr.2013021312
JUBE99_KKCI_130600_140957986.bufr.2013021312
JUCE00_KKCI_130600_140957855.bufr.2013021312
JUCE00_KKCI_130600_140957970.bufr.2013021312
JUFE00_KKCI_130600_140957882.bufr.2013021312
JUFE00_KKCI_130600_140957931.bufr.2013021312
JUJE00_KKCI_130600_140955935.bufr.2013021312
JUJE00_KKCI_130600_140955957.bufr.2013021312
JUME00_KKCI_130600_140955948.bufr.2013021312
JUME00_KKCI_130600_140955972.bufr.2013021312
JUNE00_KKCI_130600_140955947.bufr.2013021312
JUNE00_KKCI_130600_140955966.bufr.2013021312
JUOE00_KKCI_130600_140955932.bufr.2013021312
JUOE00_KKCI_130600_140955970.bufr.2013021312
JUTE00_KKCI_130600_140955997.bufr.2013021312
JUTE00_KKCI_130600_140956102.bufr.2013021312
JUTE97_KKCI_130600_140957854.bufr.2013021312
JUTE97_KKCI_130600_140957978.bufr.2013021312
JUVE00_KKCI_130600_140957887.bufr.2013021312
JUVE00_KKCI_130600_140957948.bufr.2013021312
JUWE96_KKCI_130600_140957859.bufr.2013021312
JUWE96_KKCI_130600_140957933.bufr.2013021312

For bufrascat data:

data_store/bufrascat/{YYYYMMDD}

[root@dx3-tbdw data_store]# ls /data_store/bufrascat/20130213

AWP.MAN.SMM.A2  6-29
– Hard copies uncontrolled. Verify effective date prior to use. –
Chapter 6  Ingest of NWSTG Data

00  02  04  06  08  10  12  14  16  18  20  22
01  03  05  07  09  11  13  15  17  19  21  23

[root@dx3-tbdw data_store]# ls /data_store/bufrascat/20130213/03
JSXX02_KNES_130257_289524095.bufr.2013021303
JSXX04_KNES_130257_289524066.bufr.2013021303
JSXX04_KNES_130257_289524077.bufr.2013021303
JSXX04_KNES_130257_289524079.bufr.2013021303
JSXX06_KNES_130257_289524050.bufr.2013021303
JSXX06_KNES_130257_289524065.bufr.2013021303
JSXX07_KNES_121938_46315441.bufr.2013021303

For bufrua data:
data_store/bufrua/{YYYYMMDD}

[root@dx3-tbdw data_store]# ls /data_store/bufrua/20130213
00  02  04  06  08  12  14  16  19  21
01  03  05  07  11  13  15  18  20  23

[root@dx3-tbdw data_store]# ls /data_store/bufrua/20130312/02
IUSY41_KWBC_130200_289388924.bufr.2013021302
IUSY41_KWBC_130200_289388928.bufr.2013021302
IUSY42_KWBC_130235_289447526.bufr.2013021302
IUSY43_KWBC_130200_289388922.bufr.2013021302
IUSY43_KWBC_130205_289397852.bufr.2013021302
IUSY43_KWBC_130210_289406797.bufr.2013021302
IUSY43_KWBC_130245_289463217.bufr.2013021302
IUSY44_KWBC_130205_289397846.bufr.2013021302
IUSY44_KWBC_130210_289406799.bufr.2013021302
IUSY44_KWBC_130215_289414178.bufr.2013021302
IUSY44_KWBC_130235_289471027.bufr.2013021302
IUSY48_KWBC_130205_289397859.bufr.2013021302
IUSZ51_KWBC_130255_289508743.bufr.2013021303
IUSZ53_KWBC_130200_289388983.bufr.2013021302
IUSZ53_KWBC_130205_289397843.bufr.2013021302
IUSZ54_KWBC_130200_289388973.bufr.2013021302
6.6.2 Decoding BUFR Data

Data decoding operations are performed on the EDEX (DX3/4) servers. Both EDEX servers share the load in the ingest process, which includes decoding the raw data, writing the decoded data into the Data Store, writing metadata to the database, and broadcasting a notification that the newly ingested data is available.

The processed data files are stored in /awips2/edex/data/hdf5 in the hdf5 format on the dx2f server. The processed BUFR data files are stored under /awips2/edex/data/hdf5/{BUFR}.

The basic directory structure for each model is /awips2/edex/data/hdf5/{BUFR}/model.

See Table 6.6.2-1 for directories and file names.

Table 6.6.2-1. BUFR Data: Directories and File Names

<table>
<thead>
<tr>
<th>Directory</th>
<th>Filename</th>
</tr>
</thead>
<tbody>
<tr>
<td>modelsounding</td>
<td>modelsounding-2013-02-13-00.h5</td>
</tr>
<tr>
<td>goessounding</td>
<td>goessounding-2013-02-13-08.h5</td>
</tr>
<tr>
<td>poessounding</td>
<td>poessounding-2013-02-13-08.h5</td>
</tr>
<tr>
<td>bufrhdw</td>
<td>hdw-2013-02-13-11.h5</td>
</tr>
<tr>
<td>bufrncwf</td>
<td>bufrncwf-2013-02-13-22.h5</td>
</tr>
<tr>
<td>bufrascat</td>
<td>ascat-2013-02-13-12.h5</td>
</tr>
<tr>
<td>bufrsigwx</td>
<td>SWBOTH: sigwxVTS-2013-02-13-18.h5</td>
</tr>
<tr>
<td>bufrua</td>
<td>bufrua-2011-10-25-06.h5</td>
</tr>
</tbody>
</table>

The report codes in the DataURI for bufrua data support data queries that CAVE uses when it generates menus or creates certain data displays. The codes are actually somewhat arbitrary; selected by the AWIPS II developer, they are not documented outside the code, and are not particularly meaningful outside of the development environment. Table 6.6.2-2 provides the rough meaning of the codes.
### Table 6.6.2-2. Report Code Definitions

<table>
<thead>
<tr>
<th>Report Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>2020</td>
<td>mandatory data from a level below the level having a pressure of 100MB</td>
</tr>
<tr>
<td>2021</td>
<td>significant wind data from a level below the level having a pressure of 100MB</td>
</tr>
<tr>
<td>2022</td>
<td>significant temperature data from a level below the level having a pressure of 100MB</td>
</tr>
<tr>
<td>2030</td>
<td>mandatory data from a level above the level having a pressure of 100MB</td>
</tr>
<tr>
<td>2031</td>
<td>significant wind data from a level above the level having a pressure of 100MB</td>
</tr>
<tr>
<td>2032</td>
<td>significant temperature data from a level above the level having a pressure of 100MB</td>
</tr>
</tbody>
</table>

#### 6.6.3 Data Ingest and Logging of BUFR Products

**On DX3/DX4:**

```bash
[root@dx3-tbdw ~] cd /awips2/edex/logs
```

After ingest of each data file is complete, the ingest endpoint prints a single line to the EDEX system log.

Log entries for each of the corresponding BUFR products are shown below.

**For model sounding:**

```
INFO 2014-01-13 02:49:43,747 [genericThreadPool-59] Ingest: 
EDEX: Ingest - modelsounding::
/data_store/modelsounding/20140113/02/JUSA41_KWNO_130200_46550502
.bufr.2014011302 processed in: 0.0080 (sec) Latency: 0.0160 (sec)
```

**For POES sounding:**

```
INFO 2014-01-13 03:11:29,727 [genericThreadPool-41] Ingest: 
EDEX: Ingest - poessounding::
/data_store/poessounding/20140113/03/IUTX01_KNES_130309_46582659.
bufr.2014011303 processed in: 0.0040 (sec) Latency: 0.0580 (sec)
```

**For GOES sounding:**

```
INFO 2014-01-13 03:28:30,658 [genericThreadPool-45] Ingest: 
EDEX: Ingest - goessounding::
/data_store/goessounding/20140113/03/JUTX04_KNES_130327_46589432.
bufr.2014011303 processed in: 0.0130 (sec) Latency: 0.0630 (sec)
```

**For hdw:**

```
INFO 2013-02-13 03:25:29,935 [genericThreadPool-52] Ingest: 
EDEX: Ingest - bufrhdw::
/data_store/bufrhdw/20130213/03/JJCX71_KNES_130322_289549021.bufr
.2013021303 processed in: 1.0010 (sec) Latency: 1.9010 (sec)
```

**For ncwf:**

```
INFO 2013-02-13 03:52:18,234 [genericThreadPool-80] Ingest: 
EDEX: Ingest - bufrncwf::
```
Decoding Synoptic Observation Data

Synoptic data, which is ingested from the SBN and handled by the SBN, is a specific, highly machine friendly, data format (i.e., easy for a machine to generate and decode). It also contains maritime/buoy data. Synoptic files are plain (ASCII) text files. A fairly large number of different synoptic types exist, but they are all surface observations.

Table 6.7-1 illustrates the surface obs type and data URI report type code.
### Table 6.7-1. Data URI Codes for Various Synoptic Observation Types

<table>
<thead>
<tr>
<th>Sfc Ob Type</th>
<th>Data URI Report Type Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Synoptic Fixed Land</td>
<td>1001</td>
</tr>
<tr>
<td>Synoptic Mobile Land</td>
<td>1002</td>
</tr>
<tr>
<td>Synoptic Ship</td>
<td>1003</td>
</tr>
<tr>
<td>Synoptic CMAN</td>
<td>1004</td>
</tr>
<tr>
<td>Synoptic Moored Buoy</td>
<td>1005</td>
</tr>
<tr>
<td>Drifting Buoy</td>
<td>1006</td>
</tr>
<tr>
<td>Synoptic MAROB</td>
<td>1007</td>
</tr>
</tbody>
</table>

Here are some WMO headers for maritime data.

- FZNT25 KNHC ddhhmm
- FZNT26 KNHC ddhhmm
- FZNT27 KNHC ddhhmm

**NOTE:** The Data URI for surface observations has the general form: `/sfcobs/data time/report type code/correction indicator/station ID/latitude/longitude`.

See Exhibit 6.7-1 for the flow of the acquisition and ingest of synoptic products.

### 6.7.1 Data Archive and File Naming Conventions for Synoptic Data

For sfcobs data:

```bash
data_store/synoptic/{YYYYMMDD}
```

```
[root@dx3-tbdw text]# ls /data_store/synoptic/20130520
00 02 04 06 08 10 12 14 16 18 20 22
01 03 05 07 09 11 13 15 17 19 21 23
```

```
[root@dx3-tbdw text]# ls /data_store/synoptic/20130520/12
SMCN32_CWAO_201200_128039749.2013052012
SMUS73_KWBC_201200_128055820.2013052012
SMCN32_CWAO_201200_128048912.2013052012
```
Chapter 6

Ingest of NWSTG Data

Exhibit 6.7-1. Acquire and Ingest of Synoptic Products
6.7.2 Decoding Synoptic Data

The Ingested Synoptic Data files are stored in the Data Store.

The hdf5 basic directory structure is /awips2/edex/data/hdf5/sfcobs/

[root@dx2-tbdw sfcobs]# ls /awips2/edex/data/hdf5/sfcobs/
1001 1003 1004 1005 1007

[root@dx2-tbdw sfcobs]# ls /awips2/edex/data/hdf5/sfcobs/1006
sfcobs-2013-02-05-18.h5  sfcobs-2013-02-12-00.h5  sfcobs-2013-02-13-21.h5
sfcobs-2013-02-06-00.h5  sfcobs-2013-02-12-03.h5  sfcobs-2013-02-13-22.h5
sfcobs-2013-02-06-06.h5  sfcobs-2013-02-12-06.h5  sfcobs-2013-02-13-23.h5
sfcobs-2013-02-06-12.h5  sfcobs-2013-02-12-09.h5  sfcobs-2013-02-14-00.h5
sfcobs-2013-02-06-18.h5  sfcobs-2013-02-12-12.h5  sfcobs-2013-02-14-01.h5
sfcobs-2013-02-07-00.h5  sfcobs-2013-02-12-15.h5  sfcobs-2013-02-14-02.h5
sfcobs-2013-02-07-06.h5  sfcobs-2013-02-12-17.h5

6.7.3 Data Ingest and Logging of Synoptic Products

All data ingest operations are logged in the EDEX logs on the EDEX (DX3/4) server. In a standard installation, the EDEX logs are located in /awips2/edex/logs. This directory contains several log files; the files starting “edex-” are the EDEX logs.

On DX3/DX4:

[root@dx3-tbdw ~]# cd /awips2/edex/logs
[root@dx3-tbdw logs]# date +%F
2013-02-13
[root@dx3-tbdw logs]# ls edex-*.20130213*
edex-ingest-20130213.log
edex-ingestDat-20130213.log
edex-ingest-gen_areal_ffg-20130213.log
edex-ingest-gen_areal_gpe-20130213.log
edex-ingestGrib-20130213.log
edex-ingest-purge-20130213.log
edex-ingest-radar-20130213.log
edex-ingest-satellite-20130213.log
After ingest of each data file is complete, the ingest endpoint prints a single line to the EDEX system log. The format of the log entry is

`INFO <<date-time>> [thread] Ingest: <<type>>:: <<file name>> <<statistics>>`

```
INFO 2013-05-20 00:12:19,820 [genericThreadPool-78] Ingest: EDEX: Ingest - sfcobs::
/data_store/synoptic/20130520/00/SMCN21_CWAO_200000_126419892.2013052000 processed in: 0.2340 (sec) Latency: 0.2430 (sec)
```

### 6.8 Decoding Special Sensor Microwave Imager

SSM/I data is received via the SBN.

See Exhibit 6.8-1 for the flow of the acquisition and ingest of SSM/I products.
Chapter 6 Ingest of NWSTG Data

Exhibit 6.8-1. Acquire and Ingest of SSM/I Products
6.8.1 Data Archive and File Naming Conventions for SSM/I Data

SSM/I data ingested via the SBN is written to the /data_store/bufrssmi/hh, where hh is the SSM/I data hour as obtained from the WMO header that accompanies the data.

Under the data type folder, i.e., bufrssmi, you will find a set of directories representing the various hours these products were issued. For SSM/I data, under /data_store/bufrssmi/YYYYMMDD/ you will find folders ranging from 00 through 23. For example, if you change to /data_store/20131030//19, you will find all observations issued between 19z and 20z.

6.8.2 Decoding SSM/I Data

```
[root@dx2-tbdw ssmi]# ls /awips2/edex/data/hdf5/bufrssmi/
ssmi-2013-12-31-01.h5  ssmi-2014-01-05-00.h5  ssmi-2014-01-10-00.h5
ssmi-2014-01-02-00.h5  ssmi-2014-01-06-22.h5  ssmi-2014-01-11-23.h5
ssmi-2014-01-02-22.h5  ssmi-2014-01-06-23.h5  ssmi-2014-01-12-00.h5
ssmi-2014-01-02-22.h5  ssmi-2014-01-07-01.h5  ssmi-2014-01-12-22.h5
ssmi-2014-01-02-23.h5  ssmi-2014-01-07-23.h5  ssmi-2014-01-12-23.h5
ssmi-2014-01-03-00.h5  ssmi-2014-01-08-01.h5  ssmi-2014-01-13-00.h5
ssmi-2014-01-03-23.h5  ssmi-2014-01-09-00.h5
ssmi-2014-01-04-00.h5  ssmi-2014-01-09-01.h5
```

6.8.3 Data Ingest and Logging of SSM/I Products

All data ingest operations are logged in the EDEX logs on the EDEX (DX3/4) server. In a standard installation, the EDEX logs are located in /awips2/edex/logs. This directory contains several log files; the files starting “edex-” are the EDEX logs.

The log file is a plain ASCII text file and can be viewed using any text viewing utility such as more, less, view, tail, etc.

To get a general view of processing, open a terminal session to the server and enter the following commands:

**On DX3:**

```
TYPE: cd /awips2/edex/logs
```
Chapter 6

Ingest of NWSTG Data

6.40

– Hard copies uncontrolled. Verify effective date prior to use. –

TYPE:  
ls edex-*

This will list the edex log files.

TYPE:  
tail -f edex-ingest-20131030.log

On DX4:

TYPE:  
       cd /awips2/edex/logs

TYPE:  
       ls edex-*

This will list all the edex log files.

TYPE:  
tail -f edex-ingest-20131030.log

After ingest of each data file is complete, the ingest endpoint prints a single line to the EDEX system log. The format of the log entry is

INFO <<date-time>> [thread] Ingest: <<type>>: <<file name>> <<statistics>>

A log entry will be as follows.


6.9 convSIGMET Data

6.9.1 Data Archive and File Naming Conventions for convSIGMET Data

convSIGMET data ingested via the SBN is written to the /data_store/text/{hh}, where hh is the convsigmet hour as obtained from the WMO header that accompanies the data. Nonconvsigmet and intlsigmet are also stored in the same format.

[root@dx3-tbdw data_store]# ls /data_store/convsigment/<YYYYMMDD>

[root@dx3-tbdw data_store]# ls /data_store/convsigment/20130213

00 02 04 06 08 10 12 14 16 18 21 23
01 03 05 07 09 11 13 15 17 20 22

[root@dx3-tbdw data_store]# ls /data_store/convsigment/20130213/02

WSUS31_KKCI_130255_289482017.2013021302
WSUS32_KKCI_130255_289482283.2013021302
WSUS33_KKCI_130255_289481955.2013021302

AWP.MAN.SMM.A2

– Hard copies uncontrolled. Verify effective date prior to use. –
6.9.2 **Decoding convSIGMET Data**

ConvSIGMET data is decoded by the convsigmet data decoder, which is part of the EDEX Ingest process running on the DX3/4 cluster.

**NOTE:** The decoders for nonconvsigmet and intlsigmet are **nonconvsigmet** and the **intlsigmet** decoders respectively. For the database, the tables used for **nonconvsigmet** data are **nonconvsigmet** and **nonconvsigmet_location**. For **intlsigmet**, the tables used are **intlsigmet** and **intlsigmet_location**. As with **convsigmet**, all tables are part of the AWIPS II metadata database.

6.9.3 **Data Ingest and Logging of ConvSIGMET Products**

All data ingest operations are logged in the EDEX logs on the EDEX (DX3/4) server. In a standard installation, the EDEX logs are located in /awips2/edex/logs. This directory contains several log files; the files starting “edex-” are the EDEX logs.

Open a terminal session to the server and enter the following commands:

**On DX3/DX4:**

```
[root@dx3-tbdw ~]# cd /awips2/edex/logs
[root@dx3-tbdw logs]# date +%F
2013-02-13
[root@dx3-tbdw logs]# ls edex-*20130213*
edex-ingest-20130213.log
edex-ingestDat-20130213.log
edex-ingest-gen_areal_ffg-20130213.log
edex-ingest-gen_areal_qpe-20130213.log
edex-ingestGrib-20130213.log
edex-ingest-purge-20130213.log
edex-ingest-radar-20130213.log
edex-ingest-satellite-20130213.log
edex-ingest-shef-20130213.log
edex-ingest-shef-performance-20130213.log
edex-ingest-smartInit-20130213.log
edex-ingest-text-20130213.log
edex-ingest-trigger-20130213.log
edex-ingest-unrecognized-files-20130213.log
edex-request-20130213.log
edex-request-productSrvRequest-20130213.log
edex-request-thriftSrv-20130213.log
edex-ingest-archive-20130213.log
edex-ingest-activeTableChange-20130213.log
edex-ingestGrib-performance-20130213.log
```
edex-ingest-performance-20130213.log
edex-request-performance-20130213.log
edex-ingest-gen_areal_qpe-20130213.log
edex-request-hibernate-20130213.log
edex-ingest-hibernate-20130213.log
edex-ingestGrib-hibernate-20130213.log

To monitor the ingest stream, a little understanding of the logging strategy used by EDEX is required. After ingest of each data file is complete, the ingest endpoint prints a single line to the EDEX system log. The format of the log entry is

```
INFO <<date-time>> [thread] Ingest: <<type>>:: <<file name>> <<statistics>>
```

A log entry will be as follows.
```
INFO 2013-02-13 14:56:02,850 [genericThreadPool-56] Ingest: EDEX: Ingest - convsigmet::
data_store/convsigment/20130213/14/WSUS32_KKCI_131455_290744857.2013021314 processed in: 0.0070 (sec) Latency: 0.3520 (sec)
```

A log entry for intlsigmet will be as follows.
```
INFO 2013-02-13 14:57:37,026 [genericThreadPool-34] Ingest: EDEX: Ingest - intlsigmet::
data_store/wwa/20130213/15/WSNT02_KKCI_131500_290749300.2013021314 processed in: 0.0260 (sec) Latency: 0.1270 (sec)
```

**NOTE:** The nonconvsigmet will follow the same data logging and ingest operation as is illustrated above for convsigmet and intlsigmet.

### 6.10 Tracing NWSTG (TAF) Products Using the Sequence Number

In general, to trace an SBN product, you must first identify the product’s WMO header. The following example traces a TAF through the system. The general pattern is `FTUS43 XXXX ddhhmm`. The specific WMO header being traced in the example is `FTUS43 KEAX 141500`.

**NOTE:** In the normal AWIPS II system, CP1 and CP2 are clustered as are PX 1/2 and DX 1/2. In this example, it is assumed that CP1, DX1, and DX2 are the active members of each cluster.

1. **Verify that the product was received on CPSBN (CP1).**
   
   Verify that the product was received by logging on to CP1 and examining the LDM log. To facilitate the examination, grep is used to limit the output. The LDM logs are stored in the logs directory under the LDM home directory. The active LDM logs are goes.log, nwstg2.log, nwstg.log and oconus.log. The grep command filters FTUS43 by site EAX and displays the most current entry by piping the output to tail -1.
   
   The steps to follow:
Chapter 6 Ingest of NWSTG Data

[root@cpsbn1-tbdw logs]# /usr/local/ldm/logs
[root@cpsbn1-tbdw logs]# ls -l *.log
-rw-rw---- 1 root fxalpha 1475996 Feb 14 03:28 goes.log
-rw-rw---- 1 root fxalpha  609001 Feb 14 03:26 ldmd.log
-rw-rw---- 1 root fxalpha  88872724 Feb 14 03:27 nwstg2.log
-rw-rw---- 1 root fxalpha 346572421 Feb 14 03:29 nwstg.log
-rw-rw---- 1 root fxalpha  72442342 Feb 14 03:29 oconus.log
-rw-rw---- 1 root root   4406926 Feb 14 02:57 polarsat.log
[root@cpsbn1-tbdw logs]# grep -l "FTUS43" *.log
nwstg.log
[root@cpsbn1-tbdw logs]# grep "FTUS43 KEAX" nwstg.log | tail -1
Feb 13 05:26:01 cpsbn1-box readnoaaport[5005] NOTE: FTUS43 KEAX 220000 /pTAFSTJ inserted [cat 1 type 4 ccb 2/0 seq 460087807 size 133]

The sequence number (in this case 460087807) is a sequence number that is uniquely assigned to this product. It may be used to track the product as it is ingested and stored.

2. Verify that the product was received on the CPSBN cluster and written to the data archive.

We verify that the product was received on the cp1f LDM and written to the data archive by logging onto cp1f and checking two locations: the LDM log and the data archive. In both of these cases, the key to finding the product the sequence number discovered in step (1).

The steps to follow:

Login to cpsbn1f,

[root@cpsbn1-tbdw ~]# cd ~/ldm/logs
[root@cpsbn1-tbdw logs]# ls
ldmd.log  ldmd.log.2  ldmd.log.4  ldmd.log.6  scour.log
ldmd.log.1  ldmd.log.3  ldmd.log.5  ldmd.log.7
[root@cpsbn1-tbdw logs]# grep 460087807 ldmd.log
Feb 13 00:00:19 dx2-tbdw pgact[22217] NOTE: Filed in "/data_store/text/13/02/FTUS43 KEAX_220000_460087807.20130213": 12420 20130213000017.724 IDS|DDPLUS~460087807 FTUS43 KEAX 220000 CCA

[root@dx2-tbdw data_store]# find /data_store/text/ | grep 460087807
/data_store/text/13/02/FTUS43 KEAX_220000_460087807.20130213
3. Verify that the product has been ingested.

Generally, we can verify that the product has been ingested by examining the EDEX Ingest process logs and looking for an entry identifying the file ingest.

This is complicated somewhat by the fact that the product may have been ingested on either DX3 or DX4. There is no way to predict the correct server, so we simply check them both. If the product has been ingested, it will appear in the log on either DX3 or DX4. The basic process is to grep for the sequence number in the EDEX Ingest process logs.

The steps to follow on DX3:

[root@dx2-tbdw data_store]# ssh dx3
[root@dx3-tbdw ~]# cd /awips2/edex/logs
[root@dx3-tbdw logs]# date +%F
2013-02-13
[root@dx3-tbdw logs]# ls edex*20130213*
edex-ingest-20130213.log
edex-ingestDat-20130213.log
edex-ingest-gen_areal_ffg-20130213.log
edex-ingest-gen_areal_qpe-20130213.log
edex-ingestGrib-20130213.log
edex-ingest-purge-20130213.log
edex-ingest-radar-20130213.log
edex-ingest-satellite-20130213.log
edex-ingest-shef-20130213.log
edex-ingest-shef-performance-20130213.log
edex-ingest-smartInit-20130213.log
edex-ingest-text-20130213.log
edex-ingest-trigger-20130213.log
edex-ingest-unrecognized-files-20130213.log
edex-request-20130213.log
edex-request-productSrvRequest-20130213.log
edex-request-thriftSrv-20130213.log
edex-ingest-archive-20130213.log
edex-ingestGrib-performance-20130213.log
edex-ingest-performance-20130213.log
edex-request-performance-20130213.log
edex-ingest-gen_areal_qpe-20130213.log
edex-request-hibernate-20130213.log
edex-ingest-hibernate-20130213.log
edex-ingestGrib-hibernate-20130213.log
[root@dx3-tbdw logs]# **grep 460087807 edex-ingest-20130213.log**

INFO 2013-02-13 00:07:16,288 [genericThreadPool-60] Ingest: EDEX: Ingest - taf::
/data_store/forecast/20130213/00/FTUS80_KWBC_200006_460087807. 2013021300 processed in: 0.0070 (sec) Latency: 0.0350 (sec)
# Chapter 7. Ingest of Radar Data

## Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.1</td>
<td>Overview</td>
<td>1</td>
</tr>
<tr>
<td>7.2</td>
<td>Radar Ingest Operation</td>
<td>5</td>
</tr>
<tr>
<td>7.2.1</td>
<td>SBN Radar Ingest Operation</td>
<td>5</td>
</tr>
<tr>
<td>7.2.2</td>
<td>ORPG Radar Ingest Operation</td>
<td>5</td>
</tr>
<tr>
<td>7.3</td>
<td>Data Archiving</td>
<td>5</td>
</tr>
<tr>
<td>7.3.1</td>
<td>Radar Archive Structure</td>
<td>7</td>
</tr>
<tr>
<td>7.3.1.1</td>
<td>Available SBN Radar Site Identifiers</td>
<td>8</td>
</tr>
<tr>
<td>7.3.1.2</td>
<td>Available ORPG Radar Site Identifiers</td>
<td>10</td>
</tr>
<tr>
<td>7.3.1.3</td>
<td>Available SBN Product Categories</td>
<td>10</td>
</tr>
<tr>
<td>7.3.1.4</td>
<td>Available ORPG Product Categories</td>
<td>11</td>
</tr>
<tr>
<td>7.3.2</td>
<td>EDEX Ingest Radar Data Logging</td>
<td>11</td>
</tr>
<tr>
<td>7.3.2.1</td>
<td>Radar Server Logs</td>
<td>12</td>
</tr>
<tr>
<td>7.3.2.2</td>
<td>Configuring Radar Server Logging</td>
<td>13</td>
</tr>
<tr>
<td>7.3.3</td>
<td>SBN Radar Configuration Files</td>
<td>14</td>
</tr>
<tr>
<td>7.3.3.1</td>
<td>SBN Radar Configuration Files – LDM</td>
<td>14</td>
</tr>
<tr>
<td>7.3.3.2</td>
<td>SBN Radar Configuration Files – PQACT</td>
<td>14</td>
</tr>
<tr>
<td>7.3.3.3</td>
<td>SBN Radar Configuration Files – EDEX Data Distribution Files</td>
<td>15</td>
</tr>
<tr>
<td>7.3.4</td>
<td>ORPG Radar Configuration Files</td>
<td>16</td>
</tr>
<tr>
<td>7.3.4.1</td>
<td>ORPG Radar Configuration Files – Radar Server</td>
<td>16</td>
</tr>
<tr>
<td>7.3.4.2</td>
<td>ORPG Radar Configuration Files – EDEX Data Distribution files</td>
<td>18</td>
</tr>
<tr>
<td>7.3.5</td>
<td>Storing Decoded SBN Radar Data</td>
<td>18</td>
</tr>
<tr>
<td>7.3.5.1</td>
<td>AWIPS II Metadata Database – Radar Metadata</td>
<td>19</td>
</tr>
<tr>
<td>7.3.5.2</td>
<td>AWIPS II Data Store – Radar Products</td>
<td>19</td>
</tr>
<tr>
<td>7.3.6</td>
<td>Audible Radar-Alerting</td>
<td>20</td>
</tr>
<tr>
<td>7.4</td>
<td>WAN/SBN Radar Product Distribution from Site to NCF</td>
<td>23</td>
</tr>
<tr>
<td>7.5</td>
<td>RPS list</td>
<td>27</td>
</tr>
<tr>
<td>7.6</td>
<td>Two-Way Communication</td>
<td>33</td>
</tr>
<tr>
<td>7.7</td>
<td>Central Collection Cron Schedule</td>
<td>35</td>
</tr>
</tbody>
</table>

## List of Exhibits

<table>
<thead>
<tr>
<th>Exhibit</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exhibit 7.1-1</td>
<td>Radar Ingest Data Flow</td>
<td>2</td>
</tr>
<tr>
<td>Exhibit 7.2.1-1</td>
<td>Acquire and Ingest of Radar Products</td>
<td>6</td>
</tr>
<tr>
<td>Exhibit 7.3.6-1</td>
<td>Alert Visualization Configuration dialog</td>
<td>20</td>
</tr>
</tbody>
</table>
Exhibit 7.3.6-2. Alert Visualization Configuration Sources & Priorities................................. 21
Exhibit 7.3.6-3. Alert Visualization Configuration Sources & Priorities Audio........................ 22
Exhibit 7.3.6-4. Alert Visualization Configuration Sources & Priorities Audio File............... 22
Exhibit 7.3.6-5. Alert Visualization Configuration List.......................................................... 23
Exhibit 7.5-1. RPS List Editor ............................................................................................... 28
Exhibit 7.5-2. RPS List Editor View Menu – Select List............................................................ 29
Exhibit 7.4-3. Select RPS to View.......................................................................................... 30
Exhibit 7.5-4. RPS List Editor View Menu List..................................................................... 31
Exhibit 7.5-6. RPS List Editor View Menu List New Entry ...................................................... 33
Exhibit 7.6-1. Sending Environmental Data............................................................................ 34
Exhibit 7.6-2. Sending Radar Applications.............................................................................. 35

List of Tables

Table 7.4-1. WSR-88D Products on AWIPS........................................................................... 24
7.1 Overview

The AWIPS WSR-88D/TDWR radar data ingest is responsible for providing products to be sent and receiving centrally collected radar products over the SBN; requesting/receiving data from one or more associated ORPGs/SPGs; sending Radar Precipitation Bias Table to the ORPG; and making the data available to the AWIPS workstations. More specifically, the radar ingest system is responsible for initiating and maintaining connections with multiple ORPGs/SPGs, forwarding data requests from the workstations to the ORPGs/SPGs, receiving and processing data and status messages from the ORPGs/SPGs, and forwarding status messages to the workstations.

Two additional EDEX “tasks” are involved here. They are both are part of the EDEX RPG Environmental Data Service (rpgenvdata), and are triggered by Quartz timers. They are:

- Send Environmental Data Timer: Fires every hour on the hour
- send Bias Table Timer: Fires 26 and 46 minutes after each hour

Both tasks are managed by the EDEX Ingest Process on DX3 and DX4. The triggering mechanism is “clustered quartz”; this means that the event fires on both DX3 and DX4, but a locking mechanism is employed so that only one of the servers performs the work. Messages, status, error, etc., are logged to the EDEX Ingest process log (edex-ingest-yyyymmdd.log); search for RPGEnvData. See Section 7.5 for detailed explanation.

Radar data comes from two sources: 1) the SBN for nationally distributed data; and 2) the Radar Server for locally collected radar data, including routine data and one-time as well as multiple requests from network radars.

SBN data is delivered via the Local Data Manager (LDM) on CPSBN (cp1f), and the locally collected radar data is delivered via the Radar Server on DX1/2.

Radar Ingest Data Flow is illustrated in the Exhibit 7.1-1. As shown in the exhibit, the data flow is a 12-step process. Descriptions of each step of the process follow.

1. Radar Data are received from two sources: The Radar Server on the DX1/2 cluster obtains radar data from the ORPG/SPG; and radar data obtained from the SBN is received from the LDM noaaportIngester on the CP1/2 cluster.
   a. The Radar Server and the LDM noaaportIngester write the product to a file on the Data Archive on NAS [NAS2]. (Refer to Section 7.1.)
   b. The Radar Server and LDM post their messages containing product information to the QPID Message
   c. The LDM writer instance uses the EDEX Bridge to post a message containing product information to the QPID Message Broker. The product information is the WMO header and the path to the product file.
Chapter 7

Ingest of Radar Data

Exhibit 7.1-1. Radar Ingest Data Flow
2. The EDEX Distribution endpoint obtains the WMO (World Meteorological Organization) header for the product file identified by the message. The WMO header is matched against data distribution mappings contained in the data distribution directory to determine product routing. (The WMO header is usually included in the message; if it is not, the Distribution service uses the product file name from the message to determine routing.)

**NOTE:** The files in the data distribution directory are XML (eXtensible Markup Language) files that contain regular expressions which match WMO headers with the appropriate data decoder plug-in.

a. At EDEX startup, the EDEX Distribution endpoint reads the Data Distribution files from the Data Distribution directory in the Localization Store. It uses the contents of these files to generate the routing table used to determine data routing. The Data Distribution directory is located at /awips2/edex/data/utility/edex_static/base/distribution.

b. The EDEX Distribution endpoint monitors the Data Distribution directory; when a file in that directory is modified, it rereads that file and rebuilds its routing table.

3. The EDEX Distribution Service determines the Data Decoder Plug-In registered to handle the data in the product file.

4. The EDEX Distribution Service forwards the message to the appropriate Data Decoder Plug-In via the QPID Message Broker.

5. The EDEX Data Decoder Service on the EDEX (DX) cluster pulls the message from the QPID Message broker. The Data Decoder Service reads the file pointed to by the message, decodes the data, and determines the appropriate metadata.

6. The decoded data and metadata are sent to the EDEX Persist endpoint.

7. The EDEX Persist endpoint sends the data to PyPIES and PyPies writes the data to the EDEX data store in HDF5 format.

8. The EDEX Persist endpoint sends the metadata obtained to the EDEX Index endpoint.

9. The EDEX Index endpoint sends the metadata to the PostgreSQL DBMS. The PostgreSQL DBMS writes the metadata to the database.

10. The index endpoint does not merely get the dataURI. It also persists the data object to the database. Persistence to the database is executed via a plug-in defined data access object. If the data access object does not override the persistence behavior, then a default persistence strategy is used. The default strategy for persisting data to the database is to first check to see if the data received already exists. If it already exists, then persistence to the database is not attempted for that data. If the data is unique, then it is persisted to the database.
11. (Client notification) The Data URI is forwarded to the EDEX URI Aggregator. Every 5 seconds the aggregated Data URIs are sent to the JMS Alert Topic.
CAVE monitors the JMS Alert Topic to obtain a list of available data. CAVE determines if it should render the available data.
CAVE retrieves metadata from the EDEX server via a product query sent to the EDEX Thrift endpoint.
CAVE retrieves product data from the EDEX server by accessing the EDEX data store’s HDF5 files and CAVE renders the data, updating its display as appropriate.

12. (Subscription) The Data URI is forwarded to the EDEX Subscription endpoint.
The EDEX Subscription endpoint checks the active subscription list to determine if a subscription has been registered for the product.
If no subscription has been registered for the product, no further action is taken.
The EDEX Subscription endpoint obtains the script information from the database.
The EDEX Subscription endpoint obtains a micro-engine to execute the script. The micro-engine executes the subscription script for the product.

The Radar Server manages the following types of requests (please note that the forecaster interaction is via the CAVE Radar menu):

- Routine Product Set (RPS) list
- One-Time Request (OTR)
- Alert request
- Radar Multiple Request (RMR)
- Centrally collected radar products
- Radar Precipitation Bias Table.

The **RPS list** contains the products to be sent routinely to the workstation and the products to be centrally collected for transmission over the SBN. The list is sent to the ORPG/SPGon system start-up, when the radar mode changes, or when a forecaster at the workstation modifies one of the adaption RPS lists and sends it to the ORPG/SPG. An RPS list sent to an ORPG/SPG contains both a national and a local component if a site is identified as the office responsible for the national collection and distribution of radar data. National products take precedence over local products.

The **OTR** enables a forecaster at the workstation to request that a product be sent on a one-time basis only. When requesting data from a dedicated radar, the forecaster can also request that the product be sent for more than one volume scan. The request is terminated when it has been completed.

An alert request is generated at the workstation when the forecaster wants to be notified of an alert condition. When the threshold values specified in the request are exceeded, the
ORPG sends an alert message. The forecaster can also request that a product be provided along with the alert.

The RMR enables a forecaster at the workstation to request a single product (like an OTR) or a package of products, schedule requests for products at a specified frequency, and/or request products from any number of dedicated and dial-out ORPGs/SPGs.\(^1\)

The NCF sends centrally collected radar products to all AWIPS sites over the SBN. Specified sites are tasked to collect and send certain products to the NCF via the WAN.

**Radar Precipitation Bias Tables** are available for transmission from AWIPS to an interested receiving ORPG site.

The new Radar Server takes the place of the following AWIPS I processes.

- RadarServer
- DialServer
- ORPGCommsMgr
- ORPGReqMgr.

### 7.2 Radar Ingest Operation

#### 7.2.1 SBN Radar Ingest Operation

As illustrated in Exhibit 7.2.1-1, SBN radar product ingest is performed by the EDEX Ingest processes on DX3 and DX4. EDEX operation is controlled by the edex_camel service script located in `/etc/init.d` on DX3 and DX4.

#### 7.2.2 ORPG Radar Ingest Operation

As illustrated in Exhibit 7.2.1-1, ORPG radar product ingest is controlled by the AWIPS II Radar Server, which is installed on the DX1/2 cluster. The Radar Server is controlled by the edex_rcm service script located in `/etc/init.d` on DX1 and DX2.

### 7.3 Data Archiving

Radar products are written to the Data Archive in three separate locations:

- SBN radar products are written to `/data_store/radar/XXXX` where `XXXX` is an uppercase, four- or three-character radar site identifier (customizable via the LDM).
- ORPG radar products are written to `/data_store/radar/xxxx` where `xxxx` is the lowercase, four-character radar site identifier.

\(^1\) For information on how to perform on OTR, see Section 8.3 of the AWIPS II CAVE-D2D User’s Manual (UM). Also see UM Section 8.1 for information on configuring an alert request and UM Section 8.4 for information on how to perform an RMR.
Chapter 7 Ingest of Radar Data

Config Files:
- /awips2/rcm/data/config/res/log4j
- /awips2/rcm/data/config/persist/config.xml
- /awips2/rcm/data/config/start-config

pqact.conf
ldmd.conf

/etc/init.d/edex_camel(SBN)

Log Files:
- /awips2/rcm/data/logs/radarserver.log

SBN Data

EDEX Server
(ingest, ingestGrib, ingestData, request)

QPID Message Broker

SBN CP (cp1f)

EDEX Bridge

LDM Reader

noaaportingester

PyPies

dx2f

PyPies

DX3/DX4

Log Files:
- /awips2/edex/logs/edex-ingest-radar-<yyyymmdd>.log

Data Files:
- SBN: /data_store/radar/
  XXXX/NNN
- ORPG: /data_store/radar/
  xxxx/NNN
- DPA: /data_store/dpa

SBN: /data_store/radar/

ORPG: /data_store/radar/

DPA: /data_store/dpa

Databases:
- SBN: /data_store/radar/
  XXXX/NNN
- ORPG: /data_store/radar/
  xxxx/NNN
- DPA: /data_store/dpa

(NNN is the product category)

/usr/local/ldm/logs/edexBridge.log

/awips2/edex/data/hdf5/radar

/awips2/edex/data/edex_static/base/distribution

Exhibit 7.2.1-1. Acquire and Ingest of Radar Products

- Hard copies uncontrolled. Verify effective date prior to use. -
- Digital Precipitation Array (DPA) data is potentially stored in both the SBN and the ORPG data sections. At this point, the SBN stores a subset of the radar data it receives into /data_store/dpa. This is the “DPA” data, which is used by the DPA process in EDEX. Because the DPA has been Rehosted (rather than rewritten), the options for placing the data were limited. In essence, the path (below /data_store) to the data was built into the software being rehosted. DPA products are written to /data_store/dpa and /data_store/radar.

The raw data is stored in the Data Archive. The physical directory (/data_store) is on the NAS. All the hosts mount the volume off NAS. [Note: The data_store volume is moved off the DAS (dx2 direct mount; export via nfs to all hosts) and onto NAS.]

7.3.1 Radar Archive Structure

SBN Radar Products

The general structure of the SBN radar product archive is determined by the header information on the radar products. For radar data, the archive structure is

/data_store/radar/KXXX/NNN

and the files are named:

**SDUSii\_HHMM\_XXXX\_NNN\_sequence.rad**

Where,

- **SDUSii** ⇔ product identifier
- **HHMM** ⇔ the product time stamp
- **XXXX** ⇔ product originator (transmitting wfo identifier – this is customizable via the LDM)
- **NNN** ⇔ the product category

E.g.: **SDUS21\_1320\_KCCX\_N1Q\_45929394.rad**

LDM assigns the sequence number. Its design minimizes the likelihood of file overwrites.

ORPG Radar Products

ORPG Radar products are written to the Data Archive in directories under /data_store/radar. Each site that the Radar Server is configured for has a single directory at this level. (The directory’s name is the radar site identifier in lower case. Technically, however, it does not have to be lower case; it could be in either case.) Each site directory contains several subdirectories; each of these subdirectories is the mnemonic of the products in the directory. Under each product mnemonic, products are organized into product-dependent structures. Subdirectories are generally self-explanatory. Three common directory structure patterns in the Radar Server Data Store are
Chapter 7

Ingest of Radar Data

/data_store/radar/<site>/<mnemonic>/<elevation>/<resolution>/<level>

Where site is the radar site and mnemonic is the product mnemonic.

E.g., /data_store/radar/tbwi/Z/elev1_5/res0_15/level256
/data_store/radar/<site>/<mnemonic>/<elevation>/<resolution>/<az>/<level>

E.g., data_store/radar/kgrr/Z/elev1_5/res0_25/az0_5/level256/
/data_store/radar/<site>/<mnemonic>/<layer>/<resolution>/<level>

E.g., data_store/radar/tbwi/CZ/layer0/res1/level16

The exact directory structure used by the Radar Server is determined by the Radar Server software and is unchanged from AWIPS 1 and it’s preserved primarily to support FSI.

DPA Products

The archive structure for the DPA products is similar to the ORPG Radar Products except that, since application products in the archive are DPA, the product category and radar site identifier are not included in the directory structure. For DPA products, the archive structure is /data_store/dpa and the files are named:

SDUSii_HHMM_XXXX_DPA_sequence.rad

Where,

SDUSii ⇔ product identifier
HHMM ⇔ the product time stamp
XXXX ⇔ product originator (transmitting wfo identifier – this is customizable via the LDM)
NNN ⇔ the product category

E.g.: SDUS54_0841_KHGX_DPA_191653228.rad

NOTE: The final multi-digit number in the file name will vary depending on the sequence value at the time LDM writes the file to the Data Archive.

7.3.1.1 Available SBN Radar Site Identifiers

The list of potential radar sites available in the Data Archive depends on the configuration of the LDM and the EDEX Distribution Service. It is possible to query the Data Archive to determine available site identifiers. This query is performed using a command line script; the script can run on any workstation with access to the Data Archive.

Enter the following command in a console window or x-term:

TYPE: cd /data_store/radar
**TYPE:**  
ls | grep -P "[A-Z]"

<table>
<thead>
<tr>
<th>Airport Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>KBGM</td>
</tr>
<tr>
<td>KABX</td>
</tr>
<tr>
<td>KAMA</td>
</tr>
<tr>
<td>KBLX</td>
</tr>
<tr>
<td>KCYS</td>
</tr>
<tr>
<td>KDAL</td>
</tr>
<tr>
<td>KDDC</td>
</tr>
<tr>
<td>KDFW</td>
</tr>
<tr>
<td>KDFX</td>
</tr>
<tr>
<td>KDYX</td>
</tr>
<tr>
<td>KEWX</td>
</tr>
<tr>
<td>KFDX</td>
</tr>
<tr>
<td>KFSX</td>
</tr>
<tr>
<td>KFWS</td>
</tr>
<tr>
<td>KGJX</td>
</tr>
<tr>
<td>KGLD</td>
</tr>
<tr>
<td>KGRK</td>
</tr>
<tr>
<td>KICH</td>
</tr>
<tr>
<td>KICT</td>
</tr>
<tr>
<td>KICX</td>
</tr>
<tr>
<td>KINX</td>
</tr>
<tr>
<td>KIWA</td>
</tr>
<tr>
<td>KLBB</td>
</tr>
<tr>
<td>KLNX</td>
</tr>
<tr>
<td>KMTX</td>
</tr>
<tr>
<td>KPHX</td>
</tr>
<tr>
<td>KPUX</td>
</tr>
<tr>
<td>KSGF</td>
</tr>
<tr>
<td>KSHV</td>
</tr>
<tr>
<td>KSJT</td>
</tr>
<tr>
<td>KSLC</td>
</tr>
<tr>
<td>KSRX</td>
</tr>
<tr>
<td>KTLX</td>
</tr>
<tr>
<td>KTUL</td>
</tr>
<tr>
<td>KTXW</td>
</tr>
<tr>
<td>KUEX</td>
</tr>
<tr>
<td>KVNX</td>
</tr>
<tr>
<td>KYUX</td>
</tr>
</tbody>
</table>
NOTE: This procedure provides a list of radar sites from which Radar Data has been received over the SBN. It does not provide a list of current sites or current data.

7.3.1.2 Available ORPG Radar Site Identifiers

The list of ORPG radar sites available for the Radar Server depends on the configuration of the Radar Server and the EDEX Distribution Service. It is possible to query the Data Archive to determine available site identifiers. This query is performed using a command line script that may be performed from any workstation with access to the Data Archive.

Enter the following commands in a console window or X-Term:

```
TYPE: cd /data_store/radar
TYPE: ls | grep -P "^[a-z]{4}$" > ~/radar-sites.txt
TYPE: sort ~/radar-sites.txt > ~/sorted-radar-sites.txt
TYPE: cat ~/sorted-radar-sites.txt | tr [:cntrl:] ' ' > ~/radar-sites.txt
TYPE: cat ~/radar-sites.txt ; echo
koax
```

7.3.1.3 Available SBN Product Categories

The list of Product Categories available in the Data Archive will depend on the configuration of the LDM and the EDEX distribution server. It may also vary from radar site to radar site. It is possible to query the Data Archive to get a rough list of available product categories. This query is performed using a command line script; the script may be performed from any workstation with access to the Data Archive.

Enter the following commands in a console window or X-Term:

```
TYPE: cd /data_store/radar
TYPE: echo $( for i in $(ls | grep -E "^[A-Z]{4}$"); do ls "$i"; done | sort | uniq)
DAA DHR DOD DPA DPR DSD DSP DTA DU3 DU6 DVL EET HEC N0C N0H N0K N0M N0Q N0R
N0S N0U N0V N0X N1C N1H N1K N1M N1P N1Q N1S N1U N1X N2C N2H N2K N2M N2Q
N2S N2U N2X N3C N3H N3K N3M N3Q N3S N3U N3X NAC NAH NAK NAM NAQ NBU NAX NBC
NBH NBL NBM NBQ NBU NBX NCR NET NMD NST NTP NVL NVW OHA PTA TR0 TR1 TR2 TV0
TV1 TV2 T2L
```

NOTE: This procedure provides a list of product categories for Radar Data received over the SBN. It does not provide a list of current product categories or current data.
7.3.1.4 Available ORPG Product Categories

Product categories for a specific site may be determined by listing that site’s directory in /data_store/radar. To determine product categories available regardless of site, use the following script:

```
TYPE:  cd /data_store/radar

TYPE:  echo $( for i in $(ls | grep -E "^[a-z]{4}$"); do
         ls "$i"; done | sort | uniq) (all on one line)

AAP APR CFC CZ DHR DMD DPA DUA DVL EET ET GSM HI LRM MD OHP OSD OSW PRR
PTL RCM RSS SO SPD SRM SS SSD SSW STI STP SW THP TRU TVS ULR V VIL VWP Z
```

**NOTE:** This procedure provides a list of product categories for Radar Data received over the SBN. It does not provide a list of current product categories or current data.

7.3.2 EDEX Ingest Radar Data Logging

Radar Ingest processing follows the same general strategy for logging as the remaining data decoders; both routine messages and error messages are logged to the EDEX process logs. The process logs are located in /awips2/edex/logs on each of the EDEX servers, DX3 and DX4. The logs to examine are the EDEX Ingest process log and the EDEX Radar Ingest process log.

**Note:** The EDEX Ingest process logs contain startup messages and most other messages, including errors, for the EDEX processes. The EDEX Radar Ingest process log contains messages relating to a radar type. These include both status messages and error messages.

To monitor basic EDEX ingest, log into either DX3 or DX4 and change the directory to /awips2/edex/logs.

```
TYPE:  tail -f edex-ingest-yyyymmdd.log
```

where yyyymmdd is the date stamp of the current log file.

To monitor messages specific to Radar Product ingest, log into either DX3 or DX4 and change directory to /awips2/edex/logs.

```
TYPE:  tail -f edex-ingest-radar-yyyymmdd.log
```

where yyyymmdd is the date stamp of the current log file.

The LDM writer uses patterns in pqact.conf to determine where to write data files, including the path and file name. A typical entry is something like this:

```
NNEXRAD  ^\(SDUS[234578].\)NXUS6.\)
  (K|P|T)\(LWX|BGM|CHS|RLX|TLN|CLE|AKQ|JKL|CTP|MHX|MRX|OKX|PHI\)
  \(\.|\.|\) /p\(\.|\.|\)\(\.|\).\)
```
FILE   -overwrite -close -log -edex
/data_store/radar/\2\8/\7/\5\6_\2\8_\7_(seq).rad

This results in a log entry similar to

INFO  2011-10-22 00:00:01,009 [radarThreadPool-1] Ingest: EDEX: Ingest - radar::
/data_store/radar/KDIX/NMD/SDUS31_2353_KDIX_NMD_460087421.rad processed in: 0.0330 (sec) Latency: -0.2390 (sec)

The number before the ".rad" (460087421 in this case) is set by the LDM and can be traced from EDEX. These sequence numbers (product identifiers) can be traced to the LDM logs on the cpsbn2 (goes.log, nwstg.log, etc.) in /data/ldm/logs.

NOTE: For additional information on monitoring EDEX and EDEX based ingest operations, see Chapter 25.

7.3.2.1 Radar Server Logs

The Radar Server also maintains its own logging. Radar Server logs are located in /awips2/rcm/data/logs. The current Radar Server log is radarserver.log. In normal conditions, the Radar Server creates a new log file daily and retains up to 3 old logs. Old logs are named radarserver.log.yyyy-mm-dd.

To monitor basic Radar Server operation, log onto the active member of the DX1/2 cluster and change directory to /awips2/rcm/data/logs.

TYPE:   tail -f radarserver.log.

The basic structure of the Radar Server is the same as the EDEX logging. Routine messages fit one of two patterns:

INFO <time-stamp> [thread] RadarServer: Stored <message>

INFO  00:00:01,939 [TCM Read kdmx #2] RadarServer: kdmx: message code=19 size=16468 elev=1.3 sequence=3 vs=2011-10-21 23:58:37 #74

INFO <time-stamp> [thread] RadarServer: <site>: <message>

INFO  00:00:01,968 [TCM Read kdmx #2] RadarServer: Stored message in /data_store/radar/kdmx/Z/elev1_5/res1/level16/kdmx.19.201110 21_2358

These messages will normally occur in pairs. These patterns can be used to perform a quick status check on the Radar Server.

To monitor product arrival for a specific site, log onto the active member of the DX1/2 cluster and change directory to /awips2/rcm/data/logs. Once in that directory,
TYPE: \texttt{tail -f radarserver.log | grep "RadarServer:\ <site>:"} \\
(replace <site> with the site identifier)

To monitor Radar Server writes, log onto the active member of the DX1/2 cluster and change directory to /awips2/rcm/data/logs. Once in that directory,

\textbf{TYPE:} \texttt{tail -f radarserver.log | grep "RadarServer: Stored"}

### 7.3.2.2 Configuring Radar Server Logging

Radar Server logging is controlled by a file called log4j.properties which is located in /awips2/rcm/data/config/res on each server in the DX1/2 cluster.

Unlike EDEX configuration files, this directory is not shared between the servers, so any desired changes will need to be made on both servers. The baseline version of log4j.properties for the Radar Server contains the following entries.

```
log4j.rootLogger=INFO, rolling
log4j.logger.RadarServer=INFO, rolling
log4j.additivity.RadarServer=false
log4j.appenders.console=org.apache.log4j.ConsoleAppender
log4j.appenders.console.layout=org.apache.log4j.PatternLayout
log4j.appenders.console.layout.ConversionPattern=%-5p %d [%t] %c{1}: %m%n
log4j.appenders.rolling.layout=org.apache.log4j.PatternLayout
log4j.appenders.rolling.layout.ConversionPattern=%-5p %d\{HH:mm:ss,SSS\} [%t] %c{1}: %m%n
log4j.appenders.rolling.File=${com.raytheon.rcm.logDir}/radarserver.log
log4j.appenders.rolling.DatePattern='.'yyyy-MM-dd
log4j.appenders.rolling.MaxFiles=3
```

**Default log4j.properties for Radar Server**

**NOTE:** Take extreme care in modifying this file; an error will cause logging and perhaps operational problems in the Radar Server. The only line that should be modified is the final one on the file: \texttt{log4j.appenders.rolling.MaxFiles=3}

This line specifies the number of old logs to retain. Change the 3 to the desired number of old files. Then restart the Radar Server for the changes to take effect.

The last line in the log4j.properties file specifies the number of old log files to keep. Log files are rolled daily, that is, a new log file is opened each day. Normally, as installed, three log files are retained. This means the current log file plus the previous two are retained. This also handles file purging – when the radar server is started, and each day thereafter, excess log files are deleted. For example, to retain a total of six old logs, change the final entry to "log4j.appenders.rolling.MaxFiles=6."
NOTE: The only reason to change log4j.properties (a configuration file) is to change the number of old logs being retained. This file applies only to the AWIPS II Radar Server, the application that bridges AWIPS II EDEX Ingest and the ORPG. It is located at /awips2/rcm/data/config/res on DX1 and DX2. Configuration files located in /awips2/edex/conf on both DX 3 and DX4 control all other EDEX process logging. These files are log4j.xml and log4j-ingest.xml. Any changes related to logging of other data types must be made in these files. Two caveats: 1) Each server (DX3 and DX4) holds a copy of each of these logs. Changes made to one of these files on one server must be propagated to the other. The changes are read only at EDEX startup. 2) These are XML files, which can be sensitive to errors. Edits made to these files can cause malformed XML, which could cause logging issues. It does not appear to cause EDEX to quit, but there are startup issues, and, depending on where the XML error occurs, logging can be affected. Most changes to the EDEX logging strategy need to be made to one or both of these files, but this is not recommended.

7.3.3 SBN Radar Configuration Files

There are two types of Radar Configuration Files: 1) the LDM PQACT files; and 2) EDEX Distribution Service files.

7.3.3.1 SBN Radar Configuration Files – LDM

LDM server runs only on the SBN CP. It ingests everything from the SBN.

LDM behavior is controlled by the pqact.conf file, and the patterns are matched against pqact.conf. LDM writes the product to data_store and posts a QPID message.

- pqact.conf controls how data sent to an instance of LDM is handled. This includes the data types to be handled and the location of the raw data archive.

NOTE: Modifying pqact.conf will have a severe impact on the data flow.

7.3.3.2 SBN Radar Configuration Files – PQACT

Radar data flow controlled in pqact.conf is archived into two separate areas: radar products and Digital Precipitation Array (DPA) products. In general, LDM writes radar products to directories under /data_store/radar and DPA products under /data_store/dpa.

A typical pqact.conf entry controlling radar product routing is

```
####### Added by config_pqact #########
NNEXRAD ^(SDUS[234578].|NXUS6.)(K|P|T)(MOB|BRO|FWD|LCH|CRP|SJT|HGX|EWX|SHV|LIX) (..)(..)(..) /p(...)(...) FILE -overwrite -log -close -edex
```

AWP.MAN.SMM.A2  7-14

– Hard copies uncontrolled. Verify effective date prior to use. –
Chapter 7 Ingest of Radar Data

The first part, NNEXRAD, is the FEEDTYPE.

The Regular Expression describes any product starting with SDUS and follows with any number between 2 and 8 OR starting with NXUS6.

The () store that string into the variable \1, which can be used when storing to physical disk.

(K|P|T) matches either a K, P or T and stores that to the variable \2.

(LWX|BGM|CHS|RLX|ILN|CLE|AKQ|JKL|CTP|MHX|MRX|OKX|PHI) matches any of those site IDs and stores it to variable \3.

(...) matches the next 6 characters of any type and stores them in pairs to \4, \5, and \6 respectively.

/p matches the exact string /p

(...) matches the next 6 characters of any type and stores them in trios to \7 and \8.

So the following line would match this entry in pqact and store in the following location:

Product:

NEXRAD3 56424268  SDUS51 KLWX 211529 /pTZLDCA !nids/

Storage Location:

/data_store/radar/KDCA/TZL/1529_KDCA_TZL_56424268.rad

For DPA products, the pattern is nearly identical. In this case, only DPA products are matched, so the NNN entry is hard coded as DPA.

One final note: Because the originator is specified in the first line of the entry (“KABR” in this case), there will be an entry in pqact.conf for each originator for each data type.

7.3.3.3 SBN Radar Configuration Files – EDEX Data Distribution Files

In addition to the configuration files that control the LDM, the EDEX ingest subsystem uses a set of configuration files to determine which decoder(s) will handle the data. These files are initially installed with EDEX; once EDEX has been started, they are located in the AWIPS II localization store in /awips2/edex/data/utility/edex_static/base/distribution. (Recall that to access the localization store, you log onto DX3.)

The files in /awips2/edex/data/utility/edex_static/base/distribution contain acceptance patterns for each decoder. As a general rule, the files are named <data type>.xml. For radar data, the file is radar.xml; for DPA data, the file is dpa.xml. Here is a sample version of the radar.xml file.
Chapter 7 Ingest of Radar Data

Here is a sample version of the dpa.xml file.

```xml
<requestPatterns xmlns:ns2="group">
    <regex>^SDUS[234578]. .*</regex>
    <regex>^RadarServer.*</regex>
</requestPatterns>

SDUS* entries support the SBN radar data; RadarServer entries support the ORPG ingest of local radars. Each <regex/> tag contains a Perl 5 style regular expression, which is matched against the WMO header of a data file; if the pattern matches, the decoder can handle the data in the file. As a general rule, this file should contain a pattern to correspond to each pattern in the pqact.conf file. If not, some data archived by LDM may not be processed.

Both radar.xml and dpa.xml may be edited to change data acceptance rules for the Distribution Service. This is normally performed as part of the site localization.

**NOTE:** Use care in editing these files; malformed XML can result in a failure of the EDEX Ingest process to start.

### 7.3.4 ORPG Radar Configuration Files

There are two types of configuration files: Radar Server; and EDEX Data Distribution.

#### 7.3.4.1 ORPG Radar Configuration Files – Radar Server

The radar data flow is controlled via the RCM configuration files on DX1/2. The configuration files are located in the /awips2/rcm/data/config directory tree. Basic Radar Server operation is controlled by two configuration files:

- **Start-config.** Contains basic information required for Radar Server operation. It is initialized by the AWIPS II installer; it should not be modified.
- **config.xml.** Contains configuration information defining data flow. It is located in /awips2/rcm/data/config/persist. It may be modified to change the list of radars being processed by the Radar Server. ORPG Radar Configuration Files – config.xml.

**config.xml** provides basic information the Radar Server uses to communicate with the rest of AWIPS. The basic format of config.xml is shown in the sample file on the following page; it shows two radars being accessed via Radar Server – koax and ktlx.
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<config>
  <pupID>617</pupID>
  <regionCode>3</regionCode>
  <wmoSiteID>KOAX</wmoSiteID>
  <collectionEnabled>false</collectionEnabled>
  <tdwrCollectionLimited>true</tdwrCollectionLimited>
  <decompressProducts>false</decompressProducts>
  <endpointConfig>
    <archiveRoot>/data_store/radar</archiveRoot>
    <connectionURL>amqp://guest:guest@/edex?brokerlist='tcp://cpsbn1-oma:5672'</connectionURL>
    <prefixPathWithRadar>false</prefixPathWithRadar>
    <topic>external.dropbox</topic>
  </endpointConfig>
  <radars>
    <radar>
      <radarID>koax</radarID>
      <nexradID>519</nexradID>
      <enabled>true</enabled>
      <dedicated>true</dedicated>
      <collectionEnabled>true</collectionEnabled>
      <sendEnvironmentalData>false</sendEnvironmentalData>
      <linkType>TCP_WAN</linkType>
      <links>
        <link>
          <dedicated>true</dedicated>
          <linkAddress>147.18.105.142:4501</linkAddress>
          <linkIndex>27</linkIndex>
          <maxRpsListSize>-1</maxRpsListSize>
          <tcmPassword>passwd</tcmPassword>
        </link>
      </links>
      <productAvailabilityFieldUsable>True</productAvailabilityFieldUsable>
    </radar>
    <radar>
      <radarID>ktlx</radarID>
      <nexradID>1</nexradID>
      <enabled>true</enabled>
      <dedicated>true</dedicated>
      <collectionEnabled>true</collectionEnabled>
      <sendEnvironmentalData>false</sendEnvironmentalData>
      <linkType>TCP_WAN</linkType>
      <links>
        <link>
          <dedicated>true</dedicated>
          <linkAddress>147.18.105.142:5502</linkAddress>
          <linkIndex>27</linkIndex>
          <maxRpsListSize>-1</maxRpsListSize>
          <tcmPassword>passwd</tcmPassword>
        </link>
      </links>
      <productAvailabilityFieldUsable>True</productAvailabilityFieldUsable>
    </radar>
  </radars>
</config>
7.3.4.2 ORPG Radar Configuration Files – EDEX Data Distribution files

The EDEX ingest subsystem uses a set of configuration files to determine which decoder(s) will handle data. These files are initially installed with EDEX; once EDEX has been started, they are located in the AWIPS II localization store in /awips2/edex/data/utility/edex_static/base/distribution. (Recall that to access the localization store, you log onto DX3.)

The files in /awips2/edex/data/utility/edex_static/base/distribution contain acceptance patterns for each decoder. As a general rule, the files are named <data type>.xml. For data from the Radar Server, the file is radar.xml.

Typical contents of radar.xml are:

```
<requestPatterns xmlns:ns2="group">
  <regex>^SDUS[234578]. KABR.*</regex>
  <regex>^SDUS[234578]. KCYS.*</regex>
  <regex>^SDUS[234578]. KDMX.*</regex>
  <regex>^SDUS[234578]. KGLD.*</regex>
  <regex>^SDUS[234578]. KGID.*</regex>
  <regex>^SDUS[234578]. KEAX.*</regex>
  <regex>^SDUS[234578]. KARX.*</regex>
  <regex>^SDUS[234578]. KMPX.*</regex>
  <regex>^SDUS[234578]. KLBF.*</regex>
  <regex>^SDUS[234578]. KDVN.*</regex>
  <regex>^SDUS[234578]. KUNR.*</regex>
  <regex>^SDUS[234578]. KFSD.*</regex>
  <regex>^SDUS[234578]. KTOP.*</regex>
  <regex>^SDUS[234578]. PHFO.*</regex>
  <regex>^SDUS[234578]. PGUM.*</regex>
  <regex>^koax.*</regex>
  <regex>^RadarServer.*</regex>
</requestPatterns>
```

For the Radar Server, the key is the final <regex/> tag; this specifies that a message containing a header starting with “RadarServer” is to be routed to the radar decoder. This tag is normally present in radar.xml and should not be changed. If changed, EDEX will likely cease processing radar data from the Radar Server.

7.3.5 Storing Decoded SBN Radar Data

Decoded radar data is stored in two locations: the AWIPS II Data Store and the AWIPS II metadata database

- The AWIPS II Metadata Database is hosted in PostgreSQL on the DX 1/2 cluster. The Radar metadata is stored in the radar and radar_spatial table of the awips schema of the metadata database.
- Decoded radar products are stored in HDF5 files in the AWIPS II Data Store on dx2f.
7.3.5.1 AWIPS II Metadata Database – Radar Metadata

AWIPS II software uses a PostgreSQL database to store product metadata, i.e., data that helps to identify decoded data. Physically, the database resides on the PowerVault – DAS. The PostgreSQL database Server software runs on DX1/2 cluster.

For AWIPS II, metadata is stored in the awips schema of the metadata database. Radar metadata resides in two tables: radar and radar_spatial.

The radar table is a dynamic table that includes an entry for each decoded radar product that is available on the AWIPS II system. The radar_spatial table is a static table that contains spatial data relating to the radar sites.

When Radar Data has been successfully ingested, the radar records are written to HDF5 files in the AWIPS II Data Store, and then the metadata is written to the database. At that point, the decoded data is available to client applications such as CAVE.

7.3.5.2 AWIPS II Data Store – Radar Products

Processed radar products are written to the AWIPS II Data Store. On the dx2f, the AWIPS II Data Store is mapped to /awips2/edex/data/hdf5/radar/<site>/, On the LX and XT workstations, it is also mapped to /awips2/edex/data/hdf5/radar/<site>/.

Under /awips2/edex/data/hdf5/<site>/, the decoded radar products are stored in the radar site subdirectory.

For example, under /awips2/edex/data/hdf5/radar/kakq

```
TYPE:    ls -l

drwxr-xr-x  2 awips awips  4096 Aug 15 16:28 0.0
drwxr-xr-x  2 awips awips  4096 Aug 15 16:28 0.5
drwxr-xr-x  2 awips awips  4096 Aug 15 16:28 0.9
drwxr-xr-x  2 awips awips  4096 Aug 15 16:28 10.0
drwxr-xr-x  2 awips awips  4096 Aug 15 16:28 12.0
drwxr-xr-x  2 awips awips  4096 Aug 15 16:28 14.0
drwxr-xr-x  2 awips awips  4096 Aug 15 16:28 1.5
drwxr-xr-x  2 awips awips  4096 Apr 17 16:04 16.7
drwxr-xr-x  2 awips awips  4096 Aug 15 16:28 1.8
drwxr-xr-x  2 awips awips  4096 Aug 15 16:28 19.5
drwxr-xr-x  2 awips awips  4096 Aug 15 16:28 2.4
drwxr-xr-x  2 awips awips  4096 Aug 15 16:28 3.4
drwxr-xr-x  2 awips awips  4096 Aug 15 16:28 4.3
drwxr-xr-x  2 awips awips  4096 Aug 15 16:28 5.3
drwxr-xr-x  2 awips awips  4096 Aug 15 16:28 6.0
drwxr-xr-x  2 awips awips  4096 Aug 15 16:28 7.5
```
Chapter 7 Ingest of Radar Data

Under 0.0

radar-kakq-2013-02-07-18.h5  radar-kakq-2013-02-13-17.h5
radar-kakq-2013-02-13-03.h5  radar-kakq-2013-02-13-18.h5
radar-kakq-2013-02-13-05.h5  radar-kakq-2013-02-13-20.h5
radar-kakq-2013-02-13-06.h5  radar-kakq-2013-02-13-21.h5
radar-kakq-2013-02-13-08.h5  radar-kakq-2013-02-13-23.h5
radar-kakq-2013-02-13-09.h5  radar-kakq-2013-02-14-00.h5
radar-kakq-2013-02-13-10.h5  radar-kakq-2013-02-14-01.h5
radar-kakq-2013-02-13-11.h5  radar-kakq-2013-02-14-02.h5
radar-kakq-2013-02-13-12.h5  radar-kakq-2013-02-14-03.h5

7.3.6 Audible Radar-Alerting

AlertVIZ includes the capability to use audible alerting of messages on workstations that have audio capabilities. Adding an audio component to Radar Alerts involves modifying the AlertVIZ configuration on the workstation. To do this, use the configuration dialogs built into AlertVIZ.

a. Right click on the AlertVIZ icon in the system and select Configuration. This will display the Alert Visualization Configuration dialog. (See Exhibit 7.3.6-1.)

Exhibit 7.3.6-1. Alert Visualization Configuration dialog
b. On the Alert Visualization Configuration dialog, select RADAR from the list box under Sources & Priorities at the lower left of the dialog. This will load the alerting information for RADAR into the lower control group. (See Exhibit 7.3.6-2).

![Exhibit 7.3.6-2. Alert Visualization Configuration Sources & Priorities]

```
c. In the Sources & Priorities control group, make sure RADAR in the Sources menu is selected in the list box, and then click the Audio check box under HIGH. (See Exhibit 7.3.6-3.)
```
d. Selecting the *Audio* checkbox activates the *Audio* browse button. Click the *browse* button. This will display the *Audio File Selection* dialog. This dialog displays the currently available audio files. (See Exhibit 7.3.6-4.)

![Exhibit 7.3.6-3. Alert Visualization Configuration Sources & Priorities Audio](image)

![Exhibit 7.3.6-4. Alert Visualization Configuration Sources & Priorities Audio File](image)

e. Select desired sound file and click *OK*. 

---

AWP.MAN.SMM.A2 7-22

– Hard copies uncontrolled. Verify effective date prior to use. –
f. To save the modification to the Alert VIZ configuration, click the **Save As ...** button at the bottom of the *Alert Visualization Configuration* dialog. This opens the *Configuration List* dialog in save mode. (See Exhibit 7.3.6-5.)

![Configuration List](image)

Exhibit 7.3.6-5. Alert Visualization Configuration List

Once the dialog is open, enter a name for the configuration and click **Save**. The *Alert Visualization Configuration* dialog will update to show your new configuration as the current configuration.

Once the configuration has been changed, a restart of Alert VIZ will result in the new configuration being loaded.

Note that the new configuration is saved in the AWIPS-II data localization store for the current user. As a result, the newly updated configuration is available for that user on all workstations.

7.4 **WAN/SBN Radar Product Distribution from Site to NCF**

The AWIPS WAN/SBN distribution capability allows sites to distribute radar products over the WAN to the NCF for redistribution to all AWIPS sites over the SBN. Table 7.4-1 lists the product number and name, levels, layer, resolution and range, product ID, and format for WSR-88D products on AWIPS.
## Table 7.4-1. WSR-88D Products on AWIPS

<table>
<thead>
<tr>
<th>Prod Num</th>
<th>Lvl</th>
<th>Lyr</th>
<th>Resol</th>
<th>Range</th>
<th>Prod ID</th>
<th>Product Name</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>GSM</td>
<td>General Status Msg (GSM)</td>
<td>GSM</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>PRR</td>
<td>Product Request Resp (PRR)</td>
<td>PRR</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>AAP</td>
<td>Alert Adaptation Parms (AAP)</td>
<td>AAP</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>PTL</td>
<td>Product List (PTL)</td>
<td>PTL</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>AM</td>
<td>Alert Message (AM)</td>
<td>AM</td>
</tr>
<tr>
<td>16</td>
<td>8</td>
<td>0</td>
<td>1</td>
<td>230</td>
<td>Z</td>
<td>Reflectivity (Z)</td>
<td>Radial</td>
</tr>
<tr>
<td>17</td>
<td>8</td>
<td>0</td>
<td>2</td>
<td>460</td>
<td>Z</td>
<td>Reflectivity (Z)</td>
<td>Radial</td>
</tr>
<tr>
<td>18</td>
<td>8</td>
<td>0</td>
<td>4</td>
<td>460</td>
<td>Z</td>
<td>Reflectivity (Z)</td>
<td>Radial</td>
</tr>
<tr>
<td>19</td>
<td>16</td>
<td>0</td>
<td>1</td>
<td>230</td>
<td>Z</td>
<td>Reflectivity (Z)</td>
<td>Radial</td>
</tr>
<tr>
<td>20</td>
<td>16</td>
<td>0</td>
<td>2</td>
<td>460</td>
<td>Z</td>
<td>Reflectivity (Z)</td>
<td>Radial</td>
</tr>
<tr>
<td>21</td>
<td>16</td>
<td>0</td>
<td>4</td>
<td>460</td>
<td>Z</td>
<td>Reflectivity (Z)</td>
<td>Radial</td>
</tr>
<tr>
<td>22</td>
<td>8</td>
<td>0</td>
<td>0.25</td>
<td>60</td>
<td>V</td>
<td>Velocity (V)</td>
<td>Radial</td>
</tr>
<tr>
<td>23</td>
<td>8</td>
<td>0</td>
<td>0.5</td>
<td>115</td>
<td>V</td>
<td>Velocity (V)</td>
<td>Radial</td>
</tr>
<tr>
<td>24</td>
<td>8</td>
<td>0</td>
<td>1</td>
<td>230</td>
<td>V</td>
<td>Velocity (V)</td>
<td>Radial</td>
</tr>
<tr>
<td>25</td>
<td>16</td>
<td>0</td>
<td>0.25</td>
<td>60</td>
<td>V</td>
<td>Velocity (V)</td>
<td>Radial</td>
</tr>
<tr>
<td>26</td>
<td>16</td>
<td>0</td>
<td>0.5</td>
<td>115</td>
<td>V</td>
<td>Velocity (V)</td>
<td>Radial</td>
</tr>
<tr>
<td>27</td>
<td>16</td>
<td>0</td>
<td>1</td>
<td>230</td>
<td>V</td>
<td>Velocity (V)</td>
<td>Radial</td>
</tr>
<tr>
<td>28</td>
<td>8</td>
<td>0</td>
<td>0.25</td>
<td>60</td>
<td>SW</td>
<td>Spectrum Width (SW)</td>
<td>Radial</td>
</tr>
<tr>
<td>29</td>
<td>8</td>
<td>0</td>
<td>0.5</td>
<td>115</td>
<td>SW</td>
<td>Spectrum Width (SW)</td>
<td>Radial</td>
</tr>
<tr>
<td>30</td>
<td>8</td>
<td>0</td>
<td>1</td>
<td>230</td>
<td>SW</td>
<td>Spectrum Width (SW)</td>
<td>Radial</td>
</tr>
<tr>
<td>31</td>
<td>16</td>
<td>0</td>
<td>2</td>
<td>230</td>
<td>USP</td>
<td>User Select Precip (USP)</td>
<td>Radial</td>
</tr>
<tr>
<td>32</td>
<td>256</td>
<td>0</td>
<td>1</td>
<td>115</td>
<td>DHR</td>
<td>Digital Hybrid Scan Refl (DHR)</td>
<td>Radial</td>
</tr>
<tr>
<td>33</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>8</td>
<td>0</td>
<td>1</td>
<td>230</td>
<td>CFC</td>
<td>Clutter Filter Control (CFC)</td>
<td>Radial</td>
</tr>
<tr>
<td>35</td>
<td>8</td>
<td>0</td>
<td>1</td>
<td>230</td>
<td>CZ</td>
<td>Composite Ref (CZ)</td>
<td>Raster</td>
</tr>
<tr>
<td>36</td>
<td>8</td>
<td>0</td>
<td>4</td>
<td>460</td>
<td>CZ</td>
<td>Composite Ref (CZ)</td>
<td>Raster</td>
</tr>
<tr>
<td>37</td>
<td>16</td>
<td>0</td>
<td>1</td>
<td>230</td>
<td>CZ</td>
<td>Composite Ref (CZ)</td>
<td>Raster</td>
</tr>
<tr>
<td>38</td>
<td>16</td>
<td>0</td>
<td>4</td>
<td>460</td>
<td>CZ</td>
<td>Composite Ref (CZ)</td>
<td>Raster</td>
</tr>
<tr>
<td>39</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>230</td>
<td>CZC</td>
<td>Composite Ref Contour (CZC)</td>
<td>Graphic</td>
</tr>
<tr>
<td>40</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>460</td>
<td>CZC</td>
<td>Composite Ref Contour (CZC)</td>
<td>Graphic</td>
</tr>
<tr>
<td>41</td>
<td>16</td>
<td>0</td>
<td>4</td>
<td>230</td>
<td>ET</td>
<td>Echo Tops (ET)</td>
<td>Raster</td>
</tr>
<tr>
<td>42</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>230</td>
<td>ETC</td>
<td>Echo Tops Contour (ETC)</td>
<td>Graphic</td>
</tr>
<tr>
<td>43</td>
<td>16</td>
<td>0</td>
<td>1</td>
<td>230</td>
<td>SWR</td>
<td>Svr Wx Anal - Ref (SWR)</td>
<td>Radial</td>
</tr>
<tr>
<td>44</td>
<td>16</td>
<td>0</td>
<td>0.25</td>
<td>230</td>
<td>SWV</td>
<td>Svr Wx Anal - Vel (SWV)</td>
<td>Radial</td>
</tr>
<tr>
<td>45</td>
<td>8</td>
<td>0</td>
<td>0.25</td>
<td>230</td>
<td>SWW</td>
<td>Svr Wx Anal - SW (SWW)</td>
<td>Radial</td>
</tr>
<tr>
<td>46</td>
<td>16</td>
<td>0</td>
<td>0.5</td>
<td>230</td>
<td>SWS</td>
<td>Svr Wx Anal - Shear (SWS)</td>
<td>Radial</td>
</tr>
<tr>
<td>47</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>230</td>
<td>SWP</td>
<td>Severe Wx Prob (SWP)</td>
<td>Graphic</td>
</tr>
<tr>
<td>48</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>VWP</td>
<td>VAD Wind Profile (VWP)</td>
<td>Graphic</td>
</tr>
<tr>
<td>49</td>
<td>16</td>
<td>0</td>
<td>0</td>
<td>26</td>
<td>CM</td>
<td>Combined Moment (CM)</td>
<td>Raster</td>
</tr>
<tr>
<td>50</td>
<td>16</td>
<td>0</td>
<td>1</td>
<td>230</td>
<td>RCS</td>
<td>Ref X-Sect (RCS)</td>
<td>Raster</td>
</tr>
<tr>
<td>51</td>
<td>16</td>
<td>0</td>
<td>0.5</td>
<td>230</td>
<td>VCS</td>
<td>Vel X-Sect (VCS)</td>
<td>Raster</td>
</tr>
<tr>
<td>Prod Num</td>
<td>Lvls</td>
<td>Lyr</td>
<td>Resol</td>
<td>Range</td>
<td>Prod ID</td>
<td>Product Name</td>
<td>Format</td>
</tr>
<tr>
<td>----------</td>
<td>------</td>
<td>-----</td>
<td>-------</td>
<td>-------</td>
<td>--------</td>
<td>----------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>52</td>
<td>8</td>
<td>0</td>
<td>0.5</td>
<td>230</td>
<td>SCS</td>
<td>SW X-Sect (SCS)</td>
<td>Raster</td>
</tr>
<tr>
<td>53</td>
<td>8</td>
<td>0</td>
<td>1</td>
<td>50</td>
<td>WER</td>
<td>Wk Echo Region (WER)</td>
<td>Raster</td>
</tr>
<tr>
<td>55</td>
<td>16</td>
<td>0</td>
<td>0.5</td>
<td>230</td>
<td>SRR</td>
<td>Storm Rel Vel Region (SRR)</td>
<td>Radial</td>
</tr>
<tr>
<td>56</td>
<td>16</td>
<td>0</td>
<td>1</td>
<td>230</td>
<td>SRM</td>
<td>Storm Rel Velocity (SRM)</td>
<td>Radial</td>
</tr>
<tr>
<td>57</td>
<td>16</td>
<td>0</td>
<td>4</td>
<td>230</td>
<td>VIL</td>
<td>Vert Integ Liq (VIL)</td>
<td>Raster</td>
</tr>
<tr>
<td>58</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>345</td>
<td>STI</td>
<td>Storm Track (STI)</td>
<td>Graphic</td>
</tr>
<tr>
<td>59</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>230</td>
<td>HI</td>
<td>Hail Index (HI)</td>
<td>Graphic</td>
</tr>
<tr>
<td>60</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>230</td>
<td>M</td>
<td>Mesocyclone (M)</td>
<td>Graphic</td>
</tr>
<tr>
<td>61</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>230</td>
<td>TVS</td>
<td>Tornadic Vortex Sig (TVS)</td>
<td>Graphic</td>
</tr>
<tr>
<td>62</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>460</td>
<td>SS</td>
<td>Storm Structure (SS)</td>
<td>Text</td>
</tr>
<tr>
<td>65</td>
<td>8</td>
<td>1</td>
<td>4</td>
<td>460</td>
<td>LRM</td>
<td>Lyr 1 Comp Ref Max (LRM)</td>
<td>Raster</td>
</tr>
<tr>
<td>66</td>
<td>8</td>
<td>2</td>
<td>4</td>
<td>460</td>
<td>LRM</td>
<td>Lyr 2 Comp Ref Max (LRM)</td>
<td>Raster</td>
</tr>
<tr>
<td>67</td>
<td>8</td>
<td>2</td>
<td>4</td>
<td>460</td>
<td>APR</td>
<td>Lyr 1 Comp Ref MAX (APR)</td>
<td>Raster</td>
</tr>
<tr>
<td>73</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>UAM</td>
<td>User Alert Message (UAM)</td>
<td>Text</td>
</tr>
<tr>
<td>74</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>460</td>
<td>RCM</td>
<td>Radar Coded Message (RCM)</td>
<td>Text</td>
</tr>
<tr>
<td>75</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>FTM</td>
<td>Free Text Message (FTM)</td>
<td>Text</td>
</tr>
<tr>
<td>77</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>PTM</td>
<td>PUP Text Message (PTM)</td>
<td>Text</td>
</tr>
<tr>
<td>78</td>
<td>16</td>
<td>0</td>
<td>2</td>
<td>230</td>
<td>OHP</td>
<td>One Hour Precip (OHP)</td>
<td>Radial</td>
</tr>
<tr>
<td>79</td>
<td>16</td>
<td>0</td>
<td>2</td>
<td>230</td>
<td>TPH</td>
<td>Three Hour Precip (TPH)</td>
<td>Radial</td>
</tr>
<tr>
<td>80</td>
<td>16</td>
<td>0</td>
<td>2</td>
<td>230</td>
<td>STP</td>
<td>Storm Total Precip (STP)</td>
<td>Radial</td>
</tr>
<tr>
<td>81</td>
<td>256</td>
<td>0</td>
<td>4</td>
<td>230</td>
<td>DPA</td>
<td>Digital Precip Array (DPA)</td>
<td>Raster</td>
</tr>
<tr>
<td>82</td>
<td>8</td>
<td>0</td>
<td>40</td>
<td>230</td>
<td>SPD</td>
<td>Supplemental Precip Data (SPD)</td>
<td>Text</td>
</tr>
<tr>
<td>83</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>IRM</td>
<td>Intermediate Radar Message (IRM)</td>
<td>Text</td>
</tr>
<tr>
<td>84</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>VAD</td>
<td>Vel Az Display (VAD)</td>
<td>Graphic</td>
</tr>
<tr>
<td>85</td>
<td>8</td>
<td>0</td>
<td>1</td>
<td>230</td>
<td>RCS</td>
<td>Ref X-Sect (RCS)</td>
<td>Raster</td>
</tr>
<tr>
<td>86</td>
<td>8</td>
<td>0</td>
<td>0.5</td>
<td>230</td>
<td>VCS</td>
<td>Vel X-Sect (VCS)</td>
<td>Raster</td>
</tr>
<tr>
<td>87</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>CS</td>
<td>Combined Shear (CS)</td>
<td>Raster</td>
</tr>
<tr>
<td>88</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>CSC</td>
<td>Combined Shear Contour (CSC)</td>
<td>Graphic</td>
</tr>
<tr>
<td>89</td>
<td>8</td>
<td>3</td>
<td>4</td>
<td>460</td>
<td>LRA</td>
<td>Lyr 3 Comp Ref Avg (LRA)</td>
<td>Raster</td>
</tr>
<tr>
<td>90</td>
<td>8</td>
<td>3</td>
<td>4</td>
<td>460</td>
<td>LRM</td>
<td>Lyr 3 Comp Ref Max (LRM)</td>
<td>Raster</td>
</tr>
<tr>
<td>93</td>
<td>256</td>
<td>0</td>
<td>1.00</td>
<td>115</td>
<td>DBV</td>
<td>ITWS Digital Velocity (DBV)</td>
<td>Radial</td>
</tr>
<tr>
<td>94</td>
<td>256</td>
<td>0</td>
<td>1.00</td>
<td>460</td>
<td>DZ</td>
<td>8-bit Refl Array (DZ)</td>
<td>Radial</td>
</tr>
<tr>
<td>99</td>
<td>256</td>
<td>0</td>
<td>0.25</td>
<td>230</td>
<td>V</td>
<td>8-bit Velocity Array (V)</td>
<td>Radial</td>
</tr>
<tr>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>VSDT</td>
<td>VAD Site Adapt Params (VSDT)</td>
<td>Text</td>
</tr>
<tr>
<td>101</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>STIT</td>
<td>Storm Track Alpha block (STIT)</td>
<td>Text</td>
</tr>
<tr>
<td>102</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>HIT</td>
<td>Hail Index Alpha block (HIT)</td>
<td>Text</td>
</tr>
<tr>
<td>103</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>MT</td>
<td>Mesocyclone Alpha block (MT)</td>
<td>Text</td>
</tr>
<tr>
<td>104</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>TVST</td>
<td>TVS Alpha block (TVST)</td>
<td>Text</td>
</tr>
<tr>
<td>105</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>CST</td>
<td>Combined Shear Params (CST)</td>
<td>Text</td>
</tr>
<tr>
<td>106</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>CSCT</td>
<td>Combined Shr Cntr Parms (CSCT)</td>
<td>Text</td>
</tr>
<tr>
<td>107</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>OHPT</td>
<td>1hr Rainfall Params (OHPT)</td>
<td>Text</td>
</tr>
<tr>
<td>108</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>THPT</td>
<td>3hr Rainfall Params (THPT)</td>
<td>Text</td>
</tr>
<tr>
<td>Prod Num</td>
<td>Lys</td>
<td>Lyr</td>
<td>Resol</td>
<td>Range</td>
<td>Prod ID</td>
<td>Product Name</td>
<td>Format</td>
</tr>
<tr>
<td>----------</td>
<td>-----</td>
<td>-----</td>
<td>-------</td>
<td>-------</td>
<td>---------</td>
<td>--------------</td>
<td>--------</td>
</tr>
<tr>
<td>109</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>STPT</td>
<td>Storm Total Params (STPT)</td>
<td>Text</td>
</tr>
<tr>
<td>136</td>
<td>16</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>CZFMP</td>
<td>Comp Refl Mos Filt Prec (CZFMP)</td>
<td>Raster</td>
</tr>
<tr>
<td>137</td>
<td>16</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>CZFMP</td>
<td>Comp Refl Mos Filt Prec (CZFMP)</td>
<td>Raster</td>
</tr>
<tr>
<td>139</td>
<td>16</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>CZM</td>
<td>Composite Refl Mosaic (CZM)</td>
<td>Raster</td>
</tr>
<tr>
<td>144</td>
<td>16</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>CZM</td>
<td>Composite Refl Mosaic (CZM)</td>
<td>Raster</td>
</tr>
<tr>
<td>153</td>
<td>256</td>
<td>0</td>
<td>0.25</td>
<td>460</td>
<td>HZ</td>
<td>Super-Res Reflectivity (Z)</td>
<td>Radial</td>
</tr>
<tr>
<td>154</td>
<td>256</td>
<td>0</td>
<td>0.25</td>
<td>300</td>
<td>HV</td>
<td>Super-Res Velocity (V)</td>
<td>Radial</td>
</tr>
<tr>
<td>155</td>
<td>256</td>
<td>0</td>
<td>0.25</td>
<td>300</td>
<td>HSW</td>
<td>Super-Res Spec Width (SW)</td>
<td>Radial</td>
</tr>
<tr>
<td>158</td>
<td>16</td>
<td>0</td>
<td>0.25</td>
<td>300</td>
<td>ZDR</td>
<td>Differential Refl (ZDR)</td>
<td>Raster</td>
</tr>
<tr>
<td>159</td>
<td>8</td>
<td>0</td>
<td>4</td>
<td>300</td>
<td>CZFMP</td>
<td>Comp Refl Mos Filt Prec (CZFMP)</td>
<td>Raster</td>
</tr>
<tr>
<td>160</td>
<td>16</td>
<td>0</td>
<td>1.0</td>
<td>230</td>
<td>CC</td>
<td>Correlation Coeff (CC)</td>
<td>Radial</td>
</tr>
<tr>
<td>161</td>
<td>256</td>
<td>0</td>
<td>0.25</td>
<td>300</td>
<td>CC</td>
<td>Correlation Coeff (CC)</td>
<td>Radial</td>
</tr>
<tr>
<td>162</td>
<td>16</td>
<td>0</td>
<td>1.0</td>
<td>230</td>
<td>KDP</td>
<td>Specific Diff Phase (KDP)</td>
<td>Radial</td>
</tr>
<tr>
<td>163</td>
<td>256</td>
<td>0</td>
<td>0.25</td>
<td>300</td>
<td>KDP</td>
<td>Specific Diff Phase (KDP)</td>
<td>Radial</td>
</tr>
<tr>
<td>164</td>
<td>16</td>
<td>0</td>
<td>1.0</td>
<td>230</td>
<td>HC</td>
<td>Hydrometer Class (HC)</td>
<td>Radial</td>
</tr>
<tr>
<td>165</td>
<td>256</td>
<td>0</td>
<td>0.25</td>
<td>300</td>
<td>HC</td>
<td>Hydrometer Class (HC)</td>
<td>Radial</td>
</tr>
<tr>
<td>166</td>
<td>0</td>
<td>0</td>
<td>0.0</td>
<td>230</td>
<td>ML</td>
<td>Melting Layer (ML)</td>
<td>Graphic</td>
</tr>
<tr>
<td>169</td>
<td>16</td>
<td>0</td>
<td>2.0</td>
<td>230</td>
<td>OHA</td>
<td>One Hour Accum (STA)</td>
<td>Radial</td>
</tr>
<tr>
<td>170</td>
<td>256</td>
<td>0</td>
<td>0.25</td>
<td>230</td>
<td>DAA</td>
<td>One Hour Unbiased Accum (DAA)</td>
<td>Radial</td>
</tr>
<tr>
<td>171</td>
<td>16</td>
<td>0</td>
<td>2.0</td>
<td>230</td>
<td>STA</td>
<td>Storm Total Accum (STA)</td>
<td>Radial</td>
</tr>
<tr>
<td>172</td>
<td>256</td>
<td>0</td>
<td>0.25</td>
<td>230</td>
<td>STA</td>
<td>Storm Total Accum (DSA)</td>
<td>Radial</td>
</tr>
<tr>
<td>173</td>
<td>256</td>
<td>0</td>
<td>0.25</td>
<td>230</td>
<td>DUA</td>
<td>User Select Accum (DUA)</td>
<td>Radial</td>
</tr>
<tr>
<td>174</td>
<td>256</td>
<td>0</td>
<td>0.25</td>
<td>230</td>
<td>DOD</td>
<td>One Hour Diff (DOD)</td>
<td>Radial</td>
</tr>
<tr>
<td>175</td>
<td>256</td>
<td>0</td>
<td>0.25</td>
<td>230</td>
<td>DSD</td>
<td>Storm Total Diff (DSD)</td>
<td>Radial</td>
</tr>
<tr>
<td>176</td>
<td>65536</td>
<td>0</td>
<td>0.25</td>
<td>230</td>
<td>DPR</td>
<td>Digital Inst Precip rate (DPR)</td>
<td>Radial</td>
</tr>
<tr>
<td>177</td>
<td>256</td>
<td>0</td>
<td>0.25</td>
<td>230</td>
<td>HHC</td>
<td>Hybrid Hydrometer Class (HHC)</td>
<td>Radial</td>
</tr>
<tr>
<td>199</td>
<td>16</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>OHPM</td>
<td>One Hour Precip Mosaic (OHPM)</td>
<td>Raster</td>
</tr>
<tr>
<td>200</td>
<td>16</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>OHPM</td>
<td>One Hour Precip Mosaic (OHPM)</td>
<td>Raster</td>
</tr>
<tr>
<td>204</td>
<td>16</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>ETM</td>
<td>Echo Tops Mosaic (ETM)</td>
<td>Raster</td>
</tr>
<tr>
<td>205</td>
<td>16</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>ETM</td>
<td>Echo Tops Mosaic (ETM)</td>
<td>Raster</td>
</tr>
<tr>
<td>208</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>CSAM9</td>
<td>Comb Storms Att Mosaic B9 (CSAM9)</td>
<td>Raster</td>
</tr>
<tr>
<td>209</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>CSAM</td>
<td>Comb Storms Att Mosaic (CSAM)</td>
<td>Raster</td>
</tr>
<tr>
<td>210</td>
<td>16</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>CPAM</td>
<td>Cum Precip Accum Mosaic (CPAM)</td>
<td>Raster</td>
</tr>
<tr>
<td>211</td>
<td>16</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>ODPM</td>
<td>24-Hr Precip Mosaic (ODPM)</td>
<td>Raster</td>
</tr>
<tr>
<td>212</td>
<td>16</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>CZMP</td>
<td>Comp Refl Mosaic Precip (CZMP)</td>
<td>Raster</td>
</tr>
<tr>
<td>213</td>
<td>16</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>CZMP</td>
<td>Comp Refl Mosaic Precip (CZMP)</td>
<td>Raster</td>
</tr>
<tr>
<td>216</td>
<td>16</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>CPAM</td>
<td>Cum Precip Accm Mosaic (CPAM)</td>
<td>Raster</td>
</tr>
<tr>
<td>217</td>
<td>16</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>ODPM</td>
<td>24-Hr Precip Mosaic (ODPM)</td>
<td>Raster</td>
</tr>
<tr>
<td>246</td>
<td>16</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>ZNM</td>
<td>Refl Mos Non Filt (ZNM)</td>
<td>Raster</td>
</tr>
<tr>
<td>247</td>
<td>16</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>ZNMC</td>
<td>Refl Mos Non Filt Clr (ZNMC)</td>
<td>Raster</td>
</tr>
<tr>
<td>249</td>
<td>16</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>ZM</td>
<td>Base Reflectivity Mosaic (ZM)</td>
<td>Raster</td>
</tr>
<tr>
<td>250</td>
<td>16</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>ZM</td>
<td>Base Reflectivity Mosaic (ZM)</td>
<td>Raster</td>
</tr>
</tbody>
</table>
### Ingest of Radar Data

<table>
<thead>
<tr>
<th>Prod Num</th>
<th>Lvl</th>
<th>Lyr</th>
<th>Resol</th>
<th>Range</th>
<th>Prod ID</th>
<th>Product Name</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>251</td>
<td>16</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>ZM</td>
<td>Base Reflectivity Mosaic (ZM)</td>
<td>Raster</td>
</tr>
<tr>
<td>253</td>
<td>16</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>ZMGA</td>
<td>Base Refl Mos Griff+Alb (ZMGA)</td>
<td>Raster</td>
</tr>
<tr>
<td>254</td>
<td>8</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>LRMM</td>
<td>Layer Comp Refl Low Mos (VILM)</td>
<td>Raster</td>
</tr>
<tr>
<td>255</td>
<td>8</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>LRMM</td>
<td>Layer Comp Refl Med Mos (VILM)</td>
<td>Raster</td>
</tr>
<tr>
<td>256</td>
<td>8</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>LRMM</td>
<td>Layer Comp Refl High Mos (VILM)</td>
<td>Raster</td>
</tr>
<tr>
<td>257</td>
<td>16</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>VILM</td>
<td>Vert Int Liquid Mosaic (VILM)</td>
<td>Raster</td>
</tr>
</tbody>
</table>

**NOTE:** Level III data are radar products generated from the Level II base data. The products are used to assist forecasters and others in weather analysis, predictions, and warnings. Level III products used to be recorded on WORM Optical Disks at NWS sites in the early 1990s. Currently, the required products are provided electronically in near real-time to an NWS Central Collection Facility (CCF). When products were written to the WORM Optical Disks, all products were saved and archived. Now, only required products are collected at the CCF and then archived by National Climatic Data Center (NCDC). Regardless of how the Level III products have been received, the NCDC archives the products in compressed tape archive format on the NCDC Hierarchical Data Storage System (HDSS). The Level III Products archive is available via the World Wide Web on the NCDC Home Page (http://www.ncdc.noaa.gov/).

For more information on this subject, go to:

http://www.roc.noaa.gov/WSR88D/Level_III/Level3Info.aspx

For WSR-88 D additional and discontinued products, go to

http://www.roc.noaa.gov/WSR88D/Level_III/Level3Info.aspx

Scroll down to the section titled “Changes That May Impact RPCCDS Users” and click on the notices.

### 7.5 RPS list

The RPS list is edited using a Graphical User Interface (GUI) that is part of CAVE. It is accessed via the Radar menu in the D2D perspective.

**NOTE:** When you add products to the RPS list with the AWIPS RPS List Editor, you specify the specific parameters but not the Product Number. If the product is entered in the RPS list with incorrect parameters, it will be interpreted as a different product and not distributed automatically to the WAN. It is very important that the data level, resolution, and elevation be specified correctly in the RPS list or the product will not be distributed properly.

To edit the RPS list:

1. Start CAVE and, if necessary, open the D2D perspective.
2. Select *RPS List Editor*... under the Radar menu. The RPS List Editor will display. (See Exhibit 7.5-1.)

![RPS List Editor](image)

Exhibit 7.5-1. RPS List Editor
3. Select the list you want to edit under the View Menu. (See Exhibit 7.5-2.)

Exhibit 7.5-2. RPS List Editor View Menu – Select List
4. Then select the RPG. (See Exhibit 7.5-3.)

Exhibit 7.4-3. Select RPS to View
5. After selecting the desired list, it will be displayed in the RPS List Editor. (See Exhibit 7.5-4.)

Exhibit 7.5-4. RPS List Editor View Menu List
6. To edit a list entry, select the entry and click *Edit*. Once the entry is displayed in a separate dialog, modify the appropriate values and click *OK*. (See Exhibit 7.5-5.)

![Exhibit 7.5-5. RPS List Editor View Menu List Edit](image-url)
7. To add a new list entry, click New. Once the dialog is displayed, select the appropriate entries and click OK. (See Exhibit 7.5-6.)

Exhibit 7.5-6. RPS List Editor View Menu List New Entry

8. To delete a list entry, select the entry and click Remove.

9. To submit the list, click Send.

7.6 Two-Way Communication

As illustrated in Exhibit 7.6-1, the Quartz timer task in EDEX initiates either environmental data or bias table task. The EDEX queries the RadarServer for the list of ORPGs/SPGs (For the environmental data task, EDEX queries the grib data set based on the settings in EnvironParamsLevelTable.xml and generates environmental data message. For the bias table task, EDEX looks for bias table messages that were generated by the hydro software). The EDEX sends the messages to the RadarServer and the RadarServer sends the messages to the ORPGs/SPGs.
As illustrated in Exhibit 7.6-2:

- **For OTRs:** The CAVE sends requests to the RadarServer which are forwarded to the ORPG and the RadarServer sends OTR status back to CAVE (Not shown).

- **For RMRs:** The CAVE reads and writes the list of available RMRs using the EDEX localization requests. Then the Requests to activate/deactivate RMR requests are sent to the RadarServer. The RadarServer sends the OTR requests to the ORPGs/SPGs according to the contents of the RMR and the RadarServer sends RMR status events back to CAVE while the RMR window is open (not shown).

- **For RPS:** The CAVE queries the RadarServer for the current RPS list. To activate an RPS list, CAVE first sends a request to the RadarServer which then sends an RPS message to the ORPG/SPG.

- **For Alert Requests:** The CAVE queries the RadarServer for the current alert request state and then CAVE sends the updated alert requests to the RadarServer. The RadarServer stores the alert requests in `/awips2/rcm/data/config/persist/` and then sends an alert request message to the ORPG.
7.7 **Central Collection Cron Schedule**

The two cron entries for Central Collection Cron Schedule in cronOTRs.xml are as follows.

```xml
<cronOTR cron="0 10 * * * ?" productCode="173" randomWait="300" hoursBack="3" wmo="SDUS8" nnn="DU3"/>
<cronOTR cron="0 10 12 * * ?" productCode="173" randomWait="300" hoursBack="24" wmo="SDUS8" nnn="DU6"/>.
```
Chapter 8
Local Data Acquisition and Dissemination (LDAD) System
Chapter 8. Local Data Acquisition and Dissemination (LDAD) System

Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.1</td>
<td>LDAD Functionality</td>
<td>1</td>
</tr>
<tr>
<td>8.2</td>
<td>LDAD Hardware Configuration</td>
<td>2</td>
</tr>
<tr>
<td>8.2.1</td>
<td>LAN Hub</td>
<td>2</td>
</tr>
<tr>
<td>8.2.2</td>
<td>LDAD Terminal Server</td>
<td>2</td>
</tr>
<tr>
<td>8.2.3</td>
<td>Modem Nest</td>
<td>4</td>
</tr>
<tr>
<td>8.2.4</td>
<td>LDAD Firewall</td>
<td>4</td>
</tr>
<tr>
<td>8.2.5</td>
<td>LDAD Server</td>
<td>5</td>
</tr>
<tr>
<td>8.3</td>
<td>LDAD Software Configuration</td>
<td>5</td>
</tr>
<tr>
<td>8.3.1</td>
<td>LDAD Gateway</td>
<td>7</td>
</tr>
<tr>
<td>8.3.2</td>
<td>Scheduler</td>
<td>10</td>
</tr>
<tr>
<td>8.3.3</td>
<td>Communications Server</td>
<td>11</td>
</tr>
<tr>
<td>8.3.4</td>
<td>Inter-Gateway Communication</td>
<td>11</td>
</tr>
<tr>
<td>8.3.5</td>
<td>Local Data Management</td>
<td>11</td>
</tr>
<tr>
<td>8.3.6</td>
<td>Preprocessors</td>
<td>12</td>
</tr>
<tr>
<td>8.3.7</td>
<td>Decoder and Storage</td>
<td>13</td>
</tr>
<tr>
<td>8.4</td>
<td>Rsync</td>
<td>13</td>
</tr>
<tr>
<td>8.4.1</td>
<td>Rsync with AWIPS (internal) and External (non-AWIPS) Systems</td>
<td>13</td>
</tr>
<tr>
<td>8.4.2</td>
<td>Rsync Configuration Files</td>
<td>14</td>
</tr>
<tr>
<td>8.5</td>
<td>LDAD Linux Server Heartbeat Cron Package</td>
<td>14</td>
</tr>
<tr>
<td>8.6</td>
<td>Quality Control and System Monitoring</td>
<td>16</td>
</tr>
<tr>
<td>8.6.1</td>
<td>Quality Control</td>
<td>16</td>
</tr>
<tr>
<td>8.6.2</td>
<td>System Monitoring</td>
<td>18</td>
</tr>
<tr>
<td>8.7</td>
<td>Acquire LARC/Handar 555 Data</td>
<td>19</td>
</tr>
<tr>
<td>8.7.1</td>
<td>User Interface: ldadScheduler</td>
<td>23</td>
</tr>
<tr>
<td>8.7.2</td>
<td>ldadServer</td>
<td>23</td>
</tr>
<tr>
<td>8.7.3</td>
<td>tell_co</td>
<td>26</td>
</tr>
<tr>
<td>8.7.4</td>
<td>listener</td>
<td>26</td>
</tr>
<tr>
<td>8.7.5</td>
<td>CO_serv</td>
<td>28</td>
</tr>
<tr>
<td>8.8</td>
<td>Decode LARC/Handar 555 Data</td>
<td>29</td>
</tr>
<tr>
<td>8.8.1</td>
<td>newLDADDataNotification</td>
<td>30</td>
</tr>
<tr>
<td>8.8.2</td>
<td>CO_serv</td>
<td>33</td>
</tr>
<tr>
<td>8.8.3</td>
<td>listener</td>
<td>34</td>
</tr>
</tbody>
</table>
8.8.4 preProcessLDAD.pl ................................................................. 35
8.8.5 preprocessLARC.pl ............................................................... 36
8.8.6 LDAD_ROUTER ................................................................. 36
8.8.7 DataController ................................................................. 37
8.8.8 routerLdadDecoder ........................................................... 39
8.8.9 routerStoreTextEDEX ......................................................... 40
8.8.10 routerStoreEDEX ............................................................ 41
8.8.11 routerShefEncoderEDEX .................................................. 42
8.8.12 distributeProduct ............................................................ 43

8.9 Acquire Campbell Data .......................................................... 44
8.9.1 User Interface: ldadScheduler .............................................. 49
8.9.2 ldadServer .......................................................................... 49
8.9.3 tell_co ................................................................................. 51
8.9.4 listener .............................................................................. 52
8.9.5 CO_serv ............................................................................. 53
8.9.6 process_expect ................................................................. 54
8.9.7 callSite.ksh ....................................................................... 55
8.9.8 campbell decoder .............................................................. 55

8.10 Decode Campbell Data .......................................................... 56
8.10.1 newLDADdataNotification ................................................. 57
8.10.2 CO_serv .......................................................................... 60
8.10.3 listener ............................................................................ 61
8.10.4 preprocessCAMPBELL.pl ..................................................... 63
8.10.5 LDAD_ROUTER ............................................................... 64
8.10.6 DataController ............................................................... 65
8.10.7 routerLdadDecoder .......................................................... 66
8.10.8 routerStoreEDEX ............................................................ 67
8.10.9 routerShefEncoderEDEX ................................................... 68
8.10.10 distributeProduct ............................................................ 70
8.10.11 notificationServer ............................................................ 71

8.11 Acquire Sutron Data ............................................................. 72
8.11.1 User Interface: ldadScheduler ............................................. 78
8.11.2 ldadServer ...................................................................... 78
8.11.3 tell_co ........................................................................... 79
8.11.4 listener ......................................................................... 80
8.11.5 CO_serv ....................................................................... 81
8.11.6 process_expect ............................................................. 83
8.11.7 callSite.ksh ................................................................... 83
<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.11.8</td>
<td>sutron decoder</td>
<td>83</td>
</tr>
<tr>
<td>8.12</td>
<td>Decode Sutron Data</td>
<td>84</td>
</tr>
<tr>
<td>8.12.1</td>
<td>newLDADdataNotification</td>
<td>85</td>
</tr>
<tr>
<td>8.12.1</td>
<td>CO_serv</td>
<td>88</td>
</tr>
<tr>
<td>8.12.2</td>
<td>listener</td>
<td>89</td>
</tr>
<tr>
<td>8.12.3</td>
<td>preprocessSUTRON.pl</td>
<td>90</td>
</tr>
<tr>
<td>8.12.4</td>
<td>LDAD_ROUTER</td>
<td>91</td>
</tr>
<tr>
<td>8.12.5</td>
<td>DataController</td>
<td>92</td>
</tr>
<tr>
<td>8.12.6</td>
<td>routerLdadDecoder</td>
<td>93</td>
</tr>
<tr>
<td>8.12.7</td>
<td>routerStoreEDEX</td>
<td>94</td>
</tr>
<tr>
<td>8.12.8</td>
<td>routerShefEncoderEDEX</td>
<td>95</td>
</tr>
<tr>
<td>8.12.9</td>
<td>distributeProduct</td>
<td>96</td>
</tr>
<tr>
<td>8.12.10</td>
<td>notificationServer</td>
<td>98</td>
</tr>
<tr>
<td>8.13</td>
<td>Acquire and Decode Comma Separated Variable Mesonet Data</td>
<td>100</td>
</tr>
<tr>
<td>8.13.1</td>
<td>CO_serv</td>
<td>103</td>
</tr>
<tr>
<td>8.13.2</td>
<td>newLDADdataNotification</td>
<td>104</td>
</tr>
<tr>
<td>8.13.3</td>
<td>listener</td>
<td>105</td>
</tr>
<tr>
<td>8.13.4</td>
<td>preProcessLDAD.pl</td>
<td>106</td>
</tr>
<tr>
<td>8.13.5</td>
<td>preprocessmesonet.pl</td>
<td>107</td>
</tr>
<tr>
<td>8.13.6</td>
<td>LDAD_ROUTER</td>
<td>107</td>
</tr>
<tr>
<td>8.13.7</td>
<td>DataController</td>
<td>108</td>
</tr>
<tr>
<td>8.13.8</td>
<td>routerLdadDecoder</td>
<td>110</td>
</tr>
<tr>
<td>8.13.9</td>
<td>routerStoreEDEX</td>
<td>111</td>
</tr>
<tr>
<td>8.14</td>
<td>Acquire and Process ASOS/MicroART Data (for LDAD)</td>
<td>113</td>
</tr>
<tr>
<td>8.14.1</td>
<td>suaReceiver</td>
<td>116</td>
</tr>
<tr>
<td>8.14.2</td>
<td>newLDADdataNotification</td>
<td>117</td>
</tr>
<tr>
<td>8.14.3</td>
<td>CO_serv</td>
<td>118</td>
</tr>
<tr>
<td>8.14.4</td>
<td>listener</td>
<td>119</td>
</tr>
<tr>
<td>8.14.5</td>
<td>preprocessSUA.pl</td>
<td>120</td>
</tr>
<tr>
<td>8.14.6</td>
<td>Reserved</td>
<td>121</td>
</tr>
<tr>
<td>8.14.7</td>
<td>distributeProduct</td>
<td>121</td>
</tr>
<tr>
<td>8.15</td>
<td>Using LDAD Scheduler</td>
<td>122</td>
</tr>
<tr>
<td>8.15.1</td>
<td>Using LDAD Scheduler to Modify Gauges with Session Files</td>
<td>122</td>
</tr>
<tr>
<td>8.15.2</td>
<td>Using LDAD Scheduler to Add Users</td>
<td>128</td>
</tr>
<tr>
<td>8.16</td>
<td>Alaska Profiler Data</td>
<td>133</td>
</tr>
<tr>
<td>8.16.1</td>
<td>newLDADdataNotification</td>
<td>134</td>
</tr>
<tr>
<td>8.16.2</td>
<td>CO_serv</td>
<td>136</td>
</tr>
<tr>
<td>8.16.3</td>
<td>listener</td>
<td>137</td>
</tr>
</tbody>
</table>
8.16.4 LDAD_ROUTER ................................................................................................... 138
8.16.5 DataController ................................................................................................. 139
8.16.6 routerStoreAkBlpEDEX .................................................................................. 141
8.16.7 notificationServer ........................................................................................... 142
8.17 Acquire and Process Radiosonde Replacement System (RRS) Data (for LDAD) .... 143
  8.17.1 newLDADdataNotification .............................................................................. 143
  8.17.2 CO_serv ......................................................................................................... 146
  8.17.3 listener .......................................................................................................... 147
  8.17.4 preprocessRRS.pl ......................................................................................... 148
  8.17.5 Reserved ....................................................................................................... 150
  8.17.6 distributeProduct ......................................................................................... 150
8.18 LDM Setup and Data Ingest Configuration .......................................................... 151
  8.18.1 LDM Setup ................................................................................................... 151
  8.18.2 LDM Data Ingest Configuration .................................................................. 152
  8.18.3 LDM Processes and Configuration Files ..................................................... 153
  8.18.4 LDM Product-Queue ................................................................................... 154
  8.18.5 Useful LDM Commands .............................................................................. 155
8.19 LDAD and PX2 ..................................................................................................... 156
  8.19.1 LDADinfo.txt .............................................................................................. 157
  8.19.2 PX2 Log Files .............................................................................................. 157
8.20 LDAD Troubleshooting ....................................................................................... 159
  8.20.1 LDAD Disk Removal ................................................................................... 159
  8.20.2 LDAD Server Disk Mirroring Procedure ..................................................... 159
  8.20.3 Fax Modem Troubleshooting ....................................................................... 161

List of Exhibits

Exhibit 8.2-1. LDAD Hardware Configuration ........................................................... 3
Exhibit 8.3-1. Flow of LDAD Data to EDEX ............................................................. 6
Exhibit 8.3.1-1. Generic Acquire and Decode LDAD Product (Page 1 of 2) ................. 8
Exhibit 8.3.1-1. Generic Acquire and Decode LDAD Product (Page 2 of 2) ................. 9
Exhibit 8.3.5-1. LDM Ingest ................................................................................... 12
Exhibit 8.7-1. Acquire LARC/Handar 555 Data (for LDAD) ....................................... 24
Exhibit 8.8-1. Decode LARC/Handar 555 Data (for LDAD) (Page 1 of 2) .................. 31
Exhibit 8.8-1. Decode LARC/Handar 555 Data (for LDAD) (Page 2 of 2) .................. 32
Exhibit 8.9-1. Acquire Campbell Data (for LDAD) ................................................... 50
Exhibit 8.10-1. Decode Campbell Data (for LDAD) (Page 1 of 2) ............................. 58
Exhibit 8.10-1. Decode Campbell Data (for LDAD) (Page 2 of 2) ............................. 59
Exhibit 8.11-1. Acquire Sutron Data (for LDAD) ............................................................................ 77
Exhibit 8.12-1. Decode Sutron Data (for LDAD) (Page 1 of 2) ....................................................... 86
Exhibit 8.12-1. Decode Sutron Data (for LDAD) (Page 2 of 2) ....................................................... 87
Exhibit 8.13-1. Acquire and Decode Comma Separated Variable Mesonet Data (for LDAD) (Page 1 of 2) .................................................................................................................... 101
Exhibit 8.13-1. Acquire and Decode Comma Separated Variable Mesonet Data (for LDAD) (Page 2 of 2) .................................................................................................................... 102
Exhibit 8.14-1. Acquire and Process ASOS/MicroART Data (for LDAD) (Page 1 of 2)............. 114
Exhibit 8.14-1. Acquire and Process ASOS/MicroART Data (for LDAD) (Page 2 of 2)............. 115
Exhibit 8.15.1-1. The Modify Gauges Dialog Box ........................................................................ 123
Exhibit 8.15.1-2. Campbell-Specific Information .......................................................................... 124
Exhibit 8.15.1-3. Sutron-Specific Information ............................................................................... 124
Exhibit 8.15.1-4. The Edit Storage Area Reporting Format Dialog Box ...................................... 125
Exhibit 8.15.1-5. The Choose Session File Dialog Box ................................................................. 126
Exhibit 8.15.1-6. The Edit Session Dialog Box with the Contents of a Session File .................... 127
Exhibit 8.15.1-7. The Copy File Contents Dialog Box ................................................................. 128
Exhibit 8.15.2-1. The Modify Users Dialog Box .......................................................................... 129
Exhibit 8.15.2-2. The New User Dialog Box ............................................................................... 130
Exhibit 8.15.2-3. The Status Dialog Box ..................................................................................... 132
Exhibit 8.16-1. Decode Alaska Profiler Data .............................................................................. 135
Exhibit 8.17-1. Acquire and Process RRS Data (for LDAD) (Page 1 of 2) ................................. 144
Exhibit 8.17-1. Acquire and Process RRS Data (for LDAD) (Page 2 of 2) ................................. 145

List of Tables

Table 8.2.2-1. Port Assignments ................................................................................................. 4
Table 8.3.1-1. LDAD Environment Variables on LDAD Cluster (LS) and/or PX1, PX2 .......... 7
The Local Data Acquisition and Dissemination (LDAD) system provides the means to acquire local data sets, perform quality control on the incoming data, and disseminate weather data to the external user community. It contains a number of components that reside on the internal AWIPS network (on the PX) and on the external LDAD component (on the LDAD server [LS] cluster). The internal and external components are separated by a security firewall. [Refer to Appendix L for additional information.]

The LDAD system was developed with three things in mind: the inherent site-to-site variability; site-specific information changes with time; and the need to merge older technology with current capabilities that continue to change. Thus LDAD was built using modular and generalized software component architecture. For example, all modem commands are entered as part of session files or scripts that a program reads and uses to send the appropriate commands to the modem.

The processing of the data acquisition system is shared between the local data provider and the NWS. ALERT data were previously acquired by a dedicated connection between the ALERT base-station and a computer that accepted the various weather data in the form of gauge tips for rain, voltage for temperature and pressure, etc. These values had to be converted from electronic values to meteorological values using calibrated conversions. Every time sensors were recalibrated, added, or removed by the provider, the support team had to revise and edit the appropriate sensor table. Because each provider had its own specific format, a decoder and storage module had to be developed for each data type to be acquired.

The basic LDAD concept simplifies this process for both the data providers and for the support team. LDAD uses a simple data format, ASCII Comma Separated Values Text (CSVText), which separates each data field by a comma. A set of metadata files, created and maintained by the data provider or in conjunction with site personnel, will be used by the acquisition decoder. This facilitates data processing in hydrometeorological units instead of sensor units and removes the need for conversion routines. The simplicity of the CSVText format increases the likelihood that the data provider will use this standardized format.

All LDAD acquisition data are categorized and stored into the following four classes:

- Mesonet for surface weather observations
- Hydro for rain and stream observations
- Manual for manual observations such as cooperative observers
- Upper air for multilevel observations such as profilers.

### 8.1 LDAD Functionality

The LDAD functionality supports the acquisition of the Integrated Flood Observing and Warning System (IFLOWS); ALERT; Mesonet; Profiler; RRS/Upper Air; Gauges (LARC, Handar, Campbell, Sutron); COOP (Co-operative Observations); and other data transported via LDM, Rsync, or other TCP/IP Protocols. The Data Acquisition function
is achieved when data is transmitted to the internal (trusted) AWIPS servers (i.e., PX cluster). The data is transmitted to and from the LDAD Cluster via TCP/IP protocols or RS-232 communications. The Data Dissemination function is achieved when data is transmitted to the LDAD Cluster from the internal AWIPS system and is then distributed to External Users. The data can be acquired, stored, and displayed once fully configured.

8.2 **LDAD Hardware Configuration**

The LDAD hardware configuration depicted in Exhibit 8.2-1 details the interconnection of the various components that comprise the LDAD system. The LDAD System consists of two LDAD servers (LS2/3), a LAN switch (SMC 8024), a Terminal Server (Cyclades ACS32), Modems (MultiTech MT5600BR), and a LAN DMZ (HP ProCurve 2824). The DMZ consists of two SSG 320M Firewalls, a Netgear 16 switch, and two Netgear 5 port hubs. The LDAD baseline processes run on the LDAD Cluster (DMZ) and the AWIPS PX Cluster (Internal). Other local applications may run on other internal clusters, such as DX cluster in the case of the LDAD Dissemination Server. Data is transmitted through the DMZ either to the Trusted (internal) AWIPS system or to the Untrusted (External) Users/Systems.

8.2.1 **LAN Hub**

The LDAD LAN Switch in Exhibit 8.2-1 is an SMC Networks 8024 switch with 24 1000 BaseT ports. The LAN Switch provides connectivity to external users/systems, the LDAD Terminal Server (Management/Modem RS-232), and the Untrusted side of the LDAD Firewalls (FW1/2).

8.2.2 **LDAD Terminal Server**

The LDAD Terminal Server is a Cyclades Alterpath ACS32, which provides an Ethernet interface for connection to the LDAD LAN Switch and 32 serial ports for connections to communication devices such as the modems in the Modem Nest, and Console Management ports such as the LS2/3 Servers. Each port can be configured to a maximum speed of 56 K. The terminal server in an average configuration is used to connect 10 dial-in/dial-out modems (including MicroART, RRS, and ASOS), four dedicated modems, a fax modem, and various console connections.

Port assignments are shown in Table 8.2.2-1 as an example.
Exhibit 8.2-1. LDAD Hardware Configuration
The LDAD Server system is configured with a block of IP Addresses assigned by the site. These Class C IP addresses were assigned to the site (non-AWIPS IP addresses) by the NWS, and generally a block of 15 IP addresses should be set aside specifically for LDAD usage. IP Addresses are required for the Terminal Server, software services (i.e., Radius) running on the Terminal Server, certain ports on the Terminal Server (dependent upon function), the LDAD Firewalls, and the NAT (Network Address Translation) Table stored on the firewalls. An additional IP address may be needed for the Web Server running on the LS.

### 8.2.3 Modem Nest

The LDAD modems are located in the MultiTech chassis. There are a total of 16 available slots for MultiTech modem cards (MT5600BR) and 2 power supplies (PS1600). All sites have a minimum of 10 modems for dial-out, dial-in, and fax capability.

### 8.2.4 LDAD Firewall

Two SSG 320M firewalls are employed to provide network security for the LDAD and AWIPS systems. Each AWIPS site will have two firewalls controlled by central configuration management servers. The central configuration management servers are located at the NCF and the BNCF. The control over the firewall configurations is tiered with the central configuration management server having ultimate control. The regions have a client Security manager that connects to the NCF Central Server, which in turn connects back to the firewall. Individual AWIPS sites (forecast offices) will not have direct control over their own configurations.
8.2.5 **LDAD Server**

The LS cluster is a pair of HP ProLiant DL320 Linux servers. The LS machines provide the focal point for all external communications between the AWIPS site and the community, functioning as a pass-through device for all incoming and outgoing data. Only one server is actively running the heartbeat package at a given time, and it will be the one reachable through the address 192.168.1.10 (“ls1”) via the LDAD LAN switch. In this document, “the active LS” refers to whichever machine is actively working as the LDAD server, listening on the address 192.168.1.10, and reachable through the AWIPS LAN via the hostname “ls1.” This hostname will reach the active LS from the entire AWIPS site LAN, no matter which of the LS machines is presently serving as the active LS.

The ls2/ls3 servers have unique interfaces (ls2-192.168.1.11, ls3-192.168.1.12) that are up at all times. The ls2/3 heartbeat is a private lan pair (ls2-172.16.0.2, ls3-172.16.0.1). The ls2/ls3 servers share an external IP address for access to the site local LAN and the Internet for data push/pull collection. This external ldad address is assigned by the site and configured by the region in the site's LDAD firewall. It is routed by the firewall using Mapped IP (MIP) to the active LS; there is no external network interface connection on the LDAD server.

The VIR switch provides the capability to switch asynchronous serial connections between LS2 and LS3.

The LS provides a number of services that allow for the acquisition, processing, and dissemination of information. Text, binary, graphics, Web, and model are common data sets that are ingested or disseminated.

8.3 **LDAD Software Configuration**

AWIPS II provides an interface with the existing Message Handling System (MHS). This allows products, including product reships transmitted via the LAN, to be fed to EDEX for ingest/processing. The LDAD-to-EDEX bridge is provided via three LDAD routers – routerStoreEDEX, routerShefEncoderEDEX, and routerStoreTextEDEX – which run as part of LDAD on px2f. It is important to note that AWIPS II has not changed the design and functionality of LDAD; the only change is the addition of routers to bridge the gap from LDAD to EDEX.

The flow of LDAD data to EDEX is shown in Exhibit 8.3-1.
Exhibit 8.3-1. Flow of LDAD Data to EDEX

As shown in Exhibit 8.3-1:

1. Data from the office LAN is received by LDAD(E- External to the Firewall on the LS servers) and passed through the firewall LDAD(I -Internal to the Firewall on the PX servers).
   
   **Note:** The LDAD(E) software runs on the LS servers. The LDAD(I) software runs on the PX servers.

2. The internal LDAD(I) software performs two operations on the data it receives:
   a. LDAD(I) writes the data as a file to Raw Data Storage.
   b. LDAD(I) posts a “data available” message to the QPID message broker.

3. The EDEX Ingest process obtains the “data available” message from QPID and determines the data decoder to use to ingest the data.
   a. The data decoder reads the raw data from Raw Data Storage and decodes it.

4. Once decoded, the data is persisted for later retrieval.
   a. The data is sent to the DBMS.
      i. Information about the data is persisted to the AWIPS II metadata database.
      ii. Some types of data (for example, text products) are written to tables in the AWIPS II database.
b. Certain types of data (satellite imagery, radar imagery, GRIB data, and certain point data) are written to HDF5 files in Processed Data Storage. This is performed via the PyPIES process on DX1/2.

**Note:** In some cases, EDEX will automatically run scripts on arrival of the data.

Note that AWIPS II does not change the flow of data to the PX server. Normal data verification procedures may be used to follow the data to the PX servers. Once the data is passed to EDEX, the same techniques used to trace the SBN data flow from EDEX to Processed Data Storage may be used for LDAD delivered data.

### 8.3.1 LDAD Gateway

The LDAD Gateway is a set of software components that together make up the core data acquisition and dissemination mechanism of the AWIPS LDAD. The Gateway is distributed over the LS and the PX, on both the internal and external sides of the firewall. The LDAD Gateway software links the internal and external sides of LDAD together, controlling the transfer of information across the firewall. Table 8.3.1-1 lists LDAD environmental variables and their paths. Exhibit 8.3.1-1 shows the generic data flow for acquiring and decoding LDAD products.

Gateway components include the following:

- Scheduler
- Communications Server (CO)
- Inter-Gateway Communication (IGC)
- Preprocessors
- Decoder and Storage
- Trigger
- Quality Control (QC)
- Fax.

<table>
<thead>
<tr>
<th>LDAD Environment Variable Name</th>
<th>LDAD Environment Variable Path (as user fxa on PX1, PX2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LDAD_HOME</td>
<td>/awips/fxa/ldad</td>
</tr>
<tr>
<td>LDAD_INTERNAL_HOME</td>
<td>/awips/ldad</td>
</tr>
<tr>
<td>LOG_DIR</td>
<td>/data/logs/fxa</td>
</tr>
<tr>
<td>LDAD_INTERNAL_LOGDIR</td>
<td>/awips/fxa/ldad/logs</td>
</tr>
<tr>
<td>LDAD_EXTERNAL_LOGDIR</td>
<td>/data/logs/ldad</td>
</tr>
<tr>
<td>LDAD_DATA</td>
<td>/data/fxa/LDAD</td>
</tr>
<tr>
<td>LDAD_INTERNAL_DATA</td>
<td>/awips/fxa/ldad/data</td>
</tr>
<tr>
<td>LDAD_EXTERNAL_DATA</td>
<td>/data/1dad</td>
</tr>
<tr>
<td>LDAD_DECODED_DATA</td>
<td>/data/fxa/LDAD/decoded</td>
</tr>
</tbody>
</table>
Chapter 8  
Local Data Acquisition and Dissemination (LDAD) System

Exhibit 8.3.1-1. Generic Acquire and Decode LDAD Product (Page 1 of 2)
Exhibit 8.3.1-1. Generic Acquire and Decode LDAD Product (Page 2 of 2)
New products coming into LDAD are placed into the /data/Incoming directory on the active LS machine. The newLDADdataNotification and CO_serv processes on the LS notify the listener process on the PX2 when a new product is available. The listener process transfers the file through the firewall, puts the copy on the PX2 in the SLDAD_DATA/tmp directory, and notifies CO_serv and newLDADdataNotification that the transfer has been completed. Once this notification has been received, newLDADdataNotification moves the original product from /data/Incoming to /data/ldad/Processed. If newLDADdataNotification fails to receive confirmation from the listener process within 5 minutes, it moves the file to the /data/ldad/Problem directory. If the file requires preprocessing, the listener process copies the file into the SLDAD_DATA/tmp directory and checks the SLDAD_DATA/data/LDADinfo.txt file to specify the preprocessor to call for that particular product type. The file is copied to the SLDAD_HOME/tmp directory, where it is reformatted by the preprocessor and stored in the SLDAD_DATA/Raw directory on the PX2. If the file requires no preprocessing, the listener moves the file directly into the SLDAD_DATA/Raw directory and sends a notification to the CommsRouter that a product has been preprocessed. The CommsRouter passes the information to the DataController, which in turn passes the information to the routerLdadDecoder. The routerLdadDecoder decodes all files in the SLDAD_DATA/Raw directory and writes them via serialized data files to the SLDAD_DECODED_DATA directory. Data that is processed, but not recognized are written to SFXA_DATA/LDAD/Unknown. Bad data are written to the SFXA_DATA/LDAD/Bad directory. The routerStoreEDEX process reads the files in the SLDAD_DECODED_DATA directory, converts those files that match the criteria in the LdadPatterns.txt file into an XML format, stores the converted data into the /data_store/ldad directory, and sends a notification to EDEX.

NOTE: The SLDAD_DATA/tmp directory on the PX2 works much the same as /data/Incoming on the LS; it serves as a temporary location for the file while it is being processed. So, any given file should remain in SLDAD_DATA/tmp for only the relatively short period of time that it is being processed. Once processing is complete, the file should be moved to its destination. The SLDAD_DATA/tmp directory is purged once per day, so if it contains many files, the accumulated files may fill up the /data/fxa partition. The LDAD software was designed so that the preprocessor will either delete or move the temporary files from the SLDAD_DATA/tmp directory once it completes its processing. The preprocessors that are delivered with the baselined AWIPS software remove these files. However, sites that have written their own preprocessors for specialized data types may have preprocessors that fail to clean the temporary files out of the SLDAD_DATA/tmp directory when they finish.

8.3.2 Scheduler

The Scheduler’s function is to perform specified data acquisition/dissemination events automatically at specified time intervals (in the background via the ldadServer process). The Scheduler invokes scripts to acquire/disseminate from/to remote hosts by ftp, kermit,
xmodem, ymodem, or zmodem, or automatically dials selected LARC, Handar 555, Campbell, or Sutron gauges for data at specific intervals. Configuration, manipulation, and activation of the Scheduler are accomplished via the Scheduler’s own edit menus.

The Scheduler also provides the forecaster access to the LDAD Gateway. Through this Tcl/Tk interface, the AWIPS user can start and stop the LDAD Gateway automatic processes, incorporate new types of data sources into the system, set up operating environments for external users, and initiate data acquisition on demand from an external source (e.g., a LARC gauge).

Using the Scheduler, the user can create or modify session files. These are simple, text-based files that the communications server (CO_serv) program reads and translates into the appropriate commands to execute a data transfer acquisition session with a remote source (e.g., a LARC gauge).

The Scheduler can be started from any workstation from the Cave Obs menu, Collection/Dissemination. Refer to the AWIPS II CAVE-D2D User’s Manual for additional information.

8.3.3 Communications Server

The Communications Server (CO) component is used by the other components to interface with the communication hardware; that is, the terminal server and modems. The CO uses information from the Scheduler and the session files for configuring dial-out parameters, defining the data, etc. The CO program, CO_serv, calls the script that performs the function. CO_serv is the external end of the IGC described in the next section.

8.3.4 Inter-Gateway Communication

The IGC function provides a communications link across the firewall between the internal and external sides of the Gateway. This link or Interprocess Communication (IPC) is handled by sockets, an IPC interface that allows communication across the firewall using TIS’s TCP proxy. On the PX2, a process called listener is the point of contact for all communications between processes on the internal system (PX) and the external system (active LS), where the CO_serv acts as the point of contact. All communications between the PX and the LS use the CO_serv-listener pair.

The firewall is configured to allow socket communication to a single specified port from only the LS on the external system.

8.3.5 Local Data Management

Local Data Management (LDM) is a software package that is designed to select, capture, process, and distribute data products using client/server programs. The LDM software used for the LDAD servers is copyrighted by the University Corporation of Atmospheric
Research (UCAR) in Boulder, CO. Each regional headquarters has an LDM Server that receives data from the Meteorological Assimilation Data Ingest System (MADIS). The data is then disseminated via the internet to the LDAD clients that are configured to receive LDM data. Once the data is received by LDAD, it is treated like any other LDAD product.

LDM is being used only for LDAD products that are requested from MADIS. This means that the ls2 and ls3 must be configured to ingest the products, and the px1 and px2 must be configured to process the products. Please note, for failover purposes, the ls2/ls3 must have the same LDM configuration and px1/px2 must have the same preprocessors, stations files, description files, etc. See Exhibit 8.3.5-1.

![Exhibit 8.3.5-1. LDM Ingest](image)

### 8.3.6 Preprocessors

When the listener on the PX2 receives a message that a new data file is available on the active LS, it immediately secure copies the file over to the `$LDAD_DATA/tmp` directory. The listener then checks the `$LDAD_DATA/data/LDADinfo.text` file to identify the preprocessor script that is required for the data file received. If any preprocessing for that data file is necessary, the appropriate preprocessor is called to reformat the file for the decoder, and is stored in `$LDAD_DATA/Raw`.

The preprocessors are located in the `/awips/fxa/ldad/bin` directory. The preprocessors can be written in any programming or scripting language. The only requirement is that they are executable. Sample preprocessors are installed during installation. These preprocessors were generated for generic data sets and should be treated as site-configurable. There is a potential need for many preprocessors, one for every data type. The objective is to remove the need for preprocessors by having the data providers send files in the required format.
8.3.7 Decoder and Storage

The decoder module checks the SLDAD_DATA/Raw data directory for new data sets at 1-minute intervals. If any are found, it decodes them using information from the metadata files. Using the information in the description files (*desc files), the decoder evaluates each column of data by checking for the hydrometeorological variable, the incoming units, and the storage units. After the decoding is complete, the decoder creates a data object with all the data and calls the storage module. The storage module reads the station file (*Station.txt file) for static station information such as latitude, longitude, and elevation. It stores the station information and data from the decoder into a netCDF and possibly a Standard Hydrometeorological Exchange Format (SHEF)-encoded file. The decoder process runs continuously.

8.4 Rsync

Rsync has been preconfigured on the LDAD Servers. This document details the baseline configuration of the rsync implementation on the LDAD Servers. Note that as delivered, rsync cron jobs loaded onto LS2/LS3 are syncing a few directories between the two servers. Sites may wish to alter this configuration, adding more directories to be synced or removing some that are being synced. Rsync is site configurable.

The baseline implementation of rsync consists of an rsync process initiated by root cron jobs /var/spool/cron and is limited to syncing directories between the two servers (LS2/3). The three /var/spool/cron/ rsync entries are as follows:

```
/ldad/bin/rsync_cron.sh ldad
/ldad/bin/rsync_cron.sh ldm
/ldad/bin/rsync_cron.sh root
```

The directories being synchronized between the two servers are:

- **/home** – The rsync syntax compares the /home directories of LS2/3 and saves the latest version of the files to /data/Backup directory.
- **/usr/local/ldm/data** – This synchronization is triggered by the “ldm” cron entry and synchronized the data in the /usr/local/ldm/data directory.

8.4.1 Rsync with AWIPS (internal) and External (non-AWIPS) Systems

The LDAD servers use rsync to communicate with external hosts such as web farms; and receive internal rsync connections from DX1/2 (and other AWIPS servers) in support of other programs such as AHPS. The AWIPS to LDAD rsync processes are achieved by utilizing the “-ssh” switch on rsync. Sites do not have to alter this configuration in order to initiate rsync connections from AWIPS to LS2/3 when using the “-ssh” implementation. Sites may need to modify the rsync configuration files to include external server IP Addresses assuming the “-ssh” switch is NOT being used to connect to these servers. Example of using the “ssh” switch with rsync for internal to LDAD rsync communications:
rsync -auvv -e ssh -timeout=120 <file name> ldad@ls1:/tmp

### 8.4.2 Rsync Configuration Files

The following rsync files have been verified, modified, or added to the baseline:

- `/rsync_install.sh` (rsync initial setup script)
- `/etc/rsyncd.conf` (rsync daemon config file)
- `/etc/services` (rsync tcp port assignment config file)
- `/etc/xinetd.d/rsync` (rsync daemon control file)
- `/etc/sysconfig/iptables-config` (netfilter saved configuration file)
- `/etc/sysconfig/ip6tables-config` (netfilter saved configuration file)
- `/ldad/bin/rsync_cron.sh` (cronjob script)

### 8.5 LDAD Linux Server Heartbeat Cron Package

The Heartbeat and the Cron files managed by the heartbeat package are configured on the LDAD Servers (LS2/LS3) in a manner similar to its implementation on the other AWIPS systems. Details of its implementation and procedures to localize the configuration for sites are presented in this section:

**Key Directories:**

- `/etc/ha.d/cron.d` - This directory contains two files, lscron and SITElscron. Sites should use the SITElscron file to add their local cron jobs that need to be managed by the heartbeat package.
- `/etc/ha.d/resource.d` - This directory contains the file “Crontab” which is a resource added to the `/etc/ha.d/haresources` file. This heartbeat resource file is responsible of starting the cron file listed in `/etc/ha.d/cron.d`. This file also copies the files listed in `/etc/ha.d/cron.d` to the `/etc/cron.d` directory at heartbeat startup. This script can also be run manually from the command line to check status. For example, as root on LS2, type the following from the command line:

  ```
  /etc/ha.d/resource.d/Crontab <cronfile name> status
  ```

  (where `cronfile name` could be the baseline SITElscron file name).

- `/var/spool/cron` - This directory is the standard location for user cronfiles and can still be used by sites. Baseline configuration has two cron jobs running on LS2/LS3 and they consist of the following:

  **ldm**: One entry running the `/usr/local/ldm/bin/scour` script.

  **ldad**: Runs scour daily to clean up log files

  ```
  0,30 * * * * ldad /home/AAS/test/vdot.sh
  15 * * * * ldad /home/AAS/test/vdot2.sh
  45 * * * * ldad /home/AAS/test/vdot3.sh
  30 0 * * * csh -c '${LDAD_HOME}/bin/startScour.sh >&!/data/logs/ldad/startScour.log'
  ```
Heartbeat Status

The LDAD servers are configured in a clustered pair. Only one server is actively running the heartbeat package at a given time, and it will be the one reachable through the address 192.168.1.10 (“ls1”) via the LDAD LAN switch. If the primary server loses network connectivity or had failed, the other server will take over the 192.168.1.10 ipaddress and all the LDAD processes. The `hb_stat` command can be used to determine which server is providing LDAD functionality. Notice the ipaddress configuration:

```
[root@ls2-ntcb ~]# hb_stat
Heartbeat Status Monitor

Member Status IP address
---------- ---------- ---------------
ls2-tbdw Up  192.168.1.11
ls3-tbdw Up  192.168.1.12

Resource Status
-------------------------
Resource Parameters Owner Status Start Time
-------------------------
IPaddr2 192.168.1.10/24/eth0 ls2-tbdw Running OK 2012-04-12 18:14:47
LS1IPSRCaddr ls2-tbdw OK 2012-04-12 18:14:47
apache ls2-tbdw Running OK 2012-04-12 18:14:49
csportd ls2-tbdw OK 2012-04-12 18:14:53
Xinetd wu_ftpd ls2-tbdw Running OK 2012-04-12 18:14:53
hylafax ls2-tbdw OK 2012-04-12 18:14:54
dlad ls2-tbdw OK 2012-04-12 18:14:55
dserver ls2-tbdw OK 2012-04-12 18:15:28
Crontab SITElscron ls2-tbdw OK 2012-04-12 18:15:28
Crontab lscron ls2-tbdw OK 2012-04-12 18:15:28
SITEprocesses ls2-tbdw OK 2012-04-12 18:15:29

HylaFAX Details:
HylaFAX client-server protocol server: hfaxd (pid 32423) is running...
HylaFAX queue manager process: faxq (pid 32420) is running...
```

ACTIVE LDAD Processes

The active LDAD server will have processes running similar to those listed below, depending on how the server is configured:

```
ls2-ntcb{ldad}49: ps -ef | grep nobody
nobody heartbeat: FIFO reader
nobody heartbeat: write: ucast eth1
nobody heartbeat: read: ucast eth1
nobody heartbeat: write: ping 192.168.1.1
nobody heartbeat: read: ping 192.168.1.1

ls2-tbw3{ldad}50: ps -ef | grep ldad
ldad 11680 1 0 May30 ? 00:00:05 /ldad/bin/CO_serv
ldad 11683 1 0 May30 ? 00:00:00
/ldad/bin/newLDADDataNotification
ldad 11684 1 1 May30 ? 17:01:07 /bin/bash /ldad/bin/hmingestd
ldad 11687 1 0 May30 ? 00:00:00 /usr/bin/perl
/ldad/bin/MakeLDApage /ldad/data/lda.conf
ldad 11688 1 0 May30 ? 00:00:00 /usr/bin/perl
/ldad/bin/MakeLDApage /ldad/data/ldd.conf
```
INACTIVE LDAD Processes

The inactive LDAD server will only have the heartbeat processes running, checking for connectivity to the other LDAD server. If connectivity is lost, the inactive server becomes the active server:

```
[root@ls3-ntcb ~]# ps -ef | grep nobody
nobody heartbeat: FIFO reader
nobody heartbeat: write: ucast eth1
nobody heartbeat: read: ucast eth1
nobody heartbeat: write: ping 192.168.1.1
nobody heartbeat: read: ping 192.168.1.1

[root@ls3-ntcb ~]# ps -ef | grep ldad
root 21317 21280 0 15:32 pts/0 00:00:00 grep ldad

[root@ls3-ntcb ~]# ps -ef | grep ldm
root 21313 21280 0 15:32 pts/0 00:00:00 grep ldm
```

8.6 Quality Control and System Monitoring

This section describes the Quality Control and System Monitoring.

8.6.1 Quality Control

LDAD observations are quality controlled by the AWIPS Quality Control and Monitoring System (QCMS), which in turn relies on the MAPS Surface Analysis System (MSAS) to perform automated quality control checks. MSAS/QCMS is a multischeduled system with four main subprocesses. Each subprocess runs independently based on the following crontab entries on PXs:

- The **model ingest cycle** runs twice a day.
- The **subhourly QC cycle** runs every 5 minutes.
• The hourly QC/analysis cycle runs immediately after the xx:18 subhourly run.
• The QC summary programs run near 00Z.

The QCMS is only a partial implementation of the overall requirements for the quality control of surface observations. The subhourly processing checks the validity, internal consistency, and temporal consistency of LDAD Mesonet observations of sea-level pressure, temperature, dewpoint temperature, wind, station pressure, altimeter setting, pressure change, relative humidity, visibility, and precipitation observations. The hourly QC process checks the validity, internal consistency, temporal consistency, and spatial consistency of LDAD Mesonet and NOAAPORT observations of sea-level pressure, temperature, wind, and dewpoint temperature. WFO personnel can override the results of the automated QC checks.

Two text files, /data/fxa/LDAD/fslparms/reject.txt and accept.txt, reside on the DX Network Attached Storage (NAS), a shared directory, to provide the capability to subjectively override the results of the automated QC checks. The reject list is a list of stations and associated input observations that will be labeled as bad, regardless of the outcome of the QC checks; the Accept list is the corresponding list of stations that will always be labeled as good. Applications reading the lists (e.g., MSAS) will reject or accept the specified stations. In both cases, observations associated with the stations in the lists can either be flagged individually, or be flagged in groups.

The QCMS station monitoring procedures and text summary files are not affected by the intervention lists. This allows WFO personnel to continue to monitor the performance of the stations contained in the reject and accept lists. For example, a user may notice that a station has wind observations that fail the QC checks a large percentage of the time, and choose to have that station added to the reject list. However, once the observation failure rate at the station falls back to near zero (possibly due to anemometer repair), the station can be deleted from the list.

QCMS reject and accept lists may be edited by hand (with any text editor), or through the LDAD Monitoring and Control System (see next section for additional information). For each station, both station and provider names are required. Provider names are found in the /data/fxa/LDAD/fslparms/sfchqcin.dat file. The first seven entries (SAO, MARITIME, COOP, OTHER-MTR, ASOS, NWSRAWS, UTMESNET) are national data sets, and always appear. Any others are added dynamically by the 5-minute QC program, qcstg1_2, based on dataProvider names found in /data/fxa/LDAD/mesonet/netCDF files (LDAD-collected data). A blank provider name is allowed, but it may result in multiple stations with the specified station name being rejected or accepted (e.g., when different data providers use the same station name).

The QCMS also provides the capability to accept or reject observations from all stations with a specified provider name. To utilize this option, fill in the provider and observation columns, but leave the StaID column blank.

The QCMS also calculates hourly, daily, weekly, and monthly statistics on the frequency and magnitude of the observational errors encountered for sea-level pressure,
temperature, dewpoint, and surface winds. It processes both NOAA PORT and LDAD Mesonet observations.

QCMS summary files are stored in the text database. Files are named  
<nnn>QC[HDWM]<LLL>, where <nnn> is a three-digit number (leading zeros) corresponding to a data set’s ordinal position in /data/fixa/LDAD/fslparms/sfchqcin.dat (e.g., NPN is 003); H is for hourly QC summaries, D for daily, W for weekly, and M for monthly; and <LLL> is the localization ID.

QCMS generates status messages for up to 20 providers. If more than 20 providers are found in the netCDF files, all additional providers’ QC information will be lumped into 000QC[HDWM]<LLL> files.

**8.6.2 System Monitoring**

The LDAD System Monitor and Control System is a Web-based function that allows the site to monitor the LDAD system connectivity, processes, and acquisition/dissemination status. The LDAD System Monitor checks on all LDAD processes, systems, and data to provide a visual status of LDAD performance. The LDAD Monitor Summary page is linked to the AWIPS System Monitor. The LDAD Monitor Summary page summarizes all LDAD Monitor pages and provides an interface for system administration.

The Summary Monitor application, MakeSUMMpage, runs on PX1 and uses the configuration file ~ldad/data/summary.conf for various parameters, including the monitor pages. The Internal Process and System page comes from PX1, while the other three (Data Acquisition, Data Dissemination, and External Process and System) are retrieved via HTTP protocol from active LS. By default, the monitor updates every 5 minutes. If any of the pages is out of date (i.e., older than the Run_interval specified in the summary.conf file), the MakeSUMMpage will not generate its line on the summary.

The overall/summary state of each submonitor is displayed on the left with the standard state icons. On the right are four columns, designated R, Y, B, G for Red, Yellow, Blue, and Green, respectively. The numbers below the column represent the number of items that are in each of the states for each submonitor. This gives the site administrator some indication of the relative problem and allows for prioritization for problem solving. (If any item is in ? state [e.g., a data file is not found], it does not register in any column.)

There are two Process and System pages, Internal and External, each of which displays the state of all appropriate LDAD processes and hardware systems. These pages are intended to be used primarily by the LDAD site administrator. The following three tables are on each page:

**Processes:** Displays the state of all LDAD processes on the active LS, or PXs
Systems: Displays the state of all internal or external LDAD systems
Network: Displays the state of all LDAD network systems (Gateway, Firewall, Terminal Server, etc.)

The LDAD site administrator can start/stop processes, edit system configuration files, etc.

8.7 Acquire LARC/Handar 555 Data

Data collection from LARC/Handar 555 gauges is carried out by automating a normally interactive session using the cu communication protocol. There is a Server program (CO_serv) and a Client program (tell_co). The Server program is always running, waiting for and processing requests to collect data from a particular gauge from the Client program. The Client program can be invoked from a shell script, as a cron job, or directly from a user prompt. A feature of the Client program is that it retrieves all necessary information regarding collection and storage of data from system tables in the AWIPS database. The Client and Server programs use sockets to communicate across a TCP/IP LAN. Data from a gauge are SHEF-encoded before storage on AWIPS.

Processes

LS  process_expect
LS  CO_serv
PX2  listener
PX2  tell_co
PX2  ldadServer
IX  ldadScheduler

Environment Variables

$LOG_DIR    /data/logs/fxa (as user fxa on the PX1, PX2)
            /data/logs/ldad (on the active LS or as user ldad on the PX2, PX1)
$LDAD_INTERNAL_DATA  /data/fxa/LDAD/data (as user ldad)
                     /awips/fxa/LDAD/data (as user fxa)
$LDAD_EXTERNAL_LOGDIR /data/logs/ldad (as user ldad)
$LDAD_INTERNAL_LOGDIR /data/logs/ldad (as user fxa)

Note: The Environment Variables above apply to all exhibits in this chapter.

Configuration Files (all in /data/fxa/LDAD/data)

LDADinfo.txt (for listener)
LARCStation.txt (for tell_co)
userDescriptions (for ldadScheduler)
usrpro (for ldadScheduler)
gageinfo (for ldadScheduler)
gaugeDescriptions (for ldadScheduler)
userSessList (for ldadScheduler)
gaugeSessList (for ldadScheduler)

LDADinfo.txt

The file should contain the following entry:

LARC | LARC | 89 | 90 | CVS_TYPE | hydro | preProcessLDAD.pl |

The preProcessLDAD.pl script must be located in the /awips/fxa/ldad/ directory. Once the file is installed, this line should not be modified.

LARCStation.txt

This file must be updated whenever a gauge is added to or dropped from the current LARC/Handar 555 datalogger list. To add a new LARC/Handar 555 gauge, find out the following gauge information to create a new entry in the station table format (* indicates a mandatory field):

- **Provider ID:** Data provider station ID (*)
- **AFOS(HB5) ID:** The AFOS or Handbook 5 ID (NWS assigned)
- **Station Name:** Text name of station
- **Elevation:** Station elevation (m) (*)
- **Latitude:** Station Latitude (decimal degrees, north positive) (*)
- **Longitude:** Station Longitude (decimal degrees, east positive) (*)
- **Local Time Zone:** Reporting time zone (e.g., UTC, PST/PDT) (*)
- **Location Description:** Station location/address (Text)
- **Station Type:** Station type (e.g., tower, surface, floating platform, etc.)
- **Number of Instruments:** Number of reporting instruments for the station
- **Number of Reporting Levels:** Number of data reporting levels. Level information should be provided in the instrument table
- **Maintenance/Calibration Schedule:** Frequency of maintenance/calibration
- **Site Description:** Text description of the site surroundings.

The entries are sorted alphabetically by the gauge ID.

This file must be identical on the PX2 and on the LS machines.
userDescriptions

This file is used by the ldadScheduler process to get the descriptions of users. When read in, blank lines and lines beginning with # are skipped. All other lines should have the following format:

```
<user ID>|<user description>
```

A blank user description is denoted by having the line end immediately following the | character.

This file should not be modified manually. It should only be written to by the ldadScheduler (specifically, the LdadSchedulerServer.C routine).

usrpro

The user file (usrpro) is made up of zero to n lines, each line holding the following information (note that the lines are so long that they are wrapped in the interests of readability). Columns containing “??” are not used at this time.

```
<ID>|<protocol>|<IP address>|<phone number>|<login>|<password (encrypted)>|??|<file to receive>|<file to send>|??|??|<data collection session file>|<data dissemination session file>|<local directory>|<remote directory>|??... (columns 16 through 36 have unknown meanings)...|<IP address>
```

- The <protocol> is either “F” for FTP, “X” for Xmodem, “Y” for Ymodem, “Z” for Zmodem, or “K” for Kermit.
- The <IP address> designates the host to FTP to, if the protocol is “F,” or the terminal server. <IP address> appears in two places in the above column listing: columns 3 and 37. In reality, only one of the two columns is used. For the FTP protocol, only column 3 has the address in it, and column 37 is left blank. If any other protocol is being used, column 37 has the address in it, and column 3 is left blank.
- The <phone number> field is unused if the protocol is “F.”
- The <file to receive> and <file to send> fields are mutually exclusive. Only one may be specified, and the one that is specified also indicates whether this user is a collection user (if <file to receive> is given) or a dissemination user (if <file to send> is given). All other columns are simply tracked when read in; when written out, the same values are placed in them as were originally found.

gageinfo

The gageinfo file contains a list of Handar, LARC, Campbell, and Sutron gauges. It is updated periodically as LARC gauges are added and removed with the ldadScheduler. Whenever a gauge is added or dropped from the current list, it should be updated here and in the following files:

```
$LDAD_DATA/data/LARCStation.txt
$LDAD_DATA/data/gaugeDescriptions
```
The following fields must be present for each LARC/Handar 555 gauge:

- **Gauge ID**: A five-character instrument ID.
- **Protocol Type**: C for data collection, D for dissemination.
- **Phone#**: The dial-out phone number can be in any format. However, if the gauge has the voice message first, several pauses are included in the phone number. Depending on the length of the voice message, add more pauses if the one listed here is not enough. The pause is entered by a comma (,) and is ended with a *9. If a phone number contains a pause extension, make sure the phone number is enclosed by single quotes (‘). For example, ‘1-605.886.2473,,,,,*9’.
- **Session File**: Use HANDAR_coll_sess exactly.

The HANDAR_coll_sess will be one of the following:

- **HANDAR_coll_session_1** means gauge provides river stage information only.
- **HANDAR_coll_session_2** means gauge provides total rain information only.
- **HANDAR_coll_session_12** means gauge first provides river stage information then total rain information.
- **HANDAR_coll_session_21** means gauge first provides total rain information then river stage information.
- **HANDAR_coll_session_555** means gauge provides river stage information and/or total rain information.

- **SHEF FILE**: 0=No; 1=Yes.
- **Storage Type**: S.
- **Baud Rate**: Use 9600 for LARC/Handar 555 gauges.
- **Terminal Server (tty)**: Use 7e1out for LARC gauges and 8n1out for Handar 555.

The fields are separated by vertical bars (); empty fields are for future use. The number nine (9) in front the phone number allows the user to dial out. An example for a LARC/Handar 555 gauge follows:

```
ABNG1|C|919124365917 | | | | HANDAR_coll_sess_1|1|S||9600| e | |7e1out|
SNMC1|C|93016080977 | | | | HANDAR_coll_sess_555|S||9600| e | |8n1out|
```

**NOTE:** The rcvprog field indicates the receiving program (e.g., FTP). For data collection, this field is used as the custom gauge ID if the gauge is not owned by the NWS.

**gaugeDescriptions**

The file is used to get the site descriptions and should be updated using the ldadScheduler when a new LARC/Handar 555 gauge is added. Each entry (line) should have the following format:

```
<gauge ID> | <site description>
```
For example, an entry for the BARW3 would be as follows:

BARW3 | Barron, WI

The file is sorted by state abbreviations. However, it is displayed based on the alphabetical order of the gauge IDs.

Refer to Exhibit 8.7-1 for the Acquire LARC/Handar 555 data flow diagram.

### 8.7.1 User Interface: ldadScheduler

The ldadScheduler’s function is to perform certain data acquisition/dissemination events on an automatic basis at specified time intervals (in the background). Access to the Scheduler’s functions is via a GUI invoked by selecting Collection/Dissemination... from the Surface pull-down menu.

The ldadScheduler provides a window by which the user may create, view, and manipulate scheduled LDAD requests. It allows the user to create a new request and add it to the request list. The request is activated when submitted to the ldadServer, which uses sendLDADNotification.pl to create the request. If it is a one-time request (OTR), the user can monitor its progress via a status dialog.

The ldadScheduler registers with the ldadServer providing a callback routine for update notifications. The session file contents get registered first, and the status messages and ID updates so that when registering with ldadServer, the ID lists sent by ldadServer will be caught and kept. It also registers with the File Access Controller (FAC) for notifications of changes to the existing requests file, and creates a FAC lock for the file to be used for locking it for “reads” and “writes.” The existing requests get read so as to fill the tree view with the requests that have been created.

### 8.7.2 ldadServer

The ldadServer allows multiple clients to connect to it and submit requests with intervals and durations that are honored according to their intervals until they expire.

New client initialization is performed by sending the gauge and user lists to the newly registered client. The ldadServer process creates a new message with the current active list and broadcasts it to all registered clients.

To honor the specified request, the ldadServer creates the command to be executed.

The ldadServer process receives LDAD messages from clients; messages to create and delete gauges and users; and messages requesting the contents of a session file, or specifying a session file’s contents.
Exhibit 8.7-1. Acquire LARC/Handar 555 Data (for LDAD)
To Access and Read the ldadServer Log

The ldadServer log resides on the PX2 in the $LOG_DIR directory, and records messages from the process. Perform the following commands as user ldad:

**TYPE:**
```bash
cd $LOG_DIR/<yyyymmdd>
```
- Where `<yyyymmdd>` is today’s date.

**TYPE:**
```bash
ls -ltr ldadServer*
```
- Displays the ldadServer logs from earliest to latest.

**TYPE:**
```bash
tail -f <logname>
```
- Displays the latest registered messages as the log updates.

Log Message Format

An entry at the beginning of the log looks like the following:

```
LOG-STATUS: Log file opened on host px2-rnk at Jul 12 14:28:29 2010
00:18:59.602 DataSocket.C PROBLEM: Reconnect attempt to lx1-rnk/4813/10733 failed.
00:18:59.765 DataSocket.C DEBUG: Closing the socket to lx1-rnk/4813/10733
00:18:59.766 DataSocket.C DEBUG: Creating a new object to read an orphaned socket.
00:18:59.766 ParameterizedMsg.C EVENT: Send of IPC message failed; Target not available.
  Target: lx1-rnk/4813/10733
  Module: 35
  Type: 10
00:18:59.786 ParameterizedMsg.C EVENT: Send of IPC message failed; Target not available.
  Target: lx1-rnk/4813/10733
  Module: 35
  Type: 8
00:18:59.789 DataSocket.C DEBUG: Connection to unknown IPC address has been closed by the peer.
00:18:59.790 DataSocket.C DEBUG: Closing the socket to unknown IPC address
```

The format of the log consists of the following fields:

- **Time stamp**: 00:18:59.602
- **File name**: DataSocket.C
- **Message type**: PROBLEM:
- **Message string**: Reconnect attempt to lx1-rnk/4813/10733 failed.

Here is another example that warns of a wrong host.

```
LOG-STATUS: Log file opened on host px2-tbw3 at Mon Jul 12 14:29:48 2010
14:29:48.862 Connection.C EVENT: WARNING: this process wants to use target LDAD_SCHEDULE_SERVER which is assigned to run on px2f-tbw3 but the current host is px2-tbw3
```
8.7.3 **tell_co**

The Client program (**tell_co**) retrieves all information concerning a specified gauge from the LDAD configuration files. This includes all communication details, the name of the session file to use, the location to store the data on AWIPS II, and whether SHEF-encoding is required (for LARC/Handar 555 gauges, SHEF-encoding is always required). It formats an interprocess message containing all this information, opens a socket to the **listener**, and sends a get LARC/Handar 555 Message “IPC” message to the **listener**.

- **To Access and Read** **tell_co** **Log**

  The **tell_co** log resides on the PX2 in the `$LOG_DIR` directory and records messages from the **tell_co** process. Perform the following commands as user ldad:

  ```
  TYPE: cd $LOG_DIR/<yyyymmdd>
  - Where `<yyyymmdd>` is today’s date.

  TYPE: ls -ltr tell_co*
  - Displays the **tell_co** logs from earliest to latest.

  TYPE: tail -f <logname>
  - Displays the latest registered messages as the log updates.
  ```

- **Log Message Format**

  An entry at the beginning of the log looks like the following:

  ```
  LOG-STATUS: Log file opened on host px2-tbdw at Wed May 12 19:45:55 2006
  19:45:55.368 doAlivenessTest.c EVENT: waitForReply INTERNAL  SUCCESS!!!...
  19:45:56.253 doAlivenessTest.c EVENT: waitForReply EXTERNAL  SUCCESS!!!...
  19:45:57.205 tell_co.c EVENT: TELL_CO waitForReply SUCCESS!!!...
  ```

  The format of the log consists of the following fields:

  | Time stamp | 19:45:57.205 |
  | File name  | `tell_co.c` |
  | Message type| EVENT: |
  | Message string| `TELL_CO waitForReply SUCCESS!!!` |

8.7.4 **listener**

The **listener** sets up a socket connection, binds to a local address so that the Client (**CO_serv**) or **tell_co** can send to it, and waits for the connection. Upon receipt of a data collection request from **tell_co**, the **listener** forwards information received to the
**CO_serv** process. The **ldadServer** process passes the notification to the **ldadScheduler**, which (in the case of an OTR) displays the predecoded contents of the data file to the forecaster. The **ldadScheduler** process also receives and displays the LARC/Handar 555 gauge dialing status information.

► **To Access and Read the listener Log**

The **listener** log resides on the PX2 in the directory `$LOG_DIR` and records messages from the **listener** process. Perform the following commands as user `ldad`:

**TYPE:**

```
cd $LOG_DIR/<yyyymmdd>
```

- Where `<yyyymmdd>` is today’s date.

**TYPE:**

```
ls -ltr listener*
```

- Displays the **listener** logs from earliest to latest.

**TYPE:**

```
tail -f <logname>
```

- Displays the latest registered messages as the log updates.

**Log Message Format**

An entry at the beginning of the log looks like the following:

```
20:05:42.447 listener.c EVENT: do_G_Msg: Preprocessor Command Line =
</awips/ldad/bin/preProcessLDAD.pl /data/fxa/LDAD/tmp/alert_r5.dat.1342037141
px2f 15009>
```

```
0:02:15.652 listener.c EVENT: rcp_cmd_buf3 = scp
/data/fxa/LDA/hydro/netCDF/20120711 2000 ls1:/data/ldad/hmIngest/obs-hydro-
local.20120711 2000.1342036934.incoming; ssh ls1 -l ldad "chmod 644
/data/ad/hmIngest/obs-hydro-local.20120711 2000.1342036934.incoming; mv
/data/ad/hmIngest/obs-hydro-local.20120711 2000.1342036934.incoming
/data/ad/hmIngest/obs-hydro-local.20120711 2000.1342036934; /
/ad/bin/sendLDADnocation.pl R \"obs:hydro:local /data/ad/hmIngest/obs-
hydro-local.20120711 2000.1342036934\" ls1 15007" length = 484
```

```
20:05:15.716 listener.c EVENT: child listener: sendLDADnocation completed
for /data/fxa/LDAD/mesonet/qc/20120711 2000
```

```
20:05:15.716 listener.c EVENT: rcp_cmd_buf3 = scp
/data/fxa/LDAD/mesonet/qc/20120711 2000 ls1:/data/ldad/hmIngest/obs-qcmesonet-
local.20120711 2000.1342037114.incoming; ssh ls1 -l ldad "chmod 644
/data/ad/hmIngest/obs-qcmesonet-local.20120711 2000.1342037114.incoming; mv
/data/ad/hmIngest/obs-qcmesonet-local.20120711 2000.1342037114.incoming
/data/ad/hmIngest/obs-qcmesonet-local.20120711 2000.1342037114; /
/ad/bin/sendLDADnocation.pl R \"obs:qcmesonet:local 
/data/ad/hmIngest/obs-qcmesonet-local.20120711 2000.1342037114\" ls1 15007" length = 506
```

```
20:05:42.447 listener.c EVENT: do_G_Msg: scp completed for
alert_r5.dat.1342037141
```

```
20:05:42.447 listener.c EVENT: do_G_Msg: Successfully scp’ed
alert_r5.dat.1342037141, length = 534
```

```
20:05:42.447 listener.c EVENT: do_G_Msg: Preprocessor Command Line =
</awips/ldad/bin/preProcessLDAD.pl /data/fxa/LDAD/tmp/alert_r5.dat.1342037141
px2f 15009>
```
The format of the log consists of the following fields:

- **Time stamp**: 20:05:42.447
- **File name**: listener.c
- **Message type**: EVENT
- **Message string**: do_G_Msg: Preprocessor Command Line = 
  `</awIps/ldad/bin/preProcessLDAD.pl/data/fxa/LDAD/tmp/alert_r5.dat.1342037141 px2f
  15009>`

### 8.7.5 **CO_serv**

The Server program (**CO_serv**) opens a socket and continually listens for connections from the Client program. When one arrives, a child process is spawned that reads the interprocess message, extracts the name of the “session” file (located in `/ldad/data/<id_coll_sess>`) for the gauge in question (LARC/Handar 555), and parses the file to create two temporary files: a response file and a script file. **CO_serv** uses the `process_expect` executable (communications protocol `cu`), to call the LARC/Handar 555 gauge. **CO_serv** stores the LARC/Handar 555 data in the `/data/Incoming` directory and copies the “eyewash” file to the `/data/ldad/public/point/hydro` directory. **CO_serv** passes the notification to the **listener**.

#### To Access and Read the **CO_serv** Log

The **CO_serv** log resides on the active LS in the `$LOG_DIR` directory and records messages from the **CO_serv** process. Perform the following commands as user ldad:

**TYPE:** `cd $LOG_DIR/<yyyymmdd>`
  - Where `<yyyymmdd>` is today’s date.

**TYPE:** `ls -ltr CO_serv*`
  - Displays the current day’s **CO_serv** logs from earliest to latest.

**TYPE:** `tail -f <logname>`
  - Displays the latest registered messages as the log updates.

#### Log Message Format

An entry at the beginning of this log looks like the following:

```
22:45:40.215 co_serv_main.c EVENT: CO_serv received a socket connection at: Apr 23 12 22:45:40 GMT
```

```
22:45:40.218 co_serv_main.c EVENT: V message Received by CO_serv....
```

```
```
Chapter 8  Local Data Acquisition and Dissemination (LDAD) System

22:50:39.911 co_serv_main.c EVENT: G  message Received by CO_serv.....

22:50:40.201 co_serv_main.c EVENT: CO_serv received a socket connection at: Apr 23 12 22:50:40 GMT

22:50:40.204 co_serv_main.c EVENT: V  message Received by CO_serv.....


22:55:39.903 co_serv_main.c EVENT: G  message Received by CO_serv.....


22:55:40.148 co_serv_main.c EVENT: V  message Received by CO_serv.....

The format of the log file consists of the following fields:

- Time stamp: 22:45:40.215
- File name: co_serv_main.c
- Message type: EVENT:
- Message string: CO_serv received a socket connection at: Apr 23 12 22:45:40 GMT

When the processes have performed their duties, notifications are sent to the notificationServer, which it passes to the ldadScheduler.

**NOTE:** The process_expect executable writes its messages to the CO_serv log.

### 8.8 Decode LARC/Handar 555 Data

The newLDADdataNotification script in $FXA_HOME/ldad/bin checks the LDAD products directory for LARC/Handar 555 files. When a file is present, newLDADdataNotification passes the information to the CO_serv, which informs the listener. The listener executes the $FXA_HOME/ldad/bin/preProcessLDAD.pl script and calls routerLdadDecoder to decode the file. The decoded file is stored and a notification is sent to the user stating that new data are available.

**Processes**

- LS newLDADdataNotification
- LS CO_serv
- PX2 listener
- PX2 preProcessLDAD.pl
- PX2 CommsRouter LDAD_ROUTER
- PX2 DataController LDAD_ROUTER LdadController.config
- PX2 routerLdadDecoder
- PX2 routerStoreTextEDEX
- PX2 routerStoreEDEX
- PX2 routerShefEncoderEDEX
- PX2 distributeProduct
**DX1 notificationServer**

**Configuration Files /data/liga/LDAD/data**

- LDADinfo.txt (for listener)
- Station.txt (for routerStoreEDEX)
- LdadController.config (for DataController LDAD_ROUTER)
- LdadPatterns.txt (for routerLdadDecoder, routerStoreEDEX, routerStoreText, routerShefEncoder)
- LARC.desc (for routerLdadDecoder, routerStoreEDEX, routerShefEncoderEDEX)

**Configuration Files in /awips/liga/data**

- LdadWmoAwipsId.tbl (for routerStoreTextEDEX)
- LDADWmoID.cfg (for routerShefEncoder)
- edex-info.tx (for routerStoreEDEX, routerShefEncoderEDEX)

Refer to Exhibit 8.8-1 for the Decode LARC/Handar 555 data flow diagram.

### 8.8.1 newLDADdataNotification

The `newLDADdataNotification` process uses the file’s name to determine the type of data contained in the file and checks the LDAD products directory (`/data/Incoming`) periodically for LARC/Handar 555 files.

The `newLDADdataNotification` process creates and sends an IPC message to the `CO_serv` process using `createAndSendIPCmessage`. The message notifies `CO_serv` that there are LARC/Handar 555 files in the `/data/Incoming` directory.

If `newLDADdataNotification` fails to receive confirmation from `listener` within 5 minutes or if an `fstat` on the file to transfer fails, the product gets moved to the `/data/ldad/Problem` directory using the `moveFileToProblemDir` process. Otherwise, the “processed” files are moved to the `/data/ldad/Processed` directory using the `moveFileToProcessedDir` process.

**To Access and Read the newLDADdataNotification Log**

The `newLDADdataNotification` log resides on the active LS in the `$LOG_DIR` directory and records messages from the `newLDADdataNotification` process. Perform the following commands as user ldad:

**TYPE:**

```bash
cd $LOG_DIR/<yyyymmdd>
```

- Where `<yyyymmdd>` is today’s date.

**TYPE:**

```bash
ls -ltr newLDADdataNotification*
```

- Displays the current day’s `newLDADdataNotification` logs from earliest to latest.
Exhibit 8.8-1. Decode LARC/Handar 555 Data (for LDAD) (Page 1 of 2)
TYPE: \texttt{tail -f <logname>}

- Displays the latest registered messages as the current log updates.

Log Message Format

An entry at the beginning of this log looks like the following:

```
LOG-STATUS: Log file opened on host px2-tbdw at Wed Apr 12 00:00:28 2006
00:00:28.906 newLDADdataNotification.c EVENT: newLDADdataNotification process started
00:00:28.911 newLDADdataNotification.c EVENT: Reading /data/Incoming/ every 20 seconds
00:00:28.915 newLDADdataNotification.c EVENT: Processing : /data/Incoming/RAWS.dat
00:00:29.080 newLDADdataNotification.c EVENT: File more than 24hours old, removed
/data/Incoming/satxssmipw_20060412_2300.nc.gz
13:45:37.981 newLDADdataNotification.c EVENT: Processing : /data/Incoming/RAWS.dat
13:45:41.926 newLDADdataNotification.c EVENT: waitForReply SUCCESS on
/data/Incoming/RAWS.dat
13:57:38.049 newLDADdataNotification.c EVENT: Processing :
/data/Incoming/LARCPORM2135555.dat
13:57:40.343 newLDADdataNotification.c EVENT: waitForReply SUCCESS on
/data/Incoming/LARCPORM2135555.dat
```

The format of the log file consists of the following fields:

- Time stamp
- File name
- Message type
- Message string

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time stamp</td>
<td>13:57:40.343</td>
</tr>
<tr>
<td>File name</td>
<td>newLDADdataNotification.c</td>
</tr>
<tr>
<td>Message type</td>
<td>EVENT:</td>
</tr>
<tr>
<td>Message string</td>
<td>waitForReply SUCCESS on</td>
</tr>
<tr>
<td></td>
<td>/data/Incoming/LARCPORM2135555.dat</td>
</tr>
</tbody>
</table>

**NOTE:** Handar 555 data files are similar to “55LARCSNMC1155303.dat.”

8.8.2 \textit{CO\_serv}

\textit{CO\_serv} initially sets up a (TCP/IP) socket and waits for connections. The notification received from \textit{newLDADdataNotification} is passed across the firewall to the \textit{listener} process on the internal side on the PX2.

- **To Access and Read the CO\_serv Log**

  The \textit{CO\_serv} log resides on the active LS in the \texttt{LDAD\_EXTERNAL\_LOGDIR} directory and records messages from the \textit{CO\_serv} process. Perform the following commands as user ldad:

  \textbf{TYPE:} \texttt{cd $LOG\_DIR/<yyyymmdd>}
  
  - Where \texttt{<yyyymmdd>} is today’s date.

  \textbf{TYPE:} \texttt{ls -ltr CO\_serv*}
  
  - Displays the current day’s \textit{CO\_serv} logs from earliest to latest.

  \textbf{TYPE:} \texttt{tail -f <logname>}

- Displays the latest registered messages as the current log updates.

**Log Message Format**

An entry at the beginning of this log looks like the following:

```
LOG-STATUS: Log file opened on host ls2-rnk at Wed Apr 12 00:00:17 2006
00:00:17.076 co_serv_main.c EVENT: CO_serv process started
00:05:07.791 co_serv_main.c EVENT: CO_serv received a socket connection at: Apr12 06
00:05:07 GMT
00:05:07.801 co_serv_main.c EVENT: R message Received by CO_serv.....
00:06:36.998 co_serv_main.c EVENT: CO_serv received a socket connection at: Apr12 06
00:06:36 GMT
00:06:36.995 co_serv_main.c EVENT: R message Received by CO_serv.....
00:13:48.291 co_serv_main.c EVENT: R message Received by CO_serv.....
```

The format of the log file consists of the following fields:

- **Time stamp**: 00:00:17.076
- **File name**: co_serv_main.c
- **Message type**: EVENT:
- **Message string**: CO_serv process started

### 8.8.3 listener

The listener reads the SFXA_DATA/LDAD/data/LDADinfo.txt config file, sets up a socket, binds its local address so that the client (CO_serv) can send to it, and waits for the connection from the Client process. When the notification is received, the listener retrieves the LARC/Handar 555 file from the /data/Incoming directory on the active LS and copies it to the $LDAD_DATA/tmp directory on the PX2. If the file needs no preprocessing, the listener moves the data file directly to the Raw directory. The listener sends a verification when finished.

The listener executes the SFXA_HOME/ldad/bin/preProcessLDAD.pl script to preprocess the LARC/Handar 555 data files, and receives the fully qualified data file, the hostname used by notification routine, the port number of the process requiring notification upon completion of preprocessing, and other necessary information about the LARC/Handar 555 data file format from CO_serv. The listener then sends notification to the CommsRouter LDAD_ROUTER process that a product has been preprocessed.

### To Access and Read the listener Log

The listener log resides on PX2 in the directory $LOG_DIR and records messages from the listener process. Perform the following commands as user ldad:

```
TYPE: cd $LOG_DIR/<yyyymmdd>
```
- Where <yyyymmdd> is today’s date.

**TYPE:** `ls -ltr listener*`
- Displays the current day’s listener logs from earliest to latest.

**TYPE:** `tail -f <logname>`
- Displays the latest registered messages as the current log updates.

**Log Message Format**

An entry at the beginning of this log looks like the following:

```
LOG-STATUS: Log file opened on host px2-tbdw at Wed Apr 12 00:00:12 2006
00:00:12.616 listener.c EVENT: listener process started
ls1:/data/Incoming/LARCPORM2135555.dat
/data/fxa/LDAD/tmp/01LARCPORM2135555.dat.973000658
13:57:40.266 listener.c EVENT: do_G_Msg: rcp completed for
01LARCPORM2135555.dat.973000658
13:57:40.292 listener.c DEBUG: pInfo->preprocessorScript =
<preProcessLDAD.pl>
</awips/ldad/bin/preProcessLDAD.pl
/data/fxa/LDAD/tmp/01LARCPORM2135555.dat.973000658 px 15009>
</awips/ldad/bin/preProcessLDAD.pl
/data/fxa/LDAD/tmp/01LARCPORM2135555.dat.973000658 px 15009>
```

The format of the log consists of the following fields:

- **Time stamp**
- **File name**
- **Message type** DEBUG:
- **Message string** do_G_Msg: rcp

**NOTE:** Handar 555 files are similar to “55LARCSNMC1155303.dat.1029513249.”

### 8.8.4 preProcessLDAD.pl

The `SFXA_HOME/ldad/bin/preProcessLDAD.pl` script preprocesses UDFCD ALERT (_R5, _WL, _WX), Cdot, IFLOWS, and LARC/Handar 555 data report files for the LDAD decoder. It initializes file input and output and values to the existing variables, calls the appropriate preprocessor (`preprocessLARC.pl` for LARC/Handar 555 data), and cleans up.
8.8.5 preprocessLARC.pl

The $SFXA_HOME/ldad/bin/preprocessLARC.pl script copies the file from the $SLDAD_DATA/tmp directory to the /awips/fxa/ldad/ldadtmp directory. There, it reformats the LARC/Handar 555 data file, appends the abstime string to the file name, and stores the preprocessed file in the $SLDAD_DATA/Raw directory.

8.8.6 LDAD_ROUTER

The LDAD_ROUTER (registered LDAD CommsRouter) is responsible for transporting data notifications from the data acquisition processes to the data controller processes. The main function of the LDAD_ROUTER is message passing; it does no data processing. The router provides a known location where the data acquisition processes can send data messages.

To Access and Read the CommsRouter Log

The CommsRouter log resides on PX2 in a time-stamped directory under $SLOG_DIR. Perform the following commands as user ldad:

TYPE: cd $LOG_DIR/<yyyymmdd>
    Where <yyyymmdd> is today’s date.

TYPE: ps -wef |grep ldad
    Displays the ldad processes. Notice the PID of the CommsRouter process is owned by user ldad.

TYPE: ls -rtal Comms*
    Displays the latest logs for the CommsRouter LDAD_ROUTER.

NOTE: The CommsRouter log containing the PID of the LDAD_ROUTER process is the LDAD_ROUTER’s CommsRouter log.

TYPE: tail -f <logname>
    Displays the log as it updates.

Log Message Format

An entry in the log should look like the following:

LOG-STATUS: Log file opened on host px2-tbdw at Tue Apr 18 00:00:09 2006
00:00:09.558 CommsReceiver.C EVENT: NCF_ENTRY CRMmsg Pattern:
^[0-9.][^m][^s][^a][^s][^s][^s][^q][^c]]($|)13:57:44.861 CommsReceiver.C EVENT: NCF_ENTRY DSMsg:
/data/fxa/1LDAD/Text/01LARCPORM213555.dat.973000658 1 51
The log consists of the following fields:

<table>
<thead>
<tr>
<th>Time stamp</th>
<th>14:02:47.450</th>
</tr>
</thead>
<tbody>
<tr>
<td>File name</td>
<td>CommsReceiver.C</td>
</tr>
<tr>
<td>Message type</td>
<td>EVENT:</td>
</tr>
<tr>
<td>Message string</td>
<td>NCF_ENTRY DSMsg: 01LARCPORM2140053.dat.973000958 1 31</td>
</tr>
</tbody>
</table>

The LDAD_ROUTER CommsRouter is started automatically at system reboot time or can be started manually. The CommsRouter runs continuously, exiting only when killed by the restart program.

NOTE: Handar 555 files are similar to “55LARCSNMC1155303.dat.1029513249.”

8.8.7 DataController

The DataController is responsible for creating the decoder and storage processes (as child processes), restarting them if they exit, routing appropriate messages to them, and queuing incoming data messages until the decoder and storage processes are ready. As data messages are stored in the queue, the size of the processes grows.

The DataController’s main program creates a DataRouting manager object when it first starts up and invokes a member function to start the routerLdadDecoder, routerStoreTextEDEX, routerStoreEDEX, and routerShefEncoderEDEX processes by reading and executing the LdadController.config file. The file resides on the $LDAD_INTERNAL_DATA directory and contains the names of the processes above to start/restart, a restart flag, and end of file (eof) designators to alert the Data Routing Manager that there are no more processes listed in the file.

The LDAD_ROUTER sends product-received notifications to the DataController. When the processes start up, they send messages containing their IPC addresses and process numbers to the DataController. The processes send the DataController a data request message and a ready-to-receive data message. In response, the DataController stores the processes’ ID information and data requests, and sends a message to the requesting processes (if a message is available). The DataController waits to receive another ready message from the processes before sending any more data messages. The
**DataController** queues any messages received from the **LDAD_ROUTER** until the processes send another ready message.

### To Access and Read the DataController Log

The **DataController** log resides on PX2 in a time-stamped directory under **SLOG_DIR**. Perform the following commands as user ldad:

**TYPE:**
```
cd $LOG_DIR/<yyyymmdd>
```
- Where `<yyyymmdd>` is today’s date.

**TYPE:**
```
ls -lrtal DataController*
```
- Displays the latest logs for the **DataControllers**.

**TYPE:**
```
tail -f <logname>
```
- Displays the log as it updates with the latest messages.

### Log Message Format

An entry in the log should look like the following:

```
01:30:23.120 CommsReceiver.C EVENT: NCF_ENTRY CDMsg:
data/fxa/LDAD/Raw/alert_r5.dat.1335231022 1 42
01:30:23.121 CommsReceiver.C EVENT: NCF_ENTRY CDMsg:
data/fxa/LDAD/Raw/RAWS.dat.1335231022 1 38
01:30:23.156 CommsReceiver.C EVENT: NCF_ENTRY CDMsg: 1335231022.alert_r5.decoded 1 50
01:30:23.164 CommsReceiver.C EVENT: NCF_ENTRY CDMsg: 1335231022.RAWS.decoded 1 46
01:30:25.137 CommsReceiver.C EVENT: NCF_ENTRY CDMsg: alert_r5.dat.1335231022 1 23
01:35:22.636 CommsReceiver.C EVENT: NCF_ENTRY CDMsg:
data/fxa/LDAD/Text/alert_r5.dat.1335231322 1 43
01:35:22.657 CommsReceiver.C EVENT: NCF_ENTRY CDMsg:
data/fxa/LDAD/Raw/alert_r5.dat.1335231322 1 42
01:35:22.660 CommsReceiver.C EVENT: NCF_ENTRY CDMsg: 1335231322.alert_r5.decoded 1 50
01:35:24.675 CommsReceiver.C EVENT: NCF_ENTRY CDMsg: alert_r5.dat.1335231322 1 23
```

The log file consists of the following fields:

- **Time stamp**
- **File name**
- **Message type**
- **Message string**

| Time stamp | 01:30:23.120 |
| File name | CommsReceiver.C |
| Message type | EVENT: |
| Message string | NCF_ENTRY CDMsg: data/fxa/LDAD/Raw/alert_r5.dat.1335231022 1 42 |
NOTE:  Handar 555 files are similar to “55LARCSNMC1155303.dat.1029513249.

8.8.8 routerLdadDecoder

The routerLdadDecoder process is spawned by the DataController as specified in the LdadController.config file. The routerLdadDecoder process sends a data request to the DataController for the LARC/Handar 555 products it wishes to receive by specifying product header patterns of interest. On receipt of a notification from the DataController that a product of interest has arrived, the routerLdadDecoder reads the directory containing the raw files, $LDAD_DATA/Raw.

The routerLdadDecoder process reads each file and stores it via serialized data files to the $LDAD_DECODED_DATA directory. The serialized file name is sent back through the LDAD_ROUTER to notify registered LDAD storage processes of the arrival of each LDAD decoded product.

To Access and Read the routerLdadDecoder Log

The routerLdadDecoder log resides on PX2 in a time-stamped directory under SLOG_DIR. Perform the following commands as user ldad:

TYPE:  cd $LOG_DIR/<yyyymmdd>
  •  Where <yyyymmdd> is today’s date.

TYPE:  ls -rtal routerLdadDecoder*
  •  Displays the latest routerLdadDecoder logs.

TYPE:  tail -f <logname>
  •  Displays the log as it updates with the latest messages.

Log Message Format

An entry in the log should look like the following:

20:15:42.973 LdadRoutines.C EVENT:  Processing [1/1] = RAWS.dat.1342037741
20:15:42.978 LdadRoutines.C EVENT: LdadRoutines::pingLdadRouter(),
  "$LDAD_DECODED_DATA/ 1342037741.RAWS.decoded", sent to LDAD_ROUTER

20:15:43.160 LdadRoutines.C EVENT: LdadRoutines::pingLdadRouter(),
  "$LDAD_DECODED_DATA/ 1342037741.alert_r5.decoded", sent to LDAD_ROUTER

20:16:02.252 LdadRoutines.C EVENT:  Processing [1/1] =
  1342036800.16.msas_qc.RAWS
20:16:02.256 LdadRoutines.C EVENT: LdadRoutines::pingLdadRouter(),
  "$LDAD_DECODED_DATA/ RAWS.1342036800.16.msas_qc.decoded", sent to LDAD_ROUTER

The log file consists of the following fields:

Time stamp  20:15:42.973
File name    LdadRoutines.C
Message type EVENT: Processing [1/1] = RAWS.dat.1342037741

NOTE: Handar 555 files look similar to “55LARCSNMC1155303.dat.1029513249.”

8.8.9 routerStoreTextEDEX

The routerStoreTextEDEX process is spawned by the DataController as specified in the LdadController.config file. The routerStoreTextEDEX process sends a data request to the DataController for the text products it wishes to receive by specifying product header patterns of interest. Upon receipt of a notification from the DataController that a product of interest has arrived (based on the pattern in /awips/fxa/ldad/data/LdadPatterns.txt file), the routerStoreTextEDEX reads the directory containing the decoded files, SLDAD_DECODED_DATA.

The routerStoreTextEDEX process reads each file and stores it temporarily in the SLDAD_DATA/Text directory. The routerStoreTextEDEX process reads /awips/fxa/data/LdadWmoAwipsId.tbl file to determine what WMO ID is associated with the product and using that ID to store the product into the fxatext database.

To Access and Read the routerStoreTextEDEX Log

The routerStoreTextEDEX log resides on PX2 in a time-stamped directory under SLOG_DIR. Perform the following commands as user ldad:

TYPE: cd $LOG_DIR/<yyyymmdd>
  ▪ Where <yyyymmdd> is today’s date.

TYPE: ls -rtal routerStoreTextEDEX*
  ▪ Displays the latest routerStoreTextEDEX logs.

TYPE: tail -f <logname>
  ▪ Displays the log as it updates with the latest messages.

Log Message Format

An entry in the log should look like the following:

01:35:22.650 LdadStoreText.C EVENT: LdadStoreText::getLdadProductType(), found: <alert_r5> in </data/fxa/LDAD/Text/alert_r5.dat.1226295637>
01:35:22.650 LdadStoreText.C EVENT: LdadStoreText::getLdadProductType(), found: <alert_r5> in </data/fxa/LDAD/Text/alert_r5.dat.1228995927>
01:35:22.650 LdadStoreText.C EVENT: LdadStoreText::getLdadProductType(), found: <alert_r5> in </data/fxa/LDAD/Text/alert_r5.dat.1229018126>
01:35:22.650 LdadStoreText.C EVENT: LdadStoreText::getLdadProductType(), found: <alert_r5> in </data/fxa/LDAD/Text/alert_r5.dat.1229146539>
01:35:22.651 LdadStoreText.C EVENT: LdadStoreText::getLdadProductType(), found: <alert_r5> in </data/fxa/LDAD/Text/alert_r5.dat.1229147139>
01:35:22.651 LdadStoreText.C EVENT: LdadStoreText::getLdadProductType(), found: <alert_r5> in </data/fxa/LDAD/Text/alert_r5.dat.1229426723>
8.8.10 routerStoreEDEX

The routerStoreEDEX process is spawned by the DataController as specified in the LdadController.config file. The routerStoreEDEX process sends a data request to the DataController for serialized decoded LARC/Handar 555 files in netCDF format. When notified by the DataController that a product of interest has arrived, routerStoreEDEX reads the directory containing the files, $LDAD_DECODED_DATA. The routerStoreEDEX process stores the files in the /data_store/ldad directory in XML format and sends a notification to the EDEX.

To Access and Read the routerStoreEDEX Log

The routerStoreEDEX log resides on PX2 in a time-stamped directory under $LOG_DIR. Perform the following commands as user ldad:

TYPE: cd $LOG_DIR/<yyyymmdd>
   - Where <yyyymmdd> is today’s date.

TYPE: ls -rtal routerStoreEDEX*
   - Displays the latest logs for the process. If more than one log exists, choose the log for the current routerStoreEDEX process by looking for the PID in the logname.

TYPE: tail -f <logname>
   - Displays the log as it updates with the latest messages.

Log Message Format

An entry in the log should look like the following:

LOG-STATUS: Log file opened on host px2-tbw3 at Tue Jul 13 00:30:06 2010
00:15:43.050 routerStoreEDEX.C EVENT: NCF_ENTRY
/data/exa/LDAD/decoded/1341965741.alert_r5.decoded
00:15:43.074 routerStoreEDEX.C EVENT: NCF_STOREd to
/data_store/ldad/LDAD.hydro.1341965741.alert_r5.decoded.xml
00:15:43.075 routerStoreEDEX.C EVENT: 1341965741.alert_r5.decoded processing completed
00:16:02.927 routerStoreEDEX.C EVENT: NCF_ENTRY
/data/exa/LDAD/decoded/RAWS.1341964800.16.maas_qc.decoded
The log file consists of the following fields:

- **Time stamp**: 00:15:43.050
- **File name**: routerStoreEDEX.C
- **Message type**: EVENT:
- **Message String**: /data/fxa/LDAD/decoded/1341965741.alert_r5.decoded

### NOTE: Handar 555 files are similar to “55LARCSNMC1155303.decoded.1029513249.”

#### 8.8.11 routerShefEncoderEDEX

The **routerShefEncoderEDEX** process is spawned by the **DataController** as specified in the **LdadController.config** file. The **routerShefEncoderEDEX** process sends a data request to the **DataController** for serialized decoded files. On receipt of a notification from the **DataController** that a product of interest has arrived, the **routerShefEncoderEDEX** reads the directory containing the files, `$LDAD_DECODED_DATA`. The files there are SHEF-encoded and moved to the `/data_store/ldad` directory. The **routerShefEncoderEDEX** process sends a notification to the **EDEX**. Once the SHEF-encoded data file is stored in the `/data_store/ldad` directory, the **routerShefEncoderEDEX** saves a copy of the file in the `/tmp` directory and calls **distributeProduct** process to send the file to the NCF over the WAN.

To Access and Read the **routerShefEncoderEDEX** Log

The **routerShefEncoderEDEX** log resides on PX2 in a time-stamped directory under **SLOG_DIR**. Perform the following commands as user ldad:

- **TYPE:** `cd $LOG_DIR/<yyyymmdd>`
  - Where `<yyyymmdd>` is today’s date.

- **TYPE:** `ls -rtal routerShefEncoder*`
  - Displays the latest **routerShefEncoderEDEX** logs.

- **TYPE:** `tail -f <logname>`
  - Displays the log as it is updated with the latest messages.

**Log Message Format**

An entry in the log should look like the following:

```
20:25:42.703 routerShefEncoderEDEX.C EVENT: NCF_STOREd to 
/data_store/ldad/SXUS44 KWOHshef.1342038342
```
8.8.12 *distributeProduct*

The *distributeProduct* process creates a product message and submits it for distribution across the AWIPS WAN to the addressed sites. The *distributeProduct* process first prepares the product by creating the WAN communications header and prepending the header to a temporary copy of the product.

**To Access and Read the distributeProduct Log**

The *distributeProduct* log resides on PX2 in a time-stamped directory under $LOG_DIR$. Perform the following commands as user ldad:

**TYPE:**

```bash
cd $LOG_DIR/<yyyymmdd>
```

*Where* `<yyyymmdd>` *is today’s date.*

**TYPE:**

```bash
ps -ef |grep ldad
```

*Displays the ldad processes.*

**TYPE:**

```bash
ls -ltr distributeProduct*
```

*Displays the latest logs for the distributeProduct process.*

**TYPE:**

```bash
tail -f <logname>
```

*Displays the log as it updates with the latest messages.*

**Log Message Format**

An entry in the log file should look like the following:

```
LOG-STATUS: Log file opened on host px2-tbdw at Wed Apr 12 19:30:59 2006
19:30:59.690 MHSProduct.C PROBLEM: Product (/data/fxa/ispan/hydro/temp/A_format.973020657_msg) is empty.
19:30:59.853 MHSProduct.C EVENT: Sending attachments (/data/fxa/ispan/hydro/temp/A_format.973020657) only. Attachments are assumed to contain correct communications header.
19:31:02.504 MHSProduct.C EVENT: Product request (KSTORRZLDA) was successfully submitted to the MHS request server for distribution.
```
Message Attributes:
Request ID: TBDW-25380
To: DEFAULTNCF
Priority: 0
Type: Routine
Subject:
Handling Action: 0
19:31:02.518 MHSProduct.C EVENT: TBDW-25380
~

The log file consists of the following fields:

- Time stamp: 19:31:02.504
- File name: MHSProduct.C
- Message type: EVENT:
- Message string: Product request (KSTORRZLDA) was successfully submitted to the MHS request server for distribution.

Message Attributes:
Request ID: TBDW-25380
To: DEFAULTNCF
Priority: 0
Type: Routine
Subject:
Handling Action: 0

8.9 Acquire Campbell Data

Data collection from Campbell gauges is carried out by automating a normally interactive session using the cu communication protocol. There is a Server program (CO_serv) and a Client program (tell_co). The Server program is always running, waiting for, and processing requests to collect data from a particular gauge from the Client program. The Client program can be invoked from a shell script, as a cron job, or directly from a command prompt. A feature of the Client program is that it retrieves all necessary information regarding collection and storage of data from system tables in the AWIPS database. The Client and Server programs use sockets to communicate across a TCP/IP LAN. Data from a gauge are SHEF-encoded prior to storage on AWIPS. The .dat file is moved into the Raw directory, while the SHEF file is stored into the text database, archived, and sent to the Office of Hydrology’s database during preprocessing. The .dat file is forwarded to the ldadDecoder process to further decode the Campbell data for creation of the netCDF and SHEF-encoded files.

Processes

- LS  callSite.ksh
- LS  process_expect
- LS  CO_serv
- LS  campbell
- PX2  listener
- PX2  tell_co
- PX2  ldadServer
- LX  ldadScheduler
- DX1  notificationServer
Configuration Files (all in /data/fxa/LDAD/data)

LDADinfo.txt (for listener)
CAMPBELLStation.txt (for tell_co, ldadScheduler)
userDescriptions (for ldadScheduler)
gaugeSessList (for ldadScheduler)
userSessList (for ldadScheduler)
usrpro (for ldadScheduler)
gageinfo (for ldadScheduler)
gageinfoCampbell.txt (for ldadScheduler)
gaugeDescriptions (for ldadScheduler)
SHEFcodes.cfg (for campbell)

LDADinfo.txt

The file should contain the following entry:

CAMPBELL | CAMPBELL | 89 | 90 | CVS_TYPE | hydro |
preprocessCAMPBELL.pl |

Once the file is installed, this line should not be modified.

CAMPBELLStation.txt

This file must be updated whenever a gauge is added to or dropped from the current Campbell datalogger list. To add a new Campbell gauge, find out the following gauge information to create a new entry in the station table format (* indicates a mandatory field):

- **Provider ID:** Data provider station or gauge ID (*)
- **AFOS(HB5) ID:** The AFOS or Handbook 5 ID (NWS assigned)
- **Station Name:** Text name of station
- **Elevation:** Station elevation (m) (*)
- **Latitude:** Station Latitude (decimal degrees, north positive) (*)
- **Longitude:** Station Longitude (decimal degrees, east positive) (*)
- **Local Time Zone:** Reporting time zone (e.g., UTC, PST8PDT) (*) (see Note)
- **Location Description:** Station location/address (Text)
- **Station Type:** Type of station (e.g., tower, surface, floating platform, etc.)
- **Number of Instruments:** The number of reporting instruments for the station
- **Number of Reporting Levels:** Number of data reporting levels. Level information should be provided in the instrument table
- **Maintenance/Calibration Schedule:** Frequency of maintenance/calibration
• **Site Description**: A text description of the site surroundings.

The entries are sorted alphabetically by the gauge ID.

This file must be identical on the PX2 and on LS1!

**NOTE**: The local time zone for Campbell gauges is always GMT because the Campbell Decoder already converts the observation times into GMT times.

**userDescriptions**

This file is used by the ldadScheduler process to get the descriptions of users. When read in, blank lines and lines beginning with # are skipped. All other lines should have the following format:

```
<user ID>|<user description>
```

A blank user description is denoted by having the line end immediately following the | character (a vertical bar or “pipe”).

This file should not be modified manually. It should only be written to by the ldadScheduler (specifically, the LdadSchedulerServer.C routine).

**usrpro**

The user file (usrpro) is made up of zero to n lines, each line holding the following information (note that the lines are so long that they are wrapped in the interests of readability). Columns containing “??” are not used at this time.

```
<ID>|<protocol>|<IP address>|<phone number>|<login>|<password (encrypted)>|??|<file to receive>|<file to send>|??|??|<data collection session file>|<data dissemination session file>|<local directory>|<remote directory>|??...
```

- The <protocol> is either “F” for FTP, “X” for Xmodem, “Y” for Ymodem, “Z” for Zmodem, or “K” for Kermit.
- The <IP address> designates the host to FTP to, if the protocol is “F,” or the terminal server otherwise. <IP address> appears in two places in the above column listing: columns 3 and 37. In reality, only one of the two columns is used. For the FTP protocol, only column 3 has the address in it, and column 37 is left blank. If any other protocol is being used, column 37 has the address in it, and column 3 is left blank.
- The <phone number> field is unused if the protocol is “F.”
- The <file to receive> and <file to send> fields are mutually exclusive. Only one may be specified, and which one is specified indicates also whether this user is a collection user (if <file to receive> is given) or a dissemination user (if <file to send> is given). All other columns are tracked when read in; when written out, the same values are placed in them as were originally found.
gageinfo

The gageinfo file contains a list of Handbook 5 and Campbell gauges. It is updated periodically as Campbell gauges are added and removed with the ldadScheduler. Whenever a gauge is added or dropped from the current list, it should be updated here and in the following files:

$LDAD_DATA/data/gageinfoCampbell.txt
$LDAD_DATA/data/CAMPBELLStation.txt
$LDAD_DATA/data/gaugeDescriptions

The following fields must be present for each gauge (see Note):

- **Gauge ID:** A five-character instrument ID.
- **Protocol Type:** C for data collection, D for dissemination.
- **Phone#:** The dial-out phone number can be in any format. However, if the gauge has the voice message first, several pauses are included in the phone number. Depending on the length of the voice message, add more pauses if the one listed here is not enough. The pause is entered by comma (,) and is ended with a *9. If a phone number contains a pause extension, make sure the phone number is enclosed by single quotes('. Example: ‘1-605.886.2473,,,,,*9’
- **Session File:** Use CAMPBELL_coll_sess exactly.
- **SHEF File:** 1 = Yes; 0 = No.
- **Storage Type:** S.
- **Baud Rate:** Use 9600 for Campbell gauges.
- **Terminal Server (tty):** Use 8n1out for Campbell gauges.

The fields are separated by vertical bars (|); empty fields are for future use. A number nine (9) in front the phone number allows the user to dial out. An example follows:

```
BBCW3|C| |917158843133| | | | |CAMPBELL_coll_sess|1|S| | |9600|e| |8n1out|
```

**NOTE:** The rcvprog field indicates the receiving program (e.g., FTP) for data collection. This field is used as the custom gauge ID if the gauge is owned by the NWS.

gageinfoCampbell.txt

This file should be updated along with the gageinfo file when adding/dropping Campbell gauge(s). There are at least four fields for each Campbell gauge to be entered when adding a gauge; the fields are as follows:

- **Gauge ID:** A five-character instrument ID (i.e., BRCW3).
- **Local Time Zone:** Time zone where the Campbell gauges are located, (e.g., PST, CDT, GMT, etc.).
- **Data Array ID for Storage Area 1:** Four digit array ID.
- **Observation Field Skip Bit Map for Array ID Number One:** This is an eight-bit field used to mark the observation fields to be ignored.
- **Bit Map for Observation Data Fields in PE Codes for Storage Area 1 in Hexadecimal Format:** This is a 32-bit code (see Note). The leftmost byte is designed to ignore collected data fields and the other three are used to map the type of data collected relevant to the PE codes listed in the `SHEFcodes.cfg` file.
- **Data Array ID for Storage Area 1:** Four digits array ID for gauge collection data and storing in Storage Area 2.
- **Bit Map for Collecting Data Fields in PE Codes for Storage Area 2:** Same format as Storage Area 1 if the gauge collects data and stores it in Storage Area 2.

**NOTE:** Current design can accommodate a maximum of eight data fields only for a Campbell gauge and a maximum of 32 PE codes collected by all Campbell gauges. If an individual gauge collects more than eight different data fields or if the total number of the PE codes collected by the Campbell gauges exceeds 32, these fields will have to be expanded to incorporate the changes.

**gaugeDescriptions**

The file is used to get the site descriptions and should be updated using the `ldadScheduler` when a new Campbell gauge is added. Each entry (line) should have the following format:

\[
\text{<gauge ID> | <site description>}
\]

For example, an entry for the BARW3 would be as follows:

```plaintext
BARW3 | Barron, WI
```

The file is sorted by state abbreviations. However, it is displayed based on the alphabetical order of the gauge IDs.

**SHEFcodes.cfg**

The file is designed specifically for Campbell gauges to enable creation of SHEF-encoded files (.shef). It contains the PE codes currently collected by Campbell CR10X gauges. The elements in the file are presented in alphabetical order and should be updated every time a new PE code is added to the file. When this file is modified, the LDAD system has to be shut down and restarted again.

The codes are used in constructing the 3-byte PE code bit map in the `gageinfoCampbell.txt` file. It is also used as the base for creating/updating of the `SLDAD_DATA/data/CAMPBELL.desc` file.

Refer to Exhibit 8.9-1 for the Acquire Campbell Data data flow diagram.
8.9.1 User Interface: ldadScheduler

The ldadScheduler's function is to perform certain data acquisition/dissemination events on an automatic basis at specified time intervals (in the background). Access to the Scheduler’s functions is via a GUI invoked by selecting Collection/Dissemination... from the Surface pull-down menu.

The ldadScheduler provides a window by which the user may create, view, and manipulate scheduled LDAD requests. It allows the user to create a new request and add it to the request list. The request is activated when submitted to the ldadServer, which uses sendLDADnotification.pl to create the request. If it is a one-time request (OTR), the user can monitor its progress via a status dialog.

The ldadScheduler registers with the ldadServer providing a callback routine for update notifications. The session file contents get registered first, and the status messages and ID updates so that when registering with ldadServer, the ID lists sent by ldadServer will be caught and kept. It also registers with the File Access Controller (FAC) for notifications of changes to the existing requests file, and creates a FAC lock for the file to be used for locking it for “reads” and “writes.” The existing requests get read so as to fill the tree view with the requests that have been created.

8.9.2 ldadServer

The ldadServer allows multiple clients to connect to it and submit requests with intervals and durations that are honored according to their intervals until they expire.

New client initialization is performed by sending the gauge and user lists to the newly registered client. The ldadServer process creates a new message with the current active list and broadcasts it to all registered clients.

To honor the specified request, the ldadServer creates the command to be executed.

The ldadServer process receives LDAD messages from clients; messages to create and delete gauges and users; and messages requesting the contents of a session file, or specifying a session file’s contents.
Chapter 8
Local Data Acquisition and Dissemination (LDAD) System

Exhibit 8.9-1. Acquire Campbell Data (for LDAD)
To Access and Read the ldadServer Log

The ldadServer log resides on the PX2 in the $LOG_DIR directory and records messages from the ldadServer process. Perform the following commands as user ldad:

**TYPE:**
```
    cd $LOG_DIR/<yyyymmdd>
```
Where `<yyyymmdd>` is today’s date.

**TYPE:**
```
    ls -ltr ldadServer*
```
Displays the current day’s ldadServer logs from earliest to latest.

**TYPE:**
```
    tail -f <logname>
```
Displays the latest registered messages as the log updates.

Log Message Format

An entry at the beginning of the log looks like the following:

```
LOG-STATUS: Log file opened on host px2-tbdw at Wed Apr 12 20:11:03 2006
16:11:37.334 LdadSchedulerServer.C PROBLEM: The gauge "BBCW3" specified in gageinfoCampbell.txt was not found in gageinfo.txt. This gauge will not be used.
16:11:37.345 LdadSchedulerServer.C PROBLEM: The gauge "BBCW3" specified in gageinfoCampbell.txt was not found in gageinfo.txt. This gauge will not be used.
20:11:03.470 Connection.C EVENT: WARNING: this process wants to use target LDAD_SCHEDULE_SERVER which is assigned to run on px-tbdw but the current host is px2-tbdw
20:11:03.492 LdadSchedulerServer.C PROBLEM: The gauge "BBCW3" specified in gageinfoCampbell.txt was not found in gageinfo.txt. This gauge will not be used.
20:12:12.926 SignalClient.C EVENT: Signal 15 (Terminate) received for clean shutdown
```

The format of the log consists of the following fields:

- **Time stamp**: 16:11:37.345
- **File name**: LdadSchedulerServer.C
- **Message type**: PROBLEM:
- **Message string**: The gauge “BBCW3” specified in gageinfoCampbell.txt was not found in gageinfo.txt. This gauge will not be used.

8.9.3 tell_co

The ldadServer process invokes the client program tell_co, the main controller process providing the CAMPBELL ID. The **tell_co** function retrieves all information regarding a specified gauge from the LDAD configuration files. This includes all communication details, the name of the session file to use, as well as the location to store the data on AWIPS and whether SHEF encoding is required. It formats an interprocess message.
Chapter 8 Local Data Acquisition and Dissemination (LDAD) System

containing the aforementioned information, opens a socket to the listener, and sends a
get CAMPBELL Message “IPC” message to the listener.

To Access and Read the tell_co Log

The tell_co log resides on the PX2 in the $LOG_DIR directory and records
messages from the tell_co process. Perform the following commands as user ldad:

**TYPE:**

```
cd $LOG_DIR/<yyyymmdd>
```

- Where `<yyyymmdd>` is today’s date.

**TYPE:**

```
ls -ltr tell_co*
```

- Displays the current day’s tell_co logs from earliest to
  latest.

**TYPE:**

```
tail -f <logname>
```

- Displays the latest registered messages as the log updates.

Log Message Format

An entry at the beginning of the log looks like the following:

```
LOG-STATUS: Log file opened on host px2-tbdw at Wed Apr 12 19:45:55 2006
19:45:55.368 doAlivenessTest.c EVENT: waitForReply INTERNAL  SUCCESS!!!...
19:45:56.253 doAlivenessTest.c EVENT: waitForReply EXTERNAL  SUCCESS!!!...
19:45:57.205 tell_co.c EVENT: TELL_CO waitForReply SUCCESS!!!...
```

The format of the log consists of the following fields:

- Time stamp
- File name
- Message type
- Message string

8.9.4 listener

The listener sets up a socket connection, binds to a local address so that the Client
(CO_serv) or tell_co can send to it, and waits for the connection. Upon receipt of a data
collection request from tell_co, the listener forwards information received to the
CO_serv process. If any errors were encountered during the data logging, an error
message is formulated and passed to the Scheduler for display. In addition, Campbell
gauge dialing status information is sent by the listener and displayed by the Scheduler.

To Access and Read the listener Log

The listener log resides on PX2 in the $LDAD_INTERNAL_LOGDIR directory and records messages from the listener process. Perform the following commands as user ldad:

---

AWP.MAN.SMM.A2  8-52

– Hard copies uncontrolled. Verify effective date prior to use. –
Chapter 8  Local Data Acquisition and Dissemination (LDAD) System

TYPE:  
  cd $LOG_DIR/<yyyymmdd> 
  Where <yyyymmdd> is today’s date

TYPE:  
  ls -ltr listener* 
  Displays the current day’s listener logs from earliest to latest

TYPE:  
  tail -f <logname> 
  Displays the latest registered messages as the log updates

Log Message Format

An entry at the beginning of this log looks like the following:

LOG-STATUS: Log file opened on host px2-tbdw at Wed Apr 12 00:00:12 2006
00:00:12.616 listener.c EVENT: listener process started
18:18:27.652 listener.c EVENT: child listener: rcp completed for
CAMPBELL_coll_sess
18:18:27.755 listener.c EVENT: rcp_cmd_buf= 
   /awips/ldad/bin/sendLDADNotification.pl V NOTHING px2 1780
18:18:27.807 listener.c PROBLEM: child listener: failed to sendLDADNotification
   for CAMPBELL_coll_sess

The format of the log file consists of the following fields:

<table>
<thead>
<tr>
<th>Time stamp</th>
<th>18:18:27.652</th>
</tr>
</thead>
<tbody>
<tr>
<td>File name</td>
<td>listener.c</td>
</tr>
<tr>
<td>Message type</td>
<td>EVENT:</td>
</tr>
<tr>
<td>Message string</td>
<td>child listener: rcp completed for CAMPBELL_coll_sess</td>
</tr>
</tbody>
</table>

8.9.5  CO_serv

The Server program (CO_serv) opens up a socket and continually listens for connections from the Client program. When one arrives, a child process is spawned that reads the interprocess message, extracts the name of the “session” file (session file located in /ldad/data/<id_coll_sess> ) for the gauge in question (Campbell), and parses this session file to create two temporary files: a response file and a script file. CO_serv first calculates the starting point of the data to be retrieved and how many data intervals to retrieve from the working configuration file. Using the process_expect, CO_serv retrieves the data from the Campbell gauge and stores it in a temporary file. The data are sent to the campbell decoder process for decoding. If any errors were encountered during the data logging, the error messages are formulated and sent to the user’s workstation for display.

To Access and Read the CO_serv Log

The CO_serv log resides on the active LS in the directory SLOG_DIR and records messages from the CO_serv process. Perform the following commands as user ldad:
TYPE: cd $LOG_DIR/<yyyymmdd>
   Where <yyyymmdd> is today’s date.

TYPE: ls -ltr CO_serv*
   Displays the current day’s CO_serv logs from earliest to latest.

TYPE: tail -f <logname>
   Displays the latest registered messages as the log updates.

Log Message Format

An entry at the beginning of this log looks like the following:

LOG-STATUS: Log file opened on host ls1-rnk at Wed Apr 12 00:00:17 2006
00:00:17.076 co_serv_main.c EVENT: CO_serv process started
00:05:07.791 co_serv_main.c EVENT: CO_serv received a socket connection at: May12 04 00:05:07 GMT
00:05:07.801 co_serv_main.c EVENT: R message Received by CO_serv....
00:06:36.988 co_serv_main.c EVENT: CO_serv received a socket connection at: May12 04 00:06:36 GMT
00:06:36.995 co_serv_main.c EVENT: R message Received by CO_serv....
00:13:48.283 co_serv_main.c EVENT: CO_serv received a socket connection at: Apr12 06 00:13:48 GMT
00:13:48.291 co_serv_main.c EVENT: R message Received by CO_serv....

The format of the log file consists of the following fields:

- Time stamp
- File name
- Message type
- Message string

8.9.6 process_expect

The process_expect executable processes the request for a data transfer. It parses the receive or send session file and invokes do_expect to carry out the transfer by using the script file created by the parse_session_file function as input. It stores the data in the appropriate place and deletes any temporary files used during this session.

To Access and Read the CO_serv Log

process_expect writes its messages into the CO_serv log. The log resides on the active LS in the directory $LOG_DIR. Perform the following commands as user ldad:

TYPE: cd $LOG_DIR/<yyyymmdd>
   Where <yyyymmdd> is today’s date.
8.9.7 callSite.ksh

The callSite.ksh interrogates the datalogger and passes the raw data received from the gauge to the campbell decoder for the creation of the CSV and SHEF files.

8.9.8 campbell decoder

The campbell decoder decodes the data and creates a CVS (.dat) format file and a SHEF format file using the SHEFcodes.cfg configuration file. The files are put in the /data/Incoming directory. The number of data intervals collected and the time stamp for the last data interval are written back into the working configuration file to be used by the next data retrieval.

To Access and Read the campbellDecoder Log

The campbell decoder log resides on the active LS in the directory $LOG_DIR and records messages from the campbell decoder process. Perform the following commands as user ldad:

- **TYPE:** cd $LOG_DIR/<yyyymmdd>
  - Where <yyyymmdd> is today’s date.

- **TYPE:** ls -ltr campbell*
  - Displays the current day’s logs from earliest to latest.

- **TYPE:** tail -f <logname>
  - Displays the latest registered messages as the log updates.

Log Message Format

An entry at the beginning of this log looks like the following:

```bash
process_expect.c EVENT: portLockFile==> /ldad/bin/8e1out
20:24:55.303 process_expect.c EVENT: Successfully read data from BBCW3
20:33:33.492 process_expect.c PROBLEM: process_expect: expect session timed out
process_expect.c EVENT: portLockFile==> /ldad/bin/8n1out
```

The format of the log file consists of the following fields:

- Time stamp: 20:24:55.303
- File name: process_expect.c
- Message type: EVENT:
- Message string: Successfully read data from BBCW3
Chapter 8 Local Data Acquisition and Dissemination (LDAD) System

- Displays the latest registered messages as the log updates.

**Log Message Format**

An entry at the beginning of this log looks like the following:

```
LOG-STATUS: Log file opened on host lsl-tbdw at Wed Apr 12 14:37:11 2006
14:37:11.682 Campbell.C EVENT: Start decoding Campbell data for AEFW3
```

The format of the log file consists of the following fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time stamp</td>
<td>14:37:12.018</td>
</tr>
<tr>
<td>File name</td>
<td>Campbell.C</td>
</tr>
<tr>
<td>Message type</td>
<td>EVENT:</td>
</tr>
<tr>
<td>Message string</td>
<td>Successfully decoded Campbell data for AEFW3.</td>
</tr>
</tbody>
</table>

### 8.10 Decode Campbell Data

The `newLDADdataNotification` script checks the LDAD products directory for Campbell files. When a file is available, `newLDADdataNotification` passes the information to the `CO_serv` and informs the `listener`. The `listener` moves the file to a `tmp` directory, executes the `preprocessCAMPBELL.pl` script, and moves the preprocessed file (.dat) to the `Raw` directory or stores it in the text database archives, and sends it to the OH’s database (SHEF). The `listener` process calls `routerLdadDecoder` to further decode the Campbell file (.dat) for the creation of the netCDF and SHEF-encoded files. The decoded file is stored and a notification is sent to the forecaster stating that new data are available. The SHEF-encoded file is forwarded to the NCF for distribution.

**Processes**

LS | `newLDADdataNotification`
LS | `CO_serv`
PX2 | `listener`
PX2 | `preprocessCAMPBELL.pl`
PX2 | `CommsRouter LDAD_ROUTER`
PX2 | `DataController LDAD_ROUTER LdadController.config`
PX2 | `routerLdadDecoder`
PX2 | `routerStoreTextEDEX`
PX2 | `routerStoreEDEX`
PX2 | `routerShefEncoderEDEX`
PX2 | `distributeProduct`
DX1 | `notificationServer`

**Configuration Files (all in /data/fxa/LDAD/data)**

<table>
<thead>
<tr>
<th>File</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>LDADinfo.txt</td>
<td>(for listener)</td>
</tr>
<tr>
<td>*Station.txt</td>
<td>(for routerStoreEDEX)</td>
</tr>
<tr>
<td>LdadController.config</td>
<td>(for DataController LDAD_ROUTER)</td>
</tr>
</tbody>
</table>
Chapter 8 Local Data Acquisition and Dissemination (LDAD) System

LdadPatterns.txt (for routerLdadDecoder, routerStoreEDEX, routerShefEncoder)
LARC.desc (for routerLdadDecoder, routerStoreEDEX, routerShefEncoderEDEX)

Configuration Files in /awips/fxa/data
LdadWmoAwipsId.tbl (for routerStoreTextEDEX)
LDADWmoID.cfg (for routerShefEncoderEDEX)
edex-info.txt (for routerStoreEDEX, routerShefEncoderEDEX)

CAMPBELL.desc
The $LDAD_DATA/data/CAMPBELL.desc file requires no changes unless the Campbell data logger has been modified to collect data with PE codes that are not in the current file.

The file is directly coupled with the SHEFcodes.cfg file. The sequence of data group entries should be matched with the sequence of PE codes in SHEFcodes.cfg file.

Refer to Exhibit 8.10-1 for the Decode Campbell Data diagram.

8.10.1 newLDADdataNotification

The newLDADdataNotification process uses the file name and a lookup table (LDADinfo.txt) to determine the type of data contained in the file and checks the LDAD products directory (/data/Incoming) periodically for Campbell files.

The newLDADdataNotification process creates and sends an IPC message to the CO_serv process using createAndSendIPCmessage. The message notifies CO_serv that there are Campbell files in the /data/Incoming directory.

If newLDADdataNotification fails to receive confirmation from listener within 5 minutes, or if an fstat on the file to transfer fails, the product gets moved to the /data/ldad/Problem directory using moveFileToProblemDir process. Otherwise, the
Chapter 8  Local Data Acquisition and Dissemination (LDAD) System

Exhibit 8.10-1. Decode Campbell Data (for LDAD) (Page 1 of 2)
Exhibit 8.10-1. Decode Campbell Data (for LDAD) (Page 2 of 2)
“processed” files get moved to the `/data/ldad/Processed` directory using the `moveFileToProcessDir` process.

To Access and Read the `newLDADdataNotification` Log

The `newLDADdataNotification` log resides on the active LS in the `SLOG_DIR` directory and records messages from the `newLDADdataNotification` process. Perform the following commands as user ldad:

- **TYPE:**
  ```
  cd $LOG_DIR/<yyyymmdd>
  ```
  - Where `<yyyymmdd>` is today’s date

- **TYPE:**
  ```
  ls -ltr newLDADdataNotification*
  ```
  - Displays the current day’s `newLDADdataNotification` logs from earliest to latest

- **TYPE:**
  ```
  tail -f <logname>
  ```
  - Displays the latest registered messages as the log updates

Log Message Format

An entry at the beginning of this log looks like the following:

```
LOG-STATUS: Log file opened on host ls1-tbdw at Wed Apr 12 00:00:28 2006
00:00:28.906 newLDADdataNotification.c EVENT: newLDADdataNotification process started
00:00:28.911 newLDADdataNotification.c EVENT: Reading /data/Incoming/ every 20 seconds
14:36:19.861 newLDADdataNotification.c EVENT: Processing: /data/Incoming/ftp1.txt
14:36:21.053 newLDADdataNotification.c EVENT: waitForReply SUCCESS on /data/Incoming/ftp1.txt
14:37:41.210 newLDADdataNotification.c EVENT: waitForReply SUCCESS on /data/Incoming/CAMPBELLAEFPW3311437.SHEF
```

The format of the log file consists of the following fields:

- Time stamp
- File name
- Message type
- Message string

8.10.2 CO_serv

`CO_serv` initially sets up a socket (TCP/IP) and waits for connections. The notification received from `newLDADdataNotification` is passed across the firewall to the `listener` process on the internal side on the PX2.
To Access and Read the CO_serv Log

The CO_serv log resides on the active LS in the directory $LOG_DIR and records messages from the CO_serv process. Perform the following commands as user ldad:

**TYPE:**

cd $LOG_DIR/<yyyymmdd>

* Where <yyyymmdd> is today’s date.

**TYPE:**

ls -ltr CO_serv*

* Displays the current day’s CO_serv logs from earliest to latest.

**TYPE:**

tail -f <logname>

* Displays the latest registered messages as the log updates.

Log Message Format

An entry at the beginning of this log looks like the following:

LOG-STATUS: Log file opened on host ls1-rnk at Wed Apr 12 00:00:17 2006
00:00:17.076 co_serv_main.c EVENT: CO_serv process started
00:05:07.791 co_serv_main.c EVENT: CO_serv received a socket connection at: Apr12 06 00:05:07 GMT
00:05:07.801 co_serv_main.c EVENT: R message Received by CO_serv.....
00:06:36.988 co_serv_main.c EVENT: CO_serv received a socket connection at: Apr12 06 00:06:36 GMT
00:06:36.995 co_serv_main.c EVENT: R message Received by CO_serv.....
00:13:48.283 co_serv_main.c EVENT: CO_serv received a socket connection at: Apr12 06 00:13:48 GMT
00:13:48.291 co_serv_main.c EVENT: R message Received by CO_serv.....

The format of the log file consists of the following fields:

- Time stamp
- File name
- Message type
- Message string

```
Time stamp          00:00:17.076
File name           co_serv_main.c
Message type        EVENT:
Message string      CO_serv process started
```

8.10.3 listener

The listener reads the LDADinfo.txt config file, sets up a socket, binds its local address so that the Client (CO_serv) can send to it, and waits for the connection from the Client process. When the notification is received, the listener retrieves the Campbell file from the /data/Incoming directory on the active LS and copies it to the SLDAD_DATA/tmp directory on the PX2 inside the firewall. If the file requires no preprocessing, the listener moves the data file directly to the Raw directory and sends a verification when done.

The listener executes the SFXA_HOME/ldad/bin/preprocessCAMPBELL.pl script to preprocess the Campbell data files. The listener receives the fully qualified data file, the hostname used by notification routine, the port number of the process requiring
notification upon completion of the preprocessing, and other necessary information about
the data file format from CO_serv. Finally, the listener sends notification to the
CommsRouter LDAD_ROUTER that a product has been preprocessed.

To Access and Read the listener Log

The listener log resides on PX2 in the directory $LOG_DIR and records
messages from the listener process. Perform the following commands as user
ldad:

```
 TYPE: cd $LOG_DIR/<yyyymmdd>
   Where <yyyymmdd> is today’s date.

 TYPE: ls -ltr listener*
   Displays the current day’s listener logs from earliest to latest.

 TYPE: tail -f <logname>
   Displays the latest registered messages as the log updates.
```

Log Message Format

An entry at the beginning of this log looks like the following:

```
LOG-STATUS: Log file opened on host px2-tbdw at Wed April 12 00:00:12 2006
00:00:12.616 listener.c EVENT: listener process started
14:37:40.146 listener.c DEBUG: do_G_Msg: rcp
 lsl:/data/Incoming/CAMPBELLAEFW3311437.dat.973003031
/data/fxa/LDAD/tmp/CAMPBELLAEFW3311437.dat.973003031.973003060
14:37:41.059 listener.c EVENT: do_G_Msg: rcp completed for
CAMPBELLAEFW3311437.dat.973003031.973003060
14:37:41.146 listener.c DEBUG: pInfo->preprocessorScript=
<preprocessCAMPBELL.pl>
14:37:41.146 listener.c DEBUG: makePreprocessorCommandLine: rcp_cmd_buf=
"/awips/dad/bin/preprocessCAMPBELL.pl
/data/fxa/LDAD/tmp/CAMPBELLAEFW3311437.dat.973003031.973003060 px 15009>
14:37:41.147 listener.c EVENT: do_G_Msg: Preprocessor Command Line=
"/awips/dad/bin/preprocessCAMPBELL.pl
/data/fxa/LDAD/tmp/CAMPBELLAEFW3311437.dat.973003031.973003060 px 15009>
```

The format of the log file consists of the following fields:

- **Time stamp**: 14:37:40:146
- **File name**: listener.c
- **Message type**: DEBUG:
- **Message string**: do_G_Msg: rcp
- **file name**: lsl:/data/Incoming/CAMPBELLAEFW3311437.dat.973003031
  /data/fxa/LDAD/tmp/CAMPBELLAEFW3311437.dat.973003031.973003060

AWP.MAN.SMM.A2

– Hard copies uncontrolled. Verify effective date prior to use. –
8.10.4 preprocessCAMPBELL.pl

The preprocessCAMPBELL.pl script copies the files from the SLDAD_DATA/tmp directory to the /awips/dfa/ldad/ldadtmp directory. There, it reformat the Campbell data files, appends the abstime string to the file names, and stores the preprocessed file (.dat) in the SLDAD_DATA/Raw directory and the SHEF file into the Postgres text database, archives it, and sends it to the OH database.

To Access and Read the preprocessCAMPBELL Log

The preprocessCAMPBELL log resides on PX2 in the directory $LOG_DIR and records messages from the preprocessCAMPBELL.pl process. Perform the following commands as user ldad:

TYPE: cd $LOG_DIR/<yyyymmdd>
- Where <yyyymmdd> is today’s date.

TYPE: ls -ltr preprocessCAMPBELL*
- Displays the current day’s preprocessCAMPBELL logs from earliest to latest.

TYPE: tail -f <logname>
- Displays the latest registered messages as the log updates.

Log Message Format

An entry at the beginning of this log looks like the following:

<> New Activity Log Started Wed April 12 14:37:41 UTC 2006
<> ARGV[0]: /data/fxa/LDAD/tmp/CAMPBELLAEFW3311437.dat.973003031.973003060
ARGV[1]: px
ARGV[2]: 15009
moveFile(): move /data/fxa/LDAD/tmp/CAMPBELLAEFW3311437.dat.973003031.973003060 to /data/fxa/LDAD/Raw/CAMPBELLAEFW3311437.dat.973003031.973003060
moveFile(): /data/fxa/LDAD/tmp/CAMPBELLAEFW3311437.dat.973003031.973003060 moved successfully.
Failed to move Campbell CSV format file to /raw directory

<> New Activity Log Started Wed April 12 14:37:41 UTC 2006
<> ARGV[0]: /data/fxa/LDAD/tmp/CAMPBELLAEFW3311437.SHEF.973003060
ARGV[1]: px
ARGV[2]: 15009
getPE():
getAwipsId():
getAfosPil():
afos ID =
storeInTextDB():
storeCommand =
textdb -w < /data/fxa/LDAD/tmp/CAMPBELLAEFW3311437.SHEF.973003060
Failed to store product to text database.

getSiteCommissionStatus:
commission status = 1
Site is commissioned. No need to send observation to APOS

archiveObs:
moveFile(): move /data/fxa/LDAD/tmp/CAMPBELLAEFW3311437.SHEF.973003060 to /data/fxa/archive/obs/scratch/CAMPBELLAEFW3311437.SHEF.973003060.20060412143742
moveFile(): /data/fxa/LDAD/tmp/CAMPBELLAEFW3311437.SHEF.973003060 moved successfully.
Archival was successful...
Script Done ...

The format of the log file consists of the following fields:

File name
Message string /data/fxa/LDAD/tmp/CAMPBELLAEFW3311437.SHEF.973003060 moved successfully

8.10.5 LDAD_ROUTER

The LDAD_ROUTER (registered LDAD CommsRouter) is responsible for transporting data notifications from the data acquisition processes to the DataController processes. The main function of the LDAD_ROUTER is message passing; it does no data processing. The router provides a known location where the data acquisition processes can send data messages.

▶ To Access and Read the CommsRouter Log

The LDAD_ROUTER CommsRouter log resides on PX2 in a time-stamped directory under $LOG_DIR. Perform the following commands as user ldad:

TYPE: cd $LOG_DIR/<yyyymmdd>
• Where <yyyymmdd> is today’s date.

TYPE: ps -wef |grep ldad
• Displays the ldad processes. Notice the PID of the CommsRouter process that is owned by user ldad.

TYPE: ls -ltr Comms*
• Displays the latest logs.

NOTE: The CommsRouter log containing the PID of the LDAD_ROUTER process is the LDAD_ROUTER CommsRouter log.

TYPE: tail -f <logname>
• Displays the log as it updates.
Log Message Format

An entry in the log should look like the following:

```
LOG-STATUS: Log file opened on host px2-tbdw at Wed Apr 12 00:00:09 2006
00:00:09.558 CommsReceiver.C EVENT: NCF_ENTRY CRMsg Pattern:
^[0-9.]*[^m]([^[s]([^[a]([^[s]([^[_]([^[q]([^[c]))]))]))
14:37:42.639 CommsReceiver.C EVENT: NCF_ENTRY DSMsg:
CAMPBELLAEFW3311437.dat.973003031.973003060 1 45
14:37:42.657 CommsReceiver.C EVENT: NCF_ENTRY DSMsg:
CAMPBELLAEFW3311437.SHEF.973003060 1 36
14:37:44.912 CommsReceiver.C EVENT: NCF_ENTRY DSMsg:
973003060.CAMPBELLAEFW3311437.decoded 1 62
```

The log consists of the following fields:

- **Time stamp**: 14:37:44.912
- **File name**: CommsReceiver.C
- **Message type**: EVENT:
- **Message string**: NCF_ENTRY DSMsg:
  973003060.CAMPBELLAEFW3311437.decoded 1 62

The **LDAD_ROUTER CommsRouter** is started automatically at system reboot time or it can be started manually. The **CommsRouter** runs continuously, exiting only when killed by the restart program.

### 8.10.6 DataController

The DataController is responsible for creating the decoder and storage processes (as child processes), restarting them if they exit, routing appropriate messages to them, and queuing incoming data messages until the decoder and storage processes are ready. As data messages are stored in the queue, the size of the processes grows.

The DataController’s main program creates a DataRouting manager object when it first starts up and invokes a member function to start the routerLdadDecoder, routerStoreNetcdf, and routerShefEncoder processes by reading and executing the LdadController.config file. The file resides on the $LDAD_INTERNAL_DATA directory and contains the names of the above processes to start/restart, a restart flag, and end of file (eof) designators to alert the Data Routing Manager that there are no more processes listed in the file.

The LDAD_ROUTER sends product-received notifications to the DataController. When the processes first start up, they send messages containing their IPC addresses and process numbers to the DataController. The processes send the DataController a data request message and a ready-to-receive data message. In response, the DataController stores the processes’ ID information and data requests, and sends a message to the requesting processes (if a message is available). The DataController waits to receive another ready message from the processes before sending any more data messages. The DataController queues any messages it receives from the LDAD_ROUTER until the processes send another ready message.
To Access and Read the DataController Log

The DataController log resides on PX2 in a time-stamped directory under SLOG_DIR. Perform the following commands as user ldad:

- **TYPE:**
  
  cd $LOG_DIR/<yyyymmdd>
  
  Where <yyyymmdd> is today’s date.

- **TYPE:**
  
  ls -ltr DataController*
  
  Displays the latest log for the DataController LDAD_ROUTER process.

- **TYPE:**
  
  tail -f <logname>
  
  Displays the current log as it updates with the latest messages.

Log Message Format

An entry in the log should look like the following:

```
LOG-STATUS: Log file opened on host px2-tbdw at Tue Apr 24 00:00:04 2012
00:00:04.414 DataCaptReceiver.C EVENT: DataReqMsg: pattern = (.\.[0-9]*o|\.[0-9])*\[0-9]+)\[0-9]*.msas_qc\.decoded
00:00:05.430 DataCaptReceiver.C EVENT: DataReqMsg: pattern = ^[0-9]*[m]\[[^s]\[[^a]\][^_][^q]\[c]\]).*\.decoded$
00:00:05.430 getCommsData.C EVENT: Data Request message sent to LDAD_ROUTER.
00:00:06.448 DataCaptReceiver.C EVENT: DataReqMsg: pattern = .*Text/.*[0-9]*\n00:00:06.448 getCommsData.C EVENT: Data Request message sent to LDAD_ROUTER.
00:00:07.465 DataCaptReceiver.C EVENT: DataReqMsg: pattern = .\[not_a_mesonet|alert_wl.*|alert_r5.*|..LARC.*].decoded
```

The log file consists of the following fields:

- **Time stamp**
  
  00:00:05.430

- **File name**
  
  DataCaptReceiver.C

- **Message type**
  
  EVENT:

- **Message string**
  
  EVENT: DataReqMsg: pattern = ^[0-9]*[m]\[[^s]\[[^a]\][^_][^q]\[c]\)).*\.decoded$

8.10.7 routerLdadDecoder

The routerLdadDecoder process is spawned by the DataController as specified in the LdadController.config file. The routerLdadDecoder process sends a data request to the DataController for the CAMPBELL products it wishes to receive by specifying product header patterns of interest. On receipt of a notification from the DataController that a product of interest has arrived, the routerLdadDecoder reads the directory containing the raw files, SLDAD_DATA/Raw.
The \texttt{routerLdadDecoder} process reads each file and stores them via serialized data files to the $\texttt{LDAD\_DECODED\_DATA}$ directory. The serialized file name is sent back through the \texttt{LDAD\_ROUTER} to notify registered LDAD storage processes of the arrival of each LDAD decoded product.

- **To Access and Read the routerLdadDecoder Log**

  The \texttt{routerLdadDecoder} log resides on PX2 in a time-stamped directory under \texttt{SLOG\_DIR}. Perform the following commands as user ldad:

  TYPE: \texttt{cd $LOG\_DIR/\langle yyyymmdd\rangle}
  \hspace*{1em} Where $\langle yyyymmdd\rangle$ is today’s date.

  TYPE: \texttt{ls -ltr routerLdadDecoder*}
  \hspace*{1em} Displays the latest log for the \texttt{routerLdadDecoder} process.

  TYPE: \texttt{tail -f <logname>}
  \hspace*{1em} Displays the current log as it updates with the latest messages.

**Log Message Format**

An entry in the log should look like the following:

\begin{verbatim}
LOG-STATUS: Log file opened on host px2-tbdw at Wed Apr 12 00:00:33 2006
14:33:04.580 DataCaptReceiver.C USE: ProdLen 50
14:33:04.601 DataCaptReceiver.C USE: ProdLen 58
14:33:06.556 DataCaptReceiver.C USE: ProdLen 31
14:36:22.029 DataCaptReceiver.C USE: ProdLen 18
14:37:42.662 DataCaptReceiver.C USE: ProdLen 45
14:37:44.912 LdadRoutines.C EVENT: LdadRoutines::pingLdadRouter(), 'LDAD DECODED DATA/ 973003060.CAMPBELLAEFW3311437.decoded', sent to LDAD_ROUTER
\end{verbatim}

The log file consists of the following fields:

| Time stamp | 14:37:44.912 |
| File name  | LdadRoutines.C |
| Message type | EVENT: |
| Message string | LdadRoutines::pingLdadRouter(), 'LDAD DECODED DATA/ 973003060.CAMPBELLAEFW3311437.decoded', sent to LDAD_ROUTER |

\section*{8.10.8 routerStoreEDEX}

The \texttt{routerStoreEDEX} process is spawned by the \texttt{DataController} as specified in the \texttt{LdadController.config} file. The \texttt{routerStoreEDEX} process sends a data request to the
**DataController** for serialized decoded Campbell files in netCDF format. When notified by the **DataController** that a product of interest has arrived, **routerStoreNetcdf** reads the directory containing the files, **SLDAD_DECODED_DATA**. The **routerStoreEDEX** process stores the files in the `/data_store/ldad` directory in the XML format and sends a notification to the **EDEX**.

### To Access and Read the **routerStoreEDEX** Log

The **routerStoreEDEX** log resides on PX2 in a time-stamped directory under **SLOG_DIR**. Perform the following commands as user ldad:

**Type:**
- `cd $LOG_DIR/<yyyymmdd>`
  - Where `<yyyymmdd>` is today’s date.

**Type:**
- `ls -ltr routerStoreEDEX*`
  - Displays the latest logs for the **routerStoreEDEX** process.

**Type:**
- `tail -f <logname>`
  - Displays the log as it updates with the latest messages.

### Log Message Format

An entry in the log should look like the following:

```
01:40:22.803 routerStoreEDEX.C EVENT: 1335231622.alert_r5.decoded processing completed
01:45:22.712 routerStoreEDEX.C EVENT: NCF_ENTRY /data/fxa/LDAD/decoded/1335231922.alert_r5.decoded
01:45:22.733 routerStoreEDEX.C EVENT: NCF_STOREd to /data_store/ldad/LDAD.hydro.1335231922.alert_r5.decoded.xml
01:45:22.733 routerStoreEDEX.C EVENT: 1335231922.alert_r5.decoded processing completed
01:45:22.773 routerStoreEDEX.C EVENT: NCF_ENTRY /data/fxa/LDAD/decoded/1335231922.RAWS.decoded
01:45:22.773 routerStoreEDEX.C EVENT: NCF_STOREd to /data_store/ldad/LDAD.mesonet.1335231922.RAWS.decoded.xml
01:45:22.774 routerStoreEDEX.C EVENT: 1335231922.RAWS.decoded processing completed
01:50:22.774 routerStoreEDEX.C EVENT: NCF_ENTRY /data/fxa/LDAD/decoded/1335232222.alert_r5.decoded
01:50:22.790 routerStoreEDEX.C EVENT: NCF_STOREd to /data_store/ldad/LDAD.hydro.1335232222.alert_r5.decoded.xml
01:50:22.790 routerStoreEDEX.C EVENT: 1335232222.alert_r5.decoded processing completed
```

### 8.10.9 **routerShefEncoderEDEX**

The **routerShefEncoderEDEX** process is spawned by the **DataController** as specified in the **LdadController.config** file. The **routerShefEncoderEDEX** process sends a data request to the **DataController** for serialized decoded files. On receipt of a notification from the **DataController** that a product of interest has arrived, the
routerShefEncoderEDEX reads the directory containing the files, $LDAD\_DECODED\_DATA$. The files there are SHEF-encoded and moved to the /data_store/ldad directory. The routerShefEncoderEDEX process sends a notification to the EDEX. Once the SHEF-encoded data file is stored in the /data_store/ldad directory, the routerShefEncoderEDEX saves a copy of the file in the /tmp directory and calls the distributeProduct process to send the file to the NCF over the WAN.

To Access and Read the routerShefEncoderEDEX Log

The routerShefEncoderEDEX log resides on PX2 in a time-stamped directory under $SLOG\_DIR$. Perform the following commands as user ldad:

**TYPE:**
cd $LOG\_DIR/\langle yyyymmdd\rangle

- Where \langle yyyymmdd\rangle is today’s date.

**TYPE:**
ls -rtal routerShefEncoderEDEX*

- Displays the latest logs for the routerShefEncoderEDEX process.

**TYPE:**
tail -f <logname>

- Displays the current log as it is updated with the latest messages.

Log Message Format

An entry in the log should look like the following:

```plaintext
01:50:22.775 LdadStorage.C EVENT: LdadStorage::separatePatterns () <[-999 ]> from <-999>
01:50:22.776 ShefStorage.C EVENT: TZ= GMT
01:50:22.794 routerShefEncoderEDEX.C EVENT: NCF\_STOREd to /data_store/ldad/SXUS44 KWOHshef.1335232222
01:50:22.795 checkWmoHeader.C EVENT: checkWmoHeader()
01:50:22.795 SHEF\_format.C EVENT: SHEF\_format::sendToProductWan -> data contains a WMO header.
    Creating a dummy file (zero bytes).
01:50:22.817 SHEF\_format.C EVENT: SHEF\_format::sendToProductWan ->
    system(distributeProduct -a DEFAULTNCF -e /tmp/shefwanECAx2f KSTORRZSTO /tmp/shefwanECAx2f\_msg);
01:50:22.817 SHEF\_format.C EVENT: shef data has sent to WAN successfully!
01:50:22.817 routerShefEncoderEDEX.C EVENT: Finished
```
8.10.10 distributeProduct

The **distributeProduct** process creates a product message and submits it for distribution across the AWIPS WAN to the addressed sites. The **distributeProduct** process first prepares the product by creating the WAN communications header and prepending the header to a temporary copy of the product. Distribution requests, enclosing the temporary copy of the product, are subsequently made through the **msg_send** utility program.

To Access and Read the distributeProduct Log

The **distributeProduct** log resides on PX2 in a time-stamped directory under $LOG_DIR$. Perform the following commands as user ldad:

**TYPE:**

```
cd $LOG_DIR/<yyyymmdd>
```

Where `<yyyymmdd>` is today’s date.

**TYPE:**

```
ls -ltr distributeProduct*
```

Displays the latest logs for the **distributeProduct** process.

**TYPE:**

```
tail -f <logname>
```

View the current log as it updates with the latest messages.

Log Message Format

An entry in the log should look like the following:

```
LOG-STATUS: Log file opened on host px2-tbdw at Wed Apr 12 19:30:59 2006
19:30:59.690 MHSProduct.C PROBLEM: Product (/data/fxa/ispan/hydro/temp/A_format.973020657) is empty.
19:30:59.853 MHSProduct.C EVENT: Sending attachments (/data/fxa/ispan/hydro/temp/A_format.973020657) only. Attachments are assumed to contain correct communications header.
19:31:02.504 MHSProduct.C EVENT: Product request (KSTORRZLDA) was successfully submitted to the MHS request server for distribution.
```

Message Attributes:

```
Request ID: TBDW-25380
To: DEFAULTNCF
Priority: 0
Type: Routine
Subject: 
Handling Action: 0
```

```
19:31:02.518 MHSProduct.C EVENT: TBDW-25380 ~
```

The log file consists of the following fields:

- **Time stamp**: 19:31:02.504
- **File name**: MHSProduct.C
- **Message type**: EVENT:
- **Message string**: Product request (KSTORRZLDA) was successfully submitted to the MHS request server for distribution.
8.10.11 notificationServer

The notificationServer provides information to its display clients that a new version of a display product has become available. Data acquisition processes send data notification messages to the notificationServer, which acts as a bridge between the data acquisition and workstation display components of AWIPS.

To Access and Read the notificationServer Log

The notificationServer log resides on the DX1 in the directory $LOG_DIR. This log records messages from the notificationServer process. Perform the following commands as user fxa:

**TYPE:**
cd $LOG_DIR/<yyyymmdd>
- Where <yyyymmdd> is today’s date.

**TYPE:**
ls -ltr notificationServer*
- Displays the current day’s notificationServer logs from earliest to latest.

**TYPE:**
tail -f <logname>
- Displays the latest registered messages as the log updates.

Log Message Format

An entry at the beginning of the log looks like the following:

LOG-STATUS: Log file opened on host dx1-rnk at Wed Apr 12 00:00:31 2006
00:00:31.822 NotificationServer.C DIAG: Tried to generate notifications for key: 3700 but the data notify time: 25.23 3HR does not match any times on the inventory.
00:00:35.948 GridAccessor1.C EVENT: Updating inventory info for MRF

------------------------------------------------------------------------------

Process ID of anonymous target: 29574
Host: lx2-rnk
Port number: 3231
NORMAL PRIORITY CONNECTION
Connection created: Apr 12 06 16:25:32 GMT. Initiated by connecting process.
Last time this connection was used: Apr 12 06 23:55:49 GMT
Sent messages since last stat: 4
Bytes per message (Avg/Min/Max): 544/540/548
No messages have been received since last stat report.

The format of the log file consists of the following fields:

- **Time stamp**: 00:00:31.822
- **File name**: NotificationServer.C
- **Message type**: DIAG:
- **Message string**: Tried to generate notifications for key: 3700
  but the data notify time: 25.23 3 hour does not
  match any times on the inventory.

### 8.11 Acquire Sutron Data

Data collection from Sutron gauges is carried out by automating a normally interactive
session using the **cu** communication protocol. There is a Server program (**CO_serv**) and
a Client program (**tell_co**). The Server program is always running, waiting for, and
processing requests to collect data from a particular gauge from the Client program. The
Client program can be invoked from a shell script, as a cron job, or directly from a
command line. A feature of the Client program is that it retrieves all necessary
information regarding collection and storage of data from system tables in the AWIPS
database. The Client and Server programs use sockets to communicate across a TCP/IP
LAN. Data from a gauge are SHEF-encoded prior to storage on AWIPS. The .dat file is
moved into the **Raw** directory, while the SHEF file is stored into the text database,
archived, and sent to the OH’s database during preprocessing. The .dat file is forwarded
to the **ldadDecoder** process to further decode the Sutron data for creation of the netCDF
and SHEF-encoded files.

**Processes**

- LS callSite.ksh
- LS process_expect
- LS **CO_serv**
- LS sutron
- PX2 listener
- PX2 **tell_co**
- PX2 ldadServer
- LX ldadScheduler
- DX1 notificationServer

**Configuration Files (all in /data/fxa/LDAD/data)**

- LDADinfo.txt (for **listener**)
- SUTRONStation.txt (for **tell_co**, ldadScheduler)
- userDescriptions (for **ldadScheduler**)
- usrpro (for **ldadScheduler**)
- gageinfo (for **ldadScheduler**)
- gageinfoSutron.txt (for **ldadScheduler**)
- gaugeDescriptions (for **ldadScheduler**)
- SHEFcodes.cfg (for **ldadScheduler**)

---

AWP.MAN.SMM.A2  8-72

– Hard copies uncontrolled. Verify effective date prior to use. –
LDADinfo.txt

The file should contain the following entry:

SUTRON | SUTRON | 89 | 90 | CVS_TYPE | hydro | preprocessSUTRON.pl |

Once the file is installed, this line should not be modified.

SUTRONStation.txt

This file must be updated whenever a gauge is added to or dropped from the current Sutron datalogger list. To add a new Sutron gauge, find out the following gauge information to create a new entry in the station table format (* indicates a mandatory field):

- **Provider ID**: Data provider station or gauge ID (*)
- **AFOS(HB5) ID**: The AFOS or Handbook 5 ID (NWS assigned)
- **Station Name**: Text name of station
- **Elevation**: Station elevation (m) (*)
- **Latitude**: Station Latitude (decimal degrees, north positive) (*)
- **Longitude**: Station Longitude (decimal degrees, east positive) (*)
- **Local Time Zone**: Reporting time zone (e.g., UTC, PST8PDT) (*). See Note.
- **Location Description**: Station location/address (Text)
- **Station Type**: Station type (e.g., tower, surface, floating platform, etc.)
- **Number of Instruments**: Number of reporting instruments for the station
- **Number of Reporting Levels**: Number of data reporting levels. Level information should be provided in the instrument table
- **Maintenance/Calibration Schedule**: Frequency of maintenance/calibration
- **Site Description**: Text description of the site surroundings

The entries are sorted alphabetically by the gauge ID.

This file must be identical on the PX2 and on LS1!

**NOTE**: The local time zone for Sutron gauges is always GMT since the Sutron Decoder already converts the observation times into GMT times.
userDescriptions

This file is used by the ldadScheduler process to get the descriptions of users. When read in, blank lines and lines beginning with # are skipped. All other lines should have the following format:

$user ID$|$user description$

A blank user description is denoted by having the line end immediately following the $|$ character (a vertical bar or “pipe”).

This file should not be modified manually. It should only be written to by the ldadScheduler (specifically, the LdadSchedulerServer.C routine).

usrpro

The user file (usrpro) is made up of zero to n lines, each line holding the following information (note that the lines are so long that they are wrapped in the interests of readability). Columns containing “??” are not used at this time.

$<ID>|<protocol>|<IP address>|<phone number>|<login>|<password (encrypted)>|??|<file to receive>|<file to send>|??|??|<data collection session file>|<data dissemination session file>|<local directory>|<remote directory>|??... (columns 16 through 36 have unknown meanings)...|<IP address>$

- The $<protocol>$ is either “F” for FTP, “X” for Xmodem, “Y” for Ymodem, “Z” for Zmodem, or “K” for Kermit.
- The $<IP address>$ designates the host to FTP to, if the protocol is “F,” or the terminal server otherwise. $<IP address>$ appears in two places in the above column listing: columns 3 and 37. In reality, only one of the two columns is used. For the FTP protocol, only column 3 has the address in it, and column 37 is left blank. If any other protocol is being used, column 37 has the address in it, and column 3 is left blank.
- The $<phone number>$ field is unused if the protocol is “F.”
- The $<file to receive>$ and $<file to send>$ fields are mutually exclusive. Only one may be specified. The $<file to receive>$ field indicates that the user is a collection user, and the $<file to send>$ indicates that he user is a dissemination user. All other columns are tracked when read in; when written out, the same values are placed in them as were originally found.

gageinfo

The gageinfo file contains a list of Handbook 5 and Sutron gauges. It is updated periodically as Sutron gauges are added and removed with the ldadScheduler. Whenever a gauge is added or dropped from the current list, it should be updated here and in the following files:

$LDAD_DATA/data/gageinfoSutron.txt
$LDAD_DATA/data/gaugeDescriptions
The following fields must be present for each gauge (see Note):

- **Gauge ID**: A five-character instrument ID.
- **Protocol Type**: C for data collection, D for dissemination.
- **Phone#**: The dial-out phone number can be in any format. However, if the gauge has the voice message first, several pauses are included in the phone number. Depending on the length of the voice message, add more pauses if the one listed here is not enough. The pause is entered by comma (,) and is ended with a *9. If a phone number contains a pause extension, make sure the phone number is enclosed by single quotes (‘). For example, ‘1-605.886.2473,,,,,*9’.
- **Session File**: Use SUTRON_coll_sess exactly.
- **SHEF File**: 1 = Yes; 0 = No.
- **Storage Type**: S.
- **Baud Rate**: Use 9600 for Sutron gauges.
- **Terminal Server (tty)**: Use 8n1out for Sutron gauges.

The fields are separated by vertical bars (|); empty fields are for future use. A number nine (9) in front the phone number allows the user to dial out. An example follows:

```
ADAM4|C| |916166763810| | | | |SUTRON_coll_sess|1|S|||9600|e| |8n1out|
```

**NOTE**: The rcvprog field indicates the receiving program (e.g., FTP) for data collection. This field is used as the custom gauge ID if the gauge is not owned by the NWS.

gageinfoSutron.txt

The file should be updated along with the gageinfo file when adding/dropping Sutron gauge(s). There are at least four fields for each Sutron gauge to be entered when adding a gauge; the fields are as follows:

- **Gauge ID**: A five-character instrument ID (i.e., ADAM4).
- **Local Time Zone**: Time zone where the Sutron gauges are located (e.g., PST, CDT, GMT, etc.)
- **Data Array ID for Storage Area 1**: Four digit array ID.
- **Observation Field Skip Bit Map for Array ID Number One**: This is an eight-bit field used to mark the observation fields to be ignored.
- **Bit Map for Observation data Fields in PE Codes for Storage Area 1 in Hexadecimal Format**: This is a 32-bit code (see Note). The leftmost byte is designed to ignore collected data fields and the other three are used to map the type of data collected relevant to the PE codes listed in the SHEFcodes.cfg file.
- **Data Array ID for Storage Area 1**: Four digits array ID for gauge collection data and storing in Storage Area 2.

- **Bit Map for Collecting Data Fields in PE Codes for Storage Area 2**: Same format as Storage Area 1 if the gauge collects data and stores it in Storage Area 2.

**NOTE:** Current design can accommodate a maximum of eight data fields only for a Sutron gauge and a maximum of 32 PE codes collected by all Sutron gauges. If an individual gauge collects more than eight different data fields or if the total number of the PE codes collected by the Sutron gauges exceeds 32, these fields will have to be expanded to incorporate the changes.

### gaugeDescriptions

The file is used to get the site descriptions and should be updated using the ldadScheduler when a new Sutron gauge is added. Each entry (line) should have the following format:

```
<gauge ID> | <site description>
```

For example, an entry for the ARTN6 would be as follows:

```
ARTN6 | Watertown, NY
```

The file is sorted by state abbreviations. However, it is displayed based on the alphabetical order of the gauge IDs.

### SHEFcodes.cfg

The file is designed specifically for Sutron gauges to enable creation of SHEF-encoded files (.shef). It contains the PE codes currently collected by Sutron CR10X and 8210/8200A gauges. The elements in the file are presented in alphabetical order and should be updated every time a new PE code is added to the file. When this file is modified, the LDAD system has to be shut down and restarted again.

The codes are used in constructing the 4-byte PE code bit map in the gageinfoSutron.txt file. It is also used as the base for creating and updating the SLDAD_DATA/data/SUTRON.desc file.

Refer to Exhibit 8.11-1 for the Acquire Sutron Data data flow diagram.
Exhibit 8.11-1. Acquire Sutron Data (for LDAD)
8.11.1 User Interface: ldadScheduler

The ldadScheduler’s function is to perform certain data acquisition/dissemination events on an automatic basis at specified time intervals (in the background). Access to the Scheduler’s functions is via a GUI invoked by selecting Collection/Dissemination... from the Surface pull-down menu.

The ldadScheduler provides a window with which the forecaster may create, view, and manipulate scheduled LDAD requests. The Sutron data collection can be scheduled for daily, multiple, or one time request. The Scheduler allows the forecaster to create a new request and add it to the request list. The request is submitted to the ldadServer, which uses sendLDADnotification.pl to create the request. If it is a one-time request, the forecaster can monitor its progress using a status dialog. If any errors are encountered during the data logging, an error message is formulated and sent to the forecaster workstation for display. In addition, Sutron gauge dialing status information is sent by the listener and displayed by the ldadScheduler.

The ldadScheduler registers with the ldadServer providing a callback routine for update notifications. The session file contents are registered first, and the status messages and ID updates so that when registering with ldadServer, the ID lists sent by ldadServer will be captured and kept.

It also registers with the FAC for notifications of changes to the existing requests file, and creates a FAC lock for the file to be used for locking it for “reads” and “writes.” The existing requests get read so as to fill the tree view with the requests that have been created.

8.11.2 ldadServer

The ldadServer allows multiple clients to connect to it and submit requests with intervals and durations that are honored according to their intervals until they expire.

New client initialization is performed by sending the gauge and user lists to the newly registered client. The ldadserver process creates a new message with the current active list that is broadcast all registered clients.

To honor the specified request, the ldadServer creates the command to be executed and sends the data collection request to the tell_co process.

The ldadServer receives LDAD messages from clients; messages to create and delete gauges and users; and messages requesting the contents of a session file, or specifying a session file’s contents.
To Access and Read the ldadServer Log

The ldadServer log resides on the PX2 in the $LOG_DIR directory and records messages from the ldadServer process. Perform the following commands as user ldad:

```
TYPE: cd $LOG_DIR/<yyyymmdd>
    Where <yyyymmdd> is today’s date.
```

```
TYPE: ls -ltr ldadServer*
    Displays the current day’s ldadServer logs from earliest to latest.
```

```
TYPE: tail -f <logname>
    Displays the latest registered messages as the log update.
```

Log Message Format

An entry at the beginning of the log looks like the following:

```
LOG-STATUS: Log file opened on host px2-tbdw at Wed Apr 12 20:11:03 2006
16:11:37.345 LdadSchedulerServer.C PROBLEM: The gauge "ADAM4" specified in gageinfoSutron.txt was not found in gageinfo.txt. This gauge will not be used.
16:11:37.345 LdadSchedulerServer.C PROBLEM: The gauge "ADAM4" specified in gageinfoSutron.txt was not found in gageinfo.txt. This gauge will not be used.
20:11:03.470 Connection.C EVENT: WARNING: this process wants to use target LDAD_SCHEDULE_SERVER which is assigned to run on px2-tbdw but the current host is px2-tbdw
20:11:03.492 LdadSchedulerServer.C PROBLEM: The gauge "ADAM4" specified in gageinfoSutron.txt was not found in gageinfo.txt. This gauge will not be used.
20:12:12.926 SignalClient.C EVENT: Signal 15 (Terminate) received for clean shutdown
```

The format of the log consists of the following fields:

- **Time stamp**: 16:11:37.345
- **File name**: LdadSchedulerServer.C
- **Message type**: PROBLEM:
- **Message string**: The gauge “ADAM4” specified in gageinfoSutron.txt was not found in gageinfo.txt. This gauge will not be used.

8.11.3 **tell_co**

The ldadServer process invokes the Client program tell_co, the main controller process providing the Sutron ID. The tell_co function retrieves all information regarding a specified gauge from the LDAD configuration files. This includes all communication details, the name of the session file to use, as well as the location to store the data on AWIPS and whether SHEF encoding is required. It formats an interprocess message containing the aforementioned information, opens a socket to the listener and sends a get Sutron Message “IPC” message to the listener.
To Access and Read the tell_co Log

The tell_co log resides on the PX2 in the $LOG_DIR directory and records messages from the tell_co process. Perform the following commands as user ldad:

TYPE: cd $LOG_DIR/<yyyymmdd>
   - Where <yyyymmdd> is today’s date.

TYPE: ls -ltr tell_co*
   - Displays the current day’s tell_co logs from earliest to latest.

TYPE: tail -f <logname>
   - Displays the latest registered messages as the log updates.

Log Message Format

An entry at the beginning of the log looks like the following:

LOG-STATUS: Log file opened on host px2-tbdw at Wed Apr 12 19:45:55 2006
19:45:55.368 doAlivenessTest.c EVENT: waitForReply INTERNAL  SUCCESS!!!...
19:45:56.253 doAlivenessTest.c EVENT: waitForReply EXTERNAL  SUCCESS!!!...
19:45:57.205 tell_co.c EVENT: TELL_CO waitForReply SUCCESS!!!...

The format of the log consists of the following fields:

Time stamp 19:45:57.205
File name tell_co.c
Message type EVENT:
Message string TELL_CO waitForReply SUCCESS!!!...

8.11.4 listener

The listener sets up a socket connection, binds to a local address so that the Client (CO_serv) or tell_co can send to it, and waits for the connection. Upon receipt of data collection request from tell_co, the listener forwards information received to the CO_serv process. If any errors were encountered during the data logging, an error message is formulated and passed to the Scheduler for display. In addition, Sutron gauge dialing status information is sent by the listener and displayed by the Scheduler.

To Access and Read the listener Log

The listener log resides on PX2 in the $LOG_DIR directory and records messages from the listener process. Perform the following commands as user ldad:

TYPE: cd $LOG_DIR/<yyyymmdd>
   - Where <yyyymmdd> is today’s date.

TYPE: ls -ltr listener*
Displays the current day’s listener logs from earliest to latest.

**TYPE:** `tail -f <logname>`

Displays the latest registered messages as the log updates.

### Log Message Format

An entry at the beginning of this log looks like the following:

18:24:10.419 listener.c EVENT: child listener: rcp completed for SUTRON_coll_sess
18:24:10.422 listener.c EVENT: rcp cmd buf=/awips/ldad/bin/sendLDADnotification.pl V NOTHING px2 1874
18:24:10.596 listener.c PROBLEM: child listener: failed to sendLDADNotification for SUTRON_coll_sess

The format of the log consists of the following fields:

<table>
<thead>
<tr>
<th>Time stamp</th>
<th>18:24:10:419</th>
</tr>
</thead>
<tbody>
<tr>
<td>File name</td>
<td>listener.c</td>
</tr>
<tr>
<td>Message type</td>
<td>EVENT:</td>
</tr>
<tr>
<td>Message string</td>
<td>child listener: rcp completed for SUTRON_coll_sess</td>
</tr>
</tbody>
</table>

### 8.11.5 CO_serv

The Server program (CO_serv) opens up a socket and continually listens for connections from the Client program. When one arrives, a child process is spawned that reads the interprocess message, extracts the name of the “session” file (session file located in `/ldad/data/<id_coll_sess>`) for the gauge in question (Sutron), and parses this session file to create two temporary files: a response file and a script file. CO_serv first calculates the starting point of the data to be retrieved and how many data intervals to retrieve from the working configuration file. Using the `process_expect` executable, CO_serv retrieves the data from the Sutron gauge and stores it in a temporary file. The data are sent to the sutron decoder for decoding. If any errors were encountered during the data logging, the error messages are formulated and sent to the user’s workstation for display.

#### To Access and Read the CO_serv Log

The CO_serv log resides on active LS in the directory `$LOG_DIR` and records messages from the CO_serv process. Perform the following commands as user ldad:

**TYPE:** `cd $LOG_DIR/<yyyymmdd>`

Where `<yyyymmdd>` is today’s date.

**TYPE:** `ls -ltr CO_serv*`

Displays the current day’s CO_serv log from earliest to latest.
TYPE:     tail -f <logname>
  • Displays the latest registered messages as the log updates.

Log Message Format

An entry at the beginning of this log looks like the following:

LOG-STATUS: Log file opened on host ls1-rnk at Wed Apr 12 00:00:17 2006
00:00:17.076 co_serv_main.c EVENT: CO_serv process started
00:05:07.791 co_serv_main.c EVENT: COServ received a socket connection at: Apr12 06 00:05:07 GMT
00:05:07.801 co_serv_main.c EVENT: R message Received by CO_serv......
00:06:36.988 co_serv_main.c EVENT: COServ received a socket connection at: Apr12 06 00:06:36 GMT
00:06:36.995 co_serv_main.c EVENT: R message Received by CO_serv......
00:13:48.283 co_serv_main.c EVENT: COServ received a socket connection at: Apr12 06 00:13:48 GMT
00:13:48.291 co_serv_main.c EVENT: R message Received by CO_serv......

The format of the log file consists of the following fields:

  Time stamp  00:13:48.283
  File name  co_serv_main.c
  Message type  EVENT:
  Message string  COServ received a socket connection at: Apr12 06 00:13:48 GMT

To Access and Read the co.log Log

The co.log log resides on the active LS in the $LOG_DIR directory and records messages from the CO_serv process executing the sendLDADnotification.pl script. Perform the following commands as user ldad:

TYPE:     cd $LOG_DIR/<yyyymmdd>
  • Where <yyyymmdd> is today’s date.

TYPE:     ls -ltr co.log*
  • Displays the current day’s co.log logs from earliest to latest.

TYPE:     tail -f <logname>
  • Displays the latest registered messages as the log updates.

Log Message Format

An entry at the beginning of this log looks like the following:

/awips/ldad/bin/sendLDADnotification.pl:94, socket ok
/awips/ldad/bin/sendLDADnotification.pl:104, bind ok
/awips/ldad/bin/sendLDADnotification.pl:117, connect ok
/awips/ldad/bin/sendLDADnotification.pl:127, Begin sendLDADnotification.pl
/awips/ldad/bin/sendLDADnotification.pl:38, S, SURRON coll sess 4986 px, px, 15008
/awips/ldad/bin/sendLDADnotification.pl:60, cpType = S, msgString = SURRON_coll_sess 4986 px, host= px, port = 15008
8.11.6 process_expect

The process_expect executable processes the request for a data transfer. It parses the receive or send session file and invokes do_expect to carry out the transfer by using the script file created by parse_session_file function as input. It stores the data in the appropriate place and deletes any temporary files used during this session.

To Access and Read the CO_serv Log

process_expect writes its messages into the CO_serv log. The log resides on the LS in the directory $LOG_DIR. Perform the following commands as user ldad:

 TYPE: cd $LOG_DIR/<yyyymmdd>
• Where <yyyymmdd> is today’s date.

 TYPE: ls -ltr CO_serv*
• Displays the current day’s logs from earliest to latest.

 TYPE: tail -f <logname>
• Displays the latest registered messages as the log updates.

Log Message Format

An entry at the beginning of this log looks like the following:

process_expect.c EVENT: portLockFile==> /ldad/bin/8e1out
20:24:55.303 process_expect.c EVENT: Successfully read data from ADAM4
20:33:33.492 process_expect.c PROBLEM: process_expect: expect session timed out
process_expect.c EVENT: portLockFile==> /ldad/bin/8n1out

The format of the log file consists of the following fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time stamp</td>
<td>20:24:55.303</td>
</tr>
<tr>
<td>File name</td>
<td>process_expect.c</td>
</tr>
<tr>
<td>Message type</td>
<td>EVENT:</td>
</tr>
<tr>
<td>Message string</td>
<td>Successfully read data from ADAM4</td>
</tr>
</tbody>
</table>

8.11.7 callSite.ksh

The callSite.ksh script interrogates the datalogger and passes the raw data received from the gauge to the sutron decoder for the creation of the CSV and SHEF files.

8.11.8 sutron decoder

The sutron decoder decodes the data and creates a CSV (.dat) format file and a SHEF format file using the SHEFcodes.cfg configuration file. The files are put in the /data/Incoming directory. The number of data intervals collected and the time stamp for the last data interval is written back into the working configuration file to be used by the next data retrieval.
To Access and Read the sutron Log

The **sutron** log resides on the active LS in the directory **SLOG_DIR** and records messages from the **sutron** decoder process. Perform the following commands as user ldad:

**TYPE:**

```
cd $LOG_DIR/<yyyymmdd>
```

- Where `<yyyymmdd>` is today’s date.

**TYPE:**

```
ls -ltr sutron*
```

- Displays the current day’s logs from earliest to latest.

**TYPE:**

```
tail -f <logname>
```

- Displays the latest registered messages as the log updates.

Log Message Format

An entry at the beginning of this log looks like the following:

```
LOG-STATUS: Log file opened on host ls1-tbdw at Wed Apr 12 20:24:54 2006
20:24:54.815 Sutron.c EVENT: Start decoding Sutron data for ADAM4
20:24:54.95 Sutron.c EVENT: Successfully decoded Sutron data for ADAM4
```

The format of the log file consists of the following fields:

- **Time stamp**: 20:24:54.951
- **File name**: Sutron.c
- **Message type**: EVENT:
- **Message string**: Successfully decoded Sutron data for ADAM4.

8.12 Decode Sutron Data

The **newLDADdataNotification** script checks the LDAD products directory for Sutron files. When a file is available, **newLDADdataNotification** passes the information to the **CO_serv**, which informs the **listener**. The **listener** moves the file to a **tmp** directory, executes the **preprocessSUTRON.pl** script, and moves the preprocessed file to the **Raw** directory or stores it into the text database, archives, and sends it to the OH’s database (SHEF). The **listener** process calls **routerLdadDecoder** to further decode the Sutron file for the creation of the netCDF and SHEF-encoded files. The decoded file is stored and a notification is sent to the forecaster stating that new data are available. The SHEF-encoded file is forwarded to the NCF for distribution.

Processes

```
LS  newLDADdataNotification
LS  CO_serv
PX2  listener
PX2  preprocessSUTRON.pl
PX2  CommsRouter LDAD_ROUTER
PX2  DataController LDAD_ROUTER LdadController.config
```
PX2 routerLdadDecoder
PX2 routerStoreTextEDEX
PX2 routerStoreEDEX
PX2 routerShefEncoderEDEX
PX2 distributeProduct
DX1 notificationServer

**Configuration Files** (all in /data/fxa/LDAD/data)

- LDADinfo.txt (for listener)
- LdadController.config (for DataController LDAD_ROUTER)
- LdadPatterns.txt (for routerLdadDecoder, routerStoreEDEX, routerShefEncoderEDEX)
- SUTRON.desc (for routerLdadDecoder)
- SUTRONStation.txt (for routerStoreEDEX)

**SUTRON.desc**

The SLDAD_DATA/data/SUTRON.desc file needs no change unless the Sutron datalogger has been modified to collect data with PE codes that are not in the current file. The file is directly coupled with the SHEFcodes.cfg file. The sequence of data group entries should be matched with the sequence of PE codes in SHEFcodes.cfg file.

Refer to Exhibit 8.12-1 for the Decode Sutron Data data flow diagram.

### 8.12.1 newLDADdataNotification

The newLDADdataNotification process uses the file name and a lookup table (LDADinfo.txt) to determine the type of data contained in the file and checks the LDAD products directory (/data/Incoming) periodically for Sutron files.

The newLDADdataNotification process creates and sends an IPC message to the CO_serv process using createAndSendIPCmessage. The message notifies CO_serv that there are Sutron files in the /data/Incoming directory.

If newLDADdataNotification fails to receive confirmation from listener within 5 minutes, or if an fstat on the file to transfer fails, the product gets moved to the /data/ladad/Problem directory using moveFileToProblemDir process. Otherwise, the
Exhibit 8.12-1. Decode Sutron Data (for LDAD) (Page 1 of 2)
“processed” files get moved to the /data/ldad/Processed directory using the moveFileToProcessedDir process.

To Access and Read the newLDADdataNotification Log

The newLDADdataNotification log resides on the active LS in the SLOG_DIR directory and records messages from the newLDADdataNotification process. Perform the following commands as user ldad:

**TYPE:**
```bash
cd $LOG_DIR/<yyyymmdd>
```
- Where `<yyyymmdd>` is today’s date.

**TYPE:**
```bash
ls -ltr newLDADdataNotification*
```
- Displays the current day’s newLDADdataNotification logs from earliest to latest.

**TYPE:**
```bash
tail -f <logname>
```
- Displays the latest registered messages as the log updates.

Log Message Format

An entry at the beginning of this log looks like the following:

```
20:25:24.174 newLDADdataNotification.c EVENT: Processing:
/data/Incoming/SUTRONADAM4302024.dat.999203094
20:25:24.760 newLDADdataNotification.c EVENT: waitForReply SUCCESS on
/data/Incoming/SUTRONADAM4302024.dat.999203094
```

The format of the log file consists of the following fields:

- **Time stamp**: 20:25:24.760
- **File name**: newLDADdataNotification.c
- **Message type**: EVENT:
- **Message string**: waitForReply SUCCESS on /data/Incoming/SUTRONADAM4302024.dat.999203094

**8.12.1 CO_serv**

CO_serv initially sets up a socket (TCP/IP) and waits for connections. The notification received from newLDADdataNotification is passed across the firewall to the listener process on the internal side on the PX2.

To Access and Read the CO_serv Log

The CO_serv log resides on the active LS in the directory SLOG_DIR and records messages from the CO_serv process. Perform the following commands as user ldad:

**TYPE:**
```bash
cd $LOG_DIR/<yyyymmdd>
```
- Where `<yyyymmdd>` is today’s date.
**Log Message Format**

An entry at the beginning of this log looks like the following:

```
LOG-STATUS: Log file opened on host ls1-rnk at Wed Apr 12 00:00:17 2006
00:00:17.076 co_serv_main.c EVENT: CO_serv process started
00:05:07.791 co_serv_main.c EVENT: CO_serv received a socket connection at: Apr12 06 00:05:07 GMT
00:05:07.801 co_serv_main.c EVENT: R  message Received by CO_serv....
00:06:36.998 co_serv_main.c EVENT: CO_serv received a socket connection at: Apr12 06 00:06:36 GMT
00:06:36.995 co_serv_main.c EVENT: R  message Received by CO_serv....
00:06:36.999 co_serv_main.c EVENT: R  message Received by CO_serv....
00:06:36.998 co_serv_main.c EVENT: R  message Received by CO_serv....
00:06:36.998 co_serv_main.c EVENT: R  message Received by CO_serv....
```

The format of the log file consists of the following fields:

- **Time stamp**: 00:00:17.076
- **File name**: co_serv_main.c
- **Message type**: EVENT:
- **Message string**: CO_serv process started

### 8.12.2 `listener`

The `listener` reads the `LDADinfo.txt` config file, sets up a socket, binds its local address so that the Client (CO_serv) can send to it, and waits for the connection from the Client process. When the notification is received, the `listener` retrieves the Sutron file from the `/data/Incoming` directory on the LS and copies it to the `$LDAD_DATA/tmp` directory on the PX inside the firewall. If the file requires no preprocessing, `listener` moves the data file directly to the Raw directory. The `listener` sends a verification when done.

The `listener` executes the `$FXA_HOME/ldad/bin/preprocessSUTRON.pl` script to preprocess the Sutron data files. The `listener` receives the fully qualified data file, the hostname used by notification routine, the port number of the process requiring notification upon completion of the preprocessing, and other necessary information about the data file format from CO_serv. Finally, the `listener` sends notification to the CommsRouter LDAD_ROUTER that a product has been preprocessed.

**To Access and Read the `listener` Log**

The `listener` log resides on PX2 in the directory `$LOG_DIR` and records messages from the `listener` process. Perform the following commands as user ldad:

```
TYPE: cd $LOG_DIR/<yyyymmdd>
```

Where `<yyyymmdd>` is today’s date.
Chapter 8 Local Data Acquisition and Dissemination (LDAD) System

TYPE: 
```bash
ls -ltr listener*
```
- Displays the current day’s `listener` logs from earliest to latest.

TYPE: 
```bash
tail -f <logname>
```
- Displays the latest registered messages as the log updates.

Log Message Format

An entry at the beginning of this log looks like the following:

```
    </awips/ldad/bin/preprocessSUTRON.pl /data/fxa/LDAD/tmp/SUTRONADAM4181823.dat.1069179781.1069180027 px 15009>
```

The format of the log consists of the following fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time stamp</td>
<td>18:27:29.743</td>
</tr>
<tr>
<td>File name</td>
<td>listener.c</td>
</tr>
<tr>
<td>Message type</td>
<td>EVENT:</td>
</tr>
<tr>
<td>Message string</td>
<td>do_G_Msg: rcp completed for SUTRONADAM4181823.dat.1069179781.1069180027</td>
</tr>
</tbody>
</table>

8.12.3 preprocessSUTRON.pl

The `preprocessSUTRON.pl` script copies the files from the `$LDAD_DATA/tmp` directory to the `/awips/fxa/ldad/ldadtmp` directory. There, it reformats the Sutron data files, appends the `abstime` string to the file names, and stores the preprocessed file (.dat) in the `$LDAD_DATA/Raw` directory and the SHEF file into the Postgres text database, archives it, and sends it to the OH database.

To Access and Read the preprocessSUTRON Log

The `preprocessSUTRON` log resides on PX2 in the directory `$LOG_DIR` and records messages from the `preprocessSUTRON.pl` process. Perform the following commands as user ldad:

TYPE: 
```bash
cd $LOG_DIR/<yyyymmdd>
```
- Where `<yyyymmdd>` is today’s date.

TYPE: 
```bash
ls -ltr preprocessSUTRON*
```
- Displays the current day’s `preprocessSUTRON` logs from earliest to latest.

TYPE: 
```bash
tail -f <logname>
```
- Displays the latest registered messages as the log updates.
Log Message Format

An entry at the beginning of this log looks like the following:

```
<<<<<<<<<<<
<<<<<<<<<<<
Start preprocessing
/data/fxa/LDAD/tmp/SUTRONADAM4302024.dat.999203094.999203124
ARGV[0]: /data/fxa/LDAD/tmp/SUTRONADAM4302024.dat.999203094.999203124
ARGV[1]: px
ARGV[2]: 15009
moveFile(): move /data/fxa/LDAD/tmp/SUTRONADAM4302024.dat.999203094.999203124
to /data/fxa/LDAD/Raw/SUTRONADAM4302024.dat.999203094.999203124
moveFile(): /data/fxa/LDAD/tmp/SUTRONADAM4302024.dat.999203094.999203124 moved successfully.
1
Successfully moved Sutron CSV format file from /tmp directory into /raw directory
Notified LDAD_ROUTER new Sutron data file for decoding
```

The format of the log file consists of the following fields:

- **File name**
- **moveFile():**
- **Message string**

```
File name: /data/fxa/LDAD/tmp/SUTRONADAM4302024.dat.999203094.999203124
Message string: moveFile():
Message string: Successfully moved Sutron CSV format file from /tmp directory into /raw directory
Message string: Notified LDAD_ROUTER new Sutron data file for decoding
```

8.12.4 LDAD_ROUTER

The **LDAD_ROUTER** (registered LDAD CommsRouter) is responsible for transporting data notifications from the data acquisition processes to the data controller processes. The main function of the **LDAD_ROUTER** is message passing; it does no data processing. The router provides a known location where the data acquisition processes can send data messages.

▶ To Access and Read the CommsRouter Log

The **LDAD_ROUTER CommsRouter** log resides on PX2 in a time-stamped directory under **$LOG_DIR**. Perform the following commands as user ldad:

**TYPE:**
```
    cd $LOG_DIR/<yyyymmdd>
```
- Where `<yyyymmdd>` is today’s date.

**TYPE:**
```
    ps -wef |grep ldad
```
- Displays the `ldad` processes. Notice the PID of the **CommsRouter** process that is owned by user `ldad`.

**TYPE:**
```
    ls -ltr Comms*
```
- Displays the latest logs.
NOTE: The **CommsRouter** log containing the PID of the **LDAD_ROUTER** process is the **LDAD_ROUTER CommsRouter** log.

**TYPE:**

```
tail -f <logname>
```

Displays the log as it updates.

**Log Message Format**

An entry in the log should look like the following:

```
```

The log file consists of the following fields:

<table>
<thead>
<tr>
<th>Time stamp</th>
<th>CommsReceiver.C EVENT: NCF_ENTRY DSMsg: SUTRONADAM4302024.dat.999203094.999203124 1 41</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CommsReceiver.C EVENT: NCF_ENTRY DSMsg: 999203124.SUTRONADAM4302024.decoded 1 58</td>
</tr>
<tr>
<td></td>
<td>CommsReceiver.C EVENT: NCF_ENTRY DSMsg: SUTRONADAM4302024.dat.999203094.999203124 1 41</td>
</tr>
</tbody>
</table>

The **LDAD_ROUTER CommsRouter** is started automatically at system reboot time or can be started manually. The **CommsRouter** runs continuously, exiting only when killed by the restart program.

### 8.12.5 DataController

The **DataController** is responsible for creating the decoder and storage processes (as child processes), restarting them if they exit, routing appropriate messages to them, and queuing incoming data messages until the decoder and storage processes are ready. As data messages are stored in the queue, the size of the processes grows.

The **DataController**’s main program creates a **DataRouting** manager object when it first starts up and invokes a member function to start the **routerLdadDecoder**, **routerStoreEDEX**, and **routerShefEncoderEDEX** processes by reading and executing the **LdadController.config** file. The file resides on the **SLDAD_INTERNAL_DATA** directory and contains the names of the above processes to start/restart, a restart flag, and **eof** designators to alert the Data Routing Manager that there are no more processes listed in the file.

The **LDAD_ROUTER** sends product-received notifications to the **DataController**. When the processes first start up, they send messages containing their IPC addresses and process numbers to the **DataController**. The processes send the **DataController** a data request message and a ready-to-receive data message. In response, the **DataController** stores the processes’ ID information and data requests, and sends a message to the requesting processes (if a message is available). The **DataController** waits to receive another ready message from the processes before sending any more data messages. The
**DataController** queues any messages it receives from the **LDAD_ROUTER** until the processes send another ready message.

▲ **To Access and Read the DataController Log**

The **DataController** log resides on PX2 in a time-stamped directory under **SLOG_DIR**. Perform the following commands as user ldad:

**TYPE:**

```
cd $LOG_DIR/<yyyymmdd>
```

- Where `<yyyymmdd>` is today’s date.

**TYPE:**

```
ls -ltr DataController*
```

- Displays the latest log for the **DataController** **LDAD_ROUTER** process.

**TYPE:**

```
tail -f <logname>
```

- Displays the current log as it updates with the latest messages.

**Log Message Format**

An entry in the log should look similar to the following:

```
```

The log file consists of the following fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time stamp</td>
<td>20:25:25.887</td>
</tr>
<tr>
<td>File name</td>
<td>CommsReceiver.C</td>
</tr>
<tr>
<td>Message type</td>
<td>EVENT:</td>
</tr>
<tr>
<td>Message string</td>
<td>NCF_ENTRY CDMsg: SUTRONADAM4302024.decoded 999203124.41</td>
</tr>
</tbody>
</table>

**8.12.6 routerLdadDecoder**

The **routerLdadDecoder** process is spawned by the **DataController** as specified in the **LdadController.config** file. The **routerLdadDecoder** process sends a data request to the **DataController** for the Sutron products it wishes to receive by specifying product header patterns of interest. On receipt of a notification from the **DataController** that a product of interest has arrived, the **routerLdadDecoder** reads the directory containing the raw files, **SLDAD_DATA/Raw**.

The **routerLdadDecoder** process reads each file and stores them via serialized data files to the **SLDAD_DECODED_DATA** directory. The serialized file name is sent back
through the **LDAD_ROUTER** to notify registered LDAD storage processes of the arrival of each LDAD decoded product.

### To Access and Read the routerLdadDecoder Log

The **routerLdadDecoder** log resides on PX2 in a time-stamped directory under **SLOG_DIR**. Perform the following commands as user ldad:

**TYPE:**
```
cd $LOG_DIR/<yyyymmdd>
```

Where `<yyyymmdd>` is today’s date.

**TYPE:**
```
ls -ltr routerLdadDecoder*
```

Displays the latest log for the **routerLdadDecoder** process.

**TYPE:**
```
tail -f <logname>
```

Displays the current log as it updates with the latest messages.

### Log Message Format

An entry in the log should look like the following:

```
20:25:25.884 LdadRoutines.C EVENT: LdadRoutines::pingLdadRouter(), '$LDAD_DECODED_DATA/999203124.SUTRONADAM4302024.decoded', sent to LDAD_ROUTER
```

The log file consists of the following fields:

- **Time stamp**
- **File name**
- **Message type**
- **Message string**

```
Time stamp   20:25:25.884
File name    LdadRoutines.C
Message type  EVENT:
Message string    LdadRoutines::pingLdadRouter(), ‘$LDAD_DECODED_DATA/999203124.SUTRONADAM4302024.decoded’, sent to LDAD_ROUTER
```

### 8.12.7 routerStoreEDEX

The **routerStoreEDEX** process is spawned by the **DataController** as specified in the **LdadController.config** file. The **routerStoreEDEX** process sends a data request to the **DataController** for serialized decoded Campbell files in netCDF format. When notified by the **DataController** that a product of interest has arrived, **routerStoreNetcdf** reads the directory containing the files, **SLDAD_DECODED_DATA**. The **routerStoreEDEX** process stores the files in the /data_store/ldad directory in the XML format and sends a notification to the **EDEX**.
To Access and Read the routerStoreNetcdf Log:

The **routerStoreNetcdf** log resides on PX2 in a time-stamped directory under **SLOG_DIR**. Perform the following commands as user ldad:

**TYPE:**
cd $LOG_DIR/<yyyymmdd>

- Where `<yyymmd>` is today’s date.

**TYPE:**
ls -ltr routerStoreNetcdf*

- Displays the latest logs for the **routerStoreNetcdf** process.

**TYPE:**
tail -f <logname>

- Displays the log as it updates with the latest messages.

Log Message Format

An entry in the log should look like the following:

```
20:30:31.560 routerStoreEDEX.C EVENT: routerStoreEDEX /data/fxa/LDAD/decoded/999203424.RAWS.decoded completed
```

The log file consists of the following fields:

<table>
<thead>
<tr>
<th>Time stamp</th>
<th>20:25:25.890</th>
</tr>
</thead>
<tbody>
<tr>
<td>File name</td>
<td>routerStoreEDEX.C</td>
</tr>
<tr>
<td>Message type</td>
<td>EVENT:</td>
</tr>
<tr>
<td>Message string</td>
<td>routerStoreEDEX.C,</td>
</tr>
<tr>
<td></td>
<td>putData(/data/fxa/LDAD/decoded/999203124.SUTRONADAM4302024.decoded)</td>
</tr>
</tbody>
</table>

8.12.8 **routerShefEncoderEDEX**

The **routerShefEncoderEDEX** process is spawned by the **DataController** as specified in the **LdadController.config** file. The **routerShefEncoderEDEX** process sends a data request to the **DataController** for serialized decoded files. On receipt of a notification from the **DataController** that a product of interest has arrived, the **routerShefEncoderEDEX** reads the directory containing the files, **SLDAD_DECODED_DATA**. The files there are SHEF-encoded and moved to the /data_store/ldad directory. The **routerShefEncoderEDEX** process sends a notification to the **EDEX**. Once the SHEF-encoded data file is stored in the /data_store/ldad directory, the **routerShefEncoderEDEX** saves a copy of the file in the /tmp directory and calls **distributeProduct** process to send the file to the NCF over the WAN.
To Access and Read the routerShefEncoderEDEX Log

The routerShefEncoderEDEX log resides on PX2 in a time-stamped directory under $LOG_DIR$. Perform the following commands as user ldad:

**TYPE:**
```
cd $LOG_DIR/<yyyymmdd>
```
- Where `<yyyymmdd>` is today’s date.

**TYPE:**
```
ls -rtal routerShefEncoderEDEX*
```
- Displays the latest logs for the routerShefEncoderEDX process.

**TYPE:**
```
tail -f <logname>
```
- Displays the current log as it is updated with the latest messages.

Log Message Format

An entry in the log should look like the following:

```
20:25:25.891 routerShefEncoderEDEX.C EVENT: routerShefEncoderEDEX
</data/fxa/LDAD/decoded/999203124.SUTRONADAM4302024.decoded>
```

The log file consists of the following fields:

- Time stamp
- File name
- Message type
- Message string

```
<>/data/fxa/LDAD/decoded/999203124.SUTRONADAM4302024.decoded>
```

8.12.9 distributeProduct

The distributeProduct process creates a product message and submits it for distribution across the AWIPS WAN to the addressed sites. The distributeProduct process first prepares the product by creating the WAN communications header and prepending the header to a temporary copy of the product. Distribution requests, enclosing the temporary copy of the product, are subsequently made through the msg_send utility program.
To Access and Read the distributeProduct Log

The distributeProduct log resides on PX2 in a time-stamped directory under SLOG_DIR. Perform the following commands as user ldad:

** TYPE:**
```
cd $LOG_DIR/<yyyymmdd>
```
- Where `<yyyymmdd>` is today’s date.

** TYPE:**
```
ls -ltr distributeProduct*
```
- Displays the latest logs for the distributeProduct process.

** TYPE:**
```
tail -f <logname>
```
- Displays the current log as it updates with the latest messages.

Log Message Format

An entry in the log should look like the following:

```
LOG-STATUS: Log file opened on host px2-tbdw at Wed April 12 19:30:59 2006
19:30:59.690 MHSProduct.C PROBLEM: Product (/data/fxa/isan/hydro/temp/A_format.973020657) is empty.
19:30:59.853 MHSProduct.C EVENT: Sending attachments (/data/fxa/isan/hydro/temp/A_format.973020657) only. Attachments are assumed to contain correct communications header.
19:31:02.504 MHSProduct.C EVENT: Product request (KSTORRZLDA) was successfully submitted to the MHS request server for distribution. Message Attributes:
  Request ID: TBDW-25380
  To: DEFAULTNCF
  Priority: 0
  Type: Routine
  Subject: 
  Handling Action: 0
19:31:02.518 MHSProduct.C EVENT: TBDW-25380 ~
```

The log file consists of the following fields:

<table>
<thead>
<tr>
<th>Time stamp</th>
<th>File name</th>
<th>Message type</th>
<th>Message string</th>
</tr>
</thead>
<tbody>
<tr>
<td>19:31:02.504</td>
<td>MHSProduct.C</td>
<td>EVENT:</td>
<td>Product request (KSTORRZLDA) was successfully submitted to the MHS request server for distribution. Message Attributes: Request ID: TBDW-25380 To: DEFAULTNCF Priority: 0 Type: Routine Subject: Handling Action: 0</td>
</tr>
</tbody>
</table>
8.12.10 notificationServer

The notificationServer provides information to its display clients that a new version of a display product has become available. Data acquisition processes send data notification messages to the notificationServer, which acts as a bridge between the data acquisition and workstation display components of AWIPS.

To Access and Read the notificationServer Log

The notificationServer log resides on the DX1 in the directory $LOG_DIR. This log records messages from the notificationServer process. Perform the following commands as user fxa:

TYPE: cd $LOG_DIR/<yyyymmdd>
- Where <yyyymmdd> is today’s date.

TYPE: ls -ltr notificationServer*
- Displays the current day’s notificationServer logs from earliest to latest.

TYPE: tail -f <logname>
- Displays the latest registered messages as the log updates.

Log Message Format

An entry at the beginning of the log looks like the following:

00:00:21.263 DepictableInventory.C EVENT: In purgeALittle, updating the time cache if files have been purged

00:00:21.377 DepictUpdateMgr.C DIAG: Currently there are no pending depictable notifications.

00:00:21.377 NotificationServer.C DIAG:
---Notification Server Statistics---
Size of client list (depict keys): 13
Processes per depictable (min/avg/max): 1/1.07692/2
Total data notifications processed since last stat: 0
Maximum number of pending notifications since last stat: 0
Total depict notifications sent since last stat: 0
Total depict notifications failed to match inventories: 0
Total depict notifications failed because of IPC: 0
Process ID of anonymous target: 19701
Host: lx5-tbw3
Port number: 44435

NORMAL PRIORITY CONNECTION
Connection created: Jun 06 12 20:06:03 GMT. Initiated by connecting process.
Last time this connection was used: Jun 06 12 20:06:03 GMT
No messages have been sent since last stat report.
No messages have been received since last stat report.
------------------------------------------------------------------------------
No connections have used any threads thus far.

00:00:21.263 DepictableInventory.C EVENT: In purgeALittle, updating the time cache if files have been purged
00:00:21.377 DepictUpdateMgr.C DIAG: Currently there are no pending depictable notifications.
00:00:21.377 NotificationServer.C DIAG:
---Notification Server Statistics---
Size of client list (depict keys): 13
Processes per depictable (min/avg/max): 1/1.07692/2
Total data notifications processed since last stat: 0
Maximum number of pending notifications since last stat: 0
Total depict notifications sent since last stat: 0
Total depict notifications failed to match inventories: 0
Total depict notifications failed because of IPC: 0

CACHED INVENTORY INFO
Num entries: 12
Total Size in bytes: 2544
Time spent retrieving non cached inventories: 0
20 biggest inventories:

00:00:21.377 Connection.C DIAG:

IPC STATISTICS for NOTIFICATION_SERVER from Jul 11 12 00:00:21 GMT to Jul 11 12 00:00:21 GMT
No TCP/IP socket connections have been terminated since the last stat report.
4 TCP/IP socket connections are currently active.

Process ID of anonymous target: 27720
Host: px2-tbw3
Port number: 54203
NORMAL PRIORITY CONNECTION
Connection created: May 30 12 17:18:40 GMT. Initiated by connecting process.
Last time this connection was used: Jul 10 12 23:55:42 GMT
No messages have been sent since last stat report.
No messages have been received since last stat report.

8.13 Acquire and Decode Comma Separated Variable Mesonet Data

The newLDADdataNotification script checks the LDAD products directory for Mesonet files. When a file is available, it passes the information to the CO_serv, which informs the listener. The listener executes the preProcessLDAD.pl script and calls routerLdadDecoder to decode the file. The decoded file is stored and a notification is sent to the user stating that new data are available.

Processes

LS newLDADdataNotification
LS CO_serv
PX2 listener
PX2 preProcessLDAD.pl
PX2 preprocessmesonet.pl
PX2 CommsRouter LDAD_ROUTER
PX2 DataController LDAD_ROUTER LdadController.config
PX2 routerLdadDecoder
PX2 routerStoreEDEX

Configuration Files (all in /data/fxa/LDAD/data)

LDADinfo.txt (for listener)
*Station.txt (files with station metadata for routerStoreEDEX)
*.desc (files for routerLdadDecoder)
LdadController.config (for DataController LDAD_ROUTER)
LdadPatterns.txt (for routerLdadDecoder, routerStoreEDEX)

Refer to Exhibit 8.13-1 for the Acquire and Decode Comma Separated Variable Mesonet data flow diagram.
Exhibit 8.13-1. Acquire and Decode Comma Separated Variable Mesonet Data (for LDAD) (Page 1 of 2)
Exhibit 8.13-1. Acquire and Decode Comma Separated Variable Mesonet Data (for LDAD) (Page 2 of 2)
8.13.1 CO_serv

The CO_serv process calls process_cu to process the request for a “cu” transfer. The process_cu executable parses the receive session file and invokes cu to transfer the file. The file is stored in the /data/Incoming directory. The notification received from newLDADdataNotification is passed across the firewall to the listener process on the internal side on the PX.

To Access and Read the CO_serv Log

The CO_serv log resides on the active LS in the $LOG_DIR directory. This log records messages from the CO_serv process. Perform the following commands as user ldad:

TYPE: cd $LOG_DIR/<yyyymmdd>
   ▪ Where <yyyymmdd> is today’s date.

TYPE: ls -ltr CO_serv*
   ▪ Displays the current day’s CO_serv logs from earliest to latest.

TYPE: tail -f <logname>
   ▪ Displays the latest registered messages as the log updates.

Log Message Format

An entry at the beginning of this log looks like the following:

LOG-STATUS: Log file opened on host ls1-rnk at Wed aPR 12 00:00:17 2006
00:00:17.076 co_serv_main.c EVENT: CO_serv process started
00:05:07.791 co_serv_main.c EVENT: CO_serv received a socket connection at: Apr12 06
00:05:07 GMT
00:05:07.801 co_serv_main.c EVENT: R  message Received by CO_serv....
00:06:36.988 co_serv_main.c EVENT: CO_serv received a socket connection at: Apr12 06
00:06:36 GMT
00:06:36.995 co_serv_main.c EVENT: R  message Received by CO_serv....
00:13:48.283 co_serv_main.c EVENT: CO_serv received a socket connection at: Apr12 06
00:13:48 GMT
00:13:48.291 co_serv_main.c EVENT: R  message Received by CO_serv....

The log file consists of the following fields:

<table>
<thead>
<tr>
<th>Time stamp</th>
<th>00:00:17.076</th>
</tr>
</thead>
<tbody>
<tr>
<td>File name</td>
<td>co_serv_main.c</td>
</tr>
<tr>
<td>Message type</td>
<td>EVENT:</td>
</tr>
<tr>
<td>Message string</td>
<td>CO_serv process started</td>
</tr>
</tbody>
</table>
To Access and Read the co.log Log

The co.log log resides on PX2 in the $LOG_DIR directory and records messages from the CO_serv process executing the sendLDADnotification.pl script. Perform the following commands as user ldad:

**TYPE:**

- cd $LOG_DIR/<yyyymmdd>
  - Where <yyyymmdd> is today’s date.

**TYPE:**

- ls -ltr co.log*
  - Displays the current day’s co.log logs from earliest to latest.

**TYPE:**

- tail -f <logname>
  - Displays the latest registered messages as the log updates.

Log Message Format

An entry at the beginning of this log looks like the following:

```plaintext
/awips/ldad/bin/sendLDADnotification.pl:94, socket ok
/awips/ldad/bin/sendLDADnotification.pl:104, bind ok
/awips/ldad/bin/sendLDADnotification.pl:117, connect ok
/awips/ldad/bin/sendLDADnotification.pl:27, Begin sendLDADnotification.pl
/awips/ldad/bin/sendLDADnotification.pl:38, O, 4296 px2-tbdw 4296, px, 15008
/awips/ldad/bin/sendLDADnotification.pl:60, opType = O, msgString = 4296 px2-tbdw 4296,
  host = px, port = 15008
/awips/ldad/bin/sendLDADnotification.pl:94, socket ok
/awips/ldad/bin/sendLDADnotification.pl:104, bind ok
/awips/ldad/bin/sendLDADnotification.pl:117, connect ok
/awips/ldad/bin/sendLDADnotification.pl:27, Begin sendLDADnotification.pl
/awips/ldad/bin/sendLDADnotification.pl:38, S, CAMPBELL_coll_sess 4300 px, px, 15008
/awips/ldad/bin/sendLDADnotification.pl:60, opType = S, msgString = CAMPBELL_coll_sess
  4300 px, host = px, port = 15008
/awips/ldad/bin/sendLDADnotification.pl:94, socket ok
/awips/ldad/bin/sendLDADnotification.pl:104, bind ok
/awips/ldad/bin/sendLDADnotification.pl:117, connect ok
/awips/ldad/bin/sendLDADnotification.pl:27, Begin sendLDADnotification.pl
/awips/ldad/bin/sendLDADnotification.pl:38, O, 15008 px2-tbdw 4398, px, 15008
/awips/ldad/bin/sendLDADnotification.pl:60, opType = O, msgString = 15008 px2-tbdw 4398,
  host = px, port = 15008
```

8.13.2 newLDADdataNotification

The newLDADdataNotification process uses the file’s name and a lookup table (LDADinfo.txt) to determine the type of data contained in the file and checks the LDAD products directory (/data/Incoming) periodically for Mesonet files.

The newLDADdataNotification process creates and sends an IPC message to the CO_serv process using createAndSendIPCmessage. The message notifies CO_serv that there are Mesonet files in the /data/Incoming directory.

If newLDADdataNotification fails to receive confirmation from listener within 5 minutes or if an fstat on the file to transfer fails, the product gets moved to the
To Access and Read the newLDADdataNotification Log

The `newLDADdataNotification` log resides on the active LS in the `SLOG_DIR` directory and records messages from the `newLDADdataNotification` process. Perform the following commands as user ldad:

**TYPE:**
```
cd $LOG_DIR/<yyyymmdd>
```
- Where `<yyyymmdd>` is today’s date.

**TYPE:**
```
ls -ltr newLDADdataNotification*
```
- Displays the current day’s `newLDADdataNotification` logs from earliest to latest.

**TYPE:**
```
tail -f <logname>
```
Displays the latest registered messages as the log updates.

Log Message Format

An entry at the beginning of this log looks like the following:

```
LOG-STATUS: Log file opened on host ls1-tbwd at Wed Apr 12 00:00:56 2006
00:00:56.825 newLDADdataNotification.c EVENT: newLDADdataNotification process started
00:00:56.829 newLDADdataNotification.c EVENT: Reading /data/Incoming/ every 20 seconds
00:00:56.834 newLDADdataNotification.c EVENT: Processing : /data/Incoming/RAWS.dat
00:09:37.413 newLDADdataNotification.c EVENT: Processing : /data/Incoming/alert.wx.dat
00:09:38.155 newLDADdataNotification.c EVENT: waitForReply SUCCESS on /data/Incoming/alert.wx.dat
```

The format of the log file consists of the following fields:

- **Time stamp**
- **File name**
- **Message type**
- **Message string**

```
Time stamp 00:09:38.155
File name newLDADdataNotification.c
Message type EVENT:
Message string waitForReply SUCCESS on /data/Incoming/alert.wx.dat
```

8.13.3 listener

The `listener` reads the `LDADinfo.txt` config file, sets up a socket, binds its local address so that the Client (`CO_serv`) can send to it, and waits for the connection from the Client process. When the notification is received, the `listener` retrieves the Mesonet file from the `/data/Incoming` directory on the active LS and copies it to the `SLDAD_DATA/tmp` directory on the PX2. If the file requires no preprocessing, `listener` moves the data file directly to the `Raw` directory. The `listener` sends verification when finished.

The `listener` executes the `preProcessLDAD.pl` script to preprocess the Mesonet data files. The `listener` receives the fully qualified data file, the hostname used by notification
routine, the port number of the process requiring notification upon completion of the preprocessing, and other necessary information about the Mesonet data file format from CO_serv. Finally, the listener sends notification to the LDAD **CommsRouter**, **LDAD_ROUTER**, that a product has been preprocessed.

To Access and Read the listener Log

The listener log resides on PX2 in the directory **$LOG_DIR**. This log records messages from the listener process. Perform the following commands as user ldad:

**TYPE:**

```bash
cd $LOG_DIR/<yyyymmdd>
```

Where `<yyyymmdd>` is today’s date.

**TYPE:**

```bash
ls -ltr listener*
```

Displays the current day’s listener logs from earliest to latest.

**TYPE:**

```bash
tail -f <logname>
```

Displays the latest registered messages as the log updates.

Log Message Format

An entry at the beginning of this log looks like the following:

```
00:09:39.213 listener.c EVENT: do_G_Msg: rcp completed for alert_wx.dat.97433378
```

The format of the log consists of the following fields:

- **Time stamp**: 00:09:39.263
- **File name**: listener.c
- **Message type**: EVENT:
- **Message string**: do_G_Msg: Preprocessor Command Line = </awips/ldad/bin/preProcessLDAD.pl /data/fxa/LDAD/tmp/alert_wx.dat.97433378 px 15009>

### 8.13.4 preProcessLDAD.pl

The **preProcessLDAD.pl** script preprocesses UDFCD ALERT (_R5, _WL, _WX), Cdot, IFLOWS, and LARC data report files for the LDAD decoder. It initializes file input/output and values to the existing variables, calls the appropriate preprocessor (**preprocessmesonet.pl** for Mesonet data), and cleans up.
8.13.5 preprocessmesonet.pl

The preprocessmesonet.pl script copies the file from the $LDAD_DATA/tmp directory to the $LDAD_HOME/ldadtmp directory. There, it reformats the Mesonet data file, appends the abstime string to the file name, and stores the preprocessed file in the $LDAD_DATA/Raw directory.

8.13.6 LDAD_ROUTER

The LDAD_ROUTER (registered LDAD CommsRouter) is responsible for transporting data notifications from the data acquisition processes to the data controller processes. The main function of the LDAD_ROUTER is message passing; it does not do any data processing. The router provides a known location where the data acquisition processes can send data messages.

To Access and Read the CommsRouter Log

The LDAD_ROUTER CommsRouter log resides on PX2 in a time-stamped directory under $LOG_DIR. Perform the following commands as user ldad:

```
TYPE: cd $LOG_DIR/<yyyymmdd>
  Where <yyyymmdd> is today’s date.
TYPE: ps -wef |grep ldad
  Displays the ldad processes. Notice the PID of the CommsRouter process that is owned by user ldad.
TYPE: ls -ltr Comms*
  Displays the latest logs.
```

NOTE: The CommsRouter log containing the PID of the LDAD_ROUTER process is the LDAD_ROUTER CommsRouter log.

```
TYPE: tail -f <logname>
  Displays the log as it updates.
```

Log Message Format

An entry in the log should look like the following:

```
LOG-STATUS: Log file opened on host px2-tbdw at Wed April 12 00:00:16 2006
00:01:02.726 CommsReceiver.C EVENT: NCF_ENTRY DSMsg: /data/fxa/LDAD/Raw/RAWS.dat.974332858 1 37
00:01:02.729 CommsReceiver.C EVENT: NCF_ENTRY DSMsg: RAWS.dat.974332858 1 18
```
00:01:05.511 CommsReceiver.C EVENT: NCF_ENTRY DSMsg: 974332858.RAWS.decoded 1 45
00:03:38.783 CommsReceiver.C EVENT: NCF_ENTRY DSMsg: /data/fxa/LDAD/Raw/974332800.03.msas_qc.RAWS 1 44
00:03:39.361 CommsReceiver.C EVENT: NCF_ENTRY DSMsg: RAWS.974332800.03.msas_qc.decoded 1 56
00:09:40.979 CommsReceiver.C EVENT: NCF_ENTRY DSMsg: /data/fxa/LDAD/Text/alert_wx.dat.974333378 1 42
00:09:41.462 CommsReceiver.C EVENT: NCF_ENTRY DSMsg: 974333378.alert_wx.decoded 1 49

The log consists of the following fields:

- **Time stamp**: 00:09:41.462
- **File name**: CommsReceiver.C
- **Message type**: EVENT:
- **Message string**: NCF_ENTRY DSMsg: 974333378.alert_wx.decoded 1 49

The **LDAD_ROUTER CommsRouter** is started automatically at system reboot time or can be started manually. The **CommsRouter** runs continuously, exiting only when killed by the restart program.

### 8.13.7 DataController

The **DataController** is responsible for creating the decoder and storage processes (as child processes), restarting them if they exit, routing appropriate messages to them, and queuing incoming data messages until the decoder and storage processes are ready. As data messages are stored in the queue, the size of the processes grows.

The **DataController**’s main program creates a **DataRouting** manager object when it first starts up and invokes a member function to start the **routerLdadDecoder**, and **routerStoreEDEX**, processes by reading and executing the **LdadController.config** file. The file resides on the **LDAD_INTERNAL_DATA** directory and contains the names of these decoders to start/restart, a restart flag, and **eof** designators to alert the Data Routing Manager that there are no more processes listed in the file.

The **LDAD_ROUTER** sends product-received notifications to the **DataController**. When the processes first start up, they send messages containing their IPC addresses and process numbers to the **DataController**. The processes send the **DataController** a data request message and a ready-to-receive data message. In response, the **DataController** stores the processes’ ID information and data requests, and sends a message to the requesting processes (if a message is available). The **DataController** waits to receive another ready message from the processes before sending any more data messages. The **DataController** queues any messages it receives from the **LDAD_ROUTER** until the processes send another ready message.

#### To Access and Read the DataController Log

The **DataController** log resides on PX2 in a time-stamped directory under **SLOG_DIR**. Perform the following commands as user ldad:
Chapter 8 Local Data Acquisition and Dissemination (LDAD) System

TYPE: cd $LOG_DIR/<yyyymmdd>
   ▪ Where <yyyymmdd> is today’s date

TYPE: ps -wef |grep ldad
   ▪ Displays the ldad processes. Note the PID for the DataController.

TYPE: ls -rtal DataController*
   ▪ Displays the latest logs for the DataController

TYPE: tail -f <logname>
   ▪ Displays the log as it updates with the latest messages

Log Message Format

An entry in the log should look similar to the following:

LOG-STATUS: Log file opened on host px2-tbdw at Wed April 12 00:00:14 2006
00:00:13.815 DataCaptReceiver.C EVENT: DataReqMsg: pattern = (.*/\[0-9]*|[^0-9]*\.[^0-9]*msas_qc\..*)$
00:00:14.390 DataCaptReceiver.C EVENT: DataReqMsg: pattern = ^\[0-9]*\.[^m][^s][^a][^s][^ ]([^g][^c]))))))).\..decoded$
00:00:14.716 DataCaptReceiver.C EVENT: DataReqMsg: pattern = .*//\[0-9]+.*/Text/.*/\[0-9]+.*/decoded
00:03:39.365 CommsReceiver.C EVENT: NCF_ENTRY CDMsg: RAWS.974333378.decoded 1 56
00:09:41.086 CommsReceiver.C EVENT: NCF_ENTRY CDMsg: alert_wx.dat.974333378.alert_wx.decoded 1 49
00:09:41.464 CommsReceiver.C EVENT: NCF_ENTRY CDMsg: alert_wx.dat.974333378.alert_wx.decoded 1 49
00:09:41.485 CommsReceiver.C EVENT: NCF_ENTRY CDMsg: alert_wx.dat.974333378.alert_wx.decoded 1 49

The log file consists of the following fields:

| Time stamp | 00:09:43.935 |
| File name  | CommsReceiver.C |
| Message    | EVENT: |
| Message string | NCF_ENTRY CDMsg: alert_wx.dat.974333378 1 22 |
8.13.8 routerLdadDecoder

The `routerLdadDecoder` process is spawned by the `DataController` as specified in the `LdadController.config` file. The `routerLdadDecoder` process sends a data request to the `DataController` requesting Mesonet products it wishes to receive by specifying product header patterns of interest. On receipt of a notification from the `DataController` that a Mesonet product of interest has arrived, the `routerLdadDecoder` reads the directory containing the raw files, `SLDAD_DATA/Raw`.

The description file (*`.desc`) is read by the `routerLdadDecoder`, which uses the information to evaluate each line of data by checking for the meteorological variable, the incoming data’s units of measure, and the storage units of measure.

The `.desc` format follows.

<table>
<thead>
<tr>
<th># netcdf Var Name</th>
<th>data type</th>
<th>input units</th>
<th>stored units</th>
</tr>
</thead>
<tbody>
<tr>
<td>stationId</td>
<td>STRING</td>
<td>STRING</td>
<td>STRING</td>
</tr>
<tr>
<td>observationTime</td>
<td>DATE_TIME</td>
<td>DATE_TIME_STRING</td>
<td>ABSTIME</td>
</tr>
<tr>
<td>temperature</td>
<td>FLOAT</td>
<td>F</td>
<td>Kelvin</td>
</tr>
<tr>
<td>dewpoint</td>
<td>FLOAT</td>
<td>F</td>
<td>Kelvin</td>
</tr>
<tr>
<td>precipAccum</td>
<td>FLOAT</td>
<td>mm</td>
<td>m</td>
</tr>
<tr>
<td>windSpeed</td>
<td>FLOAT</td>
<td>mph</td>
<td>m/s</td>
</tr>
<tr>
<td>windGust</td>
<td>FLOAT</td>
<td>mph</td>
<td>m/s</td>
</tr>
<tr>
<td>windDir</td>
<td>INT</td>
<td>degreeN</td>
<td>degreeN</td>
</tr>
<tr>
<td>windDirMax</td>
<td>FLOAT</td>
<td>degreeN</td>
<td>degreeN</td>
</tr>
<tr>
<td>visibility</td>
<td>FLOAT</td>
<td>ft</td>
<td>m</td>
</tr>
</tbody>
</table>

The `routerLdadDecoder` process reads each file and stores them via serialized data files to the `SLDAD_DECODED_DATA` directory. The serialized file name is sent back through the `LDAD_ROUTER` to notify registered LDAD storage processes of the arrival of each LDAD decoded product.

To Access and Read the `routerLdadDecoder` Log

The `routerLdadDecoder` log resides on PX2 in a time-stamped directory under `$LOG_DIR`. Perform the following commands as user ldad:

```
TYPE: cd $LOG_DIR/<yyyymmdd>
  - Where `<yyyymmdd>` is today’s date.

TYPE: ls -rtal routerLdadDecoder*
  - Displays the latest logs for the `routerLdadDecoder`.

TYPE: tail -f <logname>
  - Displays the log as it updates with the latest messages.
```

Log Message Format

An entry in the log should look like the following:
The log consists of the following fields:

- **Time stamp**: 00:09:41.467
- **File name**: LdadRoutines.C
- **Message type**: EVENT:
- **Message string**: LdadRoutines::pingLdadRouter(), '$LDAD_DECODED_DATA/ 974333378.alert_wx.decoded', sent to LDAD_ROUTER

### 8.13.9 routerStoreEDEX

The routerStoreEDEX process is spawned by the DataController as specified in the LdadController.config file. The routerStoreEDEX process sends a data request to the DataController requesting serialized decoded Mesonet files. On receipt of a notification from the DataController that a product of interest has arrived, routerStoreEDEX reads the directory containing the files, $LDAD_DECODED_DATA. The *Station.txt file is used by routerStoreEDEX to pull station metadata such as latitude, longitude, and elevation.

The .txt format follows.

```
ProviderId|AFOS(HB5)|StationName|StationElevation|Latitude|Longitude|localTZ|LocationDesc|
StationType|NumberOfInstruments|NumberOfLevels|Maint/CalibSchedules|
SiteDesc|
```

- **Provider ID**: Data provider station ID
- **AFOS ID**: Station AFOS (or Handbook V) ID
- **Station Name**: Text name of station
- **Elevation**: Elevation of station
- **Latitude**: Latitude of station
- **Longitude**: Longitude of station
- **local TZ**: Local time zone (i.e., log MST7MDT)
- **Location Desc**: Location of station (Text)


<table>
<thead>
<tr>
<th>Station Type</th>
<th>Type of station (i.e., logtower, surface, floating platform)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Instruments</td>
<td>The number of reporting instruments for the station</td>
</tr>
<tr>
<td>Number of Levels</td>
<td>Number of reporting levels for the data (level information should be provided in the instrument table.)</td>
</tr>
<tr>
<td>Maintenance/Calibration Schedules</td>
<td>Frequency of maintenance/calibration</td>
</tr>
<tr>
<td>Site Description</td>
<td>A text description of the site surrounding the routerStoreEDEX process.</td>
</tr>
</tbody>
</table>

The routerStoreEDEX process stores the files in the SLAD_DATA/mesonet/netCDF directory and sends a notification to the notificationServer.

**To Access and Read the routerStoreEDEX Log**

The routerStoreEDEX log resides on PX2 in a time-stamped directory under $SLOG_DIR$. Perform the following commands as user ldad:

**TYPE:**

```
cd $LOG_DIR/<yyyymmdd>
```

Where `<yyyymmdd>` is today’s date.

**TYPE:**

```
ps -wef |grep ldad
```

Displays the routerStoreEDEX process along with all other ldad-owned processes.

**TYPE:**

```
ls -rtal routerStoreEDEX*
```

Displays the latest logs for the routerStoreEDEX process.

**TYPE:**

```
tail -f <logname>
```

Displays the log as it updates with the latest messages.

**Log Message Format**

```
00:01:06.020 DataCaptReceiver.C USE: ProdLen 45
00:01:06.797 routerStoreEDEX.C EVENT: routerStoreEDEX.C, putData(/data/fxa/LDAD/decoded/974332858.RAWS.decoded)
00:01:16.599 LdadStorage.C EVENT: LdadStorage::separatePatterns () <-99., -99.00 > from <-99. -99.00>
00:01:19.301 routerStoreEDEX.C EVENT: routerStoreEDEX /data/fxa/LDAD/decoded/974332858.RAWS.decoded completed
00:03:39.839 LdadStorage.C PROBLEM: Unable to find netCDF data key for , msas_qc.RAWS
00:03:39.922 routerStoreEDEX.C EVENT: routerStoreEDEX /data/fxa/LDAD/decoded/974332800.03.msas_qc.decoded completed
00:09:41.544 DataCaptReceiver.C USE: ProdLen 49
```
The log consists of the following fields:

<table>
<thead>
<tr>
<th>Time stamp</th>
<th>00:09:41.560</th>
</tr>
</thead>
<tbody>
<tr>
<td>File name</td>
<td>routerStoreEDEX.C</td>
</tr>
<tr>
<td>Message type</td>
<td>EVENT:</td>
</tr>
<tr>
<td>Message string</td>
<td>routerStoreEDEX.C,</td>
</tr>
<tr>
<td></td>
<td>putData(/data/xfa/LDAD/decoded/974333378.alert_wx.decoded)</td>
</tr>
</tbody>
</table>

### 8.14 Acquire and Process ASOS/MicroART Data (for LDAD)

The processing and data flow for the ASOS and micro-Automatic Radio theodolite System (MicroART) products are identical; therefore, their description is combined in this section.

The `suaReceiver` process starts the data acquisition process for the ASOS/MicroART system. Data acquisition occurs through a pseudo-terminal device configured by the Xyplex Terminal Server. The `newLDADdataNotification` script checks the LDAD directory for ASOS/MicroART files. When a file is available, the `newLDADdataNotification` script passes the information to the `suaReceiver`, which informs `CO_serv`, which passes the message to the `listener`, which executes the `preprocessSUA.pl` script. The `preprocessSUA.pl` script copies the file to a tmp directory and stores it into the text database via the `textdb` command line interface. When the file is successfully stored in the database, a notification is sent to the text `notificationServer`. Finally, the data are routed to the NCF via the WAN.

**Processes**

- LS `suaReceiver`
- LS `newLDADdataNotification`
- LS `CO_serv`
- PX2 `listener`
- PX2 `preprocessSUA.pl`
- PX2 `distributeProduct`

**Configuration Files (all in /data/xfa/LDAD/data)**

- `suaCategories.txt` (for `suaReceiver`)
- `LDADinfo.txt` (for `listener`)
- `rcv_action2codes.txt` (for `distributeProduct`)
- `siteCommission.txt` (for `distributeProduct`)

Refer to Exhibit 8.14-1 for the Acquire and Process ASOS/MicroART data flow diagram.
Chapter 8  Local Data Acquisition and Dissemination (LDAD) System

Exhibit 8.14-1. Acquire and Process ASOS/MicroART Data (for LDAD) (Page 1 of 2)
Exhibit 8.14-1. Acquire and Process ASOS/MicroART Data (for LDAD) (Page 2 of 2)
8.14.1 suaReceiver

The Surface and Upper Air Receiver, suaReceiver, acquires observational data from the ASOS and MicroART stations. The suaReceiver process supports asynchronous connections from the ASOS/MicroART systems through a dial-in modem. The suaReceiver process initially sets up a TCP/IP socket and waits for connections. When a connection is made and a message arrives, the suaReceiver process parses the information contained in the message buffer. Using the parsed information, the suaReceiver process connects to the ASOS/MicroART station. Once a message is received and confirmed as a valid product message, the product is stored in the /data/Incoming directory.

To Access and Read the suaReceiver Log

The suaReceiver log resides on the active LS in a time-stamped directory under SLOG_DIR. The log file contains event messages recording product receipt and transmission, message format errors, and timeout errors in the receipt of incoming data. Perform the following commands as user ldad:

**TYPE:**
```
   cd $LOG_DIR/<yyyymmdd>
   Where <yyyymmdd> is today’s date.
```

**TYPE:**
```
   ps -ef |grep suaReceiver
   Displays the suaReceiver process. Notice the PID.
```

**TYPE:**
```
   ls -rtal suaReceiver*
   Displays the latest logs for the suaReceiver.
```

**TYPE:**
```
   tail -f <logname>
   Displays the log as it updates.
```

**NOTE:** The suaReceiver log containing the PID of the suaReceiver process is the current log.

Log Message Format

An entry at the beginning of this log looks like the following:

```
LOG-STATUS: Log file opened on host ls1-tbw4 at Wed May 12 00:01:49 2006
00:01:49.793 suaReceiver.C EVENT: suaReceiver process started ...
00:01:52.393 suaReceiver.C EVENT: Connection established with pseudo terminal device
11:00:50.333 SignalClient.C EVENT: Signal 15 (Terminate) received for clean shutdown
LOG-STATUS: Log file opened on host ls1-tbw4 at Wed May 12 00:02:01 2006
00:02:01.046 suaReceiver.C EVENT: suaReceiver process started ...
00:02:02.077 LocalDataAcq.C PROBLEM: open() failed on terminal device file.
No such file or directory, errno = 2
Exiting...
```

The format of the log file consists of the following fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time stamp</td>
<td>00:01:49 1999</td>
</tr>
<tr>
<td>File name</td>
<td>suaReceiver.C</td>
</tr>
<tr>
<td>Message Type</td>
<td>EVENT</td>
</tr>
<tr>
<td>Message string</td>
<td>Connection established with pseudo terminal device</td>
</tr>
</tbody>
</table>

### 8.14.2 newLDADdataNotification

The **newLDADdataNotification** process uses the file’s name and a lookup table (LDADinfo.txt) to determine the type of data contained in the file and checks the LDAD products directory (/data/Incoming) periodically for ASOS/MicroART files.

The **newLDADdataNotification** process creates and sends an IPC message to the CO_serv process using `createAndSendIPCmessage`. The message notifies CO_serv that there are ASOS/MicroART files in the /data/Incoming directory.

If **newLDADdataNotification** fails to receive confirmation from listener within 5 minutes or if an `fstat` on the file to transfer fails, the product gets moved to the /data/ldad/Problem directory using `moveFileToProblemDir` process. Otherwise, the “processed” file is moved to the /data/ldad/Processed directory.

To Access and Read the newLDADdataNotification Log

This log resides on the active LS in the $LOG_DIR directory and records messages from the **newLDADdataNotification** process. Perform the following commands as user ldad:

**TYPE:** `cd $LOG_DIR/<yyyymmdd>`
  - Where `<yyyymmdd>` is today’s date.

**TYPE:** `ls -ltr newLDADdataNotification*`
  - Displays the current day’s **newLDADdataNotification** logs from earliest to latest.

**TYPE:** `tail -f <logname>`
  - Displays the latest registered messages as the log updates.

Log Message Format

An entry at the beginning of this log looks like the following:

```
LOG-STATUS: Log file opened on host ls1-rnk at Wed May 12 00:00:23 2006
00:00:23.202 newLDADdataNotification.c EVENT: newLDADdataNotification process started
00:00:23.205 newLDADdataNotification.c EVENT: Reading /data/Incoming/ every 20 seconds
00:14:03.204 newLDADdataNotification.c EVENT: Processing : /data/Incoming/SUACRWRR3CIN.dat
00:14:03.880 newLDADdataNotification.c EVENT: waitForReply SUCCESS on /data/Incoming/SUACRWRR3CIN.dat
00:29:03.223 newLDADdataNotification.c EVENT: Processing : /data/Incoming/SUACLERR7ILN.dat
```
00:29:03.951 newLDADdataNotification.c EVENT: waitForReply SUCCESS on /data/Incoming/SURCLERR7ILN.dat
00:44:03.273 newLDADdataNotification.c EVENT: Processing : /data/Incoming/raindata.dat

The format of the log file consists of the following fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time stamp</td>
<td>00:00:23.202</td>
</tr>
<tr>
<td>File name</td>
<td>newLDADdataNotification.c</td>
</tr>
<tr>
<td>Message type</td>
<td>EVENT:</td>
</tr>
<tr>
<td>Message string</td>
<td>newLDADdataNotification process started</td>
</tr>
</tbody>
</table>

### 8.14.3 CO_serv

**CO_serv** initially sets up a TCP/IP socket and waits for connections. The **newLDADdataNotification** process creates and sends an IPC message to the **CO_serv** using **createAndSendIPCmessage** routine. The message notifies **CO_serv** that there are processed files in the `/data/Incoming` directory. The notification received from **newLDADdataNotification** is passed to the **listener**.

#### To Access and Read the CO_serv Log

The **CO_serv** log resides on the active LS in the directory **SLOG_DIR** and records messages from the **CO_serv** process. Perform the following commands as user ldad:

**TYPE:**

```
cd $LOG_DIR/<yyyymmdd>
```

- Where `<yyyymmdd>` is today’s date.

**TYPE:**

```
ls -ltr CO_serv*
```

- Displays the current day’s **CO_serv** logs from earliest to latest.

**TYPE:**

```
tail -f <logname>
```

- Displays the latest registered messages as the log updates.

#### Log Message Format

An entry at the beginning of this log looks like the following:

```
LOG-STATUS: Log file opened on host ls1-rnk at Wed May 12 00:00:17 2006
00:00:17.076 co_serv_main.c EVENT: CO_serv process started
00:05:07.791 co_serv_main.c EVENT: CO_serv received a socket connection at: May12 06
00:05:07 GMT
00:05:07.801 co_serv_main.c EVENT: R  message Received by CO_serv.....
00:06:36.988 co_serv_main.c EVENT: CO_serv received a socket connection at: May12 06
00:06:36 GMT
00:06:36.995 co_serv_main.c EVENT: R  message Received by CO_serv.....
00:13:48.283 co_serv_main.c EVENT: CO_serv received a socket connection at: May12 06
00:13:48 GMT
00:13:48.291 co_serv_main.c EVENT: R  message Received by CO_serv.....
```
The format of the log file consists of the following fields:

- **Time stamp**: 00:00:17.076
- **File name**: co_serv_main.c
- **Message type**: EVENT:
- **Message string**: CO_serv process started

### 8.14.4 listener

The **listener** reads the **LDADinfo.txt** config file, sets up a socket, binds its local address so that the Client (**CO_serv**) can send to it, and waits for the connection from the Client process. When the notification is received, the **listener** retrieves the ASOS/MicroART file from the `/data/Incoming` directory on the active LS and copies it to the **SLDAD_DATA/tmp** directory on the PX. The **listener** sends a verification when it is finished.

The **listener** executes the **preprocessSUA.pl** script to preprocess the ASOS/MicroART data files. The **listener** receives the fully qualified data file, the hostname used by notification routine, the port number of the process requiring notification upon completion of the preprocessing, and other necessary information about the ASOS/MicroART data file format from **CO_serv**.

#### To Access and Read the listener Log

The **listener** log resides on PX2 in the directory **$LOG_DIR** and records messages from the **listener** process. Perform the following commands as user ldad:

**TYPE:**

```bash
   TYPE: cd $LOG_DIR/<yyyymmdd>
       Where <yyyymmdd> is today’s date.
```

**TYPE:**

```bash
   TYPE: ls -ltr listener*
       Displays the current day’s listener logs from earliest to latest.
```

**TYPE:**

```bash
   TYPE: tail -f <logname>
       Displays the latest registered messages as the log updates.
```

#### Log Message Format

An entry at the beginning of this log looks like the following:

```
LOG-STATUS: Log file opened on host px2-rnk at Wed May 12 00:00:41 2006
00:00:41.606 listener.c EVENT: listener process started
00:01:39:002 listener.c EVENT: do_G_Msg: rcp completed for SUABABVIAD.dat.1029783689
00:01:39:125 listener.c EVENT: do_G_Msg: Preprocessor Command Line= /awips/ldad/bin/preprocessSUA.pl /data/fxa/LDAD/tmp/SUABABVIAD.dat.1029783679 px 15009
```
The format of the log file consists of the following fields:

- **Time stamp**: 00:01:39:002
- **File name**: listener.c
- **Message type**: EVENT:
- **Message string**: do_G_Msg: rcp completed for SUA WBABVIAD.dat.1029783689

### 8.14.5 preprocessSUA.pl

The **preprocessSUA.pl** script preprocesses ASOS/MicroART data files containing surface observations for the LDAD text decoder. These data files are pulled across the security firewall from the external side and stored on the internal side in the SLDAD_DATA/tmp directory. The file name uses the following convention: SUA<AFOS_ID>.dat, where SUA is a data type identifier corresponding to those listed in the LDADinfo.txt file and <AFOS_ID> is a seven- to nine-character AFOS product identifier.

The preprocessSUA.pl script enables WAN distribution of the processed files to the NCF. After the files are successfully disseminated over the WAN, the preprocessSUA.pl script moves the files from the $LDAD_DATA/tmp directory to the $FXA_DATA/archive/obs/scratch directory.

#### To Access and Read the preprocessSUA Log

The **preprocessSUA** log resides on PX2 in a time-stamped directory under SLOG_DIR. Perform the following commands as user ldad:

**TYPE:**

```
cd $LOG_DIR/<yyyymmdd>
```

Where `<yyyymmdd>` is today’s date.

**TYPE:**

```
ps -ef |grep preprocessSUA
```

Displays the **preprocessSUA** process. Notice the PID process.

**TYPE:**

```
ls -rtal preprocessSUA*
```

Displays the latest logs for the **preprocessSUA** process.

**NOTE:** The **preprocessSUA** log containing the PID of the **preprocessSUA** process is the current log.

**TYPE:**

```
tail -f <logname>
```

Displays the log as it updates.
Log Message Format

An entry at the beginning of the log looks like the following:

```
<><><><><><><><><><><><><><><><><><><><><><><><><><>
<>     New Activity Log Started Wed May 12 00:01:26 UTC 2006
<><><><><><><><><><><><><><><><><><><><><><><><><><>
ARGV[0]: /data/fxa/LDAD/tmp/SUAPITRR7PIT.dat.974678485
ARGV[1]: PX
ARGV[2]: 15009
getAfosPil():
    afos ID = PITRR7PIT
createAwipsID():
    Found PITRR7PIT in /awips/fxa/data/afos2awips.txt: extracting
    associated AWIPS id.
    awips ID = KPBZRR7PIT
sendWanMessage():
    tmpFilePath = /data/fxa/mhs/tmp/SUAPITRR7PIT.dat.974678485
createTmpWanFile(): was successful...
    WAN command is
        distributeProduct -a DEFAULTNCF KPBZRR7PIT
        /data/fxa/mhs/tmp/SUAPITRR7PIT.dat.974678485
    Successful message submission to WAN...
removeTmpWanFile() was successful! ...
getSiteCommissionStatus():
    commission status = 1
archiveObs():
    moveFile(): /data/fxa/LDAD/tmp/SUAPITRR7PIT.dat.974678485 to
    /data/fxa/archive/obs/scratch/SUAPITRR7PIT.dat
    .974678485.200060512_000131
    moveFile(): /data/fxa/LDAD/tmp/SUAPITRR7PIT.dat.974678485 moved successfully.
    Archival was successful...
Script Done ...
```

8.14.6 Reserved

8.14.7 distributeProduct

The `distributeProduct` process is a command-line interface to the `msg_send` utility program. The `distributeProduct` process supports most command line options provided by `msg_send`. The `distributeProduct` process instantiates a `MHSProduct` object that prepares the product for distribution and invokes the `msg_send` program. The subroutine `createTmpWanFile` creates and opens a temporary file. Only the product from the original ASOS/MicroART file is extracted and written into the temporary file, excluding the two-line communications header. This copy of the data is passed to the `distributeProduct` command line interface. The `distributeProduct` process prepends the appropriate communications header to the product before distribution across the WAN to the NCF.

To Access and Read the distributeProduct Log

The `distributeProduct` log resides on PX2 in a time-stamped directory under `SLOG_DIR`. Perform the following commands as user ldad:
**Chapter 8**

Local Data Acquisition and Dissemination (LDAD) System

---

**Log Message Format**

An entry at the beginning of the log looks like the following:

```
LOG-STATUS: Log file opened on host px2-tbdw at Wed May 12 01:09:57 2006
01:09:57.219 MHSProduct.C PROBLEM: Product (/data/fxa/ispan/hydro/temp/A_format.974682596) is empty.
01:09:57.312 MHSProduct.C EVENT: Sending attachments (/data/fxa/ispan/hydro/temp/A_format.974682596)
only. Attachments are assumed to contain correct communications header.
01:09:58.528 MHSProduct.C EVENT: Product request (KSTORRZLDA) was successfully submitted
to the MHS request server for distribution.
Message Attributes:
  Request ID: TBDW-32838
  To: DEFAULTNCF
  Priority: 0
  Type: Routine
  Subject:
  Handling Action: 0
01:09:58.529 MHSProduct.C EVENT: TBDW-32838
```

The format of the log consists of the following fields:

- **Time stamp**
- **File name**
- **Message type**
- **Message string**

---

**8.15 Using LDAD Scheduler**

---

**8.15.1 Using LDAD Scheduler to Modify Gauges with Session Files**

Selecting the **Modify Gauges...** button in the LDAD Scheduler will cause the **Modify Gauges** dialog box to launch. Allow a few moments for it to open because it is a large file. Within this dialog box, you can create or modify gauges (see Exhibit 8.15.1-1).
Exhibit 8.15.1-1. The Modify Gauges Dialog Box

- **Edit**: In the **Modify Gauges** dialog box, the **Edit** button opens the **Edit Gauge** dialog box, which allows you to modify information for a selected gauge. The options are as follows:
  
  o **ID**: The entry box is for the five-character gauge ID.
  o **Non-NWS ID (if any)**: The entry box is for a gauge ID that does not belong to the NWS, if it exists.
  o **Description**: Gauge location or other helpful information.
  o **Phone**: The dial-in phone number. You should check all gauges to verify their local or long distance number, including area codes. Be sure to include external access codes (i.e., some offices require that a 9 be dialed before an external phone number).
  o **Terminal Server**: Terminal server name. The terminal server connects your machine to the modems that dial into the gauges.
  o **Gauge Model**: This option menu specifies the type of gauge. Currently, the choices are “Handar,” “Campbell,” or “Sutron.” Depending on which option is selected, you may need to provide more information specific to the gauge model.
    
    - **Handar-Specific Information**: These elements will appear when you select “Handar” from the **Gauge Model** option menu.
      
      - **Collected Data**: This option menu specifies which data the gauge collects. The value is used to determine which session file is used. This option has no effect if you choose an alternate session file (the menu will display “Unspecified”). If the session file is being selected automatically, using the value
“Unspecified” has the same effect as using the value “River Stage.”

- **Campbell-Specific Information:** The element shown in Exhibit 8.15.1-2 will appear when you select “Campbell” from the **Gauge Model** option menu.
  - **Time Zone:** This option menu specifies the time zone used in data reported by the gauge.
  - **Setup...:** These menu buttons will open an **Edit Storage Area Reporting Format** dialog box.
  - **Has Second Storage Area:** Select this option if the gauge is equipped with a second storage area.

![Image of Campbell Specific Information](image)

Exhibit 8.15.1-2. Campbell-Specific Information

- **Sutron-Specific Information:** The elements shown in Exhibit 8.15.1-3 will appear when you select “Sutron” from the **Gauge Model** option menu.
  - **Time Zone:** This option menu specifies the time zone used in the data reported by the gauge.
  - **Setup...:** This menu button will open an **Edit Storage Area Reporting Format** dialog box.

![Image of Sutron Specific Information](image)

Exhibit 8.15.1-3. Sutron-Specific Information

- **Edit Storage Area Reporting Format:** This dialog box specifies the data reported by a Campbell or Sutron gauge and appears when you select the **Setup** buttons. It is shown in Exhibit 8.15.1-4 and has the following options:
  - **Array ID:** This entry box specifies the array ID associated with the storage area. This option is only present for Campbell gauges.
  - **Reported Data:** This list contains the data elements reported by the gauge. The order of elements in this list must match the
order of the reported data. The list may contain up to eight elements.

- **Add...:** This menu button will open a menu of the elements that gauges report. Select an item from the menu to add an element to the **Reported Data** list. Use the last menu item, “*(skip this field)*,” to indicate a reported element that you do not want to include in the collected data.

- **Remove:** This menu button removes the selected element from the **Reported Data** list.

- **Move up:** This menu button will move a selected list element one position.

![Exhibit 8.15.1-4. The Edit Storage Area Reporting Format Dialog Box](image)

When you select the “New” button from the Modify Gauges, the following three options would be provided.

- **Select Session File:** If this option is selected, the Scheduler will determine which session file should be user. A session file is a text based script that is used to collect data from an external user or disseminate data to an external user. Typically, the script is written in UNIX scripting language.

- **Use Alternate Session File:** Selecting this option will enable the following options and allows you to choose a specific session file. There are also options for creating and modifying session files.
Choose…: This menu button opens the Choose Session File dialog box, shown in Exhibit 8.15.1-5, and has the following options:

- **Entry Box**: You can enter a session file name in this entry box.
- **List of Sessions**: This is a list of session files. The list may not contain all valid session files, in which case you may simply enter the name in the Entry box.
- **OK**: Once you have selected a session file, this menu button closes the dialog box.
- **Cancel**: This menu button closes the dialog box without any changes.
- **Delete…**: In the Modify Gauges dialog box, this button is context sensitive. What it deletes depends on what is highlighted in the gauge list. A confirmation dialog box appears to verify your deletion.
- **Close**: This button closes the Modify Gauges dialog box.

**Editing Session Files**

In the Choose Session File dialog box, you can activate editing by double-clicking on a session file name in the Session File Entry Line. It opens an Edit Gauge dialog box with the gauge ID, description, etc. Choosing “Use Alternate Session File” opens the Edit Session File dialog box, where you create or modify a session file to acquire data from a gauge (refer to Exhibit 8.15.1-6).
Chapter 8 Local Data Acquisition and Dissemination (LDAD) System

LDAD session files are text-based files that are read by the LDAD Scheduler to acquire or disseminate data to or from a remote source. All of the gauges in the country were entered into a table prior to installation so you may only have to modify a preexisting gauge session file. To create a new gauge, refer to Chapter 9 of the AWIPS II CAVE-D2D User’s Manual.

The LDAD Scheduler program executes text-based script files that run the UNIX Expect scripting language to acquire data from gauges. Expect is a Tcl-based toolkit for automating interactive programs. For data collection, the LDAD Scheduler uses the Expect script session file to control the commands that are sent to the gauge to look for specific responses, and to capture data when it is sent. Here is a section of an Expect script used to retrieve data from a gauge:

```tcl
if { [checkForCancelFailure] == “True” } { exit }
    sleep 2
    expect {
        timeout {createTimeOutFile; exit}
        “OK” {send “atdt”}
    }
if { [checkForCancelFailure] == “True” } { exit }
    sleep 2
    expect {
        timeout {createTimeOutFile; exit}
        “BUSY” {createTimeOutFile; exit}
        “NO CARRIER” {createTimeOutFile; exit}
        “CONNECT”
    }
if { [checkForCancelFailure] == “True” } { exit }
    sleep 5
    expect {
        timeout {createTimeOutFile; exit}
        <I> {sleep 2}
    }
    set fileId [open <O> w]
```

Exhibit 8.15.1-6. The Edit Session Dialog Box with the Contents of a Session File
Any characters enclosed in angle braces (&lt;&gt;) will be replaced with values that are extracted from the ldadScheduler information for the particular gauge dialed.

1. &lt;I&gt;: When &lt;I&gt; (uppercase i) appears in an Expect clause, the name of the gauge that is being dialed is substituted for the &lt;I&gt; field.

2. atdt means “get the phone number for the gauge.” The Expect script executes the command atdt &lt;phone_number&gt; to dial the gauge.

3. &lt;O&gt; is replaced with the name of the temporary file to which the downloaded gauge data are stored.

4. &lt;T&gt; (not shown) is replaced with the Terminal Server machine name retrieved from the database.

- **Copy File...**: In the Edit Session File dialog box, this button opens the Copy File Contents dialog box from which you can select a session file and view its contents. Refer to Exhibit 8.15.1-7.

![Exhibit 8.15.1-7. The Copy File Contents Dialog Box](image)

- **OK**: Adds the selected session file and closes the Copy File Contents dialog box.
- **Cancel**: Closes the Copy File Contents dialog box without making any changes.

- **OK**: In the Edit Session File dialog box, this button acknowledges your modifications to the session file and closes the Edit Session File dialog box.

- **Cancel**: This button closes the Edit Session File dialog box without making any changes to a session file.

### 8.15.2 Using LDAD Scheduler to Add Users

The LDAD Scheduler allows system managers to add users and modify user information. Selecting the Modify Users... button on the LDAD Scheduler will cause a Modify Users
window to appear (see Exhibit 8.15.2-1). This window provides options that allow you to create and/or modify database entries for external users. External users are outside persons or groups to whom you can send LDAD data and from whom you can receive LDAD data. You need to have site information for external users, such as the IP address, site login, and password, as well as relevant file names. The **Modify Users** dialog box contains all the options needed to set up an external user.

![Modify Users Dialog Box](image)

Exhibit 8.15.2-1. The Modify Users Dialog Box

**Adding a New User**

Selecting the **New** button opens the **New User** dialog box, shown in Exhibit 8.15.2-2, which contains entry lines for the necessary information needed to access a new user. The fields, which are the same ones used when modifying a user, are as follows:
Exhibit 8.15.2-2. The New User Dialog Box

- **ID**: The name of the external user to be stored for the LDAD Scheduler.
- **Description**: Location and other useful information about the external user.
- **Type**: You must choose from this options menu whether this user is for collections or disseminations.
  - Choose **Collection** if you are gathering data from an external user.
  - Choose **Dissemination** if you are transmitting data or information to an external user.
- **Protocol**: An options menu with the list of LDAD-supported data transfer protocols (FTP, Kermit, Xmodem, Ymodem, and Zmodem). You and the external user need to both be using the same protocol.
- **Login**: If required, enter the external user’s login name on his/her remote machine (to replace “<L>” in a session file).
- **Password**: If required, enter the encrypted password for the external user on his/her remote machine (to replace “<P>” in a session file).
- **Local Directory**: Enter the directory path on your system to which data files are to be stored (to replace “<D>” in a session file).
• **Remote Directory:** Enter the directory name of the external user to which you send data files (to replace “<R>” in a session file).

• **File:** Enter the file name to be collected or disseminated (to replace “<N>” in a session file).

• **File Transfer Protocol (FTP) Machine/Terminal Server:** This option is modal based on your selected protocol.
  – For FTP protocol, this option is labeled **FTP Machine**. Enter the FTP machine name or IP address of the user’s machine.
  – For Kermit, Xmodem, Ymodem, or ZModem, this option is labeled **Terminal Server**. You should enter 7e1out for LARC gauges and 8n1out for Campbell, Sutron, and Handar 555 gauges in this entry line.
  – This entry will replace “<A>” or “<T>” in the session file, use “<A>” for the IP address and “<T>” for the Terminal Server.

• **Phone:** This option is dimmed for FTP transfers. It is activated for Kermit, Xmodem, Ymodem, or Zmodem transfers; enter the phone number of the user’s system.

• **Session File:** You can enter a session file name in this entry box or choose one from the **Choose Session File** dialog box.

There are two buttons at the bottom of the **New User** dialog box, **Choose**... and **Edit**, but only the **Choose**... button can be selected initially.

• **Choose...:** Opens the **Choose Session File** dialog box from which you can select a session file.
  – **Entry Box:** This allows you to specify a session file name in this entry box.
    o **List of Sessions:** This is a list of existing session files.
  – **OK:** Once you have selected a session file, this menu button adds the selected session file.
  – **Cancel:** This menu button closes the **Choose Session File** dialog box without making any changes.

• **Edit...:** This menu button is activated when a session file name is entered in the **Session File Entry Line**. It opens the **Edit Session File** dialog box.

This menu option is where a user creates a session file to collect data from an external user, or disseminate data to an external user using the data transfer protocol set under the **Add New External User Basic Information** window. As mentioned previously, session files are text-based files that are read by LDAD Scheduler to acquire or disseminate data to or from a remote source. Here is an example of a typical session file for collecting data from an external user using the UNIX FTP data transfer protocol. Any characters in angle braces (< >) are replaced by the LDAD Scheduler with values that are extracted from the database for the external user for whom the session file is saved.
1. \(<A>\) is replaced with the IP address of the external user’s machine.
2. \(<L>\) and \(<P>\) are replaced with the login name and password fields.
3. \(<D>\) is the local directory on the active LS.
4. \(<R>\) is the remote directory on the external user’s machine.
5. \(<N>\) is the name of the file that is to be sent (received) to (from) the external user.
   a. Set the string before the \(<N>\) character to \texttt{mget} if the session file is for data collection.
   b. Set it to \texttt{mput} if the session file is for data dissemination.

- OK acknowledges the entries in the \texttt{Edit User} dialog box and closes it.
- Cancel closes the \texttt{Edit User} dialog box without making any changes.
- Close closes the \texttt{Modify User} dialog box without making any changes.
- Submit sends a highlighted request to the active LS to process your request. This button is enabled only if a request containing at least one ID is highlighted in the \texttt{Existing Requests List}. The status and results of the request appear automatically in the \texttt{Status} dialog box shown in Exhibit 8.15.2-3.

Exhibit 8.15.2-3. The Status Dialog Box

- Close: The Close menu button closes the LDAD Scheduler dialog box.
8.16 Alaska Profiler Data

The Alaska profiler application was implemented to allow the Alaska NWS forecast offices to display their local boundary layer profiler data on AWIPS by translating the data they decode and store into the netCDF format required by AWIPS.

Processes

| LS | newLDADdataNotification |
| LS | CO_serv |
| PX2 | listener |
| PX2 | CommRouter LDAD_ROUTER |
| PX2 | DataController LDAD_ROUTER LdadController.config |
| PX2 | routerStoreAkBlpEDEX |
| DX1 | notificationServer |

Configuration Files (all in /data/fxa/LDAD/data)

- LDADinfo.txt (for listener)
- JNUStation.txt (for routerStoreAkBlpEDEX)
- LdadController.config (for DataController LDAD_ROUTER)
- LdadPatterns.txt (for routerStoreAkBlpEDEX)
- jnu.desc (for routerStoreAkBlpEDEX)

Configuration File in /awips/fxa/data

edex-info.txt (for routerStoreAkBlpEDEX)

LDADinfo.txt

This file should contain the following entry:

jnu  |JNU     | 88  | 0  |PROFILER |profiler | null |

Once the file is installed, this line should not be modified.

JNUStation.txt

The following is a sample of this file to show its format. To add a new profiler, use this format to create a new entry in the station table (* indicates a mandatory field):

- **Provider ID:** Data provider station ID (*)
- **AFOS(HB5) ID:** The AFOS or Handbook 5 ID (NWS-assigned)
- **Station Name:** Text name of station
• **Elevation:** Station elevation (m) (*)
• **Latitude:** Station Latitude (decimal degrees, north positive) (*)
• **Longitude:** Station Longitude (decimal degrees, east positive) (*)
• **Local Time Zone:** Reporting time zone (e.g., UTC, PST8PDT) (*)
• **Location Description:** Station location/address (Text)
• **Station Type:** Station type (e.g., tower, surface, floating platform)
• **Number of Instruments:** Number of reporting instruments for the station
• **Number of Reporting Levels:** Number of data reporting levels. Level information should be provided in the instrument table
• **Maintenance/Calibration Schedule:** Frequency of maintenance/calibration
• **Site Description:** Text description of the site surroundings.

**NOTE:** Entries are sorted alphabetically by the gauge ID.

**NOTE:** This file must be identical on the PX2 and on the LS machines.

**LdadController.config**

This file enables the startup of LDAD-related processes. The Data Controller reads this file and executes its contents.

```plaintext
routerStoreText       true
routerShefEncoderEDEX    true
routerStoreEDEX  true
routerStoreAkBlpEDEX     true
routerLdadDecoder     true
eof eof
```

Refer to Exhibit 8.16-1 for the Decode Alaska Profiler data flow diagram.

**8.16.1 newLDADdataNotification**

The `newLDADdataNotification` process uses the file’s name to determine the type of data it contains and checks the LDAD products directory (`/data/Incoming`) periodically for Alaska profiler files.

The `newLDADdataNotification` process creates and sends an IPC message to the `CO_serv` process, using `createAndSendIPCmessage`, to notify it that there are Alaska profiler files in the `/data/Incoming` directory.
Exhibit 8.16-1. Decode Alaska Profiler Data
If `newLDADdataNotification` fails to receive confirmation from `listener` within 5 minutes, or if an `fstat` on the file to transfer fails, the product is moved to the `/data/ldad/Problem` directory using the `moveFileToProblemDir` process. Otherwise, the “processed” files are moved to the `/data/ldad/Processed` directory using the `moveFileToProcessedDir` process.

To Access and Read the `newLDADdataNotification` Log

The `newLDADdataNotification` log resides on the active LS in the `$LOG_DIR` directory and records messages from the `newLDADdataNotification` process. Perform the following commands as user ldad:

- **TYPE:**
  ```bash
  cd $LOG_DIR/<yyyymmdd>
  ```
  Where `<yyyymmdd>` is today’s date.
- **TYPE:**
  ```bash
  ls -ltr newLDADdataNotification*
  ```
  Displays the current day’s `newLDADdataNotification` logs from earliest to latest.
- **TYPE:**
  ```bash
  tail -f <logname>
  ```
  Displays the latest registered messages as the log updates.

Log Message Format

An entry at the beginning of this log looks similar to the following:

```
LOG-STATUS: Log file opened on host ls1-tbdw at Wed May 12 00:00:28 2006
00:00:28.906 newLDADdataNotification.c EVENT: newLDADdataNotification process started
00:00:28.911 newLDADdataNotification.c EVENT: Reading /data/Incoming/ every 20 seconds
00:00:28.915 newLDADdataNotification.c EVENT: Processing:
/data/Incoming/jnund19.dat.1040157788
13:45:41.926 newLDADdataNotification.c EVENT: waitForReply SUCCESS on
/data/Incoming/jnund19.dat.1040157788
```

The format of the log file consists of the following fields:

- Time stamp: 00:00:28.915
- File name: `newLDADdataNotification.c`
- Message type: EVENT:
- Message string: Processing:
  ```
  /data/Incoming/jnund19.dat.1040157788
  ```

### 8.16.2 `CO_serv`

`CO_serv` initially sets up a TCP/IP socket and waits for connections. The notification received from `newLDADdataNotification` is passed across the firewall to the `listener` process on the internal side on the PX.
To Access and Read the CO_serv log

The CO_serv log resides on the active LS in the $LOG_DIR directory and records messages from the CO_serv process. Perform the following commands as user ldad:

**TYPE:**
```
cd $LOG_DIR/<yyyymmdd>
```
- Where `<yyyymmdd>` is today’s date.

**TYPE:**
```
ls -ltr CO_serv*
```
- Displays the current day’s CO_serv logs from earliest to latest.

**TYPE:**
```
tail -f <logname>
```
- Displays the latest registered messages as the current log updates.

### Log Message Format

An entry at the beginning of this log looks similar to the following:

```
00:00:17.076 co_serv_main.c EVENT: CO_serv process started
00:05:07.791 co_serv_main.c EVENT: CO_serv received a socket connection at:
   May12 06 00:05:07 GMT
00:05:07.801 co_serv_main.c EVENT: R  message Received by CO_serv.....
00:06:36.988 co_serv_main.c EVENT: CO_serv received a socket connection at:
   May12 06 00:06:36 GMT
00:06:36.995 co_serv_main.c EVENT: R  message Received by CO_serv.....
00:13:48.283 co_serv_main.c EVENT: CO_serv received a socket connection at:
   May12 06 00:13:48 GMT
00:13:48.291 co_serv_main.c EVENT: R  message Received by CO_serv.....
```

The format of the log file consists of the following fields:

- **Time stamp**
- **File name**
- **Message type**
- **Message string**

**8.16.3 listener**

The listener reads the SFXA_DATA/LDAD/data/LDADinfo.txt configuration file, sets up a socket, binds its local address so that the client (CO_serv) can send to it, and waits for the connection from the client process. When the notification is received, the listener retrieves the Alaska profiler file from the /data/Incoming directory on the active LS and copies it to the $LDAD_DATA/tmp directory on the PX. The listener sends a verification when finished.

The listener executes the SFXA_HOME/ldad/bin/null script to preprocess the Alaska profiler data files. The listener receives the fully qualified data file, the hostname used by
notification routine, the port number of the process to be notified when the preprocessing is completed, and other necessary information about the Alaska profiler data file format from CO_serv. Finally, the listener sends notification to the CommsRouter LDAD_ROUTER process that a product has been preprocessed.

To Access and Read the listener Log

The listener log resides on PX2 in the $LOG_DIR directory and records messages from the listener process. Perform the following commands as user ldad:

TYPE: cd $LOG_DIR/<yyyymmdd>

- Where <yyyymmdd> is today’s date.

TYPE: ls -ltr listener*

- Displays the current day’s listener logs from earliest to latest.

TYPE: tail -f <logname>

- Displays the latest registered messages as the log updates.

Log Message Format

An entry at the beginning of this log looks similar to the following:

00:00:12.616 listener.c EVENT: listener process started
20:43:10.068 listener.c EVENT: do_G_Msg: rcp completed for jnusd22.dat.104015778
</awips/ldad/bin/null /data/fxa/LDAD/tmp/jnusd22.dat.1040157788 px 15009>

The format of the log consists of the following fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time stamp</td>
<td>20:43:10:136</td>
</tr>
<tr>
<td>File name</td>
<td>listener.c</td>
</tr>
<tr>
<td>Message type</td>
<td>EVENT:</td>
</tr>
</tbody>
</table>
| Message string     | do_G_Msg: Preprocessor Command Line = 
</awips/ldad/bin/null /data/fxa/LDAD/tmp/jnusd22.dat.1040157788 px 15009> |

8.16.4 LDAD_ROUTER

The LDAD_ROUTER (registered LDAD CommsRouter) is responsible for transporting data notifications from the data acquisition processes to the data controller processes. The main function of the LDAD_ROUTER is message passing; it does no data processing. The router provides a known location where the data acquisition processes can send data messages.
To Access and Read the CommsRouter Log

The CommsRouter log resides on PX2 in a time-stamped directory under SLOG_DIR. Perform the following commands as user ldad:

**TYPE:**

```
cd $LOG_DIR/<yyyymmdd>
```

• here <yyyymmdd> is today’s date.

**TYPE:**

```
ps -wef |grep ldad
```

• Displays the ldad processes. Notice the PID of the CommsRouter process owned by user ldad.

**TYPE:**

```
ls -rtl Comms*
```

• Displays the latest logs for the CommsRouter.

**NOTE:** The CommsRouter log containing the PID of the LDAD_ROUTER process is the LDAD_ROUTER’s CommsRouter log.

**TYPE:**

```
tail -f <logname>
```

• Displays the log as it updates.

Log Message Format

An entry in the log should look similar to the following:

```
```

The log consists of the following fields:

<table>
<thead>
<tr>
<th>Time stamp</th>
<th>20:43:12.048</th>
</tr>
</thead>
<tbody>
<tr>
<td>File name</td>
<td>CommsReceiver.C</td>
</tr>
<tr>
<td>Message type</td>
<td>EVENT:</td>
</tr>
<tr>
<td>Message string</td>
<td>NCF_ENTRY DSMsg: jnusd25.dat.1040157788 1 22</td>
</tr>
</tbody>
</table>

The LDAD_ROUTER CommsRouter is started automatically at system reboot time or can be started manually. The CommsRouter runs continuously, exiting only when killed by the restart program.

8.16.5 DataController

The DataController is responsible for creating the decoder and storage processes (as child processes), restarting them if they exit, routing appropriate messages to them, and
queuing incoming data messages until the decoder and storage processes are ready. As data messages are stored in the queue, the size of the processes grows.

The DataController’s main program creates a DataRouting manager object when it first starts up and invokes a member function to start the routerStoreAkBlpEDEX process by reading and executing the LdadController.config file. The file resides in SLDAD_INTERNAL_DATA and contains the name of the process above (as well as the routerLdadDecoder, routerStoreEDEX, routerStoreTextEDEX, and routerShefEncoderEDEX processes) to start/restart, a restart flag, and end of file (eof) designator to alert the Data Routing Manager that there are no more processes listed in the file.

The DataController sends product-received notifications to the DataController. When the process first starts up, it sends messages containing its IPC address and process number to the DataController. The process sends the DataController a data request message and a ready-to-receive data message. In response, the DataController stores the process ID (PID) information and data requests, and sends a message to the requesting process (if a message is available). The DataController waits to receive another ready message from the process before sending any more data messages. The DataController queues any messages it receives from the DataController until the process sends another ready message.

To Access and Read the DataController Log

The DataController log resides on PX2 in a time-stamped directory under SLOG_DIR. Perform the following commands as user ldad:

TYPE: cd $LOG_DIR/<yyyymmdd>

Where <yyyymmdd> is today’s date.

TYPE: ps -wef |grep ldad

Displays the ldad processes; note the PID for the DataController process owned by user ldad.

TYPE: ls -rtal DataController*

Displays the latest logs for the DataControllers. Look for the log name that contains the PID you noted above.

TYPE: tail -f <logname>

Displays the log as it updates with the latest messages.

Log Message Format

An entry in the log should look similar to the following:

20:42:59.741 getCommsData.C EVENT: Data Request message sent to LDAD_ROUTER.
The log file consists of the following fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time stamp</td>
<td>20:43:12.052</td>
</tr>
<tr>
<td>File name</td>
<td>CommsReceiver.C</td>
</tr>
<tr>
<td>Message type</td>
<td>EVENT:</td>
</tr>
<tr>
<td>Message string</td>
<td>NCF_ENTRY CDMsg: jnusd25.dat.1040157788 1 22</td>
</tr>
</tbody>
</table>

### 8.16.6 routerStoreAkBlpEDEX

The `routerStoreAkBlpEDEX` process is spawned by the `DataController` as specified in the `LdadController.config` file. The `routerStoreAkBlpEDEX` process sends a data request to the `DataController` for the Alaska profiler products it wishes to receive by specifying product header patterns of interest. When notified by the `DataController` that a product of interest has arrived, the `routerLdadDecoder` reads the files contained in the `SLDAD_DATA/tmp` directory.

The `routerStoreAkBlpEDEX` process reads each file, decodes it and stores it in the `/data_store/ldad` directory. A notification is sent to the `notificationServer`.

#### To Access and Read the routerStoreAkBlpEDEX Log

The `routerStoreAkBlpEDEX` log resides on PX2 in a time-stamped directory under `SLOG_DIR`. Perform the following commands as user ldad:

**TYPE:**

```bash
cd $LOG_DIR/<yyyymmdd>
```

Where `<yyyymmdd>` is today’s date.

**TYPE:**

```bash
ps -wef | grep ldad
```

Displays the `ldad` process; note the PID of the `routerStoreAkBlpEDEX` process.

**TYPE:**

```bash
ls -rtal routerStoreAkBlpEDEX* 
```

Displays the latest `routerStoreAkBlpEDEX` logs; look for the log file containing the PID noted above.

**TYPE:**

```bash
tail -f <logname>
```

Displays the log as it updates with the latest messages.

#### Log Message Format

An entry in the log should look similar to the following:

```
20:42:49.149 routerStoreAkBlpEDEX.C EVENT: routerStoreAkBlp.C; main(), pattern <^jnu.*dat.[0-9]+$>
20:42:59.727 routerStoreAkBlpEDEX.C EVENT: eventLoop() pattern <^jnu.*dat.[0-9]+$>
```
The log file consists of the following fields:

- **Time stamp**: 20:43:12.123
- **File name**: storeAkBlpEDEX.C
- **Message type**: EVENT:
- **Message string**: AK BLP File:
  
  </data/fxa/LDAD/tmp/jnusd22.dat.1040157788>

### 8.16.7 notificationServer

The **notificationServer** informs its display clients that a new version of a display product has become available. Data acquisition processes send data notification messages to the **notificationServer**, and it acts as a bridge between the data acquisition and workstation display components of AWIPS.

#### To Access and Read the notificationServer Log

The **notificationServer** log resides on the DX1 in the $LOG_DIR$ directory. This log records messages from the **notificationServer** process. Perform the following commands as user fxa:

**TYPE:**

```
cd $LOG_DIR/<yyyymmdd>
```

- Where `<yyyymmdd>` is today’s date.

**TYPE:**

```
ls -ltr notificationServer*
```

- Displays the current day’s **notificationServer** logs from earliest to latest.

**TYPE:**

```
tail -f <logname>
```

- Displays the latest registered messages as the log updates.

### Log Message Format

An entry at the beginning of the log looks similar to the following:

```
00:00:31.822 NotificationServer.C DIAG: Tried to generate notifications for key: 3700 but the data notify time: 25.23 3HR does not match any times on the inventory.
00:00:35.948 GridAccessor1.C EVENT: Updating inventory info for MRF
```

---

Process ID of anonymous target: 29574
Host: lx2-rnk
Port number: 3231
NORMAL PRIORITY CONNECTION Connection created: Jan 18 06 16:25:32 GMT. Initiated by connecting process.
Last time this connection was used: Jan 18 06 23:55:49 GMT
Sent messages since last stat: 4 Bytes per message (Avg/Min/Max): 544/540/548 No messages have been received since last stat report.

The log file consists of the following fields:
8.17 Acquire and Process Radiosonde Replacement System (RRS) Data (for LDAD)

Data acquisition occurs through FTP over the local (site) LAN and by dial-in using a PPP connection to perform FTP file transfer of products. The `newLDADdataNotification` script checks the LDAD directory for RRS files. When a file is available, the `newLDADdataNotification` process passes the information to the `CO_serv` process. The `CO_serv` passes the message to the listener, which executes the `preprocessRRS.pl` script. The `preprocessRRS.pl` script copies the file to a `tmp` directory and stores it into the `fxatext` database using the `textdb` command line interface. When the file is successfully stored in the database, a notification is sent to the `textNotificationServer`. Finally, the data are routed to the NCF via the WAN.

Processes

- LS `newLDADdataNotification`
- LS `CO_serv`
- FX `listener`
- FX `preprocessRRS.pl`
- FX `distributeProduct`

Configuration Files (all in `/data/fxa/LDAD/data`)

- `LDADinfo.txt` (for listener)
- `rcv_action2codes.txt` (for distributeProduct)
- `siteCommission.txt` (for distributeProduct)
- `awips2afos.txt` (for preprocessRRS.pl)

Refer to Exhibit 8.17-1 for the Acquire and Process RRS data flow diagram.

8.17.1 `newLDADdataNotification`

The `newLDADdataNotification` process uses the file’s name and a lookup table (`LDADinfo.txt`) to determine the type of data the file contains and checks the LDAD products directory (`/data/Incoming`) periodically for RRS files.

The `newLDADdataNotification` process creates and sends an IPC message to the `CO_serv` process using `createAndSendIPCmessage`. The message notifies `CO_serv` that there are RRS files in the `/data/Incoming` directory.

If `newLDADdataNotification` fails to receive confirmation from `listener` within 5 minutes or if an `fstat` on the file to transfer fails, the product is moved to the `/data/ldad/Problem` directory using the `moveFileToProblemDir` process. Otherwise, the “processed” file is moved to the `/data/ldad/Processed` directory.
Exhibit 8.17-1. Acquire and Process RRS Data (for LDAD) (Page 1 of 2)
Exhibit 8.17-1. Acquire and Process RRS Data (for LDAD) (Page 2 of 2)
To Access and Read the newLDADdataNotification Log

This log resides on the active LS in the $LOG_DIR directory and records messages from the newLDADdataNotification process. Perform the following commands as user ldad:

**TYPE:**

cd $LOG_DIR/<yyyymmdd>

- Where <yyyymmdd> is today’s date.

**TYPE:**

ls -ltr newLDADdataNotification*

- Displays the current day’s newLDADdataNotification logs from earliest to latest.

**TYPE:**

tail -f <logname>

- Displays the latest registered messages as the log updates.

Log Message Format

An entry at the beginning of this log looks similar to the following:

00:00:23.202 newLDADdataNotification.c EVENT: newLDADdataNotification process started
00:00:23.205 newLDADdataNotification.c EVENT: Reading /data/Incoming/ every 20 seconds
00:14:03.204 newLDADdataNotification.c EVENT: Processing : /data/Incoming/RRSDENADRDEN0.man
00:14:03.880 newLDADdataNotification.c EVENT: waitForReply SUCCESS on /data/Incoming/RRSDENADRDEN0.man

The format of the log file consists of the following fields:

- Time stamp
- File name
- Message type
- Message string

8.17.2 CO_serv

CO_serv initially sets up a TCP/IP socket and waits for connections. The newLDADdataNotification process creates and sends an IPC message to the CO_serv using createAndSendIPCmessage routine. The message notifies CO_serv that there are processed files in the /data/Incoming directory. The notification received from newLDADdataNotification is passed to the listener.

To Access and Read the CO_serv Log

The CO_serv log resides on the active LS in the directory $LOG_DIR and records messages from the CO_serv process. Perform the following commands as user ldad:

**TYPE:**

cd $LOG_DIR/<yyyymmdd>
8.17.3 **listener**

The **listener** reads the **LDADinfo.txt** config file, sets up a socket, binds its local address so that the Client (**CO_serv**) can send to it, and waits for the connection from the Client process. When the notification is received, the **listener** retrieves the RRS file from the `/data/Incoming` directory on the active LS, copies it to the **SLDAD_DATA/tmp** directory on the PX, and sends a verification when it is finished.

The **listener** executes the **preprocessRRS.pl** script to preprocess the RRS data files. The **listener** receives the fully qualified data file, the hostname used by notification routine, the port number of the process requiring notification upon completion of the preprocessing, and other necessary information about the RRS data file format from **CO_serv**.

**To Access and Read the listener Log**

The **listener** log resides on PX2 in the directory **$LOG_DIR** and records messages from the **listener** process. Perform the following commands as user ldad:

**TYPE:**

```
cd $LOG_DIR/<yyyymmdd>
```

- Where `<yyyymmdd>` is today’s date.
Chapter 8  Local Data Acquisition and Dissemination (LDAD) System

TYPE:  
`
ls -ltr listener*
`  
- Displays the current day's listener logs from earliest to latest.

TYPE:  
`
tail -f <logname>
`  
- Displays the latest registered messages as the log updates.

Log Message Format

An entry at the beginning of this log looks similar to the following:

00:00:41.606 listener.c EVENT: listener process started
00:01:39:002 listener.c EVENT: do_G_Msg: rcp completed for
RRSDESTADRDEN0.man.1029783689
00:01:39:125 listener.c EVENT: do_G_Msg: Preprocessor Command
Line=/awips/ldad/bin/preprocessRRS.pl /data/fxa/LDAD/tmp/RRSDESTADRDEN0.man.1029783689
px 15009>

The format of the log file consists of the following fields:

<table>
<thead>
<tr>
<th>Time stamp</th>
<th>00:01:39.002</th>
</tr>
</thead>
<tbody>
<tr>
<td>File name</td>
<td>listener.c</td>
</tr>
<tr>
<td>Message type</td>
<td>EVENT:</td>
</tr>
<tr>
<td>Message string</td>
<td>do_G_Msg: rcp completed for RRSDESTADRDEN0.man.1029783689</td>
</tr>
</tbody>
</table>

8.17.4 preprocessRRS.pl

The preprocessRRS.pl script preprocesses RRS data files containing surface observations for the LDAD text decoder. These data files are pulled across the security firewall from the external side and are stored on the internal side in the SLDAD_DATA/tmp directory. The file name uses the following convention: 
RRS<station id><r><aaa><X><suffix>, where:

- **RRS** is a data type identifier corresponding to those listed in the LDADinfo.txt file
- **<station id>** is the RRS station identifier (e.g., IAD for Dulles International Airport)
- **<r>** is the release number
- **<aaa>** is the ascension number (1–9)
- **<X>** is the transmission number during flight (A–Z)
- **<suffix>** identifies the type of product and is one of the following:
  - MAN
  - SGL
  - ABV
  - FZL
  - ULG
The `preprocessRRS.pl` script calls the `textdb` command line interface to store the product into the `fxatext` database.

The script enables WAN distribution of the processed files to the NCF. After the files are successfully disseminated over the WAN, the `preprocessRRS.pl` script moves the files from the `SLDAD_DATA/tmp` directory to the `SFXA_DATA/archive/obs/scratch` directory.

### To Access and Read the `preprocessRRS` Log

The `preprocessRRS` log resides on DX2 in a time-stamped directory under `SLDAD_INTERNAL_LOGDIR`. As user `fxa`,

**TYPE:**
```
cd $LDAD_INTERNAL_LOGDIR/<yyyymmdd>
```
- Where `<yyyymmdd>` is today’s date.

**TYPE:**
```
ps -ef |grep preprocessRRS
```
- Note the `preprocessRRS` PID.

**TYPE:**
```
ls -lrtal preprocessRRS*
```
- Displays the latest `preprocessRRS` logs.

**NOTE:** The `preprocessRRS` log containing the PID of the `preprocessRRS` process is the current log.

**TYPE:**
```
tail -f <logname>
```
- Displays the log as it updates.

### Log Message Format

An entry at the beginning of the log looks similar to the following:

```plaintext
getFileExt(): Name of the file is: /data/fxa/LDAD/tmp/RRSDENADRDEN0.man
  first 2 letters of the file extension are: ma
ARGV[0]: /data/fxa/LDAD/tmp/RRSDENADRDEN0.man
getAfosPil(): afos ID = DENADRDEN
storeInTextDB(): storeCommand =
textdb -w DENADRDEN < /data/fxa/LDAD/tmp/RRSDENADRDEN0.man
  Successful storage to text database.
RouteProductFile() called on file: /data/fxa/LDAD/tmp/RRSDENADRDEN0.man
sendWanMessage();
getAfosPil(): afos ID = DENADRDEN
getFileHeader() from: RRSDENADRDEN.man
createAwipsID():
  Found match.
  awips ID = KBOUADRDEN

Preparing for WAN transmission...
AwipsId = KBOUADRDEN
wanHeader =TTAA97 KBOU 111200MANGJT
createTmpWanFile(): was successful...
```
Chapter 8 Local Data Acquisition and Dissemination (LDAD) System

WMD ID = KBOU
WAN command is distributeProduct -a DEFAULTNCF KBOUADRDEN
/data/fxa/mhs/tmp/RRSDENADRDEN0.man
sendWanMessage(): successful submission...
removeTmpWanFile() was successful...
archiveobs():
moveFile(): /data/fxa/scratch/RRSDENADRDEN0.man to
/data/fxa/archive/obs/scratch/RRSDENADRDEN0.man.20060512_133144
moveFile(): /data/fxa/scratch/RRSDENADRDEN0.man moved successfully.
Archival was successful...
Script Done ...

8.17.5 Reserved

8.17.6 distributeProduct

The distributeProduct process is a command-line interface to the msg_send utility program. The distributeProduct process supports most command line options provided by msg_send. The distributeProduct process instantiates a MHSProduct object that prepares the product for distribution and invokes the msg_send program. The createTmpWanFile subroutine creates and opens a temporary file. Only the product from the original RRS file is extracted and written into the temporary file, excluding the two-line communications header. This copy of the data is passed to the distributeProduct command line interface. The distributeProduct process prepends the appropriate communications header to the product before distribution across the WAN to the NCF.

▶ To Access and Read the distributeProduct Log

The distributeProduct log resides on PX2 in a time-stamped directory under $LOG_DIR. Perform the following commands as user ldad:

TYPE: cd $LOG_DIR/<yyyymmdd>
   ▪ Where <yyyymmdd> is today’s date.

TYPE: ls -litr distributeProduct
   ▪ Displays the latest distributeProduct logs.

TYPE: tail -f <logname>
   ▪ Displays the log as it updates.

Log Message Format

An entry at the beginning of the log looks similar to the following:

01:09:57.219 MHSProduct.C PROBLEM: Product (/data/fxa/ispan/hydro/temp/A_format.974682596) is empty.
01:09:57.312 MHSProduct.C EVENT: Sending attachments (/data/fxa/ispan/hydro/temp/A_format.974682596)
The format of the log consists of the following fields:

- **Time stamp**: 01:09:58.528
- **File name**: MHSProduct.C
- **Message type**: EVENT
- **Message string**: Product request (KSTORZLDA) was successfully submitted to the MHS request server for distribution.

8.18 **LDM Setup and Data Ingest Configuration**

LDM is a software package that is designed to select, capture, process, and distribute data products using client/server programs. The LDM software is copyrighted by UCAR in Boulder, CO.

The LDM Package, version 6.11.2, was preinstalled on the LDM systems. This package needs to be configured for local site usage; therefore the LDM package was installed and configured in a generic manner.

Currently, the LDM package is not supported by the AWIPS Prime Contractor. Sites that require assistance in configuring LDM will need to contact their Regional LDM/AWIPS focal point or monitor the ldad_x (or awipsinfo) listserve for further information.

Remember to COPY ALL changes made on LS2(LS3) to the other server since this cluster does not share hard drives. ALL changes made to one system must be copied to the other system. Below are the steps required to configure LDM.

8.18.1 **LDM Setup**

If a site needs to verify the configuration of LDM, perform the following steps:

- **TYPE**: ssh root@px2
- **TYPE**: ssh ls2
- **TYPE**: su - ldm
- **TYPE**: vi /usr/local/ldm/.bashrc

Verify that the following line exists in the file:

```
umask 002
```

Exit the file by typing: **esc :wq!**
Create the cron job for running the SCOUR cron. As user ldm perform the following:

**TYPE:**

```
vi /usr/local/ldm/etc/scour.conf
```

Edit the file for the desired scour frequency.

**TYPE:**

```
crontab -e
```

Verify the following line is in the file:

```
0 0,12 * * * /usr/local/ldm/bin/scour
```

Exit this file by typing: `esc :wq!`

**TYPE:**

```
vi /usr/local/ldm/etc/ldmadmin_pl.conf
```

(Verify fully qualified domain name. Example: `ls1-XXX.awips.noaa.gov`)

**TYPE:**

```
ldmadmin start
```

If there are no errors and LDM is running on LS2/LS3, then repeat the above steps for the second server in the Linux cluster (LS3/LS2), but do not start ingest with the `ldmadmin start` command. You can copy the modified files to the other server in order to save time. `pqact.conf` will also need to be edited to identify which data to ingest. These files are discussed later in this section.

### 8.18.2 LDM Data Ingest Configuration

This is an overview of what needs to happen for successful ingest of data provided through LDM. The regional AWIPS Focal Point (AFP) should be able to provide more detailed assistance if needed.

1. The LDM software should have been setup on the LDAD servers when the new ls2 and ls3 cluster was installed. If LDM was not setup, perform the steps in the previous section.

2. Contact the Regional AWIPS Focal Point, for the IP address of the regional LDM server and its hostname. The AFP should also have information on how to get the necessary MADIS files for the desired products – through the region or from the MADIS ftp site.

3. ADD the regional LDM server IP address and hostname to the `/etc/hosts` file on both ls2 and ls3. If you don’t add the IP address and hostname to the `/etc/hosts` file, you may not be able to connect to your regions LDM server.

4. Determine the LDM data types to ingest. The MADIS website, http://madis.noaa.gov, has information on what types of data is available.

5. Get the install instructions for the desired data (Instructions are available from MADIS or from your AFP) and configure the new product on the LDAD servers and the PX2 as required.

6. Make sure the new product is added to `/data/dfa/ldad/data/ldadinfo.txt`
7. If LDM is configured properly on the LDAD servers, and the product is also configured properly, it should be available in CAVE at this point.

8.18.3 LDM Processes and Configuration Files

**ldmadmin-pl.conf**

The configuration files for LDM are located in /usr/local/ldm/etc/. The ldmadmin utility is a Perl script intended for high level administration of an LDM system. This utility is site configurable by editing **ldmadmin-pl.conf**. The ldmadmin utility is an excellent tool for monitoring and troubleshooting LDM. Examples are provided in a later section.

The **ldmadmin-pl.conf** file defines the fully qualifying domain name of your LDAD. For the ls2 and ls3 cluster, ls1 is the name of the cluster pair, and must be used as the fully qualifying domain name.

```
[ldm@ls2-ntcb etc]$ more ldmadmin-pl.conf

# The fully-qualified domain-name of the computer system. If you have
to set
# this manually, then do the following:
# 1) Replace "your.hostname.here" in the following commented-out
#    statement
#    with the fully-qualified domain-name of the computer system;
#    2) Uncomment-out the same line by removing the leading hash (#)
#    character;
#    and
#    3) Comment-out the "chop" line by inserting a leading hash (#)
#    character.
#    chop($hostname = `uname -n`);
$hostname = "ls1-ntcb.cr.awips.noaa.gov";
```

**ldmd.conf**

The **ldmd.conf** file is a site configurable file that identifies the programs to run when LDM is started and which products to request from the regional LDM server. When configuring this file for a new LDAD product, only the request lines will be edited.

```
[ldm@ls2-ntcb etc]$ more ldmd.conf

####
# $Id: ldmd.conf,v 1.9 1998/10/07 16:51:16 rkambic Exp $
# Sample ldmd.conf for ldm5
####
#
# Programs that share a queue with rpc.ldmd
# are started by it and are in the same process group.
#
exec   "pqexpire"
exec   "pqbinstats"
exec   "pqact"
```
The main program for LDM is called `rpc.ldmd`. It establishes an internet port for incoming connections, reads the `ldmd.conf` file and; executes all programs specified in the EXEC entries; forks an asynchronous copy of itself for each REQUEST entry. The `rpc.ldmd` program places the new data products in the `product-queue`.

### 8.18.4 LDM Product-Queue

LDM uses a local data store called the product-queue, which is a memory mapped file that temporarily stores data products. The product queue typically holds about an hour’s worth of data. The product queue is located in `/usr/local/ldm/data/ldm.pq`. Use the `pqcat` command as user ldm to view the product-queue.

The `pqact` program reads products from the `product-queue` and writes them to the `/data/Incoming` directory. The `pqact.conf` is a site configurable file for the `pqact` program. The `pqact.conf` configuration file tells the `pqact` utility how to handle the data products.

```
[ldm@ls2-ntcb etc]$ more pqact.conf

####
# $Id: pqact.conf,v 1.8 1998/09/01 23:16:18 davis Exp $
# This is an example pqact.conf file,
# with some example entries commented out.
# The entries in this file control the local disposition
# of data. There is a whole chapter in the LDM Site Manager's
# Guide devoted to this file. We suggest you read and understand that
# material.
# http://www.unidata.ucar.edu/packages/ldm/ldm5/LDM5-44.html#HEADING44-0
...

# HIRESW
FSL5 (eastarw.*) FILE -overwrite /data/Incoming/
FSL5 (eastnmm.*) FILE -overwrite /data/Incoming/
FSL5 (westarw.*) FILE -overwrite /data/Incoming/
FSL5 (westnmm.*) FILE -overwrite /data/Incoming/
#
FSL5 (.*GribF.*) FILE -overwrite -close /data/Incoming/
```

### 8.18.4 LDM Product-Queue

LDM uses a local data store called the product-queue, which is a memory mapped file that temporarily stores data products. The product queue typically holds about an hour’s worth of data. The product queue is located in `/usr/local/ldm/data/ldm.pq`. Use the `pqcat` command as user ldm to view the product-queue.

The `pqact` program reads products from the `product-queue` and writes them to the `/data/Incoming` directory. The `pqact.conf` is a site configurable file for the `pqact` program. The `pqact.conf` configuration file tells the `pqact` utility how to handle the data products.

```
[ldm@ls2-ntcb etc]$ more pqact.conf

####
# $Id: pqact.conf,v 1.8 1998/09/01 23:16:18 davis Exp $
# This is an example pqact.conf file,
# with some example entries commented out.
# The entries in this file control the local disposition
# of data. There is a whole chapter in the LDM Site Manager's
# Guide devoted to this file. We suggest you read and understand that
# material.
# http://www.unidata.ucar.edu/packages/ldm/ldm5/LDM5-44.html#HEADING44-0
...

# HIRESW
FSL5 (eastarw.*) FILE -overwrite /data/Incoming/
FSL5 (eastnmm.*) FILE -overwrite /data/Incoming/
FSL5 (westarw.*) FILE -overwrite /data/Incoming/
FSL5 (westnmm.*) FILE -overwrite /data/Incoming/
#
FSL5 (.*GribF.*) FILE -overwrite -close /data/Incoming/
```
When ldm data is received by ldad, it is temporarily stored in /data/Incoming until px2 (or px1 in failover) pulls the data into AWIPS. Once the ldm file is placed into /data/Incoming it is treated like all other LDAD data types (RRS, ASOS, River Gauge, etc...).

### 8.18.5 Useful LDM Commands

```
[ldm@ls2-ntcb etc]$ ldmadmin restart
Flushing the LDM product-queue to disk...
Stopping the LDM server...
Waiting for the LDM to terminate
The product-queue is OK.
/usr/local/ldm/etc/pqact.conf is syntactically correct
Starting the LDM server...
[ldm@ls2-ntcb etc]$
```

```
[ldm@ls2-ntcb etc]$ ldmadmin pqactcheck
/usr/local/ldm/etc/pqact.conf is syntactically correct
[ldm@ls2-ntcb etc]$
```

```
[ldm@ls2-ntcb etc]$ ldmadmin check
LDM status report from the logs for the last 744 hours.
Currently ls2-ntcb is running 0 percent idle
load average: 0.36, 0.73, 0.82
Running version number Unknown.
LDM was restarted 1 time(s)
   Last LDM restart at Jul 03 14:39:00
```

```
[ldm@ls2-ntcb etc]$ ldmadmin watch
(Type ^D when finished)
Aug 04 21:00:35 pqutil INFO: 9420 20080804210034.232 FSL5 000 LDAD.aws.AWS.dat
Aug 04 21:00:36 pqutil INFO: 3121 20080804210034.204 FSL5 000 LDAD.raw.dat
Aug 04 21:01:28 pqutil INFO: 3586 20080804210128.180 FSL5 000 LDAD.raw.dat
Aug 04 21:02:13 pqutil INFO: 1026 20080804210212.386 FSL5 000 LDAD.raw.dat
```

```
[ldm@ls2-ntcb etc]$ ldmadmin config
hostname: ls2-tbw3.er.nws.noaa.gov
os: Linux
release: 2.6.18-194.17.1.el5PAE
ldmhome: /usr/local/ldm
bin path: /usr/local/ldm/bin
conf file: /usr/local/ldm/etc/ldmd.conf
log file: /usr/local/ldm/logs/ldmd.log
numlogs: 7
log_rotate: 1
data path: /usr/local/ldm/data
product queue: /usr/local/ldm/data/ldm.pq
queue size: 400M bytes
```

---

AWP.MAN.SMM.A2 8-155

– Hard copies uncontrolled. Verify effective date prior to use. –
queue slots: default
IP address: all
port: 388
PID file: /usr/local/ldm/ldmd.pid
LDMHOSTNAME: ls2-tbw3.er.nws.noaa.gov
PATH:
/usr/local/ldm/bin:/bin:/usr/bin:/usr/sbin:/sbin:/usr/ucb:/usr/usb:/usr/etc:/etc:/usr/local/ldm/decoders:/usr/local/ldm/util:/usr/local/ldm/bin:
/usr/lib/qt-3.3/bin:/usr/local/bin:/bin:/usr/bin:

[ldm@ls2-ntcb tmp]$ ldmadmin stop
Flushing the LDM product-queue to disk...
Stopping the LDM server...
Waiting for the LDM to terminate
[ldm@ls2-ntcb tmp]$ ldmadmin queuecheck
The product-queue is OK.
[ldm@ls2-ntcb tmp]$ ldmadmin start
The product-queue is OK.
/usr/local/ldm/etc/pqact.conf is syntactically correct
Starting the LDM server...

[ldm@ls2-ntcb ~]$ ldmping ldm.crh.noaa.gov
Feb 04 23:04:46 INFO: State Elapsed Port Remote_Host
rpc_stat
Feb 04 23:04:46 INFO: Resolving ldm.crh.noaa.gov to 204.227.XXX.XXX took 0.00021
seconds
Feb 04 23:04:46 INFO: RESPONDING 0.006546 388 ldm.crh.noaa.gov

8.19 LDAD and PX2

As with all data that is placed in /data/Incoming on the LDAD server, LDM files are pulled into AWIPS by PX2 (or PX1 in failover).

- When the listener process on px2 copies the data from LDAD, it places the file into /data/fxa/LDAD/tmp where it will stay until processed.
- If a required preprocessor is not available or if there is no entry in either the awips2afos.txt or afos2awips.txt, the file will be moved to /data/fxa/LDAD/Unknown.
- If a preprocessor cannot process the file, it will be moved into the /data/fxa/LDAD/Bad directory.
- If the data is successfully decoded, a copy of the decoded file is placed in the /data/fxa/LDAD/decoded directory.

The preprocessor scripts for LDM data should be placed on the PX2 and PX1 according to the instructions provided by MADIS. Some data types do not require preprocessors.

Note: Contact your regional AWIPS Focal Point to find out how to get the install instructions and the preprocessors.
8.19.1 LDADinfo.txt

The LDADinfo.txt file specifies which preprocessor to call for a particular data type. When a new type of data is added to the ldm ingest, a preprocessor may be needed on the px2 (and on px1 for failover). The data and its preprocessor must be added to the LDADinfo.txt file unless otherwise stated in the MADIS documentation for that preprocessor.

[root@px2-ntcd ~]# cd /data/fxa/LDAD/data

[root@px2-ntcd ~]# more LDADinfo.txt

<table>
<thead>
<tr>
<th>#data</th>
<th>netCDF</th>
<th>plot</th>
<th>LDAD</th>
<th>storage</th>
<th>preprocess</th>
</tr>
</thead>
<tbody>
<tr>
<td>dataKey</td>
<td>.txt</td>
<td>type</td>
<td>directory</td>
<td>script</td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>-----</td>
<td>------</td>
<td>--------</td>
<td>--------</td>
<td></td>
</tr>
<tr>
<td>SUA</td>
<td>SUA</td>
<td>256</td>
<td>257</td>
<td>CVS_TYPE</td>
<td>mesonet</td>
</tr>
<tr>
<td>RRS</td>
<td>RRS</td>
<td>256</td>
<td>256</td>
<td>CVS_TYPE</td>
<td></td>
</tr>
<tr>
<td>WAN_NWWS</td>
<td>OUP</td>
<td>256</td>
<td>257</td>
<td>CVS_TYPE</td>
<td>mesonet</td>
</tr>
<tr>
<td>mm5</td>
<td>mm5</td>
<td>87</td>
<td>86</td>
<td>CVS_TYPE</td>
<td>mesonet</td>
</tr>
<tr>
<td>testIPRC</td>
<td>testIFLOWS</td>
<td>89</td>
<td>90</td>
<td>CVS_TYPE</td>
<td>hydro</td>
</tr>
<tr>
<td>testMESONET</td>
<td>testMESONET</td>
<td>87</td>
<td>86</td>
<td>CVS_TYPE</td>
<td>mesonet</td>
</tr>
</tbody>
</table>

8.19.2 PX2 Log Files

The log files on PX2 (PX1 in failover) are in /data/logs/ldad/<date>.

[root@px2-ntcb ~]# cd /data/logs/ldad/(today’s Date)

[data@px2-ntcb 20080128]# pwd
/data/logs/ldad/20080128

[data@px2-ntcb 20080128]# ls

CommsRouter9208px2-ntcb000004 routerLdadDecoder9240px2-ntcb003003
DataController9234px2-ntcb000004 routerShefEncoder9238px2-ntcb003006
distributeProduct routerStoreNetcdf9239px2-ntcb003004
listener routerStoreText9237px2-ntcb003005
listener8645px2-ntcb000009 watchDogInternal9269px2-ntcb000015
pollForData21712px2-ntcb000015

listener log file

- If a product was successfully copied from LDAD, you will see an event “rcp completed” for the data file.
- If a preprocessor was caller for the file, then a “Preprocessor Command Line” event will follow immediately after the “rcp completed” entry. Notice in the example below that the log file indicates which preprocessor was used against the data file.
- If a preprocessor is not used, there will not be a “Preprocessor Command Line” event. This only indicates that no preprocessor was required for the data file.
routerLdadDecoder log file

- The routerLdadDecoder log file indicates whether or not a product was successfully decoded.
- In most cases, if the file could not be decoded, a reason will be given.
- When a file fails to be decoded, the log file indicates that it was moved into the /data/fxa/LDAD/bad directory
- When a file is successfully decoded, the log file indicates that a copy of the file was moved to /data/fxa/LDAD/decoded and the decoded file was sent to the LDAD_ROUTER
8.20 **LDAD Troubleshooting**

8.20.1 **LDAD Disk Removal**

Unlike other disks in AWIPS, the LDAD disks are **COLD-Swap**. Never attempt to remove a disk while the server is powered on (green light on the power button). Use the following procedure to remove a disk:

1. Login as root into the server that the disk will be removed from (LS2 or LS3).
2. Type `shutdown –h` now
3. When the server is down, power off the LS. (Either the orange light or no light on the power switch is ok.)
4. Remove the drive from the LS by pushing on the red button and pulling on the black lever.

8.20.2 **LDAD Server Disk Mirroring Procedure**

The LDAD Servers contain two 80GB SATA hard drives and are configured for Raid 1 (Full Mirror). The disks are mirrored via the Raid Controller, which can be accessed at boot time. If a site needs to replace one of the two disks due to a failure, perform the following steps to break the mirror, install the new disk (power down first) and create the mirror:

1. Connect an external monitor and keyboard to the back of the LS server.
2. Reboot the system. During the System boot (not O.S. but firmware booting) the “ILO” configuration displays prompting the user to press F8. WAIT until the boot process goes to the next device boot which will be the RAID controller.
3. At the RAID controller boot, the user will again be prompted to press “F8”. NOW, press **F8** to enter the RAID controller setup menu.
4. The user will now have two options to select from at this Main Menu (**HP Embedded SATA Raid Controller**):
   - Array Configuration Utility
   - Disk Utility
5. Select **Array Configuration Utility**
6. User will be presented with a new menu listing 4 Options:
   - Manage Arrays
   - Create Arrays
   - Add/Delete Hot Spare
   - Configure Drives
7. Select **Manage Arrays**.
8. The array 00 Mirror Raid1 74.3GB will be highlighted and options will be presented in the bottom left of the screen.

9. Press **Delete**. User will be presented with a Pop-Up window (“Array Properties”) querying the user with the following:

   - Delete
   - Cancel

10. Select **Delete** and press **Enter**.

11. User will be presented with another popup window (Yes/No): Type **y** and press return.

12. User will be presented with a new window (**Deleting Information**) with the following options:

   - [none] [member #0] [member #1] [Both]

13. Select **None** and press **Enter**.

14. User is presented with the status message **No Arrays Present** – Press any key to continue. Press the **space bar**. The Menu seen in step 6 returns.

15. Press **Esc** to exit to the Main Menu seen in step 4.

16. Press the **Esc** key to exit the configuration utility. User will be prompted with a Yes/No box to confirm exiting. Select **Yes** to exit back to the boot-up cycle of the server.

17. The server will begin to reboot – press the **Power Off** button immediately to power down the system.

18. Remove the defective hard drive and insert the replacement hard drive.

19. Power on the system.

20. Monitor the boot sequence and Press **F8** at the Raid configuration prompt.

21. Select **Array Configuration Utility** at the Main Menu.

22. Select **Add/Delete Hotspare** at the following Menu.

23. A status window indicating the drives and their status displays. Select the drive that was replaced. Drive 00 is the drive to the far left when facing the server. Drive 01 is the drive to the right. The status window will indicate the new drive by showing detailed status and disk drive specifications.

24. Command options will be displayed in the bottom left of the screen. Press the <INS> key to select the drive highlighted.

25. Press **Enter** to complete the selection.

26. User will be prompted with **Do you want to create a spare? (Yes/No): type y**

27. User will be returned to the previous menu as seen in step 5. Select **Manage Arrays**.
28. The highlighted array will be 00 mirror RAID1 74.3GB. Command options will be displayed in the bottom left of the screen.

29. Press Ctrl+R to rebuild the array.

30. The new disk is now being populated. Time to complete is approximately 45 minutes.

31. When done, the message Building the array is successfully completed displays – Press any key to continue. Press the space bar to continue.

32. Press the Esc key to exit to the Main Menu.

33. Press the Esc key to exit the configuration utility. Select/enter Yes to continue exiting. The system will now reboot. Monitor the boot process. The Raid Controller should indicate that TWO physical drives are present and that ONE logical drive is present/configured. If this is the case, then the system is configured properly.

This completes the LDAD Server Disk Mirroring Procedure/Setup.

8.20.3 Fax Modem Troubleshooting

As with all data that is placed in /data/Incoming on the LDAD server, LDM files are pulled into AWIPS by PX2 (or PX1 in failover).

If there seems to be a problem with the fax queue filling up with unsent faxes, run the faxstat command on the active LS. It's a good sign if the fax queue status is Running and idle or Initializing server or something like Sending job 13... However, it’s likely a bad sign if the fax queue status is Waiting for modem to come ready...

Sometimes even in normal operation the Waiting for modem to come ready message appears, but if the status line does not change even after several minutes, the modem could probably use a power-cycling. Switch it off, wait about 30 seconds, then switch it back on. Then run service heartbeat restart on the active LS, wait a minute or so for heartbeat to reset itself, and the faxes should start running again. Running faxstat now should give one of the acceptable messages rather than "Waiting for modem to come ready."

Check the fax modem dip switch settings (MultiTech Fax Modem):

   Switch: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
   Position: U U D U U U D D U D U D D U U D D

If switch settings were changed, power cycle the modem by turning the power switch off and then on.
Chapter 9
Background Applications
### Chapter 9. Background Applications

#### Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.1 Climatological Reports Formatters for NWS and NWWS</td>
<td>1</td>
</tr>
<tr>
<td>9.1.1 Configuration Files and Environment Variables</td>
<td>1</td>
</tr>
<tr>
<td>9.1.2 hmdb Database Tables</td>
<td>4</td>
</tr>
<tr>
<td>9.1.3 Processes</td>
<td>5</td>
</tr>
<tr>
<td>9.1.4 GUIs</td>
<td>16</td>
</tr>
<tr>
<td>9.1.5 Climatological Reports Formatter Setup</td>
<td>21</td>
</tr>
<tr>
<td>9.1.5.1 Configure Global Climate Parameters</td>
<td>21</td>
</tr>
<tr>
<td>9.1.5.2 Establishing the Historical Database</td>
<td>23</td>
</tr>
<tr>
<td>9.1.5.2.1 Entering Historical Data Manually</td>
<td>23</td>
</tr>
<tr>
<td>9.1.5.2.2 Importing the Data</td>
<td>32</td>
</tr>
<tr>
<td>9.1.5.3 Configuring Products</td>
<td>36</td>
</tr>
<tr>
<td>9.1.5.3.1 Report Type, Report Configuration, Select Stations for</td>
<td>39</td>
</tr>
<tr>
<td>Product</td>
<td></td>
</tr>
<tr>
<td>9.1.5.3.2 Weather Elements</td>
<td>40</td>
</tr>
<tr>
<td>9.2 Record Climate</td>
<td>44</td>
</tr>
<tr>
<td>9.2.1 Configuration Files and Environment Variables</td>
<td>45</td>
</tr>
<tr>
<td>9.2.2 Processes</td>
<td>46</td>
</tr>
<tr>
<td>9.3 Hourly Weather Roundup Formatter for the NOAA Weather Wire Service (NWWS)</td>
<td>57</td>
</tr>
<tr>
<td>9.3.1 Configuration Files and Environment Variables</td>
<td>58</td>
</tr>
<tr>
<td>9.3.2 Processes</td>
<td>60</td>
</tr>
<tr>
<td>9.3.3 Setting up the HWR NWWS Application</td>
<td>66</td>
</tr>
<tr>
<td>9.3.3.1 Identifying the Products to be Produced</td>
<td>66</td>
</tr>
<tr>
<td>9.3.3.2 Setting Up the HWR NWWS Application</td>
<td>66</td>
</tr>
<tr>
<td>9.3.3.3 Creating a Station List</td>
<td>69</td>
</tr>
<tr>
<td>9.3.3.4 Updating Station PILs</td>
<td>72</td>
</tr>
<tr>
<td>9.3.3.5 Treatment of Missing Data</td>
<td>74</td>
</tr>
<tr>
<td>9.4 Hourly Weather Roundup Formatter for the NOAA Weather Radio</td>
<td>75</td>
</tr>
<tr>
<td>9.4.1 Configuration Files and Environment Variables</td>
<td>76</td>
</tr>
<tr>
<td>9.4.2 Processes</td>
<td>77</td>
</tr>
<tr>
<td>9.4.3 Setting Up the HWR NWR Application</td>
<td>87</td>
</tr>
<tr>
<td>9.4.3.1 Identifying the Products to be Created</td>
<td>88</td>
</tr>
<tr>
<td>9.4.3.2 Selecting the HWR NWR GUI Options</td>
<td>88</td>
</tr>
<tr>
<td>9.4.3.3 Creating a Station List</td>
<td>91</td>
</tr>
<tr>
<td>9.4.3.4 Creating a Broadcast Format File</td>
<td>99</td>
</tr>
</tbody>
</table>
9.4.3.5 Creating a Default Format File ............................................................... 108
9.4.3.6 Providing Additional Variability for the NWR Broadcasts ............. 110
9.4.3.7 Treatment of Missing Data ................................................................. 114
9.4.3.8 Cautions .............................................................................................. 115

9.5 Aviation Forecast Preparation System ...................................................... 116
9.5.1 System Overview .................................................................................... 116
9.5.1.1 Name and Event Server ..................................................................... 116
9.5.1.2 Data Ingest Server ............................................................................ 118
9.5.1.3 Data Request Server ......................................................................... 118
9.5.1.4 Transmission Server ......................................................................... 119
9.5.1.5 Client Programs ................................................................................. 120

9.5.2 Configuring AvnFPS ................................................................................. 120
9.5.2.1 Files in etc ............................................................................................ 121
9.5.2.2 TAF Configuration Files .................................................................... 130
9.5.2.3 TWEB Configuration Files ................................................................ 138
9.5.2.4 X Resources Configuration Files ...................................................... 138
9.5.2.5 Climatological Data ........................................................................... 141

9.5.3 Starting and Stopping AvnFPS ................................................................. 142
9.5.3.1 Starting AvnFPS Servers .................................................................... 142
9.5.3.2 Stopping AvnFPS Servers .................................................................. 144

9.5.4 Avnsetup .................................................................................................. 145
9.5.4.1 Editing Monitoring Rules .................................................................... 146
9.5.4.2 Editing AvnFPS TAF Site Information ............................................... 148
9.5.4.3 Editing AvnFPS TAF Site Templates .................................................. 150
9.5.4.4 Editing AvnFPS TAF Product Definition ......................................... 151
9.5.4.5 Editing TWEB Route Information using AvnFPS ............................. 152
9.5.4.6 Editing the AvnFPS TWEB Route Template .................................... 153
9.5.4.7 Editing TWEB Product Definition using AvnFPS ............................. 154
9.5.4.8 Creating AvnFPS Database Triggers .................................................. 155
9.5.4.9 Using the AvnFPS Text Editor ............................................................ 156

9.5.5 Configuring IFPS to Send Gridded Data to AvnFPS ............................. 158

9.5.6 Logging .................................................................................................... 158

9.5.7 AvnFPS Troubleshooting ....................................................................... 160
9.5.7.1 Overall AvnFPS Server Health ........................................................... 160
9.5.7.1.1 Diagnostic Tools ........................................................................... 160
9.5.7.2 Log Files ............................................................................................. 161
9.5.7.3 Data Ingest ........................................................................................ 162
9.5.7.3.1 Data Ingest Troubleshooting ......................................................... 162
9.5.7.3.2 Specific Hints for Specific Data Types ........................................ 162
9.5.7.4 Product Transmission ...................................................................... 163
9.6 NOAA Weather Radio Editor ................................................................... 163
9.6.1 NWR Editor Setup and Configuration .................................................. 164
9.6.2 NWR Editor Event Logging ................................................................. 165
9.6.3 The NWR Editor GUI ........................................................................... 167
9.6.4 NWR Editor Application Design Overview ......................................... 170

List of Exhibits

Exhibit 9.1.3-1. Climatological Reports Formatter for NWR and NWWS (Daily) .... 7
Exhibit 9.1.3-2. Climatological Reports Formatter for NWR and NWWS (Monthly/Seasonal/Annual) ......................................................... 8
Exhibit 9.2.2-1. Record Climate Data Flow .................................................... 47
Exhibit 9.3.2-1. Hourly Weather Roundup Formatter for NOAA Weather Wire Service ..... 61
Exhibit 9.3.3.2-1. The Hourly Weather Roundup GUI ..................................... 66
Exhibit 9.4.2-1. Hourly Weather Roundup Formatter for NOAA Weather Radio ...... 78
Exhibit 9.4.3.3-1. Sample Station List for the NWR ........................................ 92
Exhibit 9.4.3.3-2. Symbolic Words for Use in Comment Lines ....................... 93
Exhibit 9.4.3.4-1. Sample Broadcast Format File for NWR .............................. 100
Exhibit 9.4.3.4-2. Symbolic Text for Use in Broadcast Format Files ................ 101
Exhibit 9.4.3.5-1. Sample Default Format File ............................................. 108
Exhibit 9.5.1-1. Data Flow Diagram .............................................................. 117
Exhibit 9.5.4-1. avnsetup Main GUI ............................................................. 145
Exhibit 9.5.4.1-1. AvnFPS Monitoring Criteria Editor ..................................... 146
Exhibit 9.5.4.2-1. AvnFPS TAF Site Info Editor ............................................ 149
Exhibit 9.5.4.4-1. AvnFPS TAF Product Configuration Editor ........................ 151
Exhibit 9.5.4.5-1. AvnFPS TWEB Route Info Editor ....................................... 152
Exhibit 9.5.4.7-1. TWEB Product Configuration Editor ................................... 154
Exhibit 9.5.4.8-1. AvnFPS Trigger Editor .................................................... 156
Exhibit 9.5.4.9-1. AvnFPS Setup Editor ....................................................... 157
Exhibit 9.5.4.9-2. AvnFPS Text Editor Dialog .............................................. 157
Exhibit 9.6.2-1. Sample NWR Editor nwrWish Log File Where debug=on ........ 166
Exhibit 9.6.2-2. Sample sendToNWR logStreamExpect Log File ..................... 167
Exhibit 9.6.3-1. NWR Editor GUI Access ................................................... 168
Exhibit 9.6.3-2. The NWR EditorGUI .......................................................... 169
Exhibit 9.6.4-1. NWR Editor Application Structure High-Level Design Overview .. 171
Exhibit 9.6.4-2. NWR Editor Application Structure – System View .................. 172
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.1.1-1</td>
<td>Environment Variables in CLIMATE</td>
<td>1</td>
</tr>
<tr>
<td>9.1.4-1</td>
<td>Climate GUIs and Corresponding Executables</td>
<td>16</td>
</tr>
<tr>
<td>9.1.5.3-1</td>
<td>Default Annual Periods</td>
<td>38</td>
</tr>
<tr>
<td>9.2.1-1</td>
<td>Environment Variables in Record Climate</td>
<td>45</td>
</tr>
<tr>
<td>9.5.1.2-1</td>
<td>avndis Threads</td>
<td>118</td>
</tr>
<tr>
<td>9.5.4.2-1</td>
<td>AvnFPS TAF Site Info Editor Fields and Descriptions</td>
<td>149</td>
</tr>
<tr>
<td>9.5.4.5-1</td>
<td>AvnFPS TWEB Route Info Editor Fields and Descriptions</td>
<td>153</td>
</tr>
<tr>
<td>9.6.1-1</td>
<td>File Format for the nwr.cfg File</td>
<td>164</td>
</tr>
<tr>
<td>9.6.1-2</td>
<td>Environment Variables for the NWR Editor</td>
<td>165</td>
</tr>
</tbody>
</table>
The AWIPS Background Applications are a set of applications that are designed to run unattended in the background after a manual initialization of data. Some of the applications can also be executed manually. Once the applications are set up, they run independently and only need to be monitored.

The AWIPS Background Applications currently consist of the following:

- Climatological Reports Formatters for the NOAA Weather Radio (NWR) and Weather Wire Service (NWWS)
- Record Climate
- Hourly Weather Roundup Formatter for NWR and NWWS
- Aviation Forecast Preparation System (AvnFPS)
- NOAA Weather Radio Editor.

The main data flow and setup of each application is described in the following sections.

### 9.1 Climatological Reports Formatters for NWS and NWWS

The morning, intermediate, and evening Climatological Reports Formatter creates daily climatological reports for AWIPS. These reports summarize one or more stations’ daily weather observations, records, normals, departures from normal, and last year’s values for that date. The Monthly, Seasonal, and Annual Climatological Reports Formatters are used to summarize one or more stations’ monthly, seasonal, or annual weather observations, records, normals, departures from normal, and last year’s value for the period. The formatters prepare tabular products to be disseminated over the NWWS and voice-ready products for NWR broadcast. The F6 formatter creates a month-to-date report containing climatological information that can be displayed on Web pages.

The Climate program provides the forecaster with the capability to set up and initialize output products; import climatology data; review and modify input observations; create, review, and modify the output values; and format the Climatological Reports.

#### 9.1.1 Configuration Files and Environment Variables

Table 9.1.1-1 shows the environment variables used in discussing the Climatological Reports Formatter. Variables are defined for the fxa user and individual user accounts.

<table>
<thead>
<tr>
<th>Environment Variable Name</th>
<th>Environment Variable Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLIMATE_DIR</td>
<td>/awips/adapt/climate</td>
</tr>
<tr>
<td>FXA_DATA</td>
<td>/data/fxa</td>
</tr>
<tr>
<td>LOG_DIR</td>
<td>/data/logs/fxa</td>
</tr>
</tbody>
</table>
Configuration Files

Most of the following files reside in SCLIMATE_DIR/data.

**clibrwse.conf**
This file contains the path and command to start the NWR Browser Graphical User Interface (GUI), the destination directory path for the NWWS climate products, and the path and command for the script to transfer the products to the NWR Browser or the Console Replacement System (CRS).

**clinwws.conf**
This file contains the path and command to transfer the climate products to the NWWS and the destination directory for the NWWS climate products.

**control_am/im/pm**
These are the default control files used by the **do_all_climate**, **create_climate**, and **display_climate** processes. These files contain the starting and ending reporting periods for the cooling, heating, and snow seasons for the corresponding morning, intermediate, and evening reports. They also contain instructions on whether to include the various daily climatological elements (e.g., maximum temperature during month, maximum wind speed during month) and their dates of occurrence, sunrise and sunset times, and the minimum and maximum daily temperatures and records, and their dates.

**control_mon/sea/ann**
These are the default control files used by the **do_all_climate**, **create_climate**, and **display_climate** processes. They contain the starting and ending reporting periods for the cooling, heating, and snow seasons, as well as instructions on whether to include the various monthly climatological elements (e.g., maximum temperature, average wind speed), their dates of occurrence, normals and departures from normal records, their dates of occurrence, and last year’s values and dates of occurrence.

**data_am/im/pm**
These files contain the date, number of observation stations, and the climatological data generated by the **create_climate** and **display_climate** processes. The **display_climate** process reads and updates these files. The files are also used by the **format_climate** process.
data_mon/sea/ann

These files contain the date, number of observation stations, and the climatological data generated by `create_climate`. The `display_climate` process both reads and updates these files. They are also used by `format_climate`.

gif and html files

These help files discuss the setup, review and editing of input values, execution of the Climate program, and the review and editing of the output products.

global_day

This file contains the variables that affect the climate application globally: the valid times for intermediate and evening summaries; NWWS product formatting options; and threshold values for temperature, precipitation, and snowfall. It is edited by `set_up_climate` and used by `create_climate`, `display_climate`, `format_climate`, and `qc_climate`.

header_am/im/pm/mo
n/sea/ann_<product
id>

This file contains the specific dissemination and transmission instructions for a specific product.

header_default

This file contains the default header information displayed in the NWR and NWWS configuration GUIs in the `set_up_climate` process. This information includes the effective, expiration, and creation date and time.

history_am/im/pm

These files contain the date, number of stations, and historical data from the `climate_day_norm` and `climate_mon_norm` data tables. These files are generated by the `create_climate` process and used by the `format_climate` process.

history_mon/sea/ann

These files contain the date, the number of stations, and the historical data from the `mon_climate_norm` data table. These files are generated by `create_climate` and used by `format_climate`.

info_am/im/pm

These files contain the non-meteorological data generated by the `create_climate` process and used by the `format_climate` process. The files contain valid time, number of stations, and the sunrise and sunset times for each station.

output_am/im/pm/mo
n/sea/ann_<product
id>.nwr/nwws

These are output files associated with a specific product generated by `format_climate`. Files with the suffix “nwr” are CRS products and files with the suffix “nwws” are NOAA Weather Wire Service products. Each output file is associated with a unique product ID.
This file contains station information such as the station name, four-character METAR station identifier, UTC offset, and latitude and longitude. This file is used by the format_climate process and links one or more stations with a specific product.

This file consists of two values – the data wait time and the product wait time.

DATA_WAIT_TIME defines the amount of time the display_climate program will wait after notifying the user to review data before continuing. If the user acknowledges the alert, the edit_daily_climate is displayed; this provides an opportunity to review and correct the data. If the user fails to acknowledge the alert, the derived parameters are calculated and stored in the database and the execution continues.

PRODUCT_WAIT_TIME defines the amount of time execution will wait after alerting the user. This provides an opportunity to review the NWR product prior to sending the output to the CRS. If the user fails to acknowledge the alert, the product is automatically sent to the CRS. If the user acknowledges the alert, the NWR browser is displayed, which enables the user to review NWR products.

### 9.1.2 hmdb Database Tables

These database tables are created or modified when Set Up Climate Runs is completed in its entirety. Refer to Section 9.1.3 for more information on initial climate setup.

**cli_asos_daily**

This holds the decoded values from ASOS Daily Summary Messages. The create_climate process checks this table before checking the METAR records when determining daily climate values.

**cli_asos_monthly**

This holds the decoded values from ASOS Monthly Summary Messages. The create_climate process checks this table before checking the METAR records when determining monthly climate values.

**cli_freezedates**

This holds the earliest, latest, normal, record, and last year’s freeze dates. It is updated by display_climate and used by create_climate and format_climate.

**cli_mon_season_yr**

This table consists of monthly, seasonal, and annual climatological data for several stations. These data are created by the create_climate program and displayed by the display_climate program. Data in this table can be viewed and edited by the qc_climate program.
cli_sta_product  This holds the climate product identifier and station identifier. It is created by set_up_climate and used by create_climate and format_climate.

cli_sta_setup  This holds station information such as the database identifier, name, and latitude and longitude. This table is established by the set_up_climate program.

climate_day_config  This consists of the product identifier and time of day. These data are created by the set_up_climate program and used by the format_climate program.

climate_period  This holds the beginning and ending years of the normals and extremes climate data period of record. This table is used by the init_climate program.

climo_dates  This holds beginning and ending dates for reporting snowfall and precipitation. It is created by set_up_climate and used by create_climate and format_climate.

daily_climate  This table consists of daily climatological data for each station. These data are created by the create_climate program and displayed by the display_climate program. Data in this table can be viewed and edited by the qc_climate program.

day_climate_norm  This holds daily historical climatology such as temperature, precipitation, snow, and heating/cooling degree day data. This table is established by the init_climate program and used by the create_climate program.

fss_report  This contains identifying information on decoded METAR and SCD reports that is used by the create_climate program.

mon_climate_norm  This holds monthly climatology such as temperature, precipitation, snow, and heating/cooling degree day normals, means, and extremes. It is established by the init_climate program and used by the create_climate program.

rpt  The MetarDecoder decodes these hourly observational records. They are used as input to the create_climate program.

station_location  This holds information about the climatological station (e.g., station identifier, elevation, name). It is installed at site start-up and used by the set_up_climate program.

9.1.3 Processes

When the Climate applications are initiated from the GUI, all processes run on the workstation.
do_all_climate

The do_all_climate process launches the Climate Master Menu. The forecaster may elect to initiate programs to configure climate parameters, set up/edit climate products, execute the climatological formatters, create the F6 product, and initialize and “quality control” the climate database tables from the Climate Master Menu. The do_all_climate program launches the scripts that initiate the forecaster-selected programs.

set_up_climate

The set_up_climate process controls configuration of the Climate program and its execution. This process enables the forecaster to choose and update the stations for which climatological summaries are produced, and to configure the output products for the morning, intermediate, and evening climate summaries. Configuring the output products includes setting or selecting the product identifiers and parameters that control the dissemination of NWR and NWWS products (e.g., periodicity, effective and expiration time, and listening area codes), the stations to be included in an individual product, and the content summary’s format and content.
Exhibit 9.1.3-1. Climatological Reports Formatter for NWR and NWWS (Daily)
Exhibit 9.1.3-2. Climatological Reports Formatter for NWR and NWWS (Monthly/Seasonal/Annual)
init_climate

This program establishes, edits, and updates the historical climate database contained in the tables `day_climate_norm` and `mon_climate_norm` in the `hmdb` database. The `init_climate` process can also be used to view the historical data after the data have been entered. The historical climate database consists of daily, monthly, seasonal, and annual normals, extreme values, and the years in which extreme values were observed. These data can be manually entered using the five GUls: Climatology Normals, Means, Extremes; the Daily Normals, Means, Extremes; the Monthly Normals, Means, Extremes; Seasonal Normals, Means, Extremes; and Annual Normals, Means, Extremes.

The morning climate program automatically updates the database when new records have been set.

qc_climate

The `qc_climate` process enables the forecaster to review and modify stored observations in the tables `daily_climate` and `cli_mon_season_yr` in the `hmdb` database. One GUI is provided for this purpose: the Edit Climatological Data GUI (refer to Section 9.1.4 for GUI information). The forecaster can also use the `qc_climate` process to fill in holes in the climate database.

build_f6

This program produces F6 products containing the climatological data for all stations for any requested month provided data are present in the daily climate database. The product includes the daily temperature, humidity, and wind extremes as well as the accumulated totals of precipitation, snow, and degree days.

The `create_f6_product` script generates the F6 products for the Climate program. It sets the paths to the given directories, changes the working directory to the temporary climate directory, executes the `build_f6` process, copies the `output_f6` files into the `data` directory, and removes all temporary files from the `tmp` directory.

The `build_f6` process controls the building of the daily F6 report. It gets the stations from the climate station table and the current local execution date, and it builds the report through the day before.

▶ To View the Current Day’s build_f6 Log

As user fxa or your individual user account on any workstation,

**TYPE:**

```
   cd $LOG_DIR/display/<display>/<yyymmdd>
```
Chapter 9 Background Applications

- Where `<yyyymmdd>` is today’s date and `<display>` is the current display variable (e.g., lx3-tbdw:0).

**TYPE:** `ls -ltr build_f6*`
- Displays the current day’s `build_f6` logs from earliest to latest.

**TYPE:** `tail -f <logname>`
- Displays the latest registered messages as the log updates.

**Log Message Format**

An entry in the log should look like the following:

```
build_f6 6387 974842165.042823 21:29:25.042 EVENT:
build element: missing value returned for SUM(snow)
for station 17890 period start 2006-05-01 period end 2006-05-11
database return code 0
build_f6 6387 974842165.074418 21:29:25.074 EVENT:
build element: missing value returned for SUM(snow)
for station 17890 period start 2006-05-01 period end 2006-05-11
database return code 0
build_f6 6387 974842165.115442 21:29:25.115 EVENT:
return multiple rows: missing value returned for max_wind_spd number 3
for station 17890 period start 2006-05-01 period end 2006-05-11
database return code 100
build_f6 6387 974842165.141399 21:29:25.141 EVENT:
return multiple rows: missing value returned for max_gust_spd number 2
for station 17890 period start 2006-05-01 period end 2006-05-11
database return code 100
build_f6 6387 974842165.141945 21:29:25.141 EVENT:
return multiple rows: missing value returned for max_gust_spd number 3
for station 17890 period start 2006-05-01 period end 2006-05-11
database return code 100
build_f6 6387 974842165.181658 21:29:25.181 EVENT:
return element dates: missing value returned for max_temp date 1
for station 17890 period start 2006-05-01 period end 2006-05-11
database return code 100
```

The log consists of the following fields:

<table>
<thead>
<tr>
<th>Time stamp</th>
<th>21:29:25.181</th>
</tr>
</thead>
<tbody>
<tr>
<td>Module name</td>
<td>build_f6 6387 974842165.181658</td>
</tr>
<tr>
<td>Message type</td>
<td>EVENT:</td>
</tr>
<tr>
<td>Message string</td>
<td>return element dates: missing value returned for max_temp date 1 for station 17890 period start 2006-05-01 period end 2006-05-11 database return code 100</td>
</tr>
</tbody>
</table>

**ASOS_smDecoder**

The `ASOS_smDecoder.sh` script executes the `ASOS_smDecoder` process, which decodes the ASOS daily or monthly summary messages for climate data. The `ASOS_smDecoder` uses the `ASOS_SumMsg` routine to parse the raw ASOS summary message into individual tokens and
move them into the text database. The ASOS_smDecoder uses the ASOS_DSMDecoder and ASOS_MSMDecoder routines to decode the tokens accordingly and uses the transfer_2db routine to transfer the decoded summary pages into the hmdb database where they can be used by the Climate program. Two lines of code must be contained in $FXA_HOME/data/localization/nationalData/adaptTrigger.template for the ASOS Daily Summary Message (DSM) and the ASOS Month Summary Message (MSM) to be decoded.

The first line of code triggers the decoding of the DSM. It is:

```
wwwDSMXX | | /awips/adapt/climate/bin/Linux/ASOS_smDecoder.sh
```

The second line triggers the decoding of the MSM. It is:

```
wwwMSMXX | | /awips/adapt/climate/bin/Linux/ASOS_smDecoder.sh
```

To View the Current Day’s ASOS_smDecoder Log

On PX1, as user fxa or your individual user account,

**TYPE:**

```
  cd $LOG_DIR/<yyyymmdd>
```

Where `<yyyymmdd>` is today’s date.

**TYPE:**

```
  ls -ltr ASOS_smDecoder*
```

Displays the current day’s ASOS_smDecoder logs from earliest to latest.

**TYPE:**

```
  tail -f <logname>
```

Displays the latest registered messages as the log updates.

Log Message Format

An entry in the log should look like the following:

```
08:14:06.686 ASOS_smDecoder.C PROBLEM:
ASOS DSM Decoder could not parse the product SFODSMEKA
Check product in the database
```

The log consists of the following fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time stamp</td>
<td>08:14:06.686</td>
</tr>
<tr>
<td>File name</td>
<td>ASOS_smDecoder.C</td>
</tr>
<tr>
<td>Message type</td>
<td>PROBLEM:</td>
</tr>
<tr>
<td>Message string</td>
<td>ASOS DSM Decoder could not parse the product SFODSMEKA Check product in the database</td>
</tr>
</tbody>
</table>
The `create_climate` process stores observations for the stations selected by the forecaster in the `set_up_climate` process. The observations are derived from the Daily Summary Message (DSM) data contained in the `cli_asos_daily` table, METAR data contained in the `rpt` table in the `hmdb` database, and from the Supplementary Climate Data (SCD) stored in the `fss_report` table. Parameters that are derived from observed variables, such as heating and cooling degree days, precipitation and snowfall to date for the month, season and year, are calculated by the `display_climate` process after the forecaster has had the opportunity to review and edit the data as necessary.

- The `create_climate` process obtains inputs as follows:
  - The type of report to be formatted via the `control_am/im/pm` and `global_day` files (for daily report) and `control_mon/sea/ann` and `global_day` files for monthly/seasonal/annual reports
  - The setup information from the `cli_sta_setup` table
  - The daily climate normals from the `day_climate_norm`
  - The monthly/seasonal/annual climate normals from the `mon_climate_norm`.

The `create_climate` process composes one listing, (data_am/im/pm file for daily and data_mon/sea/ann file for monthly/seasonal/annual), containing each station’s name and all of the element names for which the day’s observations and climatological information will be inserted and another for yesterday’s data (i.e., history file). The `create_climate` process also creates a file `info_am/im/pm` (for daily) and `info_mon/sea/ann` (for monthly/seasonal/annual) with sunrise and sunset times for each station (i.e., info_file).

To View the Current Day’s create_climate Log

On PX1 as user fxa or your individual user account,

**TYPE:**

```
cd $LOG_DIR/<yyyymmdd>
```

- Where `<yyyymmdd>` is today’s date.

**TYPE:**

```
ls -ltr create_climate*
```

- Displays the current day’s `create_climate` logs from earliest to latest.

**TYPE:**

```
tail -f <logname>
```

- Displays the log as it updates with the latest messages.
Log Message Format

An entry in the log should look like the following:

12:25:54.286 get_all_hourly_gusts.c EVENT: get_all_hourly_gusts
For station id 11300, the hourly wind gust weather element is missing for the hour 2006-05-11 05. STATUS_FAILURE returned by the database.

12:25:54.572 get_hourly_peak_winds.c EVENT: get_hourly_peak_winds
For station id 11300, the peak wind weather element is missing for hour 2006-05-11 06. STATUS_FAILURE returned by the database.
12:25:54.973 get_hourly_peak_winds.c EVENT: get_hourly_peak_winds
For station id 11300, the peak wind direction weather element is missing for hour 2006-05-11 06. STATUS_FAILURE returned by the database.

12:25:55.028 get_hourly_peak_winds.c EVENT: get_hourly_peak_winds
For station id 11300, the peak wind direction weather element is missing for this particular hour. STATUS_FAILURE returned by the database.

12:25:55.175 get_all_hourly_gusts.c EVENT: get_all_hourly_gusts
For station id 11300, the hourly wind gust weather element is missing for the hour 2006-05-11 06. STATUS_FAILURE returned by the database.
12:25:57.103 get_sky_cover.ec EVENT:
Bug in g: For station id 11300 at time 2006-05-10 19 the sky cover was not reported the fss_cloud_layer table.

The log consists of the following fields:

- Time stamp
- File name
- Message type
- Message string

display_climate

The display_climate process obtains daily climate values from the data_am/im/pm file, and monthly/seasonal/annual climate values from the data_mon/sea/ann file. A GUI displays the daily climate values for forecaster review and editing. After the forecaster has verified the accuracy, values for heating and cooling degree days, mean relative humidity, and accumulated precipitation and snowfall are calculated. This program displays the climate data generated by the create_climate process. The forecaster can edit the data as necessary. The display_climate process will calculate all derived fields and store them in the daily_climate table in the hmdb database (for daily) and cli_mon_season_yr table for the monthly/seasonal/annual.

To View the Current Day’s display_climate Log

On any workstation as user fxa or your individual user account,
TYPE: 
\[ cd \: \$LOG\_DIR/\text{\textasciitilde}display/<\text{\textasciitilde}display>/<yyyymmdd> \]
- Where \(<\text{\textasciitilde}yyyymmdd>\) is today’s date and \(<\text{\textasciitilde}display>\) is the current display variable (e.g., lx2-tbdw:0).

TYPE: 
\[ \text{ls} \: \text{-ltr} \: \text{display_{climate}*} \]
- Displays the current day’s \text{display_{climate}} logs from earliest to latest.

TYPE: 
\[ \text{tail} \: \text{-f} \: \langle\text{logname}\rangle \]
- Displays the log as it updates with the latest messages.

Log Message Format

An entry in the log should look like the following:

```
LOG-STATUS: Log file opened on host lx2-rnk at Wed May 12 16:09:05 2004
display_{climate} 19870 949507745.583942 16:09:05.583 PROBLEM: const char
ERROR: ERROR, SQLSTATE = 2300
display_{climate} 19870 949507745.759790 16:09:05.759 PROBLEM: Unique constraint
(informix.ul78_228) violated.
display_{climate}: 19870 949507745.583942 16:09:05.802 PROBLEM: const char
ERROR: ERROR, SQLSTATE = 2300
display_{climate} 19870 949507745.803177 16:09:05.803 PROBLEM: Unique constraint
(informix.ul78_228) violated.
display_{climate} 19870 949507745.826989 16:09:05.826 PROBLEM: const char
ERROR: ERROR, SQLSTATE = 2300
display_{climate} 19870 949507745.827594 16:09:05.827 PROBLEM: Unique constraint
(informix.ul78_228) violated.
```

The log consists of the following fields:

<table>
<thead>
<tr>
<th>Time stamp</th>
<th>16:09:05.583</th>
</tr>
</thead>
<tbody>
<tr>
<td>Module name</td>
<td>display_{climate}</td>
</tr>
<tr>
<td>Message type</td>
<td>PROBLEM:</td>
</tr>
<tr>
<td>Message string</td>
<td>const char</td>
</tr>
<tr>
<td></td>
<td>ERROR: ERROR, SQLSTATE = 2300</td>
</tr>
</tbody>
</table>

format_{climate} The \text{format_{climate}} process obtains the type of report to be formatted, station information, forecaster-reviewed daily observations and climatological data, and header information from the forecaster configuration. The \text{format_{climate}} process converts numerical values to ASCII text, builds phrases, adds instructions for the CRS, and creates tabular products for the NWWS. When a record has been established, information is piped to the \text{RecordClimateRawData.dat} and \text{RecordClimateStationInfo.dat} files, which are used by the \text{recordClimate} process.

The NWR output product is stored temporarily and the forecaster is notified of its completion. Occasionally, the NWR product may need to be edited before it is sent to the CRS. The forecaster can change the NWR product using the \text{NWRBrowser}. The forecaster can send the product directly to
the CRS once it has been edited and reviewed. If the forecaster does not send the product manually to the CRS via the **NWRBrowser**, it will be sent automatically after a user-specified or configuration-specified period of time. The default time is 20 minutes.

Before the NWWS product is distributed for broadcast over the WAN, it is displayed for the forecaster to view, edit, or delete in the **Review Climate GUI**. The **handleOUP.pl** script automatically sends the completed files for NWWS distribution. The NWWS product will always be automatically sent. Storing the NWWS product in the text database provides the forecaster with an opportunity to view the product without having to wait for it to arrive over the SBN.

**NOTE:** If the Climate program runs but you do not receive any alarms ("eyeglasses" icon and/or beeping) on the workstation, it may be that the **hmMonitorServer** process is down and needs to be restarted. The Climate program should notify you when the **hmMonitorServer** is down. Refer to Section 9.4.2 for more information on the **hmMonitorServer**, including how to restart it.

▶ **To View the Current Day’s format_climate Log**

On PX1, as user fxa or your individual user account,

**TYPE:**

```
  cd $LOG_DIR/<yyyymmdd>
  ls -ltr format_climate*
  tail -f <logname>
```

- Where `<yyyymmdd>` is today’s date.
- Displays the current day’s `format_climate` logs from earliest to latest.
- Displays the log as it updates with the latest messages.

**Log Message Format**

An entry in the log should look like the following:

```
12:45:15.015 c_format_climate.c EVENT: Running recordClimate from within format_climate...
12:45:15.198 c_format_climate.c EVENT: Recordclimate returned with code 0
12:45:30.290 c_format_climate.c PROBLEM: /data/logs/fxa/...
12:45:30.291 c_format_climate.c PROBLEM: debug log: /data/logs/fxa/20060614/handleOUP.log
12:45:30.290 c_format_climate.c PROBLEM: AFOSID = BOSCLIBOS
12:45:30.291 c_format_climate.c PROBLEM: logdIR: /data/logs/fxa/20060614/handleOUP.log
```
9.1.4 **GUIs**

Each GUI is associated with a particular program. Table 9.1.4-1 lists each GUI with its corresponding executable.

<table>
<thead>
<tr>
<th>GUI</th>
<th>Executable Program</th>
</tr>
</thead>
<tbody>
<tr>
<td>Climate Master</td>
<td>set_up_climate do_all_climate init_climate build_f6 qc_climate create_climate</td>
</tr>
<tr>
<td>Climate Preferences</td>
<td>set_up_climate</td>
</tr>
<tr>
<td>Edit Station List</td>
<td>set_up_climate</td>
</tr>
<tr>
<td>View/Edit Annual Periods</td>
<td>set_up_climate</td>
</tr>
<tr>
<td>View/Edit Reporting Periods</td>
<td>set_up_climate</td>
</tr>
<tr>
<td>Report Format</td>
<td>set_up_climate</td>
</tr>
<tr>
<td>Edit User Defined Values</td>
<td>set_up_climate</td>
</tr>
<tr>
<td>Current Products</td>
<td>set_up_climate</td>
</tr>
<tr>
<td>Climatology Normals, Means, Extremes</td>
<td>init_climate</td>
</tr>
<tr>
<td>Import Climate</td>
<td>init_climate</td>
</tr>
<tr>
<td>Choose File to Be Imported</td>
<td>init_climate</td>
</tr>
<tr>
<td>Import Daily or Monthly Climatology Data</td>
<td>init_climate</td>
</tr>
<tr>
<td>Column Placement</td>
<td>init_climate</td>
</tr>
<tr>
<td>Daily Normals, Means, Extremes</td>
<td>init_climate</td>
</tr>
<tr>
<td>Monthly Normals and Extremes</td>
<td>init_climate</td>
</tr>
<tr>
<td>Seasonal Normals and Extremes</td>
<td>init_climate</td>
</tr>
<tr>
<td>Annual Normals and Extremes</td>
<td>init_climate</td>
</tr>
<tr>
<td>Edit Climatological Data</td>
<td>qc_climate</td>
</tr>
<tr>
<td>Select Climate Run Date</td>
<td>create_climate</td>
</tr>
<tr>
<td>Display Station Climate Values</td>
<td>display_climate</td>
</tr>
<tr>
<td>Review Climate Product</td>
<td>format_climate</td>
</tr>
<tr>
<td>F6 Product Date</td>
<td>build_f6</td>
</tr>
<tr>
<td>Climate is Running</td>
<td>display_climate</td>
</tr>
<tr>
<td>Climate Error</td>
<td>display_climate</td>
</tr>
</tbody>
</table>

**Climate Master GUI**

This GUI is used to execute the following programs and scripts:

```
set_up_climate (set_up_climate_runs and set_up_climate_stations)
do_all_climate
init_climate
build_f6
qc_climate
  • create_climate
```
Climate Preferences GUI

This GUI, launched by pressing the **Configure Climate Parameters** pushbutton on the **Climate Master GUI**, is used to set the parameters that affect all products created by the climate application. These global variables include valid times for intermediate and evening reports, configuration of the station list, setting the season or alternate annual begin dates, setting the temperature, precipitation, and snowfall threshold values for monthly, seasonal, and annual stations. These global variables also establish some product format selections format for NWWS output products.

Edit Station List GUI

This GUI can be used to edit the Climate’s station list. It is launched by pressing the **Set Up Climate Station** pushbutton on the **Climate Preference GUI** or by pressing the **Edit Station List** button on the **Report Format GUI**. The **Edit Station List GUI** has three functions: add/delete stations; edit station names; and identify stations that observe standard time year round.

View/Edit Annual Periods GUI

This GUI is launched by the **Edit Alternate Annual Periods** pushbutton on the **Climate Preference GUI** or by pressing the **Edit Annual Periods** pushbutton on the **Report Format GUI**. This **View/Edit Annual Periods GUI** is used to define alternate annual or seasonal start dates for summing precipitation and snow observations.

View/Edit Reporting Periods GUI

This GUI is launched by pressing the **Edit Reporting Periods** pushbutton on the **Report Format GUI** and is used to determine when heating degree days, cooling degree days, and snow appear in the climate output. These data are displayed in the report only during the period specified by the user even though these statistics are always being calculated by the Climate program.

Report Format GUI

This GUI, launched by pressing the **Set Up/Edit Climate Products** pushbutton on the **Climate Master GUI**, is Climate’s hub for creating, editing and deleting products. The type of output (i.e., NWR or NWWS, daily morning, intermediate, evening, monthly, seasonal, or annual), the stations to be included, configuration parameters (i.e., effective and expiration times for NWR output), and weather elements to be included in the product are all selected in this GUI.
**Edit User-Defined Values GUI**

This GUI is launched by the **Edit User-Defined Thresholds** pushbutton on the **Climate Preference GUI** or by pressing the **Change Threshold Values** button on the **Report Format GUI**. This **Edit User-Defined Values GUI** is used to set the threshold values for maximum and minimum temperature, precipitation, and snowfall. The values that are entered will be displayed in the monthly, seasonal, or annual report.

**NOTE:** The Morning Climate product summarizes climate data from midnight to midnight local standard time (LST) of the previous day. It is determined from station data entered into the daily climate database. Evening climate summaries produce reports for the current day from midnight through either a valid time or the time that the climate program is executed (user’s preference). Data from evening climate summaries are not written to the database. Intermediate climate summaries are similar to evening runs and provide a third time of day for reporting climate information. Report Format is initiated by selecting **Set Up/Edit Climate Products** from the **Climate Master GUI**.

**Current Products GUI**

This GUI is launched from the **Option** pull-down menu in the **Report Format GUI**. It provides a list of climate products that have been created.

**Climatology Normals, Means, Extremes GUI**

This GUI is displayed by the **init_climate** program when **Initialize Climate Database** is selected from the **Climate Master GUI**. It is used to select the period during which the normals were calculated, records period, and the climatological input type (i.e., daily, monthly, seasonal, or annual) for one or more reporting stations. Forecasters may enter the climatological data in the **Daily Normals Means Extremes GUI**, **Monthly Normals And Extremes GUI**, **Seasonal Normals And Extremes GUI**, and the **Annual Normals And Extremes GUI**.

**Import Climate GUI**

This GUI is created by the **Climatology Normals, Means, Extremes GUI** run by the **init_climate** program. It is activated by selecting the **File** menu on the **Climatology Normals, Means, Extremes GUI** and selecting **Import Data Window**. This GUI allows forecasters to import daily, monthly, seasonal, or annual data files into the database.

**NOTE:** Some formats cannot be imported. Under these circumstances the forecaster must manually enter the data.
Choose File To Be Imported GUI

This GUI is created by the Import Climate GUI by pressing the Get New Data File Name pushbutton. It allows the forecaster to browse the directories for the import file.

Import Daily or Monthly Climatology Data GUI

This GUI is displayed when the user selects Arrange Order of Data from the Import Climate GUI. It allows the forecaster to arrange specific variables according to the format of the Weather Forecast Office’s (WFO) existing climatology file so it may be correctly entered into the climatology database.

Column Placement GUI

This GUI is created when the Place button is activated in the Import Daily or Monthly Climatology Data GUI. This GUI allows the forecaster to move the selected variable from Climatology Data Options to a specific column number under Order of Data in File.

Daily Normals, Means, Extremes GUI

This GUI is displayed by the init_climate program simultaneously with the Climatology Normals, Means, Extremes GUI when Initialize Climate Database is selected from the Climate Master GUI. It allows the forecaster to enter and modify a station’s daily climatological data. This capability will be most useful at the site startup time to establish the climatological information in the hmdb database.

Monthly Normals And Extremes GUI

This GUI is created by the init_climate program. It becomes visible when the forecaster selects Monthly on the Climatology Normals, Means, Extremes GUI. This GUI is used to enter and modify monthly climatological data for a station. This capability will be most useful at the site start-up time to establish the climatological information in the hmdb database.

Seasonal Normals And Extremes GUI

This GUI is created by the init_climate program. It becomes visible when the forecaster selects Seasonal on the Climatology Normals, Means, Extremes GUI. This GUI is used to enter and modify seasonal climatological data for a station. This capability will be most useful at the site start-up time to establish the climatological information in the hmdb database.
Annual Normals And Extremes GUI
This GUI is created by the init_climate program. It becomes visible when the forecaster selects Annual on the Climatology Normals, Means, Extremes GUI. This GUI is used to enter and modify annual climatological data for a station. This capability will be most useful at the site start-up time to establish the climatological information in the hmdb database.

Edit Climatological Data GUI
The qc_climate program displays this GUI once the forecaster selects Quality Control Climate Database from the Climate Master GUI. The forecaster can review and edit the daily, monthly, seasonal, or annual weather parameters in the GUI.

Select Climate Run Date GUI
This GUI is displayed once the forecaster executes a Climate run. The forecaster can default to the most recent time period for the climate product or choose a specific date for any given day, month, season, or year. When the forecaster executes an Intermediate or Evening Daily Climate Report, the only available option is to run the latest intermediate or latest evening climate.

Display Station Climate Values GUI
This GUI is used to review, edit, and accept the observation values prior to formatting. Modifications to a station’s data must be saved before going on to the next station; if not, the changes will be lost. This GUI is a part of the display_climate program.

F6 Product Date and Stations GUI
When the user selects Create F6 Products on the Climate Master GUI, this GUI is used to request an F6 product for any station in the station list for the current month up to the day prior to execution or for a chosen prior month. There is an additional selection for a printout of the output.

Climate Is Running GUI
This GUI is used to notify the user that Climate processes are running. When Climate is running and the GUIs disappear, this button shows the user that Climate is running and has not terminated.

Climate Error GUI
This GUI is used to notify the user that the Climate application encountered an error and has terminated. It allows the user to either view the log file or close the GUI.

9.1.5 Climatological Reports Formatter Setup
There are three main steps to configure the Climate program:
1. **Configure the global climate parameters.** Set product valid times, create the station list, set season dates, set the threshold values, and format the NWWS output.

2. **Establish a historical database for each station.** Enter data manually or import data files through the import data option.

3. **Configure climate products.** A product is a single output from the climate program, i.e., a means of dissemination (NWR or NWWS). The stations selected as well as the chosen variables and elements are included in the output.

To run Climate, the forecaster left clicks on the workspace window (menu), selects AWIPS and selects **Climate Reports** from the **Background WFO Apps** submenu in the **Workspace** menu. This will bring up the main **Climate Master GUI**.

### 9.1.5.1 Configure Global Climate Parameters

The Climate Formatter relies on several parameters that affect all output the application generates. These consist of the valid times for the intermediate and evening summaries, the stations in the product station list, the starting dates for seasonal climate summaries, the threshold values for temperature, precipitation and snowfall, and format options for the NWWS products. Selecting **Configure Climate Parameters** from the **Climate Master GUI** displays the **Climate Preferences GUI** used to set the parameters.

**Valid Time**

The forecaster has two options for setting the valid time of the intermediate and evening climate summaries: a specific time, or execution time of the Climate Formatter. To set the valid time to a specific time, select that option and type in a two-digit hour in local time. Select **AM** or **PM** from the option box next to the hour.

**Station List**

To set up the station list for the stations of interest, the forecaster must first select **Set Up Climate Stations** from the **Climate Preferences GUI**. The **Edit Station List GUI** is displayed and the forecaster can add stations to or delete stations from the station list using the following options:

**Adding a Station:** In the blank **Station ID** field, the forecaster types the four-character International Civil Aviation Organization (ICAO) identifier of the stations for which the WFO has responsibility. The ICAO identifiers must be typed in uppercase letters. The name of the station appears in the corresponding **Name** field. To edit the names, simply select the desired **Name** field and edit the text.

**NOTE:** You must use uppercase letters for station identifiers or there will not be a match between your choice and the data in the master stations table.

**Deleting a Station:** Simply remove the four-letter ID from the
appropriate **Station ID** field, and the associated station name is automatically erased.

**NOTE:** The information about stations is included in the cli_sta_setup table (hmdb database).

### Standard Time

Selecting the **Std Time All Year** box beside the station name indicates that the station does not switch to Daylight Savings Time for part of the year. This is important so the Climate Formatter can correctly adjust between local time and UTC.

**NOTE:** Choosing the **Save Stations** button saves any changes made to the list.

### Annual Periods

Another global parameter is the alternate annual periods defining the start of the seasonal sums for precipitation and snow. Selecting the **Edit Alternate Annual Periods** button displays the View/Edit Annual Periods GUI. The seasonal sums can be set to either **Current 3 Month Season** or an entire year beginning with a forecaster-defined date.

### Threshold Values

The threshold values for temperature, precipitation, and snowfall can be set by selecting the **Edit User-Defined Thresholds** button. The Edit User-Defined Values GUI pops up and the threshold values are entered into the fields in which they apply. After the threshold values are entered, choose **Save** to save the changes and close the GUI.

### Formatting Options

Several options are available to the forecaster for formatting the NWWS products produced by the Climate Formatter. For example, depending on the options the forecaster selects: Output text can be all uppercase or it can contain lowercase letters; negative values can be indicated with either a dash (-) or an upper case “M”; colons may or may not be used to separate phrases such as “Climate Normal Period” and “Climate Record Period” from the dates comprising the periods; and record values can be identified with an asterisk “*” or an “R.”

**NOTE:** Selecting the **Save** button saves any changes made via Climate Preferences GUI.
9.1.5.2 Establishing the Historical Database

Establishing the Historical Database is initiated when The Forecaster Selects Initialize Climate Database From The Climate Master GUI. The CLIMATOLOGY NORMALS, MEANS, EXTREMES GUI and the Daily Normals, Means, Extremes GUI are displayed. At this point, the forecaster has the option of entering the historical, daily, monthly, seasonal, and annual data manually or importing the data.

9.1.5.2.1 Entering Historical Data Manually

The forecaster may enter the historical data manually using the Climatology Normals, Means, Extremes GUI and one of the four resulting GUIs:

- Daily Normals, Means, and Extremes
- Monthly Normals and Extremes
- Seasonal Normals and Extremes
- Annual Normals and Extremes.

First, the forecaster must select the station name. The Station field provides the forecaster with the option to select the climatological station name for which data are to be immediately displayed in any of the above GUIs depending on what Climatology Input Type is selected. The station name must be double clicked for the second GUI to respond. The forecaster enters the Start and Ending year for the Normals and Records Periods in the Climatology Normals, Means, Extremes GUI.

Climatological Input Type

This field provides the option to select the climatological reports input data type template that will be displayed in the second GUI. The choices listed are as follows:

- Daily
- Monthly
- Seasonally
- Annual
Daily Climatology Files  If the historical data are of daily climatology, the **Daily** button should be selected. Next, in the **Daily Normals, Means, Extremes GUI** select the **Month** and **Day** corresponding to the file and enter the appropriate values in each field. The observation month and day can be entered in these two fields by two methods. The forecaster can either make a selection from the pull-down menus or increment these fields using the up and down arrows. After the new values for that day are entered, select **Modify Record** to go on to another day or **Add Record** if this is an original entry.

**Station.** This field displays the station name selected by the forecaster.

**Month and Day.** The observation month and day can be entered in these two fields by two methods. The forecaster can either make a selection from the pull-down menus or increment these fields using the up and down arrows.

**First.** This option displays the first record in the database that contains data for the selected station.

**Add Record.** This button saves the changes to the database only if the record has been modified for the first time.

**Modify Record.** This button saves the changes to the database only if changes have been made to a record that previously contained data.

**Delete Record.** The displayed day’s data are deleted from the **hmdb** database when this field is selected. The forecaster is prompted to confirm the deletion.

**Last.** This option displays the last record in the database that contains data for the selected station. The purpose of this button is to permit the forecaster to find the last data entry point when resuming a manual data entry session.

**Temperature.** Various climatological temperature data can be entered in the 11 GUI temperature fields:

- The **Mean Maximum Temperature** is a field for the day’s normal high temperature.
- The **Record Maximum Temperature** is a field for the highest temperature for the calendar day over the Records Period.
- The most recent year in which the record for the day occurred can be entered in the **Year 1 Record Maximum**
**Temperature Observed** field.
- When the record temperature for the day has occurred in additional years, the second most recent year and the third most recent year can be entered in the **Year 2 Record Maximum Temperature Observed** and the **Year 3 Record Maximum Temperature Observed** fields, respectively.
- The **Mean Minimum Temperature** is a field for the day’s normal minimum temperature.
- The **Record Minimum Temperature** is a field for the lowest temperature for the calendar day, over the Records Period.
- The most recent year in which the record for the day occurred can be entered in the **Year 1 Record Minimum Temperature Observed** field.
- When the record temperature for the day has occurred in additional years, the second most recent year and the third most recent year can be entered in the **Year 2 Record Minimum Temperature Observed** and the **Year 3 Record Minimum Temperature Observed** fields, respectively.
- The **Normal Mean Temperature** is a field for the day’s normal mean temperature.

**NOTES:**
1. If the **Year 1 Record Maximum Temperature Observed** is missing DO NOT enter any years for Year 2 or Year 3.
2. If the **Year 1 Record Minimum Temperature Observed** is missing DO NOT enter any years for Year 2 or Year 3.

**Heating/Cooling Degree Days.** These fields are for the normal heating degree days and cooling degree days for the day. Missing values are indicated by -9999, since it is possible to have values up to 25000.

**Precipitation.** Various climatological precipitation data can be entered in the five GUI precipitation fields.
- The **Average Precipitation** is a field for that day’s normal precipitation amount.
- The **Record Maximum Precipitation** is a field for the greatest precipitation for that calendar day over the
Records Period. For all of these fields, trace amounts are represented as T.

- The most recent year in which a record occurred can be entered into the **Year 1 Record Maximum Precipitation Observed** field.

- When the record maximum precipitation for the day has occurred in additional years, the second most recent year and the third most recent year can be entered in the **Year 2 Record Maximum Precipitation Observed** and the **Year 3 Record Maximum Precipitation Observed** fields, respectively.

**NOTE:** If the **Year 1 Record Maximum Precipitation Observed** is missing, do not enter any years for **Year 2** or **Year 3**.

**Snowfall.** The first five fields are analogous to the Precipitation fields. The sixth field, **Daily Snow on Ground (inches)**, is the snow depth for that day. For all of these fields, trace amounts are represented as T.

After the new values for that day are entered, select **Modify Record** to go on to another day or **Add Record** if this an original entry.

**Monthly Climatology Files**

If the historical data are of monthly climatology, the **Monthly** button should be selected. Next, in the **MONTHLY NORMALS AND EXTREMES GUI** select the Month corresponding to the file and enter the appropriate values in each field.

**Station.** This field displays the station name that was selected by the forecaster using the **CLIMATOLOGY NORMALS, MEANS, EXTREMES GUI**.

**Month.** The observation month can be entered in this field by two methods. The forecaster can either make a selection from the pull-down menus or increment the field using the up and down arrows.

**First.** This GUI option displays the data for the first day entered.

**Add Record.** This field adds data to the **hmdb** database for a day which never had any data entered before.

**Modify Record.** This GUI option provides the capability to
modify the data currently displayed for any field in the GUI and sequentially saves the data to the hmdb database.

**Delete Record.** The displayed day’s data are deleted from the hmdb database when this field is selected. The forecaster is prompted to confirm the deletion.

**Last.** When this GUI option is selected, the last day’s data entered are displayed. The purpose of this button is to permit the forecaster to find the last entry point when resuming a manual data entry session.

**Temperature.** The temperature fields are divided into a Normals and a Records section.

- The normal maximum monthly temperature can be entered into the first field, the **Maximum Temperature** \( (^\circ \text{F}) \) field. This value represents the average of the highest monthly temperature in each year of the Normals Period.
- The normal average maximum monthly temperature can be entered into the **Average Maximum Temperature** \( (^\circ \text{F}) \) field.
- The normal number of days per month when daily maximum temperatures are greater than or equal to 90 degrees Fahrenheit can be entered into the **Days with Maximum Temperature GE 90\(^\circ\text{F}\)** field.
- The normal number of days per month when daily maximum temperatures are less than or equal to 32 degrees Fahrenheit can be entered into the **Days with Maximum Temperature LE 32\(^\circ\text{F}\)** field.
- The normal minimum monthly temperature can be entered into the **Minimum Temperature** \( (^\circ \text{F}) \) field. This value represents the average of the lowest monthly temperature in each year of the Normals period.
- The normal average minimum monthly temperature can be entered into the **Average Minimum Temperature** \( (^\circ \text{F}) \) field.
- The normal number of days per month when daily minimum temperatures are less than or equal to 32 degrees Fahrenheit can be entered into the **Days with Minimum Temperature LE 32\(^\circ\text{F}\)** field.
- The normal number of days per month when daily minimum temperatures are less than or equal to 0 degrees Fahrenheit can be entered into the **Days with Minimum Temperature LE 0\(^\circ\text{F}\)** field.
Temperature LE 0°F field.

- The normal monthly mean temperature can be entered into the Mean Temperature (°F) field.

Additional climatological temperature data are calculated by the Climate program from the daily database. The following GUI Records section temperature fields can be viewed but not edited:

- The first temperature field in the Records section is the Maximum Temperature (°F) field which is for the record highest temperature in that month.
- The Date 1 Maximum Temperature Observed field is for the most recent day and year in which the record maximum temperature was observed.
- If the record was tied in additional years, the second most recent day and year can be entered into the Date 2 Maximum Temperature Observed field, and the third most recent day and year can be entered into the Date 3 Maximum Temperature Observed field.
- The next temperature field in the Records section is the Minimum Temperature (°F) field in which the record minimum temperature in that month can be entered.
- The most recent year in which the record minimum temperature was observed can be entered into the Date 1 Minimum Temperature Observed field.
- If the record was tied in additional years, the second most recent year can be entered into the Date 2 Minimum Temperature Observed field, and the third most recent year can be entered into the Date 3 Minimum Temperature Observed field.

Precipitation. The Precipitation fields are divided into a Normals section and a Records section.

In the Normals section:

- The first field is the Total Precipitation (inches) field in which the normal total monthly precipitation can be entered.
- The normal average amount of precipitation for a day in that month can be entered into the Daily Average Precipitation (inches) field.
- The normal number of days in the month with precipitation greater than or equal to 0.01 inches can be
entered into the Days with Precipitation GE 0.01 Inches field.

The normal number of days in the month with precipitation greater than or equal to 0.10 inches can be entered into the Days with Precipitation GE 0.10 Inches field.

The normal number of days in the month with precipitation greater than or equal to 0.50 inches can be entered into the Days with Precipitation GE 0.50 Inches field.

The normal number of days in the month with precipitation greater than or equal to 1.00 inch can be entered into the Days with Precipitation GE 1.00 Inch field.

In the Records section:

The first precipitation field is the Maximum Total Precipitation (inches) field in which the record maximum monthly precipitation amount can be entered. The most recent year in which the record maximum monthly precipitation was observed can be entered into the Date 1 Maximum Precipitation Observed field. When the record has occurred in additional years, the second most recent year can be entered into the Date 2 Maximum Precipitation Observed field, and the third most recent can be entered into the Date 3 Maximum Precipitation Observed field.

The next precipitation field is the Minimum Total Precipitation (inches) field in which the record minimum monthly precipitation amount can be entered. The most recent year in which the record minimum monthly precipitation was observed can be entered into the Date 1 Minimum PrecipitationObserved field. When the record has occurred in additional years, the second most recent year can be entered into the Date 2 Minimum Precipitation Observed field, and the third most recent can be entered into Date 3 Minimum Precipitation Observed field.

Snow. The snow fields are also divided into a Normals section and a Records section.

In the Normals section:

The first field is the Total Snowfall (inches) field in
which the normal total monthly snowfall can be entered. The water equivalent of the normal total monthly snow that falls can be entered into the **Total Water Equivalent (inches)** field.

The normal amount of snow on the ground for any day in the month can be entered into the **Snow Depth (inches)** field.

The normal number of days on which snow occurs in the month can be entered into the **Days with Any Snowfall** field.

The normal number of days on which snowfall is equal to or greater than 1 inch can be entered into the **Days with Snowfall GE 1.0 inch**.

In the **Records** section:

The first field is the **Total Snowfall (inches)** field in which the record maximum total amount of snow that has ever fallen during that month can be entered.

The most recent year in which this record occurred can be entered into the **Date 1 Total Snowfall Observed** field.

When this record has occurred in additional years, the second most recent year and third most recent year can be entered into the **Date 2 Total Snowfall Observed** and the **Date 3 Total Snowfall Observed** fields, respectively.

The record maximum 24-hour snowfall can be entered into the **Total 24-Hour Snowfall (inches)** field.

The beginning day and month of the 24-hour record snowfall event can be entered into the **Date 1: Begin Date** field and the ending day, month, and year of the event can be entered into the **End Date and Year** fields.

When the record total 24-hour snowfall has occurred in additional years, the second most recent year’s and the third most recent year’s event dates can be entered into the **Date 2: and the Date 3: fields**, respectively.

The record snow depth at any time during the month can be entered into the **Snow Depth (inches)** field.

The date of the most recent event can be entered into the **Date 1 Snow Depth Observed** field.

When the record snow depth has occurred in additional years, the second most recent year’s and the third most
recent year’s event dates can be entered into the **Date 2 Snow Depth Observed** and the **Date 3 Snow Depth Observed** fields, respectively.

After the new values for the selected month are entered, select **Modify Record** and go on to another month, or **Add Record** if this is an original entry.

**Degree Days.** The degree days fields are divided into the Degree Days Normals, Freeze Dates Normals, and Records sections. The first field in the Degree Days Normals section is **Heating Degree Days** in which the normal number heating degree days for the month can be entered. The last field in this section is the **Cooling Degree Days** in which the normal number of cooling degree days for the month can be entered.

The first field in the Freeze Dates Normals section is the **Early Freeze Date.** The normal date of the earliest freeze can be entered in this field. The last field in this section is the **Late Freeze Date.** The normal date of the latest freeze can be entered into this field.

The first field in the Records section is the **Early Freeze Date.** The date of the record earliest freeze can be entered into this field. The last field is the **Late Freeze Date.** The date of the record latest freeze can be entered into this field.

**Seasonal Climatology Files**

If the historical data are for seasonal climatology, the **Seasonal** button should be selected in the **Climatology Normals, Means, Extremes GUI**. Next, in the **Seasonal Normals And Extremes GUI** use the **Month** button to select the 3-month season (DJF = December, January, February) corresponding to the file and enter the appropriate values in each field. After the new values for the selected month are entered, select **Modify Record** and go on to another season, or **Add Record** if this is an original entry. Fields are similar to the monthly ones.
Chapter 9  

Annual Climatology Files  

If the historical data are of annual climatology, the Annual button should be selected in the Climatology Normals, Means, Extremes GUI. Next, select the Start Year and Ending Year corresponding to the file and enter the appropriate values in each field in the Annual Normals And Extremes GUI. After the new values for the selected month are entered, select Modify Record, or Add Record if this is an original entry. Fields are similar to the monthly ones.

When all the data are entered, select Close in the Climatology Normals, Means, Extremes GUI.

NOTE: For all of the Climatology Files, the changes will be lost if Modify Record or Add Record is not selected before going on to another day, month, season, or year.

9.1.5.2.2 Importing the Data

The forecaster may import the historical data by selecting Import Data Window within the File menu in the Climatology Normals, Means, Extremes GUI. The Import Climate GUI launches and the forecaster may begin the process of importing files.

Station Name  

This field displays the stations currently in the climate database.

Station Name(s) in Header(s)  

This button allows the forecaster to specify that the station name(s) appears in the header(s) of the import file. This is useful when the file contains more than one station’s data separated by headers.

Station Name(s) in Data  

This button allows the forecaster to specify that the station name(s) appears in the data of the import file.

Enter Data Delimiters  

This field allows the forecaster to insert any delimiter that appears in the imported climatology file. The delimiter defaults are “*,” “space,” and “/.” If any other delimiter is used, the forecaster must enter the appropriate character(s) into the field. There is no specific format for this; if entering more than one delimiter, neither spaces nor commas are needed.
### File Name
This field displays the name of the import file. The file name will only appear after the forecaster goes through the process of **Getting a New Data File Name**. This button launches the **Choose File To Be Imported GUI** which allows the forecaster to browse the directories for the import file.

### Data Type
The **Daily** button and the **Monthly** button specify whether the import file is daily climatology or monthly climatology.

### Date Format
This field lists the months of January through December and allows the forecaster to select what month(s) to which the file(s) pertains.

### Month Name(s) in Header(s)
This button allows the forecaster to specify if the month name(s) appears in the header(s). This is useful when a file contains more than one month’s data separated by headers. The program searches for a match using the first three letters of the month. Consequently, abbreviations will be accepted.

### Month Located in Date Field
This button allows the forecaster to specify whether the month name is located in the data field. If so, the forecaster must select **Month First** if the format for the date places the month first, **Day First** if the format for the date places the day first, **Julian Days** if the date is recorded in Julian days, or **Julian Days (Feb29=60)** if the date is recorded in Julian days and the 29th of February is denoted as the 60th day. If you use **Julian Days**, the date field for the first 9 days of the year must be in the 01–09 format for the data to be successfully imported.

### Arrange Order of Data
This button allows the forecaster to specify the format of each data file. Once activated, this button will bring up the **Import Daily or Monthly Climatology Data GUI** where the forecaster arranges the climatology data according to the order of the data in the import file. **NOTE**: The format must be arranged before the file can be imported.

### Add File to List
This button adds the specified file to the pending list on the right. The specific file name, station, data type, data format and delimiter values become visible.

### Save New Values for File
This button saves any format changes made to the highlighted file located in the pending list.

### Delete from List
This button deletes the highlighted file from the import list.
Import Data from File(s)  This button imports the data from all of the files contained in the import list.

NOTE: To view the imported data the forecaster must double-click on the Station ID in the CLIMATOLOGY NORMALS, MEANS, EXTREMES GUI. The data will appear in the appropriate fields within the DAILY/MONTHLY NORMALS, MEANS, EXTREMES GUIs.

Cancel  The Cancel button closes the Import Climate GUI without saving any changes. The same effect is achieved by toggling off the Import Climate Window option in the File menu of the CLIMATOLOGY NORMALS, MEANS, EXTREMES GUI.

The file formats supported by the Import function are described in Appendix M. Appendix M also describes the import procedure in more detail.

NOTE: It is helpful to have the file you are importing open in a separate window when you do this.

Retrieving Files

To get the file, the forecaster must select Get New Data File Name, which brings up the Choose File To Be Imported GUI and allows the forecaster to browse for the import file. Once a file name is found and selected, the OK button should be clicked and the file name will appear in the field below File Name.

File Type  The forecaster must either select the Daily button if the data file is of daily climatology or the Monthly button if the data file is of monthly climatology. The forecaster must select the station(s) that is (are) included in the file and select Station Name(s) in Header(s). If the Station Name(s) appear in the data, the corresponding button should be selected. The Station Names do not need to be manually selected for this option. If the import file contains delimiters other than the indicated defaults, they need to be entered in the field provided; otherwise, the field may be left blank. The next step is to select the month(s) included in the file and select Month Name(s) in Header(s). If the Month is located in the data field the forecaster should select Month(s) located in data field specify how the month appears in the file.

NOTE: If the month(s) are located in the data field, the months do not need to be manually selected for this option.
Configuring the File

Next, the forecaster should select **Arrange Order of Data** which brings up the **Import Daily** or **Monthly Climatology Data GUI** described as follows:

**Climatology Data Options**
Includes a list of variables that may be contained in the WFO’s daily or monthly climatology, dependent upon the format chosen for the file (daily or monthly). The file must contain a column with the Date, although the date may be “month/day,” “day/month,” or Julian date.

**Order of Data in File**
Once a data element is selected in the **Climatology Data Options** list, it is displayed with its associated column number in this field. An element may be selected by double-clicking the left mouse button or by highlighting the element and single-clicking the **Append** button.

**Append**
Moves selected element from **Climatology Data Options** to **Order of Data in File**. Append can also be activated by double-clicking the left mouse button.

**Place**
Opens the **Column Placement GUI** and allows the forecaster to move the selected element from **Climatology Data Options** to a specific column number in the **Order of Data in File** field. This is helpful if the forecaster leaves an element out in the initial configuration, because that element can be added without starting over. The element placed will not remove any other elements, but will push the remaining list down one column.

**Append All**
Moves all of the remaining elements in **Climatology Data Options** to the **Order of Data in File** field. The elements will keep the order in which they appear under **Climatology Data Options**. “Other data not listed” will not move unless directly selected and appended.

**Remove**
Takes the selected element in the **Order of Data in File** and places it back in the **Climatology Data Options** field shifting the list up one place. If no element is selected, the element listed in column 1 is the default and will be removed. **Remove** can also be activated by double-clicking the left mouse button.

**Remove All**
Selects all of the elements in the **Order of Data in File** field and places them in the **Climatology Data Options** field.

**Accept**
Saves the changes and returns the forecaster to the **Import Climate GUI**.
Cancel

Closes the GUI without saving the values and returns forecaster to the Import Climate GUI.

Listing the File Name

To list the file that was configured, select Add File to List and the file name will appear in the pending list to the right. If additional files need to be imported, the forecaster may do so at this time by repeating the procedures above.

Once all the files in the pending list are ready to be imported, the forecaster needs to select Import Data from File(s). The Import Climate GUI will close and return the forecaster to the CLIMATOLOGY NORMALS, MEANS, EXTREMES GUI and the DAILY NORMALS, MEANS, EXTREMES GUI. Selecting Close will return the user to the Climate Master GUI.

Notes:

1. After you have imported the data, double-click on the station name in the CLIMATOLOGY NORMALS, MEANS, EXTREMES GUI. This fetches the data you have just imported from the database and allows you to review the results.
2. When new records are set, they are automatically updated in the database.

9.1.5.3 Configuring Products

To configure Climate products, choose Set Up/Edit Climate Products from the Climate Master GUI. (NOTE: If Set Up/Edit Climate Products does not execute on the text workstation, try using the graphics workstation.) The Report Format GUI is displayed. It is divided into four sections for product manipulation:

- Report Type
- Report Configuration
- Select Stations for Product
- Weather Elements

► To Create a New Product

Select: File from the top-line menu
Select: New Product from the pull-down menu
► Initializes the fields on the Report Format GUI to default values.

► To View Existing Products

Select: File from the top-line menu
Select: Open Existing Product

Submenus allow selection of any existing products by time period (Daily [morning, intermediate, evening]; Monthly, Seasonal, Annual).

The other top-line menu items either provide the means to display additional information or choices other than the buttons.

The top-line menu option Edit provides another means of editing the Reporting Periods, Annual Periods, Threshold Values, and Station list.

Also in this menu are Select All Elements and Deselect All Elements which select and deselect all of the weather elements, respectively.

Under Options, choosing Current Products Window causes a small window to appear which lists the products that have already been created and what type of report they produce. Double clicking the product ID will display that product’s information in the Report Format GUI.

Selecting the buttons performs many of the same functions as the menu options.

Saving a product, deleting a product, and closing Report Format can all be done via the buttons.

Also, buttons are provided for editing the annual periods, the station list (refer to Section 9.1.5.1), and the reporting periods.

When the Edit Reporting Periods button is selected, the View/Edit Reporting Periods GUI is displayed. Reporting periods control when the heating and cooling degree days and snow sections appear in the Climate output.

When the Edit Annual Periods button is pressed, the View/Edit Alternate Annual Periods is displayed. The forecaster can use defaults or change the annual reporting period for snow and liquid precipitation.

Reporting Periods

These three fields are for the entry of the snow, heating degree days (HDD), and cooling degree days (CDD) reporting periods. The month can be selected by clicking on the Month field and choosing from the pull-down menu. The forecaster can either directly type the Day or can increment/decrement using the up and down arrows to select the day.

Reporting Periods define the time period in which snow, heating degree days, and cooling degree days are reported.

Alternate Annual Period (Season)

These two fields are for the entry of the beginning month and day of the seasonal periods. The month can be selected by clicking on the Month field and choosing from the pull-
down menu. The forecaster can either directly type the Day or can increment/decrement the day using the up and down arrows. This will create an alternate annual period. Choosing the “Current 3 Month Season” results in a 3-month summary for the season to date total.

**Definition of Annual Periods**

**Fixed.** These parameters cannot be changed and will always be the dates for the ‘year to date’ (YTD).

<table>
<thead>
<tr>
<th>HDD</th>
<th>1 July - 30 June</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDD</td>
<td>1 January - 31 December</td>
</tr>
<tr>
<td>Snow</td>
<td>1 July - 30 June</td>
</tr>
</tbody>
</table>

**Default.** These parameters can be changed and will be seen in the Alternate Annual Period box.

<table>
<thead>
<tr>
<th>Precip</th>
<th>1 January - 31 December</th>
</tr>
</thead>
<tbody>
<tr>
<td>Snow</td>
<td>1 July - 30 June</td>
</tr>
</tbody>
</table>

The forecaster can create alternate annual periods for snow and precipitation using the Alternate Annual Periods box. These changes will appear in the NWWS products and are controlled by the Season to Date (STD) flag. The output header line will contain the words “Since 1 Aug” where 1 August is the start of the alternate year defined by the user.

If the forecaster creates an alternate annual period, it will be used in place of the normal seasonal values. That is to say, the “STD” flags will control the output of the alternate annual period. The “starting” date of the alternate annual period will be included in the NWWS output to make it clearer to the users.

If the forecaster does not change the default annual periods, the seasonal output for these variables will be for seasons defined in Table 9.1.5.3-1.

**Table 9.1.5.3-1. Default Annual Periods**

<table>
<thead>
<tr>
<th>Season</th>
<th>Months</th>
</tr>
</thead>
<tbody>
<tr>
<td>Winter</td>
<td>Dec, Jan, Feb</td>
</tr>
<tr>
<td>Spring</td>
<td>Mar, Apr, May</td>
</tr>
<tr>
<td>Summer</td>
<td>Jun, Jul, Aug</td>
</tr>
<tr>
<td>Fall</td>
<td>Sep, Oct, Nov</td>
</tr>
</tbody>
</table>

**NOTE:** The forecaster cannot change this field; it is for information purposes only.
9.1.5.3.1  **Report Type, Report Configuration, Select Stations for Product**

**NOTE:** Morning reports produce climatology for the previous day. Intermediate and evening reports produce climatology for the current day.

- **To Configure NWR**

  In the **Report Type** area, select the **NWR** button.

  In the **Report Configuration**, enter a three-character **Product ID**, usually the station’s three-character METAR ID.

  **NOTE:** The product ID must be unique amongst morning or evening products. The same product ID can be used for a morning and an evening product. For example, DCA can be chosen for an NWR morning product and an NWR evening product; however, DCA cannot be chosen for both NWR morning and NWWS morning products.

Enter the following information:

- **Periodicity**: The number of minutes in between broadcasts of the Daily Climatological Report
- **Effective Time**: The earliest allowable broadcast time of the Daily Climatological Report
- **Expiration Time**: The latest allowable broadcast time of the Daily Climatological Report
- **Listening Area Code (LAC)**: Notice that the LAC should be terminated by a “c.” The program will inform you if you miss it.

  **NOTE:** For the time fields, the forecaster can either directly type in the time or can increment/decrement using the up and down arrows to reach the desired value.

In the **Report Type** area, select the **NWWS** button. Select a corresponding button for Daily Morning, Daily Intermediate, Daily Evening, Monthly, Seasonal, or Annual.

In the **Report Configuration**, enter a three-character **Product ID**. The product ID can be any three characters. However, for this product ID to be transmitted, it must be a valid ID and recognized by the WAN.
NOTE: The product ID must be unique amongst morning or evening products. The same product ID can be used for a morning and an evening product. For example, DCA can be chosen for an NWR morning product and an NWR evening product; however, DCA cannot be chosen for both NWR morning and NWWS morning products.

The Address field is for the dissemination of the product. It may be stored locally (000), sent to another addressee, or sent to all addressees (ALL).

Once the Report Type and Report Configuration have been determined, move to the Select Stations for Product area. Use the mouse and click on the station codes to select stations for the climate product.

9.1.5.3.2 Weather Elements

The remainder of the Report Format GUI allows the forecaster to configure the climatology summary by toggling the weather element buttons on and off.

This section is further subdivided into two parts, the Categories list and the Subcategories field. The Categories list contains 10 general categories of elements available for the output product. Upon selecting an element from the list, available options for the variable appear in the Subcategories portion of the interface. For example, temperature is further divided into the maximum, minimum, and mean values. Also under Categories, toggle buttons are provided as shortcuts to either Include All or None of the subcategories within a given element. If Include All is chosen for temperature, for example, all temperature data will be included in the output product. Conversely, Include None results in no temperature data.

In the Subcategories section, the first item in the option lists is always the measured value for yesterday or today depending on the type of report chosen. The measured value has to be chosen before any other options such as the time of occurrence, record, record date, normal, departure, and previous year’s value can be chosen. If the measured value is not chosen, the list of options beneath it will disappear.

Available Subcategories

- Maximum Temperature
- Minimum Temperature
- Mean Temperature
- Total Precipitation
- Total Snowfall
- Snow Depth
- Heating Degree Days
- Cooling Degree Days
• Maximum Wind
• Maximum Wind Gust
• Resultant Wind (NWWS only)
• Average Wind Speed
• Maximum Relative Humidity (not available for mon/sea/ann NWR)
• Minimum Relative Humidity (not available for mon/sea/ann NWR)
• Mean Relative Humidity (not available for mon/sea/ann NWR)
• Possible Sunshine (NWWS only) (percent of possible sunshine for day based on total minutes of sun and total minutes of sun possible)
• Sky Cover (NWWS only) (average for the day)
• Weather Types (NWWS only) (observed weather for the day)
Also available are temperature normals and extremes and sunrise and sunset for the day following the observation day.

The subcategories are further divided into options which are described below. Some of the options are common to several subcategories while a few are only available to certain subcategories.

**Common Options**

• **Time/Date of Occurrence.** The forecaster can specify to include the weather elements observation time via this option.

  **NOTE:** For wind and wind gust, time/date in the maximum column will be selectable for the NWR Daily Report, but not for the NWWS Daily Report.

• **Normal.** This option is for the inclusion of daily normal/mean values.
• **Departure.** This option is to include the observed weather element’s departure from normal.
• **Record.** The extreme/record for the day can be included via this option.
• **Date/Year.** This option is for the inclusion of the year(s) in which record events occurred. Up to 3 years are stored in the database.
• **Last Year’s.** Last year’s value for the chosen element.

  **NOTE:** For Daily NWR products, normal values cannot be included without the departure from normal value.
Other Options

- **Include Temp in deg C?** Select this option to add the temperature in degrees Celsius to the Daily Climatological Report. This will only appear when configuring an NWR report; it is not available for NWWS reports.

**NOTE:** This only displays temperature in degrees Celsius in the CRS output and does not affect the NWWS output.

- **Month to Date (MTD).** This option is available for precipitation, snowfall, and degree days and includes the total amount of the chosen element during the month up to and including the observation day.

- **Season to Date (STD).** This option is available for precipitation, snowfall, and degree days and includes the total amount of the chosen element for the current season. Remember that the precipitation season can be defined as an alternate annual period, such as 1 September through 31 August.

- **Year to Date (YTD).** This option is available for precipitation, snowfall, and degree days and includes the total amount of the chosen element for the current year. Notice that this option appears as “Since July 1” for heating degree days and “Since January 1” for cooling degree days.

**NOTE:** The toggle buttons Normal, Departure, and Last Year’s control the normal, departure, and the previous year’s reports for MTD, STD, and YTD sums in addition to the measured values. For instance, choosing the MTD sum as well as the departure will result in the MTD sum and departure from normal of the sum both being included.

Additional Options

You can choose from several additional options when setting up the monthly, seasonal, and annual climate product. They include Temperature, Precipitation, Snowfall, and Degree Days options.

**Temperature Options**

- **Avg. Daily Max.** Include the average daily maximum temperature for the mon/sea/ann.

- **Avg. Daily Min.** Include the average daily minimum temperature for the mon/sea/ann.

- **Days Max GE 90° F.** Include the number of days where the temperature was greater than or equal to 90 degrees Fahrenheit. This value is a set value within Climate.

- **Days Max LE 32° F.** Include the number of days where the temperature was less than or equal to 32 degrees Fahrenheit. This is a set value within Climate.

- **Days GE <forecaster-defined_value>° F.** Include the number of days the maximum temperature was greater than or equal to a forecaster-defined value. The value can be...
set by accessing the Edit User-Defined Values GUI from the Climate Parameters GUI or the Report Format GUI.

- **Day LE <forecaster-defined_value>° F.** Include the number of days the maximum temperature was less than or equal to a forecaster-defined value. The value can be set by accessing the Edit User-Defined Values GUI from the Climate Parameters GUI or the Report Format GUI.

- **Days Min LE 32° F.** Include the number of days where the minimum temperature was less than or equal to 32 degrees Fahrenheit. This value is a set value within Climate.

- **Days Min LE 0° F.** Include the number of days where the minimum temperature was less than or equal to 0 degrees Fahrenheit. This value is a set value within Climate.

- **Days GE <forecaster-defined_value> F.** Include the number of days the minimum temperature was greater than or equal to a forecaster-defined value. The value can be set by accessing the Edit User-Defined Values GUI from the Climate Parameters GUI or the Report Format GUI.

- **Days LE <forecaster-defined_value> F.** Include the number of days the minimum temperature was greater than or equal to a forecaster-defined value. The value can be set by accessing the Edit User-Defined Values GUI from the Climate Parameters GUI or the Report Format GUI.

**Precipitation Options**

- **Average Daily.** Include the average daily amount of precipitation.

- **Storm Max.** Include the storm maximum amount of precipitation.

- **24 Hour Max.** Include the 24-hour maximum amount of precipitation.

- **Days GE 0.01 in.** Include the number of days where the amount of liquid precipitation was greater than or equal to 0.01 inches.

- **Days GE 0.10 in.** Include the number of days where the amount of liquid precipitation was greater than or equal to 0.10 inches.

- **Days GE 0.50 in.** Include the number of days where the amount of liquid precipitation was greater than or equal to 0.50 inches.

- **Days GE 1.00 in.** Include the number of days where the amount of liquid precipitation was greater than or equal to 1.00 inch.

- **Days GE <forecaster-defined_value> F.** Include the number of days where the amount of liquid precipitation was greater than or equal to a forecaster-defined value (in inches). The value can be set by accessing the Edit User-Defined Values GUI from the Climate Parameters GUI or the Report Format GUI.

**Snowfall Options**

- **Depth Average.** Include the average snow depth during the mon/sea/ann.
• **Tot Since Jul 1.** Include the total snowfall since 1 July.
• **Depth Maximum.** Include the deepest snow depth during the mon/sea/ann.
• **Days Any Snow.** Include the number of days with snowfall greater than a trace.
• **Days GE 1.0 in.** Include the number of days the snowfall was greater than or equal to 1.0 inch. This is a set value within Climate.
• **Days GE <forecaster-defined_value>F.** Include the number of days the snowfall was greater than or equal to a forecaster-defined value (in inches) of liquid precipitation. The value can be set by accessing the Edit User-Defined Values GUI from the Climate Parameters GUI or the Report Format GUI.
• **Water Equiv.** Include the water equivalent amount of snowfall.
• **WE Since Jul 1.** Include the water equivalent amount of snowfall since 1 July.
• **Storm Max.** Include the storm maximum amount of snowfall.
• **24 Hour Max.** Include the 24-hour maximum amount of snowfall.

*Degree Days Options*
• **Freeze Dates.** Include the earliest and latest freeze dates for the mon/sea/ann.
• **Average Sky Cover.** Include the average sky cover for the mon/sea/ann.

**Creating Products**

All toggle buttons can be turned on and off to achieve the desired set of elements for the product being created. When editing an existing product, simply toggle the elements which need to be changed.

Once all parameters for the product have been entered or chosen, select the **Save Product** button to accept parameters.

If a product is no longer needed, choose **Delete Product** to remove it from the database.

Selecting **Close** will return the forecaster to the **Climate Master GUI**.

### 9.2 Record Climate

The **recordClimate** process performs the basic “record report” (RER) functionality required by AWIPS. **recordClimate** is an extension to the current **Climate** functionality. If a Record is set in the **Climate** process, **recordClimate** will recognize this and produce a formatted RER and store it in the **fxatext** database. A forecaster can then retrieve this product from the database via the **Text WS** or **nwrEditor** for data verification and subsequent offsite dissemination.

The recordClimate process performs three basic operations:

1. **Formats of Record Climate Reports**
2. Stores Record Climate Reports in the `fxatext` database (CCCRERXXX)
3. Notifies forecaster (local/System) that a record has been set.

**NOTE:** Distribution and verification of products from the AWIPS `fxatext` database are accomplished manually via the TextWS (WAN) and nwrEditor (CRS).

### 9.2.1 Configuration Files and Environment Variables

Table 9.2.1-1 shows the environment variables used in discussing the Climatological Reports Formatter. These variables are defined for user fxa and individual user accounts.

<table>
<thead>
<tr>
<th>Environment Variable Name</th>
<th>Environment Variable Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLIMATE_DIR</td>
<td>/awips/adapt/climate</td>
</tr>
<tr>
<td>FXA_DATA</td>
<td>/data/fxa</td>
</tr>
<tr>
<td>FXA_HOME</td>
<td>/awips/fxa</td>
</tr>
<tr>
<td>LOG_DIR</td>
<td>/data/logs/fxa</td>
</tr>
<tr>
<td>CLIMATE_BIN_DIR</td>
<td>/awips/adapt/climate/bin/Linux</td>
</tr>
</tbody>
</table>

The following files are used by the `recordClimate` process.

- **recordClimate.cfg**
  - This file is located in the `SCLIMATE_DIR/data` directory and is used for forecaster notification configuration settings on initial startup. The configuration file consists of the following data:
    - `<Display Type>`
    - `<Importance>`
    - `<Audio Alert File>`
  - Each is on a line by itself as shown in this example:
    ```
    SYSTEM
    URGENT
    /awips/fxa/data/sounds/beep.au
    ```

- **RecordClimateRawData.dat**
  - This file will be created in the `SCLIMATE_DIR/tmp` directory. It is used to pass data from Climate to `recordClimate` when a climatic record event occurs. The file will not be created if no records are broken, and it will be deleted after `recordClimate` has read it.

- **RecordClimateStationInfo.dat**
  - This file will be created in the `SCLIMATE_DIR/tmp` directory. It is used to pass data from Climate to `recordClimate` when a climatic record event occurs. The file will not be
created if no records are broken. Upon completion of `recordClimate`, the file will be deleted.

```
afos2awips.txt
awipsPriorities.txt
NWWS_exclude_XXX.txt
nwr.cfg
nwwsDeviceName.txt
rcv_handler.tbl
textNotificationServerClientList.txt
NWWSSchedulerHost.txt
```

### 9.2.2 Processes

<table>
<thead>
<tr>
<th>LX</th>
<th>Climate</th>
</tr>
</thead>
<tbody>
<tr>
<td>LX</td>
<td>recordClimate</td>
</tr>
<tr>
<td>PX1</td>
<td>textNotificationServer</td>
</tr>
<tr>
<td>LX</td>
<td>fxaAnnounce</td>
</tr>
<tr>
<td>LX</td>
<td>textWish</td>
</tr>
<tr>
<td>PX1</td>
<td>MhsServer</td>
</tr>
<tr>
<td>PX1</td>
<td>NWWSProduct</td>
</tr>
<tr>
<td>PX1</td>
<td>NWWSSchedule</td>
</tr>
<tr>
<td>LX</td>
<td>nwrEditorWish</td>
</tr>
<tr>
<td>LX</td>
<td>transferNWR</td>
</tr>
<tr>
<td>PX1</td>
<td>nwrTrans.pl</td>
</tr>
<tr>
<td>PX1</td>
<td>senttoNWR</td>
</tr>
</tbody>
</table>

Refer to Exhibit 9.2.2-1 for the Record Climate data flow diagram.

**Climate**

Exhibit 9.1.3-1, Climatological Reports Formatter for NWR and NWWS (Daily), shows the **Climate** process. When a record is set in the **Climate** process, the information necessary to form Record Reports is written to the two data files, `RecordClimateRawData.dat` and `RecordClimateStationInfo.dat`.

**recordClimate**

The **recordClimate** process is launched during each climate run. It will check for existence of two files, `RecordClimateRawData.dat` and `RecordClimateStationInfo.dat`, created by `create_climate`. Upon encountering these two files, **recordClimate** will create and store a record report (RER) product in the `fxatext` database and notify the forecaster that a record report exists.
Chapter 9 Background Applications

Config Files:
- /data/fxa/workFiles/wanMsgHandling/NWWS_exclude_XXX.txt
- $FXA_HOME/data/awips2awips.txt
- /data/fxa/workFiles/nwr/nwr.cfg
- $CLIMATE_DIR/RecordClimateStationInfo.dat
- $LOG_DIR/display/<display>/<yyyyymmdd>/logStreamExpect
- $CLIMATE_DIR/RecordClimateMaxData.dat
- /data/fxa/workFiles/nwr/reply
- /data/fxa/workFiles/nwr/nwwsDeviceName.txt
- /data/fxa/workFiles/nwws/NWWSSchedulerHost.txt
- /data/fxa/workFiles/nwws/rcv_handler.tbl
- /data/fxa/archive/OUP/scratch
- /data/fxa/archive/OUP/scratch
- /data/fxa/workFiles/nwr/pending
- /data/fxa/workFiles/nwr/sent
- /data/fxa/workFiles/nwr/nwrTrans.pl
- /data/fxa/workFiles/nwws/NWWSSchedulerHost.txt
- /data/fxa/workFiles/nwws/rcv_handler.tbl
- $FXA_HOME/data/awipsPriorities.txt
- /data/fxa/workFiles/nwws/nwwsDeviceName.txt
- /data/fxa/workFiles/nwws/rcv_handler.tbl
- $FXA_HOME/data/awipsPriorities.txt
- /data/fxa/workFiles/nwws/NWWSSchedulerHost.txt
- /data/fxa/workFiles/nwws/rcv_handler.tbl

Log Files:
- $LOG_DIR/display/<display>/<yyyyymmdd>/nwrEditorWish<logid>
- $LOG_DIR/display/<display>/<yyyyymmdd>/textWish<logid>
- $LOG_DIR/display/<display>/<yyyyymmdd>/logStreamExpect
- $LOG_DIR/display/<display>/<yyyyymmdd>/recordClimate<logid>

Data Files:
- /data/fxa/workFiles/nwr/ready
- /data/fxa/workFiles/nwr/nwrEditorWish<logid>
- /data/fxa/workFiles/nwr/reply
- /data/fxa/workFiles/nwr/sent
- /data/fxa/workFiles/nwr/nwrTrans.pl
- /data/fxa/workFiles/nwws/NWWSSchedulerHost.txt
- /data/fxa/workFiles/nwws/rcv_handler.tbl

Databases:
- fxatext Database
- $LOG_DIR/<yyyyymmdd>/MhsServer<logid>
- $LOG_DIR/<yyyyymmdd>/textNotificationServer<logid>
- $LOG_DIR/<yyyyymmdd>/NWWSProduct<logid>

Exhibit 9.2.2-1. Record Climate Data Flow

-- Hard copies uncontrolled. Verify effective date prior to use. --
To View the Current Day’s recordClimate Log

On LX1 as user fxa,

**TYPE:** cd $LOG_DIR/display/<display>/<yyyymmdd>
- Where <yyyymmdd> is today’s date and <display> is the current display variable (e.g., lx1-tbdw:0).

**TYPE:** ls -ltr recordClimate*
- Displays the current day’s recordClimate logs from earliest to latest.

**TYPE:** tail -f <logname>
- Displays the log as it updates with the latest messages.

Log Message Format

An entry in the log should look like the following:

```
LOG-STATUS: Log file opened on host lx1-tbdw3 at Mon Aug 29 12:28:55 2006
12:28:55:824 recordClimate.C EVENT: Checking to see if records are broken...
12:28:55:837 recordClimate.C EVENT: No need to run recordClimate...Skipping...
```

The log consists of the following fields:

- Time stamp: 12:28:55:837
- File name: recordClimate.C
- Message type: EVENT
- Message string: No need to run recordClimate...Skipping...

The logs are purged by the **scour** script after 2 days and the empty directories are removed by the **master.purge** script.

**textNotificationServer**

The **textNotificationServer** process runs on PX1 and sends out notification messages to each of its registered clients for every text product written to the text database. The notification message identifies each product received by its nine-letter product ID. The process creates a data file, **textNotificationServerClientList.txt**, which identifies the processes registered with the **textNotificationServer**. When a product of interest is stored in the text database, it notifies **recordClimate**.
To View the Current Day’s textNotificationServer Log

On PX1 as user fxa,

TYPE: cd $LOG_DIR/<yyyymmdd>
  Where <yyyymmdd> is today’s date.

TYPE: ls -ltr textNotificationServer*
  Displays the current day’s textNotificationServer logs from earliest to latest.

TYPE: tail -f <logname>
  Displays the log as it updates with the latest messages.

Log Message Format

An entry in the log should look like the following:

18:42:47.480 TextNotificationServer.C EVENT: SFORERSFO notif rcvd

The log consists of the following fields:

Time stamp  18:42:47.480
File name  TextNotificationServer.C
Message type  EVENT:
Message string  SFORERSFO notif rcvd

The logs are purged by the scour script after 2 days and the empty directories are removed by the master.purge script.

fxaAnnounce

The fxaAnnounce process produces the text that displays on the appropriate fxa status line according to the “displayer” (LOCAL, SYSTEM, RADAR) and “importance” (SIGNIFICANT, ROUTINE, URGENT) characteristics.

TextDB_Server

The recordClimate process makes a request to the EDEX plug-in called TextDBSrv. It runs in the EDEX Request. The log messages can be found on edex-request.log.

textWish

The textWish process uses textWindow.tcl to create a text window so the forecaster can retrieve the record report product from the database. Upon verification of the product, it calls MhsServer to disseminate the report to the WAN.
Chapter 9 Background Applications

TYPE: `cd $LOG_DIR/display/<display>/<yyyymmdd>`
  - Where `<yyyymmdd>` is today’s date and `<display>` is the current display variable (e.g., lx1-tbdw:0).

TYPE: `ls -ltr textWish*`
  - Displays the current day’s `textWish` logs from earliest to latest.

TYPE: `tail -f <logname>`
  - Displays the log as it updates with the latest messages.

Log Message Format

An entry in the log should look like the following:

```
textWish 8803 1030732899.962823 18:41:39.962 EVENT: textWish 8803 app initialized
```

The log consists of the following fields:

- **Time stamp**: `18:42:45.365`
- **File name**: `textWish`
- **Message type**: `EVENT:`
- **Message string**: `Reading /data/fxa/workFiles/wanMsgHandling/NWWS_exclude_STO.txt`

The logs are purged by the `scour` script after 2 days and the empty directories are removed by the `master.purge` script.

MhsServer

The `MhsServer` process on PX1 registers a call-back function that will handle products sent from the text workstation, declares an IPC address, and enters a dispatch loop waiting for messages. A notification is sent to the `MhsServer` process consisting of the WMO header and the file name slated for distribution. Module `MhsWfoProduct` sends the product to the WAN, NWWSUP while it archives the file in the `/data/fxa/archive/OUP/scratch` directory.

▶ To View the Current Day’s `MhsServer` Log

On the PX1 as user `fxa` or your individual user account,
TYPE: `cd $LOG_DIR/<yyyymmdd>`  
   - Where `<yyyymmdd>` is today’s date.

TYPE: `ls -ltr MhsServer*`  
   - Displays the current day’s MhsServer logs from earliest to latest.

TYPE: `tail -f <logname>`  
   - Displays the log as it updates with the latest messages.

Log Message Format

An entry in the log should look like the following:

```
18:42:46.660 MhsWfoProduct.C EVENT:/data/fxa/archive/OUP/scratch/KSFORERSFO.wan 20060512_184245 successfully archived
19:00:19.755 DataSocket.C DEBUG: Connection to lx1-tbdw/2573/8803 has been closed by the peer.
19:00:19.774 DataSocket.C DEBUG: Closing the socket to lx1-tbdw/2573/8803
```

The log consists of the following fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time stamp</td>
<td>18:42:46.660</td>
</tr>
<tr>
<td>File name</td>
<td>MhsWfoProduct.C</td>
</tr>
<tr>
<td>Message type</td>
<td>EVENT:</td>
</tr>
<tr>
<td>Message string</td>
<td>/data/fxa/archive/OUP/scratch/KSFORERSFO.wan 20060512_184245 successfully archived</td>
</tr>
</tbody>
</table>

The logs are purged by the `scour` script after 2 days and the empty directories are removed by the `master.purge` script.

NWWSProduct

The process stores the record report it received from MhsServer in the `/data/fxa/workFiles/nwws/rcvq` directory and opens a socket to NWWSSchedule to transfer the message.

To View the Current Day’s NWWSProduct Log

On PX1 as user fxa,

```
TYPE: `cd $LOG_DIR/<yyyymmdd>`  
   - Where `<yyyymmdd>` is today’s date

TYPE: `ls -ltr NWWSProduct*`  
   - Displays the current day’s NWWSProduct logs from earliest to latest

TYPE: `tail -f <logname>`
```
• Displays the log as it updates with the latest messages

**Log Message Format**

An entry in the log should look like the following:

18:42:52.580 nwmsProduct.C EVENT: Received message: 0056 SFORERSFO /data/fxa/workFiles/nwws/rcvq/TBDW-153708.001

18:42:52.867 nwmsProduct.C EVENT: Product sent to NWSSSschedule via socket

The log consists of the following fields:

- **Time stamp**: 18:42:52.867
- **File name**: nwmsProduct.C
- **Message type**: EVENT:
- **Message string**: Product sent to NWSSSschedule via socket

The logs are purged by the **scour** script after 2 days and the empty directories are removed by the **master.purge** script.

**NWSSSschedule**

The **NWSSSschedule** process creates a socket to communicate and receive products with **NWSSProduct**, validates product ID, maintains product queues, disseminates the products by forking **NWSSTransmit** and cleans up product files upon successful transmission.

▶ **To View the Current Day’s NWSSSschedule Log**

**On PX1 as user fxa,**

**TYPE:**

```
cd $LOG_DIR/<yyyymmdd>
```

• Where `<yyyymmdd>` is today’s date.

**TYPE:**

```
ls -ltr NWSSSschedule*
```

• Displays the current day’s **NWSSSschedule** logs from earliest to latest.

**TYPE:**

```
tail -f <logname>
```

• Displays the log as it updates with the latest messages.

**Log Message Format**

An entry in the log should look like the following:


18:42:52.936 nwmsScheduler.C EVENT: RER is not found in nwmsProductPriority table. Default priority is used


18:42:54.267 nwmsScheduler.C EVENT: SFORERSFO /data/fxa/workFiles/nwws/rcvq/TBDW-153709.001 has been successfully transmitted
The log consists of the following fields:

- Time stamp
- File name
- Message type
- Message string

The logs are purged by the `scour` script after 2 days and the empty directories are removed by the `master.purge` script.

**nwrEditorWish**

The `nwrEditorWish` process has been extended to include NWR application-specific commands and executes the `nwrEditor` script to provide the GUI access. The forecaster is allowed to remotely control NWR from an AWIPS workstation to monitor product transfer from AWIPS to NWR. The `nwrEditorWish` process provides a last-minute edit capability for products which are to be sent to the NWR from AWIPS.

▶ **To View the Current Day’s nwrEditorWish Log**

On the LX,

- **TYPE:** `cd $LOG_DIR/display/<display>/<yyyymmdd>`
  - Where `<yyyymmdd>` is today’s date and `<display>` is the current display variable (e.g., lx2-tbdw:0).

- **TYPE:** `ls -ltr nwrEditorWish*`
  - Displays the current day’s `nwrEditorWish` logs from earliest to latest.

- **TYPE:** `tail -f <logname>`
  - Displays the log as it updates with the latest messages.

**Log Message Format**

An entry in the log should look like the following:

```
LOG-STATUS: Log file opened on host lx2-tbdw at Wed May 12 18:49:26 2006
nwrEditorWish 9100 1030733366.848139 18:49:26.848 PROBLEM: unable to send voice ready product to crs _sendToCrs failed
nwrEditorWish 9100 1030733366.851436 18:49:26.851 PROBLEM: could not send voice ready product to crs
nwrEditorWish 9100 1030733366.851809 18:49:26.851 PROBLEM: voice ready product NOT SENT to crs
nwrEditorWish 9100 1030733989.540121 18:59:49.540 EVENT: textWish interpreter 9100 exiting.
```

The log consists of the following fields:

- Time stamp
- File name
- Message type
Message string unable to send voice-ready product to crs
_sendToCrs failed

The logs are purged by the scour script after 2 days and the empty directories are removed by the master.purge script.

transferNWR

The transferNWR script is responsible for transferring voice-ready products to a holding area where they will be picked up and sent to NWR. The transferNWR script is a command-line interface, which provides the forecaster with three options:

transferNWR -a [filename] This option will automatically transfer the file from the source directory to the /data/fxa/workFiles/nwr/ready directory. The full pathname of the file must be given so that the file can first be copied into the source directory. The transferNWR script then monitors the $FXA_DATA/workFiles/nwr/pending directory to verify that the product has been successfully sent to CRS by the sendToNWR script.

transferNWR -d [filename] This option will also ensure that the file is copied into the source directory, but the file will not be transferred. This option is for cases where the forecaster would first like to inspect or edit the product before it is sent.

transferNWR –n This option sends a warning message to the local workstation that there has been a problem in transmitting the file. This message is sent at the forecaster’s discretion, based upon the return and exit values issued by this script.

The transferNWR script is forked by a parent process, NWRBrowser. The transferNWR script returns one of two values to the parent process. The values are as follows:

Value = 0 indicates all was successful
Value = 1 indicates that the transfer may have failed

Log files can be viewed from the NWRBrowser. The most current log is listed last. Logs can also be viewed by other text editors, such as the vi editor.

To View the Current Day’s transferNWR Log

On the workstation,

TYPE: cd $LOG_DIR/display/<display>/<yyyymmdd>

Where <yyyymmdd> is today’s date and <display> is the current display variable (e.g., lx1-tbw4:0).
TYPE: `ls -ltr logStreamExpect*`

- Displays the current day’s `transferNWR` logs from earliest to latest.

TYPE: `tail -f <logname>`

- Displays the log as it updates with the latest messages.

**Log Message Format**

An entry in the log should look like the following:

```
LOG-STATUS: Log file opened on host lx1-tbw4 at Wed May 12 18:19:05 2006
18:19:05.358 TclLogStream.C EVENT: File /data/fxa/workFiles/nwr/pending/PITHWRNW1 will be automatically transferred
18:19:05.359 TclLogStream.C EVENT: /data/fxa/workFiles/nwr/pending/PITHWRNW1 has been copied to data/fxa/workFiles/nwr/ready/PITHWRNW1 for processing by sendToNWR.
```

**NOTE:** Check the logStreamExpect log on PX for information about the transfer of `data/fxa/workFiles/nwr/pending/PITHWRNW1` to CRS.

The log consists of the following fields:

- Time stamp: 18:19:05.358
- File name: TclLogStream.C
- Message type: EVENT:
- Message string: File /data/fxa/workFiles/nwr/pending/PITHWRNW1 will be automatically transferred

The scour script purges the logs after 2 days, and the empty directories are removed by the `master.purge` script.

**nwrTrans.pl**

The `nwrTrans.pl` (on PX1) Perl script is a persistent script that looks in `/data/fxa/workFiles/nwr/ready` every couple of seconds to find files that have been placed there by either the `NWRBrowser` or `nwrEditor` GUIs when a user requests to send a product to the CRS for broadcast on the NOAA Weather Radio. This script then calls `transferNWR` for any files that were found. Previously, `transferNWR` was called directly from the `NWRBrowser` or `nwrEditorWish`.

**sendToNWR**

Voice-ready text is transmitted from AWIPS to the NWR by the `sendToNWR Expect` script. In addition to transferring the product, copies can be kept in one of two directories:

```
$FXA_DATA/workFiles/nwr/pending
$FXA_DATA/workFiles/nwr/sent
```
The sendToNWR script is called from the nwrTrans.pl script, which is a persistent fxa process that checks the /data/fxa/workFiles/nwr/ready directory every 2 seconds for products placed there by the transferNWR script. When nwrTrans.pl finds a file, or files, in this directory, it calls sendToNWR for each file.

The sendToNWR script is responsible for transferring voice-ready products from AWIPS to the NWR. The sendToNWR script is a command-line interface that provides the forecaster with three options:

- **sendToNWR -a [filename]**
  This option will automatically transfer the file from the source directory on AWIPS to the destination directory on the CRS. The full pathname of the file must be given so that the file can first be copied into the source directory.

- **sendToNWR -d [filename]**
  This option will also ensure that the file is copied into the source directory, but the file will not be transferred. This option is for cases in which the forecaster would first like to inspect or edit the product before it is sent.

- **sendToNWR –n**
  This option sends a warning message to the local workstation that there has been a problem in transmitting the file. This message is sent at the forecaster’s discretion, based upon the return and exit values issued by this script.

**NOTE:** The transferNWR script also may continue to be used for the –d and –n options.

When sendToNWR is invoked, the product is automatically copied to the pending directory. If successfully sent, the product is moved to the sent directory. The contents of each directory can be viewed in the windowpane dedicated to that directory. Products not successfully transmitted will remain in the pending directory.

**To View the Current Day’s sendToNWR Log**

On PX1, as user fxa,

**TYPE:**

cd $LOG_DIR/<yyyymmdd>

- Where <yyyymmdd> is today’s date.

**TYPE:**

ls -ltr logStreamExpect

- Displays the current day’s sendToNWR log.

**TYPE:**

tail -f <logname>

- Displays the log as it updates with the latest messages.
Log Message Format

An entry in the log should look like the following:

18:19:05.358 TclLogStream.C EVENT: File /data/fxa/workFiles/nwr/pending/PITHWRNW1 will be automatically transferred
18:19:06.012 TclLogStream.C EVENT: Successful ping to 0MP
18:19:06.368 TclLogStream.C EVENT: Successful ping to 5MP
18:19:09.802 TclLogStream.C EVENT: Master processor is 0MP
18:19:11.746 TclLogStream.C EVENT: transfer complete to 0MP
18:19:11.899 TclLogStream.C EVENT: Successful transfer of /data/fxa/workFiles/nwr/pending/PITHWRNW1 to 0MP

The log consists of the following fields:

<table>
<thead>
<tr>
<th>Time stamp</th>
<th>18:19:05.358</th>
</tr>
</thead>
<tbody>
<tr>
<td>File name</td>
<td>TclLogStream.C</td>
</tr>
<tr>
<td>Message type</td>
<td>EVENT:</td>
</tr>
<tr>
<td>Message string</td>
<td>File /data/fxa/workFiles/nwr/pending/PITHWRNW1 will be automatically transferred</td>
</tr>
</tbody>
</table>

The scour script purges the logs after 2 days and the empty directories are removed by the master. purge script.

9.3 Hourly Weather Roundup Formatter for the NOAA Weather Wire Service (NWWS)

The HWR is a summary of the current weather observations for locations in and around the local area including marine observations for those WFOs with coastal responsibilities.

The HWR NWWS formatter application prepares tabular products for the NWWS. The formatted products are automatically sent to the NWWS.

The HWR uses:

- METAR observations
- Marine observations
- SCD
- Satellite Cloud Product (SCP).

The HWR NWWS application (hwrnwws) performs calculations on hourly weather data, both land and marine, constructs the product, and stores the resultant text files into a temporary target directory. From the temporary directory, the product is automatically sent to the NWWS. The transferNWWS.pl script sends the completed files over the WAN to the site’s NWWS uplink site for distribution to the NWWS via the uplink site’s NWWS asynchronous interface port.
NOTE: The forecaster-entered METARINT value in the nwws.config file specifies the number of minutes before the hour at which to start including the METAR observations in the HWR. The HWR considers any observations from H-METARINT up to but not including H+(60-METARINT) as an observation for the nominal hour. For example, if the METARINT value is 15, observations at 0845Z are considered 0900Z observations, but observations at 0840Z are considered 0800Z observations. So, if an observation comes in at H+0:40, then at the top of the hour HWR will consider that observation to be from the last hour and reject it.

The NWWS selection in the Product Setup section of the HWR main GUI permits the forecaster to create a station listing and make configuration and threshold selections.

9.3.1 Configuration Files and Environment Variables

Environment Variables

The following environment variables are referenced in this chapter for user fxa and individual user accounts; their full paths are provided as follows:

$LOG_DIR /data/logs/fxa
$FXA_HOME /awips/fxa
$FXA_DATA /data/fxa

Configuration Files

nwws.config

The nwws.config configuration file contains the list of products to be produced for the NWWS. It resides in the /awips/adapt/hwr/etc/nwws directory. An example of an nwws.config file is shown in Exhibit 9.3.1-1.

```
SCRIPT /awips/fxa/bin/transferNWWS.pl
DESTDIR /data/fxa/workFiles/nwws
# product list. The following product will be generated:
METARINT 15 # minutes before the hour at which to start using METAR's in HWR
PRODUCT PITSWRPA
PRODUCT PITSWRPA1
PRODUCT PITSWRPA2
```

Exhibit 9.3.1-1. Example nwws.config File

awipsPriorities.txt

The awipsPriorities.txt configuration file contains a table with AFOS product categories (NNN) and corresponding priorities. The values in the first column are the product categories; those in the second column are the priorities. The file resides in the $FXA_HOME/data directory. An example of an awipsPriorities.txt file follows:

```
ACR 1
AWG 2
AWU 1
```
afos2awips.txt

The afos2awips.txt configuration file is used to convert the AFOS site ID into the AWIPS site ID. The first column contains the AFOS ID. The second column contains the World Meteorological Organization (WMO) header ID. The third column contains the AWIPS site ID. The file resides in the $FXA_HOME/data directory. An excerpt of this file follows:

```
ABNNPWAPN WWUS45 KAPN
ABNNPWAPN WWUS46 KAPN
ABQADAABQ NOUS45 KABQ
ABQADAABQ NOUS45 KABQ
ABQADAROW NOUS45 KROW
ABQADAABQ NOUS45 KZAB
ABQADAABQ NOUS45 KABQ
```

siteCommission.txt

This configuration file contains the AWIPS site’s commissioning status and is a holdover from the precommissioned phase. The file contains a 1 if the site is commissioned and a 0 if it is not yet commissioned. All operational sites should have a 1 in this file.

wmoSiteId.txt

This configuration file is located in /awips/fxa/data/localizationDataSets/LLL (where LLL is the localization ID, usually the same as the site ID) and identifies the site’s four letter AWIPS site ID.

NWWS_exclude_XXX.txt

This configuration file provides a list with NWWS products that will not be sent to the NWWS.

siteCommission.txt

This configuration file contains the AWIPS site’s commissioning status and is a holdover from the precommissioned phase. The file contains a 1 if the site is commissioned and a 0 if it is not yet commissioned. All operational sites should have a 1 in this file.

wmoSiteId.txt

This configuration file is located in /awips/fxa/data/localizationDataSets/LLL (where LLL is the localization ID, usually the same as the site ID) and identifies the site’s four letter AWIPS site ID.

NWWS_exclude_XXX.txt

This configuration file provides a list with NWWS
products that will not be sent to the NWWS.

9.3.2 Processes

PX1  hwrnwws
LX   hwrgui
LX   transferNWWS.pl
LX   handleOUP.pl
LX   distributeProduct

Exhibit 9.3.2-1 diagrams the Hourly Weather Roundup Formatter for NWWS data flow.

hwrnwws

The hwrnwws process requires the following input:

- Decoded marine observations from netCDF files
- Decoded METAR observations from the text database (fxatext)
- SCD and SCP data from the text database (fxatext)
- Setup data, which includes the station lists and marine selections from the CCCSWRXXX.setup, and CCCSWRXXX.product files
- Configuration information from the nwws.config file.

The hwrnwws process performs calculations on both land and marine hourly weather data, constructs the product, and stores the resultant text files in a temporary target directory. The transferNWWS.pl script sends the completed files to all sites across the AWIPS WAN via the NCF which distributes them to the NWWS primary and backup sites where it is uplinked.

▶ To View the Current Day’s hwrnwws Log

On PX1, as user fxa or your individual user account,

TYPE: cd $LOG_DIR/<yyyymmdd>
* Where <yyyymmdd> is today’s date.

TYPE: ls -ltr hwrnwws*
* Displays the current day’s hwrnwws logs from earliest to latest.

TYPE: tail -f <logname>
* Displays the log as it updates with the latest messages.
Exhibit 9.3.2-1. Hourly Weather Roundup Formatter for NOAA Weather Wire Service
Log Message Format

An entry in the log should look like the following:

Traceback (most recent call last):
File “/awips/adapt/hwr/py/hwrnwws.py”, line 52, in ?
wxRoundup.getFiles(‘nwws’, names)
File “/awips/adapt/hwr/py/wxRoundup.py”, line 100, in getFiles
pils = Pils.restorePils()
File “/awips/adapt/hwr/py/Pils.py”, line 18, in restorePils
return cPickle.load(open(PILFile))
IOError: [Errno 2] No such file or directory: ‘etc/pil.data’
~
~

The logs are purged by the scour script after 2 days and the empty directories are removed by the master.purge script.

hwrgui

The hwrnwws process enables the forecaster to enter required/allowed configuration and threshold information through a GUI controlled by processes launched by the hwrgui process. Forecaster selections are validated and stored in the CCCSWRXXX.setup file.

The NWWS product setup process uses the station format editor to create the station list for each output product. Each list contains the METAR and SCD station identifier, the SCP station identifier, a symbol to indicate how to treat missing data, and the station name.

The product IDs and the station lists are stored in the CCCSWRXXX.product file.

To View the Current Day’s hwrgui Log

On the workstation as user fxa or your individual user account,

**TYPE:**
```
    cd $LOG_DIR/display/<display>/<yyyymmdd>
    Where <display> is the current display variable (e.g., lx1-tbw4:0) and <yyyymmdd> is today’s date.
```

**TYPE:**
```
    ls -ltr hwrgui*
    Displays current day’s hwrgui logs from earliest to latest.
```

**TYPE:**
```
    tail -f <logname>
    Displays the log as it updates with the latest messages.
```

Log Message Format

An entry in the log should look like the following:

hwrgui 21509 924279088.572954 16:11:28.572 EVENT: HWRNWWSSetup::update() succeeded
~
The log consists of the following fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time stamp</td>
<td>16:11:28.572</td>
</tr>
<tr>
<td>Module name</td>
<td>hwrgui</td>
</tr>
<tr>
<td>Message type</td>
<td>EVENT:</td>
</tr>
<tr>
<td>Message string</td>
<td>HWRNWWSSetup::update() succeeded</td>
</tr>
</tbody>
</table>

The logs are purged by the **scour** script after 2 days and the empty directories are removed by the **master.purge** script.

**transferNWWS.pl**

The **transferNWWS.pl** script prepares a given product for WAN distribution by removing the one line communications header. The $SFXA_HOME/data/afos2awips.txt file is read and used to convert the AFOS product identifier, extracted from the product contents, to an equivalent AWIPS identifier. The **transferNWWS.pl** script invokes the **handleOUP.pl** script.

### To View the Current Day’s transferNWWS Log

On the workstation as user fxa or your individual user account,

**TYPE:**    cd $LOG_DIR/display/<display>/<yyyymmdd>

- Where <display> is the current display variable (e.g., lx1-tbw4:0) and <yyyymmdd> is today’s date.

**TYPE:**    ls -ltr transferNWWS*

- Displays the current day’s **transferNWWS.pl** logs from earliest to latest.

**TYPE:**    tail -f <logname>

- Displays the log as it updates with the latest messages.

**Log Message Format**

An entry in the log should look like the following:

```
product = /data/fxa/workFiles/nwws/SFOSWRCA
AFOS ID = SFOSWRCA
AFOS routing node = DEF
    -- Handling OUP
mapToAwipsID():
    Found match.
awips ID = KSOSWRCA
wan Command =
    handleOUP.pl -r DEF KSOSWRCA /data/fxa/workFiles/nwws/SFOSWRCA
Handling of OUP was successful.
Script done...
```

The logs are purged by the **scour** script after 2 days and the empty directories are removed by the **master.purge** script.
handleOUP.pl

The handleOUP.pl script performs tasks associated with the handling of an Official User Product, including distribution across the WAN, transmission over the NWWS uplink, local storage, and archiving. The handleOUP.pl script submits the message for WAN distribution via the distributeProduct command line interface. Once the product is disseminated, the handleOUP.pl script locally stores the product in the text database.

To View the Current Day’s handleOUP.pl Log

On the workstation as user fx a or your individual user account,

**TYPE:** cd $LOG_DIR/display/<display>/<yyyymmdd>

- Where <display> is the current display variable (e.g., lx1-tbw4:0) and <yyyymmdd> is today’s date.

**TYPE:** ls -ltr handleOUP*

- Displays the current day’s handleOUP.pl logs from earliest to latest.

**TYPE:** tail -f <logname>

- Displays the log as it updates with the latest messages.

Log Message Format

An entry in the log should look like the following:

```
<> New Activity Log Started Wed May 12 12:10:27 UTC 2006
<>
awipsID = KSFOSWRCA
mapToAfosAndWmoIds():
   ID = KSFOSWRCA
   Found match -- awips ID is valid.
   afos ID = SFOSWRCA
   wmo abbrev ID = ASUS46 KSFO
product = /data/fxa/workFiles/nwws/SFOSWRCA
getCategory():
   Product Category = SWR
getPriority():
   Priority = 0
getSiteCommissionStatus():
   commission status = 1
createNWWSAbbrevHeader():
   wmo abbrev header = ASUS46 KSFO 121210
getProductContents():
sendWANMsg():
   wanCommand = distributeProduct -p 0 -a DEFAULTNCF -c 0 KSFOSWRCA
   /data/fxa/workFiles/nwws/SFOSWRCA
   Message to DEFAULTNCF successfully submitted.
isNWWSProduct():
   KSFOSWRCA is an NWWS product.
sendWANMsg():
   wanCommand = distributeProduct -p 0 -a NWWSUP -c NWWS_UPLINK -s
   SPOSWRCA KSFOSWRCA /data/fxa/workFiles/nwws/SFOSWRCA
```
distributeProduct

The **distributeProduct** process creates a product message and submits it for distribution across the AWIPS WAN to the addressed sites. The **distributeProduct** process first prepares the product by creating the WAN communications header and prepending the header to a temporary copy of the product. Distribution requests, enclosing the temporary copy of the product, are subsequently made to the X.400 Message Handling System through the `msg_send` utility program.

**TYPE:**

```shell
cd $LOG_DIR/display/<display>/<yyyymmdd>
```

- Where `<display>` is the current display variable (e.g., `lx1-tbw4:0`) and `<yyyymmdd>` is today’s date.

**TYPE:**

```shell
ls -ltr distributeProduct*
```

- Displays the current day’s **distributeProduct** logs from earliest to latest.

**TYPE:**

```shell
tail -f <logname>
```

- Displays the log as it updates with the latest messages.

**Log Message Format**

An entry in the log should look like the following:

```
12:10:42.004 MHSProduct.C EVENT: Product request (KSFOSWRCA) was successfully submitted to the MHS request server for distribution.
Message Attributes:
- Request ID: TBDW-35537
- To: NWWSUP
- Priority: 0
- Type: Routine
- Subject: SFO\n\n12:10:42.050 MHSProduct.C EVENT: TBDW-35537
```

~
9.3.3 **Setting up the HWR NWWS Application**

Before it can be used, the HWR NWWS application has to be set up (localized) for your site. There are two steps involved in setting up this application:

1. Specify the HWR NWWS Setup Product options.
2. Create a Station List.

**NOTE:** Setting up the HWR NWWS application only needs to be done for the initial application installation and when creating new product identifiers.

9.3.3.1 **Identifying the Products to be Produced**

To identify the products to be produced, press the NWWS button in the Product Setup section of the hwr GUI. Either select an existing product and press Load or create a new product by typing the new product ID in the Product Actions section and press Add.

9.3.3.2 **Setting Up the HWR NWWS Application**

To access the HWR Editor for NWWS from the System Control Menu, select Background WFO Apps, Hourly Weather Roundup (HWR) from the cascading menu. This will display the Hourly Weather Roundup GUI shown in Exhibit 9.3.3.2-1.

![Hourly Weather Roundup GUI](image)

**Exhibit 9.3.3.2-1. The Hourly Weather Roundup GUI**

Next, select NWWS under the Product Setup on the HWR GUI. The **Hourly Weather Roundup** Editor for NOAA Weather Wire Service Product Setup (shown in Exhibit 9.3.3.2-2) will display.
Exhibit 9.3.3.2-2. The Hourly Weather Roundup for NOAA Weather Wire Service Product Setup

Once the HWR Editor for NWWS has opened you will see the following setup options.

**Products**

This scrolled list displays the products that may be converted to output products. Each product must contain a Station List file (editable via the Station Editor). Select a product and press **Load**. New products may also be created (see **Product Actions** below).

**Editors**

**Station.** This editor pushbutton displays the Station List for review and edit.
**Product Actions**

- **New/Add.** Create a new product by typing the product ID in this field and pressing **Add**.
- **Verify.** After you have created a new product, select **Verify** to check that the required files exist for the product.
- **Save.** This selection saves all the setup and product data from the GUI.
- **Delete.** You may delete a product by selecting the product ID from the scrolled list and hitting **Delete**.
- **Help.** This selection displays general information about the various sections in the GUI.

- **Close.** Closes the GUI.

**Addressee**

This is a three-letter identifier for the product routing address.

**Window for Marine Obs(hours):**

By default, if observations are more than 1 hour old, they are considered invalid and are not used in the roundup. However, since C-MAN, buoy, and ship observations are not necessarily reported every hour, an option to use older observations is included in the GUI. Marine observations, up to the number of hours entered in this window, will be considered valid and used by the HWR NWWS application.

**Include temperature in deg C?:**

This option is used to output the temperature, in degrees Celsius, in the Remarks field. However, just because this field is selected it does not mean the temperature in degrees Celsius will always be output. The Remarks section can contain a maximum of two weather elements based on a fixed priority order, with the temperature in degrees Celsius having the lowest priority.

**Express sky conditions when SCP missing?:**

ASOS METAR cloud observations do not exceed 12,000 feet, so the sky/weather phrase may not be generated accurately when there is no significant weather and the ASOS cloud observation is not cloudy. Satellite cloud observations are used to supplement ASOS observations in these conditions. If this option is selected, and the SCP is missing while the ASOS observation is clear or partly cloudy, the phrase “FAIR” will display. Otherwise, when the SCP is missing, the sky/weather will be output as missing.

**Time Zone:**

The forecaster can specify the time zone to be used in the product via this option.

**Std time all year?:**

This Standard Time All Year option must be selected for those areas that do not switch to Daylight Savings Time.
**Thresholds**

**Heat Index.** The forecaster can specify the minimum heat index value at or above which a heat index phrase is to be output.

**Wind Chill Index.** For the wind chill index to be calculated, the observed temperature must be at or below the value entered in **Max Temp(deg F)** and the observed wind speed must be at or above the value entered in **Min Wind Speed(mph)**. Also, the calculated wind chill index must be at or below the value entered in **Max WCI** for a wind chill index to be output.

**Marine Fields**

To determine the HWR NWWS data line length, select either **69** or **80**, which represent the number of characters to be formatted per line, and choose the marine fields to be included in the HWR NWWS product (from **Field selection**). The **Station ID** field is automatically selected; the field is necessary for this product. You may select from one up to all of the rest of the fields provided in the GUI. The GUI provides a running total, named **Field size**, of the number of characters comprising the marine fields you selected, which includes the blank spaces that will be inserted between each marine field. When the **Field size** exceeds the **Line length**, the resulting output product will contain marine fields that are wrapped on to a second line for stations reporting those fields.

---

**9.3.3.3 Creating a Station List**

The Station Format Editor is used to create the Station List. An example of an HWR NWWS Station List is shown in Exhibit 9.3.3.3-1.

The software delivery includes a template (product ID CCCSWRXX) that you may cut, paste, and edit to match your requirements. The station list begins with the AFOS PIL, WMO Header, and the Mass Media Header followed by comment lines. The comment lines are followed by universal generic code with expiration date/time. Another comment follows to identify the part(s) of the state in which the succeeding group of stations, called a segment, are located. Next, the stations in the group are listed.

**Comment Lines**

Any line in the Station List that begins with a tilde “~” is interpreted as information which will appear verbatim in the HWR NWWS output product. Symbolic words, converted into data observations (refer to Exhibit 9.3.3.3-2), can be inserted into these comment lines. Any line starting with the pound symbol (#) is interpreted as being an internal Station List comment and is not included in the HWR NWWS output product.
Chapter 9

Background Applications

Exhibit 9.3.3.3-1. Sample Station List for the NWWS

| %DATE% | - the current date (e.g., August 6). |
| %DAY% | - the current day of the week (e.g., Thursday). |
| %TIME% | - the current hour (e.g., 9 AM). |
| %TIMEZ% | - the current nominal hour and time zone (e.g., 9AM Pacific Daylight Time). |
| %TIMEW% | - the product creation time in numerical format (e.g., 251908, which means 25th day of the month, 19th hour, 8th min). |
| %HEADDT% | - the NWWS header date/time (e.g., 900 AM PACIFIC DAYLIGHT TIME THURSDAY AUGUST 6, 1998). |
| %DDHHMM% | - the current date of the month, nominal hour, and minute of the observation (e.g., 251900, which means 25th day of the month, 19th hour, 0th min). |

Exhibit 9.3.3.3-2. Symbolic Words
Fixed Phrase Stations

Enter !!!!!FF_STATIONS to identify that fixed phrase land stations follow in the station list. Fixed phrase land stations are identified by their four-character ICAO identifiers (e.g., KCRW) and are listed in the order they are to be output, one station per line.

Place a slash “/” on each side of the surface station (METAR or marine observation) identifier.

Next, place another four-character ICAO identifier for the representative SCP station followed by a slash, unless the station identifier is identical to the surface station identifier. In such cases, it is not necessary to repeat the surface station identifier. Instead, place another slash after the previous one.

Next, place the missing station flag which is either the plus sign “+,” minus sign “-,” the equal sign “=,” or the ampersand “&.”

The “+” indicates that the corresponding station will always be output, even if the weather elements are missing (i.e., “NOT AVBL”).

The “-” indicates that the station will be ignored if all the weather elements for the station are missing.

The “=” indicates that the corresponding land station will always be output, even if all weather elements are missing. This symbol also indicates that additive data are to be appended immediately below the station’s hourly data.

The “&” indicates that the corresponding land station will be dropped if all weather elements are missing. This symbol also indicates that additive data are to be appended immediately below the station’s hourly data.

Enter the fixed phrase station name after the missing station flag. As many as 14 characters may be used for land station names. Sample fixed phrase station lines follow:

```
/ KCGI // Cape Girardeau
/ KMKC/KMCI/-KC Downtown
WS/ -tbw4
/tmp
```

Land stations are grouped together in segments consisting of up to 10 stations. Each segment ends with ~$$ as shown in the following example.

```
!!!FF_STATIONS
/KMCI/+KC INTL APT
/KMKC/KMCI/+KC DOWNTOWN
/KGVW/+GRANDVIEW
/KSTJ/+ST JOSEPH
/KP35/+SPCKARD
~$$
# ~MD2079>081-088-090>094-101-102%DDHHMM% ~...Southwest Missouri...
11111FF_STATIONS
/KSGF/-SPRINGFIELD
```

AWP.MAN.SMM.A2  9-71

– Hard copies uncontrolled. Verify effective date prior to use. –
Marine Observations

Enter **!!!!MAR_STATIONS** to identify that marine stations follow in the station list. Marine stations are identified by their alphanumeric identifier (e.g., WECB, MISM1, 41002). They are listed in the order they are to be output, one station per line.

Place one slash “/” before the identifier and two slashes after it.

After the second slash, place either the plus sign “+” or minus sign “-” missing station flags as described under the Fixed Phrase Stations. Enter the marine station name after the missing station flag. As many as 18 characters may be used for the station name.

Use the keyword **LATLON** surrounded by percent signs (%) in place of the marine station name to output the station latitude and longitude instead of its name.

Marine stations are grouped together in segments of up to 10 stations. Each segment ends with ~$$ as shown in the following example.

```
~!MOZ038>042-044>050-056>058-069>072-082-083-%DDHHMM%
~...East Coast...
!!!!MAR_STATIONS
/MISM1//-MARTINICUS ISLE
/MDRM1/+MT DESERT ROCK
/SAUF1//-ST. AUGUSTINE
~$$
~!MOZ008>010-017-018-061-063>065-%DDHHMM%
~...West Coast...
!!!!MAR_STATIONS
/POTA2//-POTATO POINT
/FFIA2/+FIVE FINGER LH
/BLIA2/+LATLON%
/TTWI1/+TATOOSH ISLAND
/41002/+LATLON%
~$$
```

### 9.3.3.4 Updating Station PILs

**Hourly Weather Roundup PILs Editor**

The **Hourly Weather Roundup PILs Editor** is the interface that reads the `/awips/adapt/hwr/etc/pil.data` file and creates a listing of NWR and NWWS stations where each station mapped to their respective AFOS site ID and SCD ID.

The `/awips/adapt/hwr/etc/pil.data` configuration file contains a list of AWIPS station IDs, AFOS METAR IDs, and SCD IDs derived from the `SFXA_HOME/data/localization/nationalData/afosMasterPIL.txt` configuration file. It is used to allow data retrieval from the text database by providing the full AFOS IDs for METAR and SCD reports for each AWIPS site ID listed. The values in the first column are AWIPS site IDs. The values in the second column are AFOS METAR IDs. The
values in the third column are SCD site IDs. If no AFOS or SCD site ID is available, the value of ‘XXXXXXXXXX’ may be added its respective columns.

The pil.data file should be updated every time a new product is added or a station is added or deleted from a product. Also, the pil.data file should only be updated through the HWR PILs Editor GUI. Saving changes writes to the /awips/adapt/hwr/etc/pil.data file. The afosMasterPIL.txt file is not affected.

To access the HWR PILs Editor from the System Control Menu, select Background WFO Apps, Hourly Weather Roundup (HWR) from the cascading menu. This will display the Hourly Weather Roundup GUI shown in Exhibit 9.3.3.2-1.

Select: PILs pushbutton

The PILs Editor shown in Exhibit 9.3.3.4-1 will appear.

![PILs Editor GUI](image)

Exhibit 9.3.3.4-1. PILs Editor

Once the PILs Editor has opened, you have the following options:

Menu Options

File

<table>
<thead>
<tr>
<th>Print:</th>
<th>Invokes the print dialog.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Update:</td>
<td>Updates AFOS site IDs for all NWR and NWWS products based on the SFXA_HOME/data/localization/nationalData/afosMasterPIL.txt file. Update will search the afosMasterPIL.txt file for CCCMTRXXX and CCCSCDXXX entries where XXX is the AWIPS site ID. If a match is not found, the AFOS site ID is displayed as XXXXXXXXXX (and the retrieval of METAR/SCD report is not attempted). If a station entry is listed in the text widget, the METAR and SCD IDs are not updated.</td>
</tr>
</tbody>
</table>
Replace: Replaces the entire list based on the
/data/fxa/nationalData/afosMasterPIL.txt file. All entries listed in the
text widget are replaced.
Close: Exits from the PILs Editor GUI.

Edit
Cut: Removes highlighted text from the widget and saves to a buffer.
Copy: Makes a copy of highlighted text from the widget to a buffer.
Paste: Copies text from the buffer to the widget.
Find: Search and replace utility for finding text within the widget list.

Button Options
Clear Clears all text from the widget.
Open Loads PILs data from /awips/ adapt/hwr/etc/pil.data file.
Save Saves content of the text window to the /awips/ adapt/hwr/etc/pil.data
file.
Insert Toggles on and off text insert mode.

9.3.3.5 Treatment of Missing Data

The HWR NWWS application treats missing data as described below.

If the entire observation for the station is missing and the station is marked with a “+” or
“=” symbol in the station list, the station is output as “NOT AVBL.”

If the entire report for a station is missing and the station is marked with a “-” or “&”
symbol in the station list, the station is not output.

If wind speed is missing, “MISG” is output in the wind field. If sky/weather, temperature
in degrees Fahrenheit, dew point, relative humidity, or pressure is missing, “N/A” is
output in the respective field. Up to two weather elements may be output as remarks in a
fixed priority order. Thus, if one element is missing, the weather element with the next
highest priority is output. If only one of the remark field’s weather elements is available,
only that element will be output. If none of the remark’s section weather elements are
available, this field is left blank. If none of the 6-hour, additive data, weather elements
are available, these fields will be skipped.

Low-level, cloud cover observations are supplemented with satellite-derived middle and
upper-level cloud cover observations which result in the handling of missing data as
follows:

- If the ASOS cloud cover observation is missing, but the SCP product, which
  contains information on only middle and high level cloud cover is available, the
  cloud cover is considered missing.
Chapter 9 Background Applications

- If the ASOS observation shows overcast conditions, the SCP product is not needed, since the sky conditions will be reported as cloudy.

- If the ASOS observation shows broken clouds, the SCP product will be accessed to determine if the sky is overcast. If the SCP product is missing, the sky conditions will be reported as **PARTLY SUNNY** or **MOSTLY CLOUDY**, depending on the time of day.

### 9.4 Hourly Weather Roundup Formatter for the NOAA Weather Radio

The HWR is a summary of the current weather observations for locations in and around the local area including marine observations for those WFOs with coastal responsibilities.

The HWR NWR formatter application prepares voice-ready products for NWR broadcast. The products are automatically sent to the NWR across the AWIPS LAN to the CRS LAN or via a computer-to-computer link to a site-provided PC running Bubble and Airwave, which will transfer the products to the CRS, but the forecaster is given the opportunity to review and edit them.

The Hourly Weather Roundup uses
- METAR observations
- Marine observations
- SCD
- SCP.

The HWR NWR application (**hwrnwr**) uses METAR and marine observations, the SCP, and SCD to perform required calculations; build phrases; and add instructions for the CRS. The output product is stored temporarily and the WFO is notified of its completion.

**NOTE:** The forecaster-entered **METARINT** value in the **nwr.config** file specifies the number of minutes before the hour at which to start including the METAR observations in the HWR. The HWR considers any observations from H-METARINT up to but not including H+(60-METARINT) as an observation for the nominal hour. For example, if the METARINT value is 15, observations at 0845Z are considered 0900Z observations but observations at 0840Z are considered 0800Z observations. So, if an observation comes in at H+0:40, HWR will consider that observation to be from last hour and reject it.

At this point, the forecaster uses the NWR Browser to:
- View the list of products that are available for viewing and editing but have not yet been sent to the CRS.
- View the list of products that have been sent to the CRS.
- Select a product from the **Delayed** or **Sent lists** and **View** (not edit) in a separate window.
• Select a product from the Delayed or Sent lists and Edit the product in the NWR editor.
• Send the selected product to the CRS.
• Refresh the Delayed and Sent lists.
• Display a list of log files. When the Log File list is displayed, the View Logs button becomes a Refresh Log Window button.

9.4.1 Configuration Files and Environment Variables

Environment Variables

The following environment variables are referenced in this chapter for user fxa and individual user accounts and their full paths are provided below.

$LOG_DIR  /data/logs/fxa
$FXA_HOME  /awips/fxa
$FXA_DATA  /data/fxa

Configuration Files

nwr.config

The nwr.config configuration file contains:

• The list of products to be produced for the NWR
• The paths to the NWRBrowser, transferNWR script, and destination directory
• The allotted forecaster review period before the product is sent to the CRS
• The number of minutes before the hour at which to start including METAR observations in the HWR.

The file resides in the /awips/adapt/hwr/etc/nwr directory. An example of a nwr.config file follows.

```
# nwr.config
# configuration file for hwrnwr
#
GUI  /awips/fxa/bin/NWRBrowser
SCRIPT /awips/fxa/bin/transferNWR -a
DESTDIR /data/fxa/workFiles/nwr/pending
REVIEW 7
METARINT 15 # minutes before the hour at which to start using METARS in HWR
#
TEMP_PHRASE 0 # 1 = Descriptive phrasing; 0 = Specific numbers
# Flags associated with PRODUCT ID below should only rarely be changed. # See users' manual
# Product list. The following products will be generated:
PRODUCT HFOHWRNW1 AD_ 0
PRODUCT HFOHWRNW2 AD_ 1
```
The `nwr.cfg` file is a dedicated configuration file that holds the username, a line of dummy text as a placeholder (this will be the NWRCRS password until CRS Build 10 is released) and the password for Telnet and FTP and the connectivity type (PC or LAN). The information must be kept exactly in the previously mentioned order. The file resides in the `$FXA_DATA/workFiles/nwr` directory.

### 9.4.2 Processes

- FX1 `hwrnwr`
- LX `hwrgui`
- LX `hmMonitor.tcl`
- LX `hmMonitorWish`
- PX1 `hmMonitorServer`
- LX `NWRBrowser`
- LX `transferNWR`
- PX1 `nwrTrans.pl`
- PX1 `sendToNWR`

Refer to Exhibit 9.4.2-1 for the HWR Formatter for NWR data flow diagram.

#### hwrnwr

The **hwrnwr** process requires the following input:

- Decoded marine observations from netCDF files
- Decoded METAR observations from the text database (`fxatext`)
- SCD and SCP data from the text database (`fxatext`)
- Setup data, which includes the station lists, broadcast records, and defaults records from the `CCCHWRXXX.setup` and `CCCHWRXXX.product` files
- Configuration information and threshold selections from the `nwr.config` file.

The hwrnwr process performs required calculations on land and marine hourly weather data and surface and satellite, builds phrases, and adds instructions for the CRS. The output product is stored temporarily and the forecaster is notified of its completion. The forecaster may change the text product using the NWRBrowser before the product is sent to the CRS. The product must be sent to the CRS using a pushbutton (NWREditor) on the browser which invokes the transferNWR script. If the forecaster does not view, edit, and send the product manually to the CRS via the browser, hwrnwr will, after a forecaster-specified period of time, send the product automatically to the CRS.
Exhibit 9.4.2-1. Hourly Weather Roundup Formatter for NOAA Weather Radio
To View the Current Day’s hwrnwr Log

On PX1 as user fxa or your individual user account,

TYPE:  cd $LOG_DIR/<yyyymmdd>
       Where <yyyymmdd> is today’s date.

TYPE:  ls -ltr hwrnwr*
       Displays the current day’s hwrnwr logs from earliest to latest.

TYPE:  tail -f <logname>
       Displays the log as it updates with the latest messages.

Log Message Format

An entry in the log should look like the following:

LOG-STATUS: Log file opened on host px1-tbw4 at Wed May 12 04:10:38 2006
04:10:37.770 C_hmHMU_logError.C PROBLEM:
WARNING message from function/subroutine “get_METARs”: No reports were retrieved by get_report.

04:10:45.578 C_hmHMU_logError.C PROBLEM:
WARNING message from function/subroutine “get_METARs”: No reports were retrieved by get_report.

The log consists of the following fields:

- Time stamp
- File name
- Message type
- Message string

hwrgui

For hwrnwr to require/allow configuration and threshold information from the forecaster, it enables the forecaster to make selections using the GUI controlled by processes launched by the hwrgui process. The forecaster selections are validated and stored in the CCCHWRXXX.setup file.

The hwrgui process uses the station format editor to create the station list for each output product. Each list contains the METAR and SCD station identifier, the SCP station identifier, a symbol to indicate how to treat missing data, and the station name. The product IDs and the station lists are stored in the CCCHWRXXX.product file. The hwrgui process stores the broadcast and default broadcast records in the CCCHWRXXX.n.format table.

To View the Current Day’s hwrgui Log

On the workstation, as user fxa or your individual user account,
Log Message Format

An entry in the log should look like the following:

```
hwrgui 3611 975425655.214865 15:34:15.214 PROBLEM: WARNING message from function/subroutine “get_METARs”: No reports were retrieved by get_report.
```

The log consists of the following fields:

- **Time stamp**: 15:34:19.125
- **Module name**: hwrnwr
- **Message type**: PROBLEM:
- **Message string**: WARNING message from function/subroutine “get_METARs”: No reports were retrieved by get_report

hmMonitor.tcl

The Monitoring Controller process is the GUI by which a user is alerted to the receipt of a notification message sent by the `hmMonitorServer`. The `hmMonitor.tcl` script executes `hmMonitorWish` to create the `hmMonitor GUI`. The product ready notification is passed to the GUI by the `hmMonitorServer`. When the HWR button is pressed in the Monitoring Controller GUI, the CRS editor containing the HWR message is started and a message is sent to the Monitoring Controller to delete the notification message from other monitoring controllers. The `hmMonitor GUI` notifies the `hmMonitorServer` when new notifications arrive.

Restart Monitor Controller Window

The Monitoring Controller Window displayed on the text workstation is used to monitor HWR and Climate applications. If this window is closed, it can be restarted by executing the following command on the workstation as user `fxa` or your individual user account:

```
TYPE: setenv DISPLAY xt<#>-<siteID>:0.0
```

Where `<#>` is the text workstation number and `<siteID>` is the site ID (e.g., xt2-hgx).
The `hmMonitorWish` process is the main process for the Monitoring Controller GUI and also is responsible for all necessary initializations.

**To View the Current Day’s `hmMonitorWish` Log**

On the workstation, as user fxa or your individual user account,

**TYPE:**

```
cd $LOG_DIR/display/<display>/yyyyymmdd
```

Where `<display>` is the current display variable (e.g., lx1-mrx:0) and `<yyyyymmdd>` is today’s date.

**TYPE:**

```
ls -ltr hmMonitorWish*
```

Displays the current day’s `hmMonitorWish` logs from earliest to latest.

**TYPE:**

```
tail -f <logname>
```

Displays the log as it updates with the latest messages.

**Log Message Format**

An entry in the log should look like the following:

```
hmMonitorWish 10489 979037926.356198 10:58:46.356 PROBLEM: Unable to retrieve METARS for @bM-
```

```
hmMonitorWish 10489 979041527.857008 11:58:47.857 PROBLEM: Unable to retrieve METARS for Wed 12 May 2006 06:58 EST
```

```
hmMonitorWish 10489 979035409.064146 10:16:49.064 PROBLEM: Unable to retrieve METARS for Wed 12 May 2006 10:16 UTC
```

```
hmMonitorWish 10489 979037923.611689 10:58:43.611 BUG: Call to UpdateTextWindow failed: can’t read "w": no such variable
```

```
LOG-STATUS: Recent traceback from same location in file suppressed.
```

```
hmMonitorWish 10489 979035409.064146 10:16:49.064 PROBLEM: Unable to retrieve METARS for Wed 12 May 2006 06:58 EST
```

```
hmMonitorWish 10489 979035409.064146 10:16:49.064 PROBLEM: Unable to retrieve METARS for Wed 12 May 2006 10:16 UTC
```

```
hmMonitorWish 10489 979035409.064146 10:16:49.064 PROBLEM: Unable to retrieve METARS for Wed 12 May 2006 10:16 UTC
```

```
hmMonitorWish 10489 979035409.064146 10:16:49.064 PROBLEM: Unable to retrieve METARS for Wed 12 May 2006 10:16 UTC
```

```
hmMonitorWish 10489 979035409.064146 10:16:49.064 PROBLEM: Unable to retrieve METARS for Wed 12 May 2006 10:16 UTC
```

```
hmMonitorWish 10489 979035409.064146 10:16:49.064 PROBLEM: Unable to retrieve METARS for Wed 12 May 2006 10:16 UTC
```

```
hmMonitorWish 10489 979035409.064146 10:16:49.064 PROBLEM: Unable to retrieve METARS for Wed 12 May 2006 10:16 UTC
```

```
hmMonitorWish 10489 979035409.064146 10:16:49.064 PROBLEM: Unable to retrieve METARS for Wed 12 May 2006 10:16 UTC
```

```
hmMonitorWish 10489 979035409.064146 10:16:49.064 PROBLEM: Unable to retrieve METARS for Wed 12 May 2006 10:16 UTC
```

```
hmMonitorWish 10489 979035409.064146 10:16:49.064 PROBLEM: Unable to retrieve METARS for Wed 12 May 2006 10:16 UTC
```

```
hmMonitorWish 10489 979035409.064146 10:16:49.064 PROBLEM: Unable to retrieve METARS for Wed 12 May 2006 10:16 UTC
```

```
hmMonitorWish 10489 979035409.064146 10:16:49.064 PROBLEM: Unable to retrieve METARS for Wed 12 May 2006 10:16 UTC
```

```
hmMonitorWish 10489 979035409.064146 10:16:49.064 PROBLEM: Unable to retrieve METARS for Wed 12 May 2006 10:16 UTC
```

```
hmMonitorWish 10489 979035409.064146 10:16:49.064 PROBLEM: Unable to retrieve METARS for Wed 12 May 2006 10:16 UTC
```

```
hmMonitorWish 10489 979035409.064146 10:16:49.064 PROBLEM: Unable to retrieve METARS for Wed 12 May 2006 10:16 UTC
```

```
hmMonitorWish 10489 979035409.064146 10:16:49.064 PROBLEM: Unable to retrieve METARS for Wed 12 May 2006 10:16 UTC
```

```
hmMonitorWish 10489 979035409.064146 10:16:49.064 PROBLEM: Unable to retrieve METARS for Wed 12 May 2006 10:16 UTC
```

```
hmMonitorWish 10489 979035409.064146 10:16:49.064 PROBLEM: Unable to retrieve METARS for Wed 12 May 2006 10:16 UTC
```

```
hmMonitorWish 10489 979035409.064146 10:16:49.064 PROBLEM: Unable to retrieve METARS for Wed 12 May 2006 10:16 UTC
```

```
hmMonitorWish 10489 979035409.064146 10:16:49.064 PROBLEM: Unable to retrieve METARS for Wed 12 May 2006 10:16 UTC
```

```
hmMonitorWish 10489 979035409.064146 10:16:49.064 PROBLEM: Unable to retrieve METARS for Wed 12 May 2006 10:16 UTC
```

```
hmMonitorWish 10489 979035409.064146 10:16:49.064 PROBLEM: Unable to retrieve METARS for Wed 12 May 2006 10:16 UTC
```

```
hmMonitorWish 10489 979035409.064146 10:16:49.064 PROBLEM: Unable to retrieve METARS for Wed 12 May 2006 10:16 UTC
```

```
hmMonitorWish 10489 979035409.064146 10:16:49.064 PROBLEM: Unable to retrieve METARS for Wed 12 May 2006 10:16 UTC
```

```
hmMonitorWish 10489 979035409.064146 10:16:49.064 PROBLEM: Unable to retrieve METARS for Wed 12 May 2006 10:16 UTC
```

```
hmMonitorWish 10489 979035409.064146 10:16:49.064 PROBLEM: Unable to retrieve METARS for Wed 12 May 2006 10:16 UTC
```

```
hmMonitorWish 10489 979035409.064146 10:16:49.064 PROBLEM: Unable to retrieve METARS for Wed 12 May 2006 10:16 UTC
```
The log consists of the following fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time stamp</td>
<td>01:01:52.600</td>
</tr>
<tr>
<td>Module name</td>
<td>hmMonitorWish</td>
</tr>
<tr>
<td>Message type</td>
<td>BUG:</td>
</tr>
<tr>
<td>Message string</td>
<td>Call to UpdateTextWindow failed: can’t read</td>
</tr>
<tr>
<td></td>
<td>“w”: no such variable</td>
</tr>
</tbody>
</table>

hmMonitorServer

The Monitoring Controller process is the GUI by which a forecaster is alerted to the receipt of a notification message sent by the hmMonitorServer.

If the value of reviewPeriod in the /awips/adapt/hwr/etc/nwr/nwr.config configuration file is defined as greater than zero, hwrnwr sends a message to hmMonitorServer, which in turn notifies hmMonitor GUI. When the forecaster is notified and selects the HWR button, the NWR browser will appear. The product may be viewed, edited, and transmitted as the user wishes.

Under normal operations, hmMonitorServer starts as part of the site startup process. If it dies, the Climate program will notify users, and it can be restarted manually.

To Restart the hmMonitorServer Process

On PX1, as user fxa,

**TYPE:**

- `cd /awips/fxa/bin`
  - Changes to the directory where the hmMonitorServer resides.

**TYPE:**

- `hmMonitorServer &`
  - Starts the hmMonitorServer as a background process.

To verify the process is running:

**TYPE:**

- `ps -ef | grep hmMon`
  - Displays the hmMonitorServer process, as shown in the following example:

```
fxa 29894     1  1  May  7  ?        24:42 /awips/fxa/bin/hmMonitorServer
```

**NOTE:** Unless the hmMonitorServer process is started on PX1 by user fxa, there will be no communications with the clients.

If the hmMonitorServer shuts down, the server maintains a list of its current clients in the file hmMonitorServerClientList.txt located in SFXA_DATA/workFiles. When the server is restarted, it rereads this file to re-register any previous clients. If those clients are still running, they will continue to receive notifications once the hmMonitorServer process has restarted. If those clients are no longer running, they will be removed from the client list once the first attempt to send a message to that client fails.
To View the Current Day’s hmMonitorServer Log

On PX1 as user fxa or your individual user account,

**TYPE:**

- `cd $LOG_DIR/<yyyymmdd>`
  - Where `<yyyymmdd>` is today’s date.

**TYPE:**

- `ls -ltr hmMonitorServer*`
  - Displays the current day’s hmMonitorServer logs from earliest to latest.

**TYPE:**

- `tail -f <logname>`
  - Displays the log as it updates with the latest messages.

Log Message Format

An entry in the log should look like the following:

```
16:08:05.094 DataSocket.C DEBUG: Connection to lx1-tbw4/3037/18692 has been closed by the peer.
16:08:05.142 DataSocket.C DEBUG: Closing the socket to lx1-tbw4/3037/18692
16:11:56.586 HmMonitorServer.C EVENT: Saving client state...
16:11:56.643 HmMonitorServer.C EVENT: Received client notification message from px1-tbw4/4635/25150
  Target: lx1-tbw4/3037/18692
  Module: 26
  Type: 0
16:11:56.668 HmMonitorServer.C EVENT: Try again later -- unable to send update notification to client lx1-tbw4/3037/18692
```

The log consists of the following fields:

- **Time stamp** 16:08:05:142
- **File name** DataSocket.C
- **Message type** DEBUG:
- **Message string** Closing the socket to lx1-tbw4/3037/18692

NWRBrowser

The NWRBrowser is a GUI with a threefold purpose:

1. It allows the forecaster to remotely control the NWR from an AWIPS workstation.
2. It allows the forecaster to monitor product transfer from AWIPS to the NWR.
3. It provides last-minute edit capability for products that are to be sent to the NWR from AWIPS.
The `sendToNWR` Expectscript transmits voice-ready text from AWIPS to the NWR. In addition to transferring the product, copies can be kept in one of two directories:

- `SFXA_DATA/workFiles/nwr/pending`
- `SFXA_DATA/workFiles/nwr/sent`

When `sendToNWR` is invoked, the product is automatically copied to the `pending` directory. If successfully sent, the product is moved to the `sent` directory. The contents of each directory can be viewed in the window pane dedicated to that directory. Products not successfully transmitted will remain in the `pending` directory.

**transferNWR**

The `transferNWR` script is responsible for transferring voice-ready products to a holding area where they will be picked up and sent to NWR. The `transferNWR` script is a command-line interface that provides the forecaster with three options:

- **transferNWR -a [filename]**
  This option will automatically transfer the file from the source directory to the `/data/fxa/workFiles/nwr/ready` directory. The full pathname of the file must be given so that the file can first be copied into the source directory. The `transferNWR` script then monitors the `SFXA_DATA/workFiles/nwr/pending` directory to verify that the product has been successfully sent to CRS by the `sendToNWR` script.

- **transferNWR -d [filename]**
  This option will also ensure that the file is copied into the source directory, but the file will not be transferred. This option is for cases where the forecaster would first like to inspect or edit the product before it is sent.

- **transferNWR –n**
  This option sends a warning message to the local workstation that there has been a problem in transmitting the file. This message is sent at the forecaster’s discretion, based upon the return and exit values issued by this script.

The `transferNWR` script is forked by a parent process, `NWRBrowser`. The `transferNWR` script returns one of the following two values to the parent process:

- **Value = 0** indicates all was successful
- **Value = 1** indicates that the transfer may have failed

Log files can be viewed from the `NWRBrowser`. The most current log is listed last. Logs can also be viewed by other text editors, such as the `vi` editor.

- **To View the Current Day’s transferNWR Log**
  On the workstation, as user fxa or your individual user account,
TYPE:  cd $LOG_DIR/display/<display>/<yyyymmdd>
    Where <yyyymmdd> is today’s date and <display> is the
    current display variable (e.g., lx1-mrx:0).

TYPE:  ls -ltr logStreamExpect*
    Displays the current day’s transferNWR logs from earliest
    to latest.

TYPE:  tail -f <logname>
    Displays the log as it updates with the latest messages Log
    Message Format.

An entry in the log should look like the following:

LOG-STATUS: Log file opened on host pxl-tbw4 at Wed May 12 18:19:05 2006
18:19:05.358 TclLogStream.C EVENT: File /data/fxa/workFiles/nwr/pending/PITHWRNW1 will
be automatically transferred
18:19:05.359 TclLogStream.C EVENT: /data/fxa/workFiles/nwr/pending/PITHWRNW1 has been
copied to data/fxa/workFiles/nwr/ready/PITHWRNW1 for processing by sendToNWR.

NOTE:  Check the logStreamExpect log on PX for information about transfer of
data/fxa/workFiles/nwr/pending/PITHWRNW1 to CRS.

The log consists of the following fields:

<table>
<thead>
<tr>
<th>Time stamp</th>
<th>18:19:05.358</th>
</tr>
</thead>
<tbody>
<tr>
<td>File name</td>
<td>TclLogStream.C</td>
</tr>
<tr>
<td>Message type</td>
<td>EVENT:</td>
</tr>
<tr>
<td>Message string</td>
<td>File /data/fxa/workFiles/nwr/pending/PITHWRNW1 will be automatically transferred</td>
</tr>
</tbody>
</table>

nwrTrans.pl

The nwrTrans.pl Perl script is a persistent script that looks in
/data/fxa/workFiles/nwr/ready every couple of seconds to find files that have been
placed there by either the NWRBrowser or nwrEditor GUIs when a user requests to
send a product to the CRS for broadcast on the NOAA Weather Radio. This script then
calls transferNWR for any files that were found. Previously, transferNWR was called
directly from the NWRBrowser or nwrEditorWish. transferNWR is always run as
user fxa to ease maintenance of the NWRCRS’s authorized keys file on the CRS. This
change was required by the move to secure shell (ssh) and individual user accounts.

sendToNWR

The sendToNWR Expect script transmits voice-ready text is transmitted from AWIPS to
the NWR. In addition, copies of the product can be kept in one of two directories:

$FXA_DATA/workFiles/nwr/pending
$FXA_DATA/workFiles/nwr/sent
The `sendToNWR` script is called from the `nwrTrans.pl` script, which is a persistent `fxa` process that checks the `/data/fxa/workFiles/nwr/ready` directory every 2 seconds for products placed there by the `transferNWR` script. When `nwrTrans.pl` finds a file, or files, in this directory, it calls `sendToNWR` for each file.

The `sendToNWR` script is responsible for transferring voice-ready products from AWIPS to the NWR. The `sendToNWR` script is a command-line interface that provides the forecaster with three options:

- **`sendToNWR -a [filename]`**
  - This option will automatically transfer the file from the source directory on AWIPS to the destination directory on the CRS. The full pathname of the file must be given so that the file can first be copied into the source directory.

- **`sendToNWR -d [filename]`**
  - This option will also ensure that the file is copied into the source directory, but the file will not be transferred. This option is for cases where the forecaster would first like to inspect or edit the product before it is sent.

- **`sendToNWR –n`**
  - This option sends a warning message to the local workstation that there has been a problem in transmitting the file. This message is sent at the forecaster’s discretion, based upon the return and exit values issued by this script.

**NOTE:** The `transferNWR` script also may continue to be used for the `-d` and `-n` options.

When `sendToNWR` is invoked, the product is automatically copied to the pending directory. If successfully sent, the product is moved to the sent directory. The contents of each directory can be viewed in the windowpane dedicated to that directory. Products not successfully transmitted will remain in the pending directory.

To View the Current Day’s `sendToNWR` Log

On PX1, as user `fxa` or your individual user account,

**TYPE:**

- `cd $LOG_DIR/<yyyymmdd>`
  - Where `<yyyymmdd>` is today’s date.

- `ls -ltr logStreamExpect`
  - Displays the current day’s log.

- `tail -f <logname>`
  - Displays the log as it updates with the latest messages Log Message Format.

An entry in the log should look similar to the following:
The log consists of the following fields:

- **Time stamp**: 00:12:10.543
- **File name**: TclLogStream.C
- **Message type**: EVENT
- **Message string**: File /data/fxa/workFiles/nwr/pending/BOSHWRNW2 will be automatically transferred

The logs are purged by the **scour** script after 2 days and the empty directories are removed by the **master.purge** script.

### 9.4.3 Setting Up the HWR NWR Application

Before it can be used, the HWR NWR application must be set up (localized) for your site. Specify options and thresholds using the HWR NWR Setup Product options:

- Create a Station List
- Create a Broadcast Format
- Create a Default Format
- Update the Station PILs list.
**NOTE:** Setting up the HWR NWR application only needs to be done upon the initial application installation and when creating a new product identifier.

### 9.4.3.1 Identifying the Products to be Created

To identify the products to be produced, press the **NWR** button in the **Product Setup** section of the HWR GUI. Either select an existing product and press **Load**, or create a new product by typing in the **Product Actions** section and press **Add**.

### 9.4.3.2 Selecting the HWR NWR GUI Options

To access the HWR Editor for NWWS from the root or **applauncher** menu, select **Background WFO Apps, Hourly Weather Roundup (HWR)** from the cascading menu. This will display the **Hourly Weather Roundup** GUI shown in Exhibit 9.3.3.2-1.

Select: NWR pushbutton in the **Product Setup** section.

- The Hourly Weather Roundup Editor for the NOAA Weather Radio shown in Exhibit 9.4.3.2-1 will appear.

Once the editor for NWR has opened, you will see the following setup options:

**Products.** This pushbutton displays the list of products that may be converted to output products. Each product must contain a Station List file (via the Station Editor), at least one Broadcast and one Default Format file. New products may also be created, as discussed in the Products Actions below.

**Editors**

- **Station**
  This editor pushbutton displays the Station List for review and edit.

- **Broadcast**
  A Product may have multiple Broadcast Formats. The Broadcast Format files are numbered zero through nine. The numbered pushbutton displays the Broadcast Format file numbers for selection by the forecaster. The Editor pushbutton displays the selected Broadcast Format file for review and edit. Note that Broadcast file zero is the default format, which is accessed whenever one or more of the weather element requested in the associated station file is missing.
Exhibit 9.4.3.2-1. The Hourly Weather Roundup Editor for the NOAA Weather Radio

Product Actions

Add Create a new product by typing the product ID in this field and pressing Add.

Verify After you have created a new product, select Verify to check that the required files exist for the product.

Save This selection saves all the setup and product data from the GUI.

Delete You may delete a product by selecting the product ID from the scrolled list and hitting Delete.

Help This selection displays general information about the various sections in the GUI.

Close Closes the GUI.

LAC. This is the Listening Area Code (LAC) field. Enter the zones for which the product must be transmitted. The CRS maps these zones to the correct transmitter.

NOTE: It is beyond the scope of this document to describe how the CRS maps the LAC text to the correct transmitters.
Expiration Time. The number of minutes entered into this field will be added to the nominal hour to determine the expiration time of the product.

Effective Time. The number of minutes entered into this field will be added to the nominal hour to determine the effective time of the product.

Temp Range (deg F). The number entered into this field determines the range of temperatures at or below that the temperatures for a group of stations in summary format will be summarized.

Window for Marine Obs (hours). By default, observations that are more than 1 hour old are considered invalid and are not used by the HWR application when preparing products. However, since many C-MAN, buoy, and ship observations are not reported every hour, you may specify a longer valid period for the Marine Observations. Marine observations, up to the number of hours entered in this window, will be considered valid and used in the product.

Express sky conditions as fair when SCP missing? Since ASOS METAR cloud observations do not exceed 12,000 feet, the sky/weather phrase may not be generated accurately when there is no significant weather and the ASOS cloud observation is not cloudy. Satellite cloud observations (the SCP) are used to supplement the ASOS observations under these conditions. If this option is selected, and the SCP is missing while the ASOS observation is clear or partly cloudy, the phrase “FAIR” will be expressed. Otherwise, when the SCP is missing, the sky/weather will be output as missing in the product.

Use descriptive phrasing for temperatures? In summary format, you can describe the temperature for a group of stations descriptively (e.g., mid 60's) numerically (63 to 65).

Time Zone. Specify the time zone to be used in the product.

Std Time All Yr? The Standard time all year option must be selected for those areas that do not switch to Daylight Savings Time.

CRS Flag. There are three flags used by the CRS that are user-selectable. The first flag was originally designed to permit the users to send a product to the CRS and either activate (A) its broadcast or store it (I). The C flag is used to indicate that the product is meant for the nearest version of console hardware. The second flag indicates whether the product should be deleted (D) after it expires, or saved (S). The last flag indicates whether the CRS should confirm (C) receipt of the product or not (blank).

Thresholds. The following threshold fields are used in conjunction with the Broadcast Format file for each product:

- Wind Speed. Specify the minimum wind speed at or above which a wind speed phrase is to be output.
- Wind Gust. Specify the minimum wind gust speed at or above which a wind gust phrase is to be output.
- **Visibility(mi)**. Specify the maximum visibility at or below which a visibility phrase is to be output.
- **Dew Point**. Specify the maximum dew point at or above which a dew point phrase is to be output.
- **Heat Index**. Specify the minimum heat index value at or above which a heat index phrase is to be output.
- **Wind Chill Index**. For the wind chill index to be calculated, the observed temperature must be at or below the value entered in **Max Temp(deg F)** and the observed wind speed must be at or above the value entered in **Min Wind Speed (mph)**. The calculated wind chill index must be at or below the value entered in **Max WCI** for a wind chill index phrase is to be output.

**Visibility Thresholds.** Click on the selection box next to an obscuration in order to set the visibility threshold value for that obscuration. This results in that obscuration (e.g., fog) being output in the HWR product when the visibility is at or below the selected value. A selection of Unlimited results in the obscuration always being output no matter what the visibility is.

### 9.4.3.3 Creating a Station List

The Station List is created using the Station Format Editor. You must create a Station List for each product. The Station List specifies the stations that are included in the product. A station may appear in multiple Station Lists.

**To Create a Station List**

**Access:** The HWR for NOAA Weather Radio Product Setup

**Select:** A Product

**Select:** The Station Editor button from the Editor’s section.

- Displays the default HWR NWR text editor for you to type the Station List for the selected product.

An example of a Station List is shown in Exhibit 9.4.3.3-1.

The contents and format of a Station List are described below. The software delivery contains a template under the CCCHWRXXX product ID that will help you create Station Lists for your site’s products.
Comment Lines

The NWR HWR application interprets any line in the Station List that begins with a tilde (~) as information which will appear verbatim in the broadcast message. Such lines may appear anywhere in the Station List. Symbolic words (shown in Exhibit 9.4.3.3-2) which the HWR NWR application converts into text (according to the list discussed below), can be inserted into these lines. A comment line may contain references to two time zones. This is useful for WFOs that span two time zones. It is also useful for sites within the same time zone when one of the sites is in Arizona or Indiana, which are always on standard time. Place brackets around words that are to be output only during Daylight Savings Time to alleviate repetitive phrasing of the time zones during Standard Time.
Any line starting with the pound symbol “#” is interpreted as being an internal Station List comment and is not included in the HWR NWR output product.

Each Station List may contain several different sections:

- A Fixed Station Format Section identified by the !!!!!FF_STATIONS line.
- A Marine Format section identified by the !!!!!MAR_STATIONS line.
- A Summary Format section identified by the !!!!!SUM_STATIONS line.
- A Broadcast Frequency section identified by the !!!!!BRDCST_FREQ line.

The “@” may be used instead of the “~” at the beginning of the Fixed Format Station section, Marine Observations section, and Station Summary Group section of a station list. The “@” may also be used inside the Station Summary Group section of the HWR NWR as is discussed in the Station Summary Group section of this document. However, a maximum of 15 comment lines starting with either “@” or “~” may be used in a station list. When all stations are marked with a “~,” the “@” is used for all comments associated with a section, and all of the stations in the section are missing, nothing will be broadcast for this section, not even the comment line.

Fixed Format Stations

1. Each Fixed Format Station section begins with !!!!!FF_STATIONS. Fixed format stations are identified by their four-character ICAO identifier (e.g., KCRW) in the order they are to be broadcast, one station per line.

2. Place a slash “/” on each side of the surface station METAR identifier.
3. Next, place the four-character ICAO identifier for the most representative SCP station followed by a slash, unless the SCP station ID is identical to the surface station ID. In such cases, it is not necessary to repeat the ID. Instead, place another “/” after the previous one.

4. Next, place the missing station flag which is either the plus sign “+” or the minus sign “-”.

   The “+” indicates that the corresponding station will always be broadcast, even if the weather elements are missing (e.g., “The report from Missoula was not available”).

   The “-” indicates that the station will be ignored if all the weather elements for the station are missing.

5. Enter the fixed phrase station name after the missing station flag. As many as 30 characters may be used for the station name.

A sample Fixed Format Station section follows:

```
# Fixed Format Station section
!!!!!!FF_STATIONS
~These are the %TIMEZ% observations for the local area, on ~%DAY%, %DATE%.
/KDCA/KDCA/*Reagan National Airport;/KADW/+Andrews Air Force Base
/KBIW/-Fort Belvoir
/KADW/+B W I Airport
/KADW/KDCA/+Andrews Air Force Base
/KAKR/HRIC/+Wakefield
```

**NOTE:** A Fixed Format Station section may contain a maximum of 20 station lines; however, you may have multiple Fixed Format Station sections.

**Station Substitution**

If all of a station’s observations are missing, another station may be used as a substitute. To specify a station substitution in the station list, place an asterisk after the first station identifier, a semicolon between the stations, and the missing station flag after the second station identifier. The “+” after the second station indicates that the first station will be output as missing if both stations are missing. A “-” after the second station indicates that both stations will be ignored if both stations are missing. For example,

```
/KDCA/*/Reagan National Airport;/KADW/+ANDREWS AIR FORCE BASE
```

In the example, Reagan National Airport will normally be broadcast. If its observations are missing, the observation from Andrews Air Force Base will be broadcast. If both stations are missing, the output will state that the report from Reagan National Airport is not available.
Marine Observations

1. Enter !!!!!MAR_STATIONS to identify that marine stations follow in the Station List. Marine stations are identified by their alphanumeric identifiers, which may be more than four characters (e.g., WECB, MIS51, 41002). They are listed in the order they are to be broadcast, one station per line.

2. Place one slash “/” before the identifier and two slashes after it.

3. After the second slash, place the missing station flag which is either the plus sign “+” or the minus sign “-.” For a discussion of the missing station flags, see the Fixed Format Stations subsection above.

4. Enter the marine station name after the missing station flag. As many as 30 characters may be used for the station name.

An sample Marine Observations Section follows.

```
# Marine Format
~Here are some CMAN reports.
!!!!!!MAR_STATIONS
/BLIA2//-Bligh Reef
/TTIW1/+Tatoosh Island
/MIS1/-Matinicus Isle
/SAUF1/-St. Augustine
#
~Here are some buoy reports.
!!!!!!MAR_STATIONS
/46006/+Buoy 46006
/44014/-Buoy 44014
/42019/-Buoy 42019
/51003/+Buoy 51003
```

NOTE: A maximum of 10 stations is permitted in each !!!!!MAR_STATIONS section; however, you may have multiple sections.

Station Summary Groups

Insert a tilde “~” and a comment line. This comment line will always be broadcast even if all data are missing or weather conditions are not similar and summary phrases are not created. Use the “@” instead of “~” to indicate to the program that the comment should be ignored if all stations within the summary group are missing.

1. Enter !!!!!WX_CONDS to identify a Station Summary group.

2. On the second line, list the weather elements to be summarized. The following weather elements may be summarized: sky/weather (element identifier SW), temperature in degrees Fahrenheit (element identifier TF), and temperatures in degrees Celsius (element identifier TC).

3. On the third line, enter !!!!!SUM_STATIONS to identify that the stations that follow it are to be summarized. List at least 3 but no more than 10 station identifiers, one
station per line, identified by their four-character ICAO identifier. Place a slash “/” on each side of the station identifier.

4. Next, place another four-character ICAO identifier for the closest SCP station followed by a slash, unless the station is the same as the first one on the line. In such cases, do not insert the ICAO identifier again, just place the second slash.

5. After the second slash, place the missing station flag which is either the plus sign “+” or the minus sign “-”

The “+” indicates that the corresponding station will always be broadcast, even if the weather elements are missing. An example of the output in such cases is “The report from Baton Rouge was not available.”

The “-” indicates that the station will be ignored if all the weather elements for the station are missing. When using the “@” at the beginning of the comment line outside of the summary group, also use the minus sign “-” for the missing station flags for each station.

6. Enter the station name after the missing station flag. As many as 30 characters may be used for the station name.

Inside the summary group, the “@” symbol is used to indicate that a comment follows. Comments may be placed anywhere in the summary phrase group. These comments within the summary group are only broadcast when similarities of weather conditions force the creation of a summary phrase. For example,

```
# Summary Format
~Here are some observations outside the metro area.
!!!!!WX_CONDS
SW TF TC
!!!!!SUM_STATIONS
@Throughout the region,
/KHGR//Hagerstown; /KNHK/+Patuxent River
/KMRY/+Martinsburg
/KSBY/+Salisbury
/KNHK/-Dover
/KCHO/-Charlottesville
/KADM/+Andrews Air Force Base
/KRIC/+Richmond
```

In this example, there’s a comment line identified by the tilde “~” outside of the summary group and a comment line inside the summary group identified by the “@.” The comment outside the group is always broadcast, whereas the comment inside the group is broadcast only when similarities of weather conditions force the creation of a summary phrase. Because the only weather elements eligible for summarizing, in this example and in this program, are sky/weather (SW), temperature in degrees Fahrenheit (TF), and temperatures in degrees Celsius (TC), there are four possible outcomes:

1. If both temperature and sky/weather conditions are uniform, the output will read one of two ways.

   The first option is for the output to read as follows:
“Here are some observations outside the metro area. Throughout the region, skies ranged from cloudy to mostly cloudy, and temperatures were in the mid and upper 30s.”

In this example, the temperatures are phrased as a range rather than specific values. The forecaster is provided with a temperature phrasing option in the GUI. These criteria are that the temperature range is less than 5 degrees Fahrenheit and all of the temperatures are higher than 13 degrees Fahrenheit and lower than 99 degrees Fahrenheit.

The second option is for the output to read as follows:

“Here are some observations outside the metro area. Throughout the region, skies ranged from cloudy to mostly cloudy, and temperatures were between 34 and 39 degrees.”

In this example, the temperatures are phrased as being bounded by the specific lowest observed temperature and the specific highest observed temperature.

2. If the sky/weather conditions are uniform (sky conditions do not differ by more than one category) across the region but not the temperatures, the output will read as follows:

“Here are some observations outside the metro area. Throughout the region, temperatures were between 84 and 88 degrees. It was sunny at Hagerstown, Salisbury, and Andrews Air Force Base. It was mostly sunny at Martinsburg and Richmond. At Dover, it was partly sunny. At Charlottesville, it was cloudy.”

In this example, the temperatures are worded as being bounded by the specific lowest observed temperature and the specific highest observed temperature. The forecaster is provided with a temperature phrasing option in the GUI.

The second option is for the output to read as follows:

“Here are some observations outside the metro area. Throughout the region, temperatures were in the middle and upper 80s. It was sunny at Hagerstown, Salisbury, and Andrews Air Force Base. It was mostly sunny at Martinsburg and Richmond. At Dover, it was partly sunny. At Charlottesville, it was cloudy.”

**NOTE:** The HWR NWR application arranges only three to five stations in a sentence.

3. If the temperatures are uniform (within the range specified in the HWR Editor for NWS) but not the sky/weather conditions, the output will read in one of two ways.

The first option is for the output to read as follows:

“Here are some observations outside the metro area. Throughout the region, temperatures were between 84 and 88 degrees. It was sunny at Hagerstown, Salisbury, and Andrews Air Force Base. It was mostly sunny at Martinsburg and Richmond. At Dover, it was partly sunny. At Charlottesville, it was cloudy.”

In this example, the temperatures are worded as being bounded by the specific lowest observed temperature and the specific highest observed temperature. The forecaster is provided with a temperature phrasing option in the GUI.

The second option is for the output to read as follows:

“Here are some observations outside the metro area. Throughout the region, temperatures were in the middle and upper 80s. It was sunny at Hagerstown, Salisbury, and Andrews Air Force Base. It was mostly sunny at Martinsburg and Richmond. At Dover, it was partly sunny. At Charlottesville, it was cloudy.”
In this example, the temperatures are phrased as a range (e.g., temperatures were in the middle and upper 80s) rather than the specific values.

4. If neither temperatures nor sky/weather conditions are uniform, the output will read as follows:

“Here are some observations outside the metro area. It was sunny, with a temperature of 34 at Hagerstown, 28 at Dover, and 26 at Richmond. Under mostly sunny skies, Martinsburg reported 22 degrees, and Charlottesville 35. At Salisbury, it was partly sunny, with a temperature of 36. At Andrews Air Force Base, it was cloudy with a temperature of 34.”

NOTE: In the last example, the phrase “Throughout the region” was not part of the broadcast statement as that phrase is only included when there are similarities in the weather conditions.

Broadcast Frequency Section

There is also a way to obtain additional variability from hour to hour for Fixed Phrase Format stations. Multiple broadcast records for each station list record may be created, each with a different fixed format for each station. Consequently, the probability with which each file is to be accessed must be provided in the station record. The program will randomly access the files according to the specified probability distribution. The total of the frequencies MUST be 100 (representing 100 percent). If only one broadcast record for a transmitter is needed, either omit the frequency from the station list record, or enter “100.” In the example below, the “10, 40, 50” values indicate that there are three broadcast records associated with this station list record. For example, assume this record is named CCCHWRNW5 (indicating that this is the file associated with the fifth output product). Also, assume the hwrnwr program is aware of three broadcast records, one of which will be selected this hour for product 5. hwrnwr will randomly generate a number to choose one of the three files. Thus, after multiple executions of the program, one would expect that broadcast record CCCHWRNW5.1 would be selected 10 percent of the time, while record CCCHWRNW5.2 would be selected 40 percent, and record CCCHWRNW5.3 would be selected 50 percent of the time. For example,

```
!!!!!BRDCST_FREQ
10 40 50
# Fixed Station Format section
!!!!!FF_STATIONS
# These are the %TIME% observations for the local area, on %DAY%, %DATE%.
/KDCA/KDCA/+Reagan National Airport
/KDAA/-Fort Belvoir
/KDWI/+B W I Airport;/KADW/+Andrews
/KADW/KDCA/+Andrews
/KAKQ/KRIC/+Wakefield
#
# Summary Format
# Here are some observations outside the metro area.
!!!!!WX_CONDS
SW TF
!!!!!SUM_STATIONS
```
@Throughout the region,
/WMFB//Martinsburg
/KHGR//Hagerstown, /CDEV/+Dover
/KNNK//Patuxent River
/KING/+Wilmington
/KSBY/+Salisbury
#
# Marine Format
~Here are some CMAN reports.
!!!!MAR_STATIONS
/BLIA2//-Bligh Reef
/TITI1/+Tatoosh Island
/MISM1//-Matinicus Isle
/SAUF1//-St. Augustine
~Here are some buoy reports.
#
!!!!MAR_STATIONS
/46006/+Buoy 46006
/44014//-Buoy 44014
/42019//-Buoy 42019
/51003/+Buoy 51003

9.4.3.4 Creating a Broadcast Format File

A Broadcast Format file (Exhibit 9.4.3.4-1) is created using the HWR NWR Broadcast Editor. Broadcast Format files contain Broadcast Format paragraphs that are used to create the broadcast text that is sent to the CRS. Broadcast Format paragraphs begin with the Station ID (enclosed in “/ /”) and end with “$$.” Broadcast Format paragraphs consist of verbatim text (i.e., to be broadcast verbatim) and symbolic text (i.e., to be replaced with data observations). The verbatim and symbolic text may be mixed in any manner that produces coherent broadcast text. Exhibit 9.4.3.4-2 shows the symbolic text that may be used.

You must have at least one Broadcast Format paragraph for each station (Fixed Format and Marine) in the associated Station List. Do not include Broadcast Format paragraphs for station summary groups. You will get an error message when the application runs, if the Broadcast Format file does not contain a Broadcast Format paragraph for a fixed format and Marine station in the associated Station List. Extra Broadcast Format paragraphs (that is, the station is not in the Station List, but is in the Broadcast Format file) are ignored.

NOTE: Broadcast Format files may contain a second Broadcast Format paragraph for some stations, refer to Section 9.4.3.6, Providing Additional Variability for the NWR Broadcasts.
Chapter 9  Background Applications

Exhibit 9.4.3.4-1. Sample Broadcast Format File for NWR
Exhibit 9.4.3.4-2. Symbolic Text for Use in Broadcast Format Files

It is very helpful to view the Station List file while editing the Broadcast Format file. This way you can easily view both files simultaneously.

For each product, multiple Broadcast Format files (1 through 9) may be created, each containing different Broadcast Format paragraphs for a given station. In this manner, when the HWR application selects different Broadcast Format files from hour to hour, there will be broadcast variation for the same station.

When initially installed, Product ID CCCHWRXXX contains a Broadcast Format template that may be used to create additional Broadcast Formats. Cut and paste as needed from the template.

To Create a Broadcast Format File

Select: A Product

Select: The Broadcast Format File number (1–9)

Select: Broadcast editor button

- The default HWR NWR text editor appears displaying the Broadcast Format File for the selected Product and Broadcast
Format File number. If no file exists, a pop-up will ask if you want to create one and will open a blank file in the text editor if you select Yes.

The following approach explains how to create Broadcast Format paragraphs from scratch.

**To Create a Sample Hourly Weather Roundup Report**

1. Write out a sample of an Hourly Weather Roundup station report for all land and marine stations which will have a fixed format. With the land observations, start with the assumption that rain was observed, and no thresholds were met. Sentences don’t need to be entirely grammatically correct, since they may be somewhat abbreviated in the normal broadcast fashion. However, commas must be inserted where a pause is required. For example,

   “At Reagan National Airport, rain was falling. The temperature was 57 degrees, the dew point 56, and the relative humidity 97 percent. The pressure, 29.92 inches and falling.

   The temperature was 56 degrees with rain at B W I Airport. The dew point 55, and the relative humidity 97 percent. The pressure, 29.97 inches and falling.

   Rain was observed at Dulles Airport. The temperature 56 degrees, the dew point 56, and the relative humidity 100 percent. The pressure, 29.89 inches and steady.

   Middle Nomad Buoy located 60 miles southeast of Norfolk recorded a south wind of 22 knots, swells 6 feet, air temperature 71 degrees, and water temperature 77 degrees.

   Ship Discovery located at 38 degrees north latitude and 73 degrees west longitude recorded southeast winds at 15 knots, air temperature 59 degrees, water temperature 63 degrees, with 5 foot waves.”

2. Open the desired Broadcast Format file and enter the text for the statements. Start each Broadcast Format Paragraph with the Station ID and end it with $$. See the example below.

   /KDCA/At Reagan National Airport, rain was falling. The temperature was 57 degrees, the dew point 56, and the relative humidity 97 percent. The pressure, 29.92 inches and falling.$$  

   /KBWI/The temperature was 56 degrees with rain at B W I Airport. The dew point 55, and the relative humidity 97 percent. The pressure, 29.97 inches and falling.$$  

   /KIAD/Rain was observed at Dulles Airport. The temperature 56 degrees, the dew point 56, and the relative humidity 100 percent. The pressure, 29.89 inches and steady.$$
/BLIA2/Middle Nomad Buoy located 60 miles southeast of Norfolk recorded a south wind 22 knots, swells 6 feet, air temperature 71 degrees, and water temperature 77 degrees.$$

/WBEC/Ship Discovery located at 38 degrees north latitude and 73 degrees west longitude recorded southeast winds at 15 knots, air temperature 59 degrees, water temperature 63 degrees, with 5 foot waves.$$

This is a good time to compare the Station List with the Broadcast Format. You must have at least one Broadcast Format Paragraph for each Station (Fixed Format and Marine) in the Station List.

Create Phrasing for Sky/Weather Conditions

The phrasing for the sky/weather conditions is the most complicated task. You must create four phrases that produce a coherent description of the sky/weather conditions. The HWR NWR application selects the correct format based upon the current sky/weather conditions.

1. Substitute, in turn, each of the following weather conditions in place of rain in the above examples, ensuring that the sentences still sound acceptable: sleet, snow, hail, drizzle, light snow, light rain, heavy snow, and heavy rain. If necessary, change any phrase so that all of these weather conditions, including rain, sound correct.

2. Place curly braces ({{}}) around the phrases with rain in the examples. Inside the curly braces, place a semicolon after the phrase with rain, followed by a phrase for the weather, assuming fog was reported instead of rain. For example,

   /KDCA/At Reagan National Airport, {{rain was falling; fog was reported}}. The temperature...

   /KBWI/The temperature was 56 degrees {with rain; with fog} at B W I airport...

   /KIAD/{{Rain was observed; Fog was observed}} at Dulles Airport. The temperature ...”

3. Substitute each of the following weather conditions in place of fog in the above examples, ensuring that the sentences still sound acceptable: thunder, volcanic ash, dust, sand, haze, smoke, spray, a dust devil, a squall, a thunderstorm, a thundershower, blowing snow, blowing dust, blowing sand, and blowing spray. If necessary, change the phrase so that it sounds correct with any of these weather elements. When substituting thunder for fog, you might not like the phrase, “thunder was observed” for the Dulles report. If not, change the phrase to something acceptable for all of the weather elements in this category, such as “thunder was reported.”

4. Place another semicolon after the phrase with fog. Create a phrase for sunny sky conditions. The CFS pauses for both periods and commas, thus the
incomplete sentence in the following example will sound correct, because it connects well with the phrase that follows it. For example,

/KDCA/At Reagan National Airport, {rain was falling; fog was reported; under sunny skies}. The temperature ...

/KBWI/The temperature was 56 degrees {with rain; with fog; with sunny skies} at B W I Airport...

/KIAD/{Rain was observed; Fog was reported; It was sunny} at Dulles Airport. The temperature ...

5. Substitute each of the following sky conditions in place of sunny in the above examples to ensure that the sentences still sound acceptable: cloudy, mostly cloudy, partly cloudy, clear, partly sunny, mostly sunny, and fair. If necessary, change any phrase so that all of these sky conditions, including “sunny,” sound correct.

6. Place another semicolon after the phrase with sunny sky conditions. Create a phrase for weather conditions stated in plural form (e.g., snow showers), or for multiple weather conditions (e.g., drizzle and fog). To make the file easier to read, you may place carriage returns anywhere. However, don’t drop the necessary spaces and punctuation marks. For example,

/KDCA/At Reagan National Airport, {rain was falling; fog was reported; under sunny skies; snow showers were reported}. The temperature was 57 degrees, the dew point 56, and the relative humidity 97 percent. The pressure 29.92 inches and falling.$$

/KBWI/ The temperature was 56 degrees {with rain; with fog; with sunny skies; with snow showers} at B W I airport. The dew point 55, and the relative humidity 97 percent. The pressure 29.97 inches and falling.$$ #

/KIAD/{Rain was observed; Fog was reported; It was sunny; Snow showers were observed} at Dulles Airport. The temperature 56 degrees, the dew point 56, and the relative humidity 100 percent. The pressure 29.89 inches and steady.$$ #

/BLIA2/Middle Nomad Buoy located 60 miles southeast of Norfolk recorded a south wind 22 knots, swells 6 feet, air temperature 71 degrees, and water temperature 77 degrees.$$ #

/WEBC/Ship Discovery located at 38 degrees north latitude and 74 degrees west longitude recorded southeast winds at 15 knots, air temperature 59 degrees, water temperature 63 degrees, with 5 foot waves.$$ 


7. Substitute each of the following weather conditions in place of snow showers in the above examples, ensuring that the sentences still sound acceptable: sleet showers, rain showers, and snow flurries. If necessary, change any phrase so that all of these weather conditions, including snow showers, sound correct. Also, try some combinations of multiple weather conditions to ensure that these combinations also sound acceptable (e.g., “snow and blowing snow were observed”).

- **To Add Threshold Phrases**

1. Decide which weather elements should be broadcast only if their threshold values have been met (e.g., Wind Speed, Wind Gust, Visibility, Heat Index, and Wind Chill Index).

2. Place in parentheses the phrases describing all the threshold-dependent weather elements which you are requesting, in the correct locations in the paragraph for each station. Be certain to determine if the punctuation marks and spaces belong inside or outside of the parentheses. It’s also possible to embed parentheses within parentheses. This is necessary because the entire sentence may be dependent on the threshold of a particular weather element, and if that threshold is met, another weather element may control the broadcast of a phrase within the sentence. For example,

```plaintext
/KDCA/At Reagan National Airport,
{rain was falling; fog was reported; under sunny skies; snow showers were reported}
(, which reduced the visibility to 3/4 of a mile).
The temperature was 57 degrees,
(, the dew point 56,)
and the relative humidity 97 percent
(, producing a heat index of 104 degrees).
(The wind northeast at 12 miles an hour [, gusting to 27])
[ , producing a wind chill of 8 degrees].
The pressure 29.92 inches and falling.$$
#
/KBWI/The temperature was 56 degrees,
{with rain; with fog; with sunny skies; with snow showers}
(, and a visibility of ½ mile) at B W I Airport. The dew point 55, and
the relative humidity 97 percent
(, yielding a heat index of 98).
(The wind [, east at 14 miles an hour], with gusts to 29 [, which makes
the temperature feel like 17 degrees].)
The pressure 29.97 inches and falling.$$#
/KIAD/(Rain was observed; Fog was reported; It was sunny; Snow showers
were observed)
(, which reduced the visibility to 1/4 of a mile) at Dulles Airport.
The temperature 56 degrees
(, the dew point 56,)
and the relative humidity 100 percent.
(The combination of temperature and humidity feels like 97 degrees.)
```
(Strong winds from the north [, at 11 miles an hour] [, gusting to 22], are producing a wind chill of 23 degrees.)
The pressure 29.89 inches and steady.$$
#
/BLIA2/Middle Nomad Buoy located 60 miles southeast of Norfolk recorded a south wind 22 knots, swells 6 feet, air temperature 71 degrees, and water temperature 77 degrees.$$
#
/WEBC/Ship Discovery located at 38 degrees north latitude and 74 degrees west longitude recorded southeast winds at 15 knots, air temperature 59 degrees, water temperature 63 degrees, with 5 foot waves.$$

► To Add Symbolic Weather Elements

Replace values with symbolic words, using the key in Exhibit 9.4.3.4-2. Place a U (for units) after the symbolic word to indicate that both the value and the units are to be output where desired. For example, if the temperature was 53 degrees, %TF%U would result in 53 degrees. On the other hand, %TF% would only be replaced with 53. Sky/weather, wind direction, heat index, wind chill index, swell direction, and pressure tendency have no units and should not be used with the U character. Also, the plural form of the unit is used whenever the value is not 1 or -1.

► To Add Substitute Stations

Create paragraphs for stations which substitute for another station, even if the format is identical. In our example, Andrews Air Force Base is a substitute station for Reagan National Airport.

/KDCA/At Reagan National Airport, {
%SW% was falling; %SW% was reported; under %SW% skies; %SW% were reported
(), which reduced the visibility to %V.V.%U.
The temperature was %TF%U, (the dew point %DF%), and the relative humidity %RR%U
(), producing a heat index of %HF%.
(The wind %DD% at %FF%U, (gusting to %GG%), (producing a wind chill of %WF%). )
The pressure %PP%U and %PT%.$$
#
/KADW/At Andrews Air Force Base,
{%SW% was falling; %SW% was reported; under %SW% skies; %SW% were reported
(), which reduced the visibility to %V.V.%U.

The temperature was %TF%U, (the dew point %DF%), and the relative humidity %RR%U
(), producing a heat index of %HF%U).
(The wind %DD% at %FF%U, (gusting to %GG%) [, producing a wind chill of %WF%U].)
The pressure %PP%U and %PT%.$$

/KBWI/The temperature was %TF%U (with %SW%; with %SW% skies; with %SW%)
(), and a visibility of %V.V.%U at B W I Airport. The dew point %DF%, and the relative humidity %RR%U, yielding a heat index of %HF%.}
The wind %DD% (at %FF%U), with gusts to %GG% (which makes the temperature feel like %WF%).

The pressure %PP%U and %PT%.$$
#
/KIAD/ (%SW% was observed; %SW% was reported; it was %SW%; %SW% were observed)
(, which reduced the visibility to %V.V.%U) at Dulles Airport.
The temperature %TF%U (the dew point %DF%), and the relative humidity %RR%U.
(The combination of temperature and humidity feels like %HF%U.) (Strong winds from the %DD% (at %FF%U), gusting to %GG%), are producing a wind chill of %WF%U.

The pressure %PP%U and %PT%.$$
#
/BLIA2/Middle Nomad Buoy located 60 miles southeast of Norfolk recorded %DD% winds %FK%U, swells %HS%U, air temperature %TF%U, and water temperature %SF%U.$$
#
/WEBC/Ship Discovery located at 38 degrees north latitude and 74 degrees west longitude recorded %DD% winds at %FK%U, air temperature %TF%U, water temperature %SF%U, with %HW% foot waves.$$

To Choose Weather Elements to Control Threshold Phrases

At the beginning of each threshold phrase, indicate by placing into brackets, the weather element which controls the reporting of the phrase. This is necessary, because some phrases contain two elements for which there are thresholds. For example,

/KDCA/At Reagan National Airport,
{ %SW% was falling; %SW% was reported; under %SW% skies; %SW% were reported}
([V.V.], which reduced the visibility to %V.V.%U).
The temperature was %TF%U( [DF], the dew point %DF%), and the relative humidity %RR%U
([HF], producing a heat index of %HF%U).
([FF] The wind %DD% at %FF%U([GG], gusting to %GG%)
([WF], producing a wind chill of %WF%U). )
The pressure %PP%U and %PT%.$$
#
/KADW/At Andrews Air Force Base,
{ %SW% was falling; %SW% was reported; under %SW% skies; %SW% were reported}
([V.V.], which reduced the visibility to %V.V.%U).
The temperature was %TF%U( [DF], the dew point %DF%), and the relative humidity %RR%U
([HF], producing a heat index of %HF%U).
([FF] The wind %DD% at %FF%U([GG], gusting to %GG%)
([WF], producing a wind chill of %WF%U). )
The pressure %PP%U and %PT%.$$
#
/KBWI/The temperature was %TF%U (with %SW%; with %SW%; with %SW% skies; with %SW%) 
([V.V.], and a visibility of %V.V.%U) at B W I Airport. The dew point %DF%, and the relative humidity %RR%U 
([HF], yielding a heat index of %HF%).
([GG] The wind %DD% ([FF] at %FF%U),
with gusts to %GG%
([WF], which makes the temperature feel like %WF%). )
The pressure %PP%U and %PT%.$$
9.4.3.5 Creating a Default Format File

For each product, you must also create a Default Format file. This file appears as Brdcst #0 in the NWR GUI. The Default Format is output only when one or more weather elements are missing from a decoded station. The Default Format is needed since the Broadcast Format permits the forecaster to compose output in an almost unlimited manner, and the HWR NWR application will not necessarily be able to coherently drop missing weather elements.

Consequently, the Default Format MUST conform to more rigid standards. Whereas the Broadcast Format File requires an entry for each station, the Default Format requires only one Broadcast Format Paragraph for land stations and one for marine stations. An example of a Default Format file is shown in Exhibit 9.4.3.5-1.

Exhibit 9.4.3.5-1. Sample Default Format File

The initial installation of the HWR NWR included a template for the default format. You may edit this template to match your requirements, or you may wish to follow the steps described below.

▲ To Create a Default Format File

Select: A Product
Select: The Broadcast Editor button, Brdcst #0

- The default HWR NWR text editor appears so that you can type in the Default Format file for the selected Product

To Create Sample Default Phrases for Land and Marine Observations

1. Write out a sample HWR for a land station and one for a marine station. Make sure that the HWR is constructed so that a punctuation mark follows the expression of each weather element. For example,

   “At Reagan National Airport, rain was reported. The temperature was 57 degrees, the dewpoint 56, and the relative humidity 97 percent. The pressure 29.92 inches, and falling.

   Middle Nomad Buoy recorded, a south wind, 22 knots, swells 6 feet, air temperature 59 degrees, water temperature 63 degrees, with wave height 5 feet.”

   In the above example, notice that if any of the weather elements are missing, the program can skip to the next punctuation mark, and the sentence is still readable, although it may not be grammatically correct. For example, if the temperature is missing at Reagan National Airport, the following phrase will be broadcast:

   “At Reagan National Airport, rain was reported. The dewpoint 56. The pressure 29.92 inches, and falling.”

   If the pressure is missing, the entire sentence is skipped, because the next punctuation mark is the end of the sentence.

   **NOTE:** The program cannot calculate the relative humidity without the temperature or a pressure tendency without a current or previous pressure.

2. Via the Default Format Editor, enter the text to match the HWR in Step 1. Wherever the sky/weather conditions are requested, substitute phrases for obstructions to vision, sky conditions, and multiple sky/weather conditions, in an identical manner to the technique described in Section 9.3.3.4. For example,

   At Reagan National Airport,
   
   { rain was falling; fog was reported; under sunny skies; snow showers were reported}.

   The temperature was 57 degrees(, the dewpoint 56,) and the relative humidity 97 percent.

   The pressure was 29.92 inches and falling.

   Middle Nomad Buoy recorded a south wind, 22 knots, swells 6 feet, air temperature 59 degrees, water temperature 63 degrees, with wave height 5 feet.
3. Threshold values may be added, but be careful that the sentence will make sense if any weather element is missing.

At Reagan National Airport,
{ rain was falling; fog was reported; under sunny skies; snow showers were reported}.

The temperature was 57 degrees, the dewpoint 56, and the relative humidity 97 percent (producing a heat index of 104 degrees).
The pressure was 29.92 inches and falling.

Middle Nomad Buoy recorded, a south wind, 22 knots, swells 6 feet, air temperature 59 degrees Fahrenheit, water temperature 63 degrees Fahrenheit, with wave height 5 feet.

4. Insert symbolic words, using the key in Exhibit 9.4.3.4-2, where observation values are to be placed. Indicate the weather element that controls the reporting of the threshold phrase. Examples are as follows:

At %ID%,
{ %SW% was falling; %SW% was reported; under %SW% skies; %SW% were reported}.

The temperature was %TF%U([DF], the dewpoint %DF%), and the relative humidity %RR%U
([HF], producing a heat index of %HF% degrees).

The pressure was %PP%U and %PT%.
%ID% recorded, a %DD% wind, %FK%U, swells %HS%U, air temperature %TF%U,
water temperature %SF%U, with wave height %HW%U.

5. Finally, add the section identifiers and end markers $$$. You may also add additional comments as needed.

# This is the Default Format File for CCCHWRNW1
#
#Fixed Format Station Section
!!!!FF_STATIONS
At %ID%,
{ %SW% was falling; %SW% was reported; under %SW% skies; %SW% were reported}.

The temperature was %TF%U([DF], the dewpoint %DF%), and the relative humidity %RR%U
([HF], producing a heat index of %HF% degrees).

The pressure was %PP%U and %PT%.
%ID% recorded, a %DD% wind, %FK%U, swells %HS%U, air temperature %TF%U,
water temperature %SF%U, with wave height %HW%U.$$$

#Marine Station Section
!!!!MAR_STATIONS
%ID% recorded, a %DD% wind, %FK%U, swells %HS%U, air temperature %TF%U,
water temperature %SF%U, with wave height %HW%U.$$$

9.4.3.6 Providing Additional Variability for the NWR Broadcasts

The capability is also provided to modify the output in three additional ways:
1. By producing variability in the phrases for a station from hour to hour;
2. By including a station in both the fixed phrase and summary group stations; and
3. By recapping the weather conditions for one or more stations.

To Create Hour to Hour Variability

There is also a way to obtain additional variability from hour to hour for the Fixed Phrase Format stations. Multiple Broadcast Format files, each with a different fixed phrase format for the stations in the Station List, may be created. To create different Broadcast Format files for a product, specify a different number before selecting the Broadcast Format Editor button.

When multiple Broadcast Format files are provided for a product, the probability with which each file is to be accessed must be provided in the Station List. To create broadcast variability, create a frequency section at the beginning of the Station List.

The first line in the section contains the frequency section indicator, "!!!!BRDCST_FREQ." The next line contains the values that determine the frequency of use. The total of the frequencies MUST be 100 (representing 100 percent). If only one broadcast record for a transmitter is needed, either omit the frequency section from the station list record, or enter “100.”

In the example below, the “10, 40, 50” values indicate that there are three broadcast records associated with this Station List. The HWR NWR application randomly generates a number to choose one of the three files. Assuming that this is the beginning of the Station List for Product CCCWRNW5, you would expect, after multiple executions of the application that

- Broadcast Format File CCCHWRNW5.1 would be selected 10 percent of the time.
- Broadcast Format File CCCHWRNW5.2 would be selected 40 percent of the time.
- Broadcast Format File CCCHWRNW5.3 would be selected 50 percent of the time.

For example,

```
!!!!BRDCST_FREQ
10 40 50
# Fixed Station Format section
!!!!FF_STATIONS
~These are the %TIME2% observations for the local area, on %DAY%, %DATE%.
/KDCA/KDCA/+Reagan National Airport ...
```
To Add Stations to Both Fixed Phrase and Summary Groups

Sometimes, when the geography warrants, it may be desirable to place a station into the fixed format section, and also to include it in a summary group. In this manner, a summary which encompasses the region in which the fixed phrase format station is a part will be more accurate. For example, if the Baltimore Washington International Airport (BWI) is in a fixed phrase broadcast format, a group of stations which constitutes Maryland would not be entirely complete without its observations. Thus, BWI should appear twice in the station list record, once as a Fixed Phrase Format station, and once within the group that comprises Maryland. Therefore, temperature and sky/weather conditions specifically for BWI will be broadcast twice when the sky/weather or temperature in degrees Fahrenheit are within the summary group thresholds. For example,

```plaintext
!!!!!!BRCST_FREQ
10 40 50
# Fixed Format Station section
!!!!!!FF_STATIONS
~These are the %TIMEZ% observations for the local area, on ~%DAY%, %DATE%.
/KDCA/KDCA/+Reagan National Airport
/KDAA/+Fort Belvoir
/KBWI/+Baltimore Washington International Airport
/KADW/+Andrews
/KARQ/KRIC/+Wakefield
#
# Summary Format
~Here are some observations for Maryland.
!!!!!!WX_CONDS
SW TF
!!!!!!SUM_STATIONS
@Throughout the region,
KBWI/+Baltimore Washington International Airport
/KHGR/+Hagerstown
/KNHK/-Patuxent River
/KSBY/+Salisbury
#
# Marine Format
~Here are some CMAN reports.
!!!!!!MAR_STATIONS
/BLIA2/-Bligh Reef
/TTIW1/+Tatoosh Island
~Here are some buoy reports.
#
!!!!!!MAR_STATIONS
/46006/+Buoy 46006
/44014/-Buoy 44014
#
```

To Recap the Weather Conditions

At many WFOs, it is a common practice to repeat the temperature and weather for a major city or cities at the end of a roundup. To set this up in the Broadcast Format file, enter a second Broadcast Format paragraph for these stations using the Format guidelines described previously. In the associated Station List, re-enter
the stations at the end of the list, this time using a station flag of “&.” This
instructs the program to search for that station’s second Broadcast Paragraph in
the Broadcast Format file. If the observation is missing, that station will be treated
as a “-” station and ignored. For example,

!!!!!!BRDCST_FREQ
10 40 50
# Fixed Station Format section
!!!!!!FF_STATIONS
/KDCA/KDCA/+Reagan National Airport
/KADW/KDCA/-Andrews Air Force Base
/KBWI/+B W I Airport
/KIAD/+Dulles
/KDCA/&Reagan National Airport

The corresponding Broadcast Format file needs to be modified to incorporate the
second entry for this station. For example,

/KDCA/At Reagan National Airport,
{ %SW% was falling; %SW% was reported; under %SW% skies; %SW% were reported}
((V.V.), which reduced the visibility to %V.V.%U).
The temperature was %TF%U([DF], the dew point %DF%), and the relative humidity
%RR%U
([HF], producing a heat index of %HF%U).
([FF]The wind %DD% at %FP%U([GG], gusting to %GG%)
([WF], producing a wind chill of %WF%U). )
The pressure %PP%U and %PT%.

/KADW/At Andrews Air Force Base,
{ %SW% was falling; %SW% was reported; under %SW% skies; %SW% were reported}
((V.V.), which reduced the visibility to %V.V.%U).
The temperature was %TF%U([DF], the dew point %DF%), and the relative humidity
%RR%U
([HF], producing a heat index of %HF%U).
([FF]The wind %DD% at %FP%U([GG], gusting to %GG%)
([WF], producing a wind chill of %WF%U). )
The pressure %PP%U and %PT%.

/KBWI/The temperature was %TF%U{ with %SW%; with %SW%; with %SW% skies; with
%SW%}
((V.V.), and a visibility of %V.V.%U) at B W I Airport. The dew point %DF%, and
the relative humidity %RR%U
([HF], yielding a heat index of %HF%).
([GG]The wind %DD% ([FF]at %FP%U)
with gusts to %GG%
([WF], which makes the temperature feel like %WF%). )
The pressure %PP%U and %PT%.

/KIAD/{%SW% was observed; %SW% was reported; It was %SW%; %SW% were observed}
((V.V.), which reduced the visibility to %V.V.%U) at Dulles Airport. The
temperature was %TF%U([DF], the dew point %DF%), and the relative humidity
%RR%U.
([HF]The combination of temperature and humidity feels like %HF%).
([WF]Strong winds from the %DD%
([FF]at %FP%U([GG], gusting to %GG%), are producing a wind chill of %WF%U.
) The pressure %PP%U and %PT%.

/KDCA/Once again, at Reagan National Airport it was %TF%U {with %SW%; with
%SW%;under %SW% skies;with %SW%}.
9.4.3.7 Treatment of Missing Data

The HWR NWR application treats missing data in the various ways described below.

**For Fixed Format Station Observations**

If the entire station is missing and the station is marked in the station list record with a “+” symbol, the site is broadcast as “not available.”

If the entire station is missing and the station is marked with a “-” symbol, the site is dropped from the broadcast.

If the entire station is missing and the station is marked with a “*” symbol, the second station substitutes for the first. If the second station is also missing, the FIRST station is broadcast as “not available,” or not broadcast at all, depending on the symbol with the second station.

If one or more non-threshold weather elements are missing, the fixed format established, in the Broadcast Format file, for that station are ignored, and the program reverts to the default format. One or more missing weather elements can be handled in conjunction with the default format. A default format is necessary because there is virtually an unlimited number of phrases that can be established by the users, and the HWR NWR application cannot realistically be programmed to structure a new sentence required by the absence of the missing weather elements.

**For Summary Group Stations**

Any station marked with a “+” whose observations are missing is reported as “not available,” and the observations for the remaining stations are grouped but not summarized.

A station marked with a “-” whose observations are missing is dropped from the broadcast, and the summary for the remaining stations is grouped, but not summarized.

If the observations for a station are missing and the station is marked with a “*” symbol, the second station is substituted for the first. If both stations are missing, the FIRST station is reported as “not available” or not reported at all, depending on the symbol associated with the second station. The observations for the remaining stations are grouped but not summarized.

If only one of the two (sky/weather and temperature) weather elements is missing for any station, the missing element is not summarized for the remaining stations, but the available weather element remains eligible for summarizing.

When all stations are marked with a “-,” the “@” is used for all comment lines associated with the summary phrase, and all of the stations in the summary group are missing, nothing will be broadcast for this summary phrase, not even the comment line.
For ASOS Stations

For ASOS stations, all of the above rules hold. However, there is an additional consideration, because low-level cloud cover observations are supplemented with satellite-derived middle- and upper-level cloud cover observations. The additional considerations are as follows:

If the ASOS cloud cover observation is missing but the SCP product, which contains information on only middle and high level cloud cover is available, the cloud cover is considered **missing**.

If the ASOS observation shows overcast conditions, the SCP product is not needed, since the sky conditions will be reported as **cloudy**.

If the ASOS observation shows broken clouds, the SCP product will be accessed to determine if the sky is overcast. If the SCP product is missing, the sky conditions will be reported as “**PARTLY SUNNY**” or “**MOSTLY CLOUDY**,” depending on the time of day.

If the ASOS observation shows clear skies or scattered clouds, the SCP product will be accessed to determine if the higher level sky conditions are cloudier. If the SCP product is missing, the sky conditions will either be considered missing, or reported as “**FAIR**,” depending on the forecaster’s preformat selection.

9.4.3.8 Cautions

Some important points that should be considered while setting up the HWR NWR application follow.

1. Punctuation is very important, since the voice synthesizer will be confused by spaces preceding commas and periods. Therefore, be sure to check the printout when initially setting up and testing your files.

2. When it is desirable to repeat an observation for a station within the same fixed format group, be sure that the first entry contains a station flag of “+” or “-,” and that only the second entry in the station list record contains the “&” station flag.

3. There must be **four** sky/weather condition phrases for each fixed format station, even if two or more of the four sky/weather types are to be broadcast identically.

4. One or more missing or erroneous weather elements from a fixed format station’s observation cause the application to revert to a default format. This is necessary because the program is otherwise incapable of patching together the remaining phrases to form a coherent report. Check the station’s reported observation first whenever an unexpected output is encountered.

5. A maximum of 15 comment lines starting with either “@” or “~” may be used in a station list. When all stations are marked with a “-,” the “@” instead of a “~” is used at the beginning of a section, and all of the stations in the section are missing, nothing will be broadcast for this section, not even the comment line.
9.5 **Aviation Forecast Preparation System**


9.5.1 **System Overview**

AvnFPS is implemented with a series of server and client processes that communicate via TCP sockets. The servers run on hosts where the data are readily available. Clients run on workstations which do not need direct access to the data. The client may be located on a non-AWIPS system and does not have to be within the same network segment. Data request servers (avndrs) are responsible for delivering data to clients. Data ingest servers (avndis) monitor data received by AWIPS and store them in a modified format in the AvnFPS database. A notification system (avnserver) is responsible for alerting the clients when new data are received. The AvnFPS database can be distributed among several hosts.

AvnFPS is written mostly in the Python programming language. Remote communication features are implemented with the help of Python Remote Objects (PYRO). PYRO is an advanced and powerful Distributed Object Technology system, written entirely in Python, which resembles Java's Remote Method Invocation (RMI). TAF and METAR decoders utilize Toy Parser Generator (TPG), also a Python module. Graphical interfaces are written in Tkinter. The AvnFPS distribution comes with its own version of the above tools, and is independent of AWIPS COTS.

Network Attached Storage eliminates the need for a distributed server configuration, as all the data are visible from a single host. The standard configuration is shown in Exhibit 9.5.1-1.

| NOTE: | The data flow diagram shows servers running on PX2, however, any Red Hat™ host on an AWIPS network would do. The standard installation puts the servers on PX2. |

9.5.1.1 **Name and Event Server**

The avnserver process stores the names and locations of all objects used by AvnFPS servers, and it provides these names and locations to all other AvnFPS processes. Additionally, avnserver other task is to provide a notification mechanism between servers and clients. Thus, avnserver combines PYRO's name and event servers. One and only one instance of avnserver should run within a WFO cluster. A client first connects to avnserver to get information about the locations of data request servers and to subscribe for notifications. Client processes do not have to run on hosts at the same WFO as the data servers although this feature is generally not used. An access list of valid hosts, maintained in a configuration file, is used to provide host based access control. avnserver must be running before any other AvnFPS servers or clients can be successfully started.
The application **avninit** is used to launch AvnFPS servers and in a certain order. Users typically do not have to be concerned about the servers. It is **avninit**'s job to ensure that all servers are running and restart them if they fail. However, command argument details for each server are provided here for their usefulness in troubleshooting, should it be necessary to start them by hand.

The **avnserver** program accepts the following command line arguments:

**TYPE:**

```
avnserver [-d] -n host
```

- where *host* is the host name where the process will run.

Obviously, this value could be retrieved from the operating system. This argument, however, is needed to properly handle failover situations. The optional flag `-d` tells the program to not detach itself from the terminal session, in other words, to run in the foreground.
9.5.1.2 Data Ingest Server

An instance of *avndis* monitors AWIPS data directories via *gamin* [Gamin]. *avndis* is a threaded application: for each data source, there is a dedicated thread that processes the data. Which server instance processes which data is determined by the server configuration file `etc/server.cfg`. *avndis* receives a notification from the *gamin* server when a file in a monitored directory is created or modified. The server, or parent process, then passes the file name to the appropriate thread. The thread processes the file's content, writes data to AvnFPS database and returns information back to the parent process. The parent process then sends notification to *avnserver*. A client that subscribed to the server will then receive this notification. The threads that are currently implemented are listed in Table 9.5.1.2-1.

<table>
<thead>
<tr>
<th>Thread</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>text</td>
<td>Processes text products: TAFs, METARs, TWEBs and CCFPs.</td>
</tr>
<tr>
<td>ltg</td>
<td>Processes lightning data.</td>
</tr>
<tr>
<td>rltg</td>
<td>Processes regression-based lightning forecasts data.</td>
</tr>
<tr>
<td>llws</td>
<td>Calculates Low Level Wind Shear value. The data sources are radar data (VWP) and METAR observations.</td>
</tr>
<tr>
<td>file</td>
<td>Monitors files written by an external program in a format that can be processed directly by AvnFPS clients. Currently used to process IFPS grids.</td>
</tr>
<tr>
<td>guid</td>
<td>Monitors directories where guidance sources arrive over the AWIPS WAN. Typically, BUFR MOS and raw model output from NCEP.</td>
</tr>
</tbody>
</table>

Every 30 seconds, *avndis* sends an ALIVE message that is used by clients to monitor the state of AvnFPS servers and alert users when something goes wrong.

The program accepts the following arguments:

```
TYPE:  avndis [-d] -n host

where host is the host name where the process will run. The optional flag -d tells the program to not detach itself from the terminal session (i.e., to run in the foreground). See Section 9.5.3.1, “Starting AvnFPS Servers,” for an example of a ps listing.
```

9.5.1.3 Data Request Server

*avndrs* is mainly responsible for providing data to AvnFPS clients. The data source is the AvnFPS database, however, it can also directly access AWIPS files such as netCDF data in the /data/fxa tree. The other function of *avndrs* is to write forecasts prepared by the clients to a queue where they are then processed by the transmission server *avnxs*. Each instance of *avndrs* is capable of processing all data sources.
Every 30 seconds, **avndrs** sends an **ALIVE** message that is used by clients to monitor the state of the AvnFPS servers and alert users when something is wrong.

The program accepts the following arguments:

```
TYPE: avndrs [-d] -n host
```

- where **host** is the host name where the process will run. The optional flag **-d** tells the program to not detach itself from the terminal session (i.e., to run in the foreground). See Section 9.5.3.1, “Starting AvnFPS Servers,” for an example of a **ps** listing.

### 9.5.1.4 Transmission Server

The transmission server, **avnxs**, provides a bridge between AvnFPS and the AWIPS transmission system. avnxs supports delayed transmissions, i.e., issuing forecasts that have been prepared by the forecaster before the transmission window opens.

Forecasts prepared by the AvnFPS forecast editor are written to the directory `xmit/pending` as text files with a particular naming convention. The filename is then used to determine the disposition of the product. Files in the `xmit/pending` directory are named as follows:

```
TYPE: fid-cccnmnxxx-wmoid-wfoid-ymmdhhmm-typ-ttttttttttt
```

- where
  - **fid** - Forecaster ID Number
  - **cccnmnxxx** - AWIPS id, argument to handleOUP.pl
  - **wmoid** - WMO Header
  - **wfoid** - WMO ID if the WFO issuing the product
  - **ymmdhhmm** - Full Timestamp, as year (modulo 100), month (Jan=1), day of the month, UTC hour and minute
  - **typ** - Forecast Type and Version, one of the following
    - ___ Routine Forecast
    - RRX Delayed Forecast
    - AAx Amendment
    - CCx Correction

where **x** is the version: A through Z.

- **tttttttttt** - Earliest transmission time, in Unix seconds. For routine forecasts this is the start of the transmission window or an arbitrary time set by the forecaster. For other forecasts this is the time the forecaster presses the Send button.
**avnxs** monitors files in the `xmit/pending` directory to the current system clock. The requested transmission time, $T_{xmit}$, and the creation time, $T_{post}$, are retrieved from the filename and from the OS, respectively. If the current time $T_{cur}$ is within range: $T_{xmit} < T_{cur} < T_{post} + N$ hours, **avnxs** will call an AWIPS program (currently **handleOUP.pl**) to send the forecast out to the WAN. $N$, the number of hours before a pending file is considered 'old', is now a configurable item in `etc/xmit.cfg` file. If the forecast file is too old, it will be moved to `xmit/bad` directory. The value of $N$ (usually 3 hours) is set in transmission server configuration file. Files that do not match transmission file format are discarded. **avnxs** checks for the return code from **handleOUP.pl**. The status is written to a file `xmit/DoW`, where `DoW` is a 3-letter abbreviation for the Day Of the Week. Here is a sample entry from `xmit/DoW`:

SUCCESS002-KPBZTWB072-FRUS41-KPBZ-0410270100-___-1098840000

The first word is either SUCCESS or FAILnnn, and nnn is the error code returned by **handleOUP.pl**. If **handleOUP.pl** fails to transmit the forecast, the file will be moved to `xmit/bad`. The transmission status will be announced via **avnserver** to all client GUIs, see Section 5, “Forecast Transmission”.

The program accepts the following arguments:

**TYPE:**

```
avnxs [-d] -n host
```

where `host` is the host name where the process will run. The optional flag `-d` tells the program to not detach itself from the terminal session (i.e., to run in the foreground). See Section 9.5.3.1, “Starting AvnFPS Servers,” for an example of a `ps` listing.

The transmission server also collects information on forecasts sent by the office for verification purposes. A special product is sent periodically to the Verification Branch at NWS Headquarters that contains information that identifies which forecaster is responsible for which forecast. A configuration file `etc/xmit.cfg` specifies WMO and AWIPS headers and the frequency of transmission.

### 9.5.1.5 Client Programs

**AvnWatch** is the main client application. It provides forecast preparation and monitoring functionality. **Avnclimate** is a set of GUIs accessing archived climatological data. Current functionality provides display of historical data and statistics that can help by producing a short-term forecast based on current weather conditions.

### 9.5.2 Configuring AvnFPS

This section describes directory layout, configuration files used by AvnFPS and steps needed to download climatological data. Generally, this section will only be used as an aid in troubleshooting problems with the configuration of AvnFPS. While it is possible to
configure AvnFPS by editing these files directly, most people will find it much easier to use the GUIs described in Section 9.5.4, “Avnsetup”.

After successful installation of AvnFPS 3.2, the top level directory /awips/adapt/avnfps should be as follows:

    lx2-wfo: /bin/ls -la

`drwxrwxrwx 8 fxa fxalpha 4096 Jan  8  2010 .`
`drwxrwxr-x 14 awips fxalpha 4096 Sep 12  2012 ..`
`lrwxrwxrwx  1 fxa   fxalpha    9 Aug 29  2012 bin -> OB9.2/bin`
`lrwxrwxrwx  1 fxa   fxalpha    9 Aug 29  2012 etc -> OB9.2/etc`
`drwxrwxr-x  4 fxa   fxalpha 4096 Mar 24  2009 gamin-0.1.10`
`drwxrwxr-x  6 fxa   fxalpha 4096 Jan 20  2006 hdf5-1.6.4`
`drwxr-xr-x  9 fxa   fxalpha 4096 Dec 19  2008 OB9`
`drwxr-xr-x  9 fxa   fxalpha 4096 Jan  8  2010 OB9.2`
`drwxrwxr-x  7 fxa   fxalpha 4096 Mar 24  2009 Python-2.4.1`
`drwxrwxr-x  1 fxa   fxalpha 4096 Dec 25  2006 .sqlcmdlog`
`drwxrwxr-x  5 fxa   fxalpha 4096 Jun  9  2005 tcltk8.4.5`

The installation process does attempt to remove versions of the software older than 3.1. It is up to you to remove any remaining cruft that might be in this directory.

All configuration files are located in the directory /awips/adapt/avnfps/3.2/etc and its subdirectories. Most of the files (those with an extension .cfg) are in Windows™ ini format:

```
[section]
keyword=argument(s)
```

Lines starting with # are comments.

Files containing specifications for graphical user GUIs follow a X11 standard:

```
*pattern: value
```

where pattern is a hierarchy of one or more widget classes/names, separated by *. These files are located in the subdirectory app-resources.

The file etc/forecasters contains forecaster IDs and names in plain ASCII text format.

### Files in etc

**etc/forecasters.** Contains list of forecaster ids and names/initia ls. The numbers must be in the range 1 through 999. Lines starting with # are comments. Blank lines are allowed in the file. The format is the same as in previous releases. An example:

```
# etc/forecasters
# List of forecasters
# Format:
# number name
1 Amanda
2 Bailing
3 Belinda
```

**etc/logging.cfg.** This file configures the logging behavior of all AvnFPS programs. You
should not need to modify it. Not changed since AvnFPS 3.0.

```python
# logging.cfg
# configuration file for Python logging utility

[loggers]
keys=root

[handlers]
keys=hdl

[formatters]
keys=fmt

[logger_root]
level=NOTSET
handlers=hdl

[handler_hdl]
class=WeeklyFileHandler
args=('%s/logs' % os.environ['TOP_DIR'], os.path.basename(sys.argv[0]).split('.')[0])
formatter=fmt

[formatter_fmt]
format=%(asctime)s %(levelname)-5s [%(process)5d:%(thread)6d] %(module)s: %(message)s

NOTES:
1. Log level is set in the section [logger_root]. Available values in decreasing order of information output are: DEBUG or NOTSET (these two are equivalent), INFO, WARNING, ERROR, CRITICAL.
2. [handler_hdl] specifies output directory and file name. The class WeeklyFileHandler appends an abbreviated day of the week to file name.
3. [formatter_fmt] specifies format of a line written to the log file. Refer to Python Documentation for details.
```

`etc/server.cfg`. This file specifies values used by the data ingest servers.

```bash
# etc/server.cfg
# server configuration file for OB7
# last update: 03/01/06

# Name servers
# Either local or ip address. Local means that broadcast is used to locate
# the server, which means the server must be on the same subnet
# The list of tafs is displayed in the startup (menu) GUI

# Data ingest servers
[dis]
tag=text2,ltg,rltg,file,llws,guid

# etc/server.cfg
# etc/server.cfg
# server configuration file for OB7
# last update: 03/01/06

# Name servers
# Either local or ip address. Local means that broadcast is used to locate
# the server, which means the server must be on the same subnet
# The list of tafs is displayed in the startup (menu) GUI

# Data ingest servers
[dis]
tag=text2,ltg,rltg,file,llws,guid

# Data ingest servers
[dis]
tag=text2,ltg,rltg,file,llws,guid

# Data ingest servers
[dis]
tag=text2,ltg,rltg,file,llws,guid

# Data ingest servers
[dis]
tag=text2,ltg,rltg,file,llws,guid

# Data ingest servers
[dis]
tag=text2,ltg,rltg,file,llws,guid
```

AWP.MAN.SMM.A2

– Hard copies uncontrolled. Verify effective date prior to use. –
#module=TextThread
#nhours=24

[dis_text]
name=text
source=/awips/adapt/avnfps/data/text
module=TriggerThread
nhours=24

[dis_ltg]
name=ltg
source=/data/fxa/point/binLightning/netcdf
module=LtgThread
# distance in miles, age in minutes
distance=20
age=15

[dis_rltg]
name=rltg
source=/data/fxa/img/SBN/netCDF/LATLON/3hr/LTG
module=RadLtgThread

[dis_llws]
name=llws
radar=/data/fxa/radar
profiler=/data/fxa/point/profiler/netcdf
module=LLWSThread

[dis_file]
tag=pxl
name=File
source=/awips/adapt/avnfps/data/grids,
module=FileThread

[dis_guid]
name=guid
ngmmos=/data/fxa/point/mos/NGM/netcdf
avnmmos=/data/fxa/point/mos/AVN/netcdf
gfsmos=/data/fxa/point/mos/GFS/netcdf
etamos=/data/fxa/point/mos/ETA/netcdf
eta=/data/fxa/point/model/ETA/netcdf
module=GuidanceThread
nhours=24

# access control
# hosts allowed to access data
[valid]
hosts=XXX.XXX.58.*

NOTES:

1. Section [dis] specifies which ingest threads should run. For each item tag on
the list there is a corresponding [dis_tag] section which defines the thread.
Each section comprises several items listed below. There are two ways of
accessing text products. TextThread reads data files written by acqserver.
However, the AWIPS ingest system has to be changed from baseline
configuration in order to use this thread. Thus, it is commented out and
TriggerThread is used in AWIPS instead.
name is one of predefined data sources which is specific to each module. Do not modify this value.
- source is the directory where AWIPS data files reside. This directory is monitored by `gamin` server daemon, `gam_server`. In this case, this is the directory where `avntrigger.sh` writes files extracted from text database. A special case is LLWSThread which relies on radar and profiler data. The relevant directories are specified by keywords radar and profiler.
- module specifies Python source code file. Do not change.
- nhours is a module specific parameter, in this case it is the number of hours of data that should be preserved in AvnFPS database.

2. Section `[dis_text]` defines an experimental setup that bypasses AWIPS decoders and databases. TAFs, TWEBs and METARs are retrieved from files dumped by `acqserver` into a directory. This configuration is not recommended for novices.

3. Item `[dis_text]` defines thread processing reports retrieved from Postgres `fxatext` by `avntrigger.sh` via the trigger mechanism. This is delivered as the default technique to get text products.

4. `[dis_ltg]` tag controls the processing of lightning observations. The tag distance defines an area of interest. Only lightning strikes within a square centered at the TAF site of size twice the provided value (in miles) will be extracted from AWIPS data files.

5. `[dis_rltg]` tag is for the radar-based 3 hour cloud-to-ground lightning probability forecast which is currently available in the lower 48 states. OCONUS sites may safely remove this tag from the list in the `[dis]`section.

6. `[dis_llws]` controls Low Level Wind Shear setup. The directories are the base radar file location (the site id and product type VWP will be appended by the thread). The profiler tag can be a comma-delimited list of directories. This allows LDAD directories as well as AWIPS ones to be listed.

7. The `[dis_file]` describes a thread that reads the content of files in the directories listed on the source line. The base directory of each file is passed to the client, which can thus determine the data type. Currently this thread reads files produced by a GFE text formatter.

8. The `[dis_guid]`, specifies guidance data: MOS and high-resolution point NAM-WRF model data. You may comment out some lines if a particular data is not available in your area. For instance, NGM MOS isn't available in Alaska and Pacific Regions, so ngmmos tag can be removed.

9. The access control section `[valid]` restricts access to AvnFPS server to hosts with IP addresses matching those on the hosts line. A shell type match characters can be used: * matches any string, ? matches single character
etc/px2finit.cfg. This file specifies which servers should be started by avnint and their startup order and should not be changed.

```bash
# px2finit.cfg
[avnserver]
name=avnserver
order=0
wait=5

[avndrs]
name=avndrs
order=1
wait=3

[avndis]
name=avndis
order=2
wait=1

[avnxs]
name=avnxs
order=3
```

**NOTE:**

For each server there is a separate section. The tag must be unique. A section will have the following items:

- ‘name’ specifies server name
- ‘order’ determines server startup sequence. Servers with lower order start first.
- ‘wait’ is a wait time in seconds after a particular server is started (forked), before starting the next server on the list.

etc/gui.cfg. Specifies the values used by GUI shared by all users.

```bash
# etc/gui.cfg
# specifies available monitor options for avnwatch

# Name servers
# Either local or ip address. Local means that broadcast is used to locate
# the server, which means the server must be on the same subnet
# The list of tafs is displayed in the startup (menu) GUI
[ns]
tags=local,

# colors in main GUI: 7 items
[colors]
tags=green3, grey, pale green, yellow, orange, red, purple

# editor tags
[editor_tags]
fatal=red
error=orange
warning=forest green

# miscellaneous features
[features]
```
tafeditor=1
twbeditor=1

# viewers: taf and metar must be present, taf must be first on the list
[viewers]
tags=taf,metar,gfsmos,etamos,ngmmos,etabuf,grids

[viewer_taf]
module=TafViewer

[viewer_metar]
module=MetarViewer
label=Metars

[viewer_gfsmos]
module=MosViewer
label=GFS-MOS
model=gfsmos

[viewer_gfslamp]
module=MosViewer
label=GFSLAMP
model=gfslamp

[viewer_ngmmos]
module=MosViewer
label=NGM-MOS
model=ngmmos

[viewer_etamos]
module=MosViewer
label=ETA-MOS
model=etamos

[viewer_etabuf]
module=EtaViewer
label=NAM-WRF-profile

[viewer_grids]
module=GridViewer
label=Grids

[viewer_avnmos]
module=MosViewer
label=AVN-MOS
model=avnmos

[menus]
number=7
[menu_0]
items=metar
[menu_1]
items=persistence_1hr,persistence_2hr,persistence_3hr
[menu_2]
items=ltg
[menu_3]
items=rltg
[menu_4]
items=ccfp

[menu_5]
items=grids

[menu_6]
items=llws

[monitor_metar]
menu=METAR
module=MetarMonitor
items=tempo,vsby,wind,wx,sky
labels=tpo,vis,wnd,wx,cig

[monitor_persistence_1hr]
menu=persistence 1hr
module=PersistMonitor
nhours=1
items=vsby,wind,wx,sky
labels=vis,wnd,wx,cig

[monitor_persistence_2hr]
menu=persistence 2hr
module=PersistMonitor
nhours=2
items=vsby,wind,wx,sky
labels=vis,wnd,wx,cig

[monitor_persistence_3hr]
menu=persistence 3hr
module=PersistMonitor
nhours=3
items=vsby,wind,wx,sky
labels=vis,wnd,wx,cig

[monitor_ltg]
menu=ltg
module=LtgMonitor
items=wx
labels=ts

[monitor_rltg]
menu=rltg
module=RadLtgMonitor
items=wx
labels=ts

[monitor_ccfp]
menu=ccfp
module=CCFPMonitor
items=wx
labels=ts

[monitor_grids]
menu=grid
module=GridMonitor
from=2
to=6
items=vsby,wind,wx,sky
labels=vis,wnd,wx,sky

[monitor_llws]
menu=llws
module=LLWSMonitor
items=wind
labels=ws

NOTES:

1. Section [colors] lists colors used to indicate discrepancies between forecasts and observations/guidance, in increasing order of severity.

2. Section [viewers] lists data viewers available in the TAF Editor window. For each tag listed, there must be a [viewer_tag] section.

3. Each viewer requires certain parameters. Module specifies a Python source code file, label defines text displayed in the window’s tab. Some modules require additional values. You should not modify any of those values, with the exception of the labels.

4. Section [menus] defines data areas in the main monitoring GUI. For each area there must be a section [menu_#] listing items that will be monitored. Count starts from 0.

5. Items is a list of comma separated tags, each of the tags must have a corresponding [monitor_item] section. If the list consists of one item, it must be terminated by a comma. If 2 or more items are on the list, the user can select what is monitored.

6. Each monitor requires certain parameters. module specifies a Python source code file, menu defines text displayed when the popup menu is invoked. items is a list of monitored weather elements, labels determine displayed text. To keep the GUI nicely aligned, keep label names 2 or 3 characters long.

7. “nhours” is the number of hours ahead assuming the weather does not change.

8. “from, to define” interval of time the grids are monitored against the forecast.

etc/wxplot.cfg. This is a configuration file for the weather plot GUI.

    # wxplot.cfg
    [print]
    cmd=convert - tmp/%s.jpg

    [hours]
    back=24
    forward=24

    [vsby]
    bot=0.125
    top=10.0

    [cig]
    bot=100
    top=8000

    [viewers]
    tags=taf,metar,gfsmos,etamos,ngmmos,eta
[selected]

tags=taf,metar

[viewer_taf]
module=TafPlot
label=TAF
color=blue

[viewer_metar]
module=MetarPlot
label=METARS
color=red

[viewer_gfsmos]
module=MosPlot
label=GFS-MOS
color=forest green
model=avn

[viewer_etamos]
module=MosPlot
label=ETA-MOS
color=green
model=gfs

[viewer_ngmmos]
module=MosPlot
label=NGM-MOS
color=dark green
model=ngm

[viewer_eta]
module=EtaPlot
label=NAM-WRF
color=brown

NOTES:

1. Command to be executed when the Print is pressed. A window dump file (xwd format) is passed via standard input stream.

2. Number of hours of data to display, counted from the current hour.

3. Ceiling and visibility plot limits.

4. This section lists data viewers available in the Weather Plot window. For each tag listed, there must be a [viewer_tag] section.

5. Data viewer toggles selected by default.

6. Each viewer requires certain parameters. 'module' tag specifies a Python source code file, 'label' tag defines text displayed as a toggle button name. Some modules require additional values. You should not modify any of those values, with the exception of the labels.

etc/xmit.cfg

This configuration file is used by avnxs which sends forecast and verification products
Chapter 9 Background Applications

out for dissemination.

    # etc/xmit.cfg
    [verification]
    wmo = NXUS98 KPIT
    awips = KPBZVFPBZ
    fcstid = 0
    period = 6

    [transmission]
    # how often checks for new files (seconds)
    frequency = 15
    # reject older than (hours)
    old = 3

NOTES:

1. This section defines parameters used by the program to transmit the NXUS98 verification product to NWS headquarters. wmo and awips are the WMO and AWIPS headers. These headers must exist in the file /awips/afx/data/afos2awips.txt. The transmission file name contains (reserved) forecaster id 0. This should not be changed. The verification product is transmitted every 6 hours.

2. You may want to adjust the old value. It defines the cutoff time for early transmission.

etc/triggerTemplate. Made by avnsetup. This file is processed during trigger localization on the data server. avnsetup also creates a file etc/triggerTemplate.good while updating Postgres triggers directly. The latter is not used by the localization script.

9.5.2.2 TAF Configuration Files

The TAF configuration files reside in directory etc/tafs. For each TAF site CCCC there is a subdirectory etc/tafs/CCCC containing template files, one per each forecast period and site info data etc/tafs/CCCC/info.cfg.

Site Info File. This file is normally created by the Taf Site Editor GUI, normally invoked from avnsetup.

    # etc/tafs/KPIT/info.cfg
    [headers]
    wmo = FTUS41 KLWX
    afos = WBCTAFIAD

    [thresholds]
    cig = 200,600,1000,2000,3100
    vsby = 0.5,1.0,2.0,3.0,6.0
    radar_cutoff = 600,0
    profiler_cutoff =

    [sites]

NOTES:

1. Section [thresholds] defines ceiling and visibility categories used by the monitoring system. A new feature of AvnFPS 3.2 is the cutoff thresholds for the VWP sources. Cutoff values correspond to heights in meters below which data will be ignored. As in the example above, for the KLWX radar, any VWP data below 600 meters will be ignored. However, all VWP data from the surface up to 2000 feet from the KDOX radar will be used. This feature is useful if your VWP source(s) is (are) routinely contaminated by non-meteorological phenomenon near the site.

2. Section [sites] defines IDs used by other data sources. Some TAF sites may have more than one associated observation. In that case, separate the ids by a comma.

3. Section [geography] provides location of the airport. elev is in meters, runway directions in degrees true north are used to calculate cross winds.

Monitoring Rules. The default configuration files for all monitors are located in the directory etc/tafs/XXXX. Currently these are: grids.cfg, ltg.cfg, mtrs.cfg, rltg.cfg, ccfp.cfg. All can be edited with the "Monitoring Rules" GUI via avnsetup. Here is an example of rule definition file for monitoring of METARs:

```
# etc/tafs/XXXX/mtrs.cfg

[rules]1
active = 0,1,2,3,4,5,6,7,8,9,10,11

[rule_1]
severity = 3
msg = Freezing precipitation in TAF and not in obs
unique = 0
type = wx
method = WxTafDelta

[rules]2
active = 0,1,2,3,4,5,6,7,8,9,10,11

[rule_11]
severity = 5
msg = Visibility difference of 3 or more categories
unique =
type = vsby
method = VsbyCatDelta
ncat = 3

[geography]3
lat = 38.96
runway = 60,90,
lon = -77.45
elev = 98
```
NOTES:

1. Each [rule_N] provides arguments to the monitoring function, specified by the method keyword. The following tags must exist for each rule:
   - severity: a number in the range 1 through 5 corresponding to colors defined in gui.cfg, see 1.
   - msg: message displayed by the monitoring GUI.
   - type: one of wind, vsby, wx, sky
   - unique: 0 or 1.

   Each function may accept additional arguments, such as ncat. In this case ncat is the number of categories the visibilities in a TAF/METAR pair must differ for the rule to activate. Thus you may specify the same method with different arguments, ncat=2 in our case.

2. Section [rules] defines the list of rules to monitor. For each number there must be a corresponding [rule_N] section.

If you want to have rules dependent on the TAF site, copy the relevant file to the TAF site directory and modify it there. The monitoring program first tries to find the files in the site directory, if it can't find it there, uses the one from etc/tafs/XXXX.

Important! There are no default hardcoded rules. The files in etc/tafs/XXXX must exist.

Impact QC Definition File. Impact QC is a new feature in AvnFPS3.2. The forecaster can be alerted when a prepared TAF has a significant impact on airport operations. This file is optional. Currently this file cannot be modified with avnsetup, you must use a text editor instead. Shown below is a configuration file with additional rules added that are specific to Pittsburgh International Airport, etc/tafs/KPIT/impact.cfg:

```cfg
# impact.cfg
[conditions]
items=cond_1,cond_2,cond_3,cond_4,cond_5

[cond_1]
tag=FC
level=1
text=Impact: Fuel-Alternate vsby<3 or cig<2000
expr=vsby<2.95 or cig<2000

[cond_2]
tag=FC
level=2
text=Impact: LIFR conditions vsby<1 or cig<500
expr=vsby<0.95 or cig<500

[cond_3]
tag=ws
level=1
text=Wind shift vs previous group resulting in >6kt change along PIT runway 28R/10L, may affect runway config
```

wx = FZRA, FZDZ, PL
expr=wind[0].shift and wind[0].runway>6

[cond_4]
tag=xw
level=1
text=Cross wind component >14KT on PIT runway 28R/10L
expr=wind[0].cross>14

[cond_5]
tag=xw
level=2
text=Cross wind component >24KT on PIT runway 28R/10L
expr=wind[0].cross>24

NOTES:

1. Section [conditions] defines the list of conditions to check. For each tag [cond_N] there must be a corresponding section.

2. Each [cond_N] provides arguments to the impact check function. The check is done for each TAF period. The following tags must exist for each rule:
   - **expr**: A Python expression to be evaluated. This expression can contain following variables: vsby, cig and wind. vsby is in miles, ceiling in feet. wind is an array that has length equal to the number of runways. Each element of this array contains 3 values: runway, cross and shift. The wind vector is represented in a Cartesian coordinates along and perpendicular to the runway. The absolute values of these components are wind[N].runway and wind[N].cross, where N is runway number, counted from 0, as listed in the site configuration file, see section: : “Site Info File”. wind[N].shift is a Boolean value, indicating that the runway component changed sign from one TAF period to another.
   - **text**: Message to be displayed when the expression evaluates to True.
   - **level**: Highlight background color, 1 - green, 2 - orange.
   - **tag**: Used to group messages, so only the message with the highest level is displayed. In this example, if visibility is 1/2SM, only the LIFR message is shown in a popup window of the TAF Editor.

Climate QC Definition File. The file etc/tafs/XXXX/climqc.cfg is used to define category thresholds used while creating climate QC count files. Currently this file cannot be modified with avnsetup, you must use a text editor. An example:

```
# etc/tafs/XXXX/climqc.cfg

[ARGS]
alpha=0.3
showdetails=0

[THRESHOLDS]
vsby=0.27,1.1,2.9,6.1
cig=205,605,1005,3105,40000
ff=4.1,10.1,20.1
dd=22.5,67.5,112.5,157.5,202.5,247.5,292.5,337.5
```
NOTES:

1. The parameter alpha is used to determine likelihood of occurrence of a combination of weather elements in the forecast based on conditional probabilities. Alpha should be a positive number, less than one. The showdetails flag (0=no;1=yes) turns detailed statistics display, generally disliked by forecasters.

2. The thresholds determine category ranges. Each of arguments is a list of values $v_1 < v_2 < \ldots < v_n$, with implicit $v_1 = 0$ and $v_{n+1} = \infty$. Category $k$ consists of all values $v$ such that $v_k \leq v < v_{k+1}$.

The value 40 000 ft on the ceiling list serves to distinguish between limited and unlimited ceiling, which is coded as 99 998 ft.

If you want to have thresholds dependent on the TAF site, copy the file to the TAF site directory and modify it there. The program avnqstats which creates the count files first tries to find the file in the site directory, if it can't find it there, uses the one from etc/tafs/XXXX.

TAF Product Definition Files. Located in etc/tafs. Normally created by the Taf Product GUI invoked from avnsetup. An example file:

```plaintext
# etc/tafs/PBZ_TAFS.cfg
# list of TAF sites
[sites]
workpil = PITWRKTAF
idents=KFTT,KAGC,KBVI,KLBE,KHLG,KMGW,KZZV,KFKL,KDUJ
```

Default Product File. The optional DEFAULT file contains name of the configuration file, sans the .cfg extension. That should be loaded on startup of the monitoring GUI. If not present, the first one in alphabetical order is used.

TAF Formatter Configuration Files. The TAF Formatter requires three configuration files. These files are located in the directory etc/tafs/XXXX. With AvnFPS3.2, flt_cat.cfg and grp_taf.cfg files can now be customized for each TAF site and moved to its respective etc/taf/CCCC directory.

File flt_cat.cfg contains the threshold values of visibility and ceiling height that are used to determine aviation flight categories. The default values are identical to those provided in Directive 10-813. The thresholds can be changed to site-specific values.

```plaintext
# etc/tafs/XXXX/flt_cat.cfg
[cig]
thresholds=200,500,1000,3000,3500,4000,5000,8000,10000,12000,15000,18000,20000
[vis]
thresholds=0.5,1,3,5
```

NOTE:

1. These thresholds divide the spectrum of ceiling and visibility into categories with an implied 0 at the beginning and 'infinity' at the end of each list. Thus as
defined here, ceiling category 1 corresponds to cloud heights from 0 to 200 feet, category 2 from 200 to 500 feet, etc. Similarly for visibility, with category 1 corresponding to visibilities LTE to 1/2 mile. Category 5 corresponds to visibility greater than 5 miles.

File grp_taf.cfg contains the configurable threshold values that are used to form concise TAFs. Detailed description for each category has been provided inside the file.

```plaintext
# This file contains thresholds that control the algorithm that converts numerical/tabular guidance into 'guidance' TAFs.
# Probabilities (POT and POP06) above which precipitation and/or thunderstorms will be included in the prevailing group.
[prev]
pop=80
tstm=80
# Probabilities (POT and POP06) above which the tempo group is formed containing precipitation and/or thunderstorms.
[tempo]
tstm=36
pop=36
# Boolean (yes/no) if PROB30 groups are wanted in guidance TAFs
[prob30]
value=yes
# The longest time span in hours that makes this group combinable with a longer, adjacent (in time) group. If the duration of a group is longer than this value, this group stays alone in the TAF.
[short_dt]
value=1
# The shortest time span in hours that allows its shorter duration neighbor to be combined with it.
[long_dt]
value=15
# Maximum category difference between the long and the short duration groups. If the difference in flight category between the two groups is greater than this value, they are not combined.
# Categories are treated separately for visibility and ceilings.
# To see the breakpoints for the ceiling and visibility categories, see etc/tafs/XXXX/flt_cat.cfg (flight category) configuration file.
# First the visibility
[dc_vis]
value=1
```

File grp_taf.cfg contains the configurable threshold values that are used to form concise TAFs. Detailed description for each category has been provided inside the file.
# Lowest visibility category to which the combining algorithm is applied
# [low_c_vis]$^4$
value=5
#
# Highest visibility category to which the combining algorithm is applied
# [high_c_vis]$^5$
value=5
#
# Now, the ceiling . . .
# [dc_cig]$^6$
value=1
#
# The lowest ceiling category to which the combining is applied
# [low_c_cig]$^7$
value=13
#
# Highest ceiling category to which the combining is applied
[high_c_cig]$^8$
value=14
#
# Earliest forecast time in hrs to which the combining algorithm can be applied
[low_p]$^9$
value=12
#
# Maximum allowable ceiling height in hundreds of feet when thunderstorms
# are forecasted
[cb_hi]
value=50
#
# Maximum speed which is considered to be not a calm wind.
[wind]
calmwd=5
#
# Wind speed and direction change in wind speed above which combining will not
# be applied.
value=10
delta=4$^{10}$

NOTES:
Items 1 through 9 control how two dissimilar (in visibility and/or ceiling) groups,
one 'short', the other 'long', can be further combined.

1. The maximum duration of the smaller group that can be combined with the
   larger, dissimilar one. When combined, the larger group values are not
   adjusted by the smaller group's values.

2. The minimum length of time to be considered the 'large' group candidate for
   combining with smaller ones.

3. The largest visibility category difference allowed between TAF groups and
   still be able to combine.
4. The lowest visibility category allowed to be considered for combining. This value is related to the values assigned to the vis thresholds in flt_cat.cfg file.

5. The highest visibility category allowed to be considered for combining. This value is related to the values assigned to the vis thresholds in flt_cat.cfg file. Usually set to the number of visibility 'breakpoints' in vis thresholds.

6. The largest ceiling category difference allowed between TAF groups and still be able to combine.

7. The lowest ceiling category allowed to be considered for combining. This value is related to the values assigned to the cig thresholds in flt_cat.cfg file.

8. The highest ceiling category allowed to be considered for combining. This value is related to the values assigned to the cig thresholds in flt_cat.cfg file. Usually set to the number of ceiling 'breakpoints' in cig thresholds.

9. Number of hours when the combining algorithm will be applied to groups valid beyond the TAF's beginning valid time.

10. Maximum allowable difference in speed (knots) and direction (tens of degrees) between groups allowed to be combined.

Finally, grid_probs.cfg contains configuration data on how the probability attributes on IFPS grids are translated into numerical probability values that will be used to form TAFs. Remember that precipitation is only included in the TAFs when certain probability threshold is exceeded. This file cannot be customized to individual TAF sites.

```
#tables used to convert prob symbols in grids to a pop value
[before9hr]
S=30
IS=30
WS=30
SC=50
O=50
C=50
D=70
WP=70
NM=70
L=70

[after9hr]
S=20
IS=20
WS=20
SC=30
O=30
C=30
D=50
WP=50
NM=50
L=50
```
9.5.2.3  **TWEB Configuration Files**

The TWEB configuration files reside in etc/twbs.

**Route Configuration Files**

For each TWEB route or synopsis there is a subdirectory etc/twbs/NNN containing template file route/synopsis info data etc/twbs/NNN/info.cfg. This file is normally created by the setup program avnsetup. An example file:

```
# etc/twbs/073/info.cfg

[headers]
wmo = FRUS41 KPBZ
afos = PITTWB073

[sites]
metars = KCLE, KYNG, KPIT, KJST, KAOO, KMEE, KMFB, KDCA
tafs = KCLE, KYNG, KPIT, KJST, KAOO, KMEE, KMFB, KDCA
```

Section [sites] lists TAFs and METARs that should be displayed in the TWEB Editor window.

**TWEB Product Configuration Files.** These files are located in etc/twbs. They are normally created by avnsetup. An example file:

```
# etc/twbs/PIT_TWBS.cfg

[sites]
workpil = PITWRKTWB
idents = 072,073
```

**Default Product File.** The optional DEFAULT file contains name of the product that should appear selected in the product loader in the TWEB Editor window. If not present, the first product in alphabetical order is chosen.

9.5.2.4  **X Resources Configuration Files**

The default and individual resource files are located in the directory etc/app-resources. To edit the default file etc/app-resources/X you can use any text editor. Individual files have naming convention X.N, where N is the forecaster number from etc/forecasters. These files are created by a graphical editor available from the AvnWatch GUI menu.

Here is the listing of the default file etc/app-resources/X.

```
!! default X resources configuration file
!! last modified July 10, 2005
!! user editable options
!! default font
!! change 100 to 140 if you want bigger default font
*font: -adobe-helvetica-bold-r-normal-*-100-*-p-*-iso8859-9
!! default colors
```
*background: grey70
*foreground: black
!! text window font
*Text.font: 7x14
!! text window colors
*Text.background: #2f3f6f
*Text.foreground: #ffffff
!! other text window specific options
*Text.width: 74
*Text.height: 24
*Text.cursor: hand1
!! forecast editor font - may want bigger
*textEditor.font: 8x16
!! forecast editor colors - may want different
*textEditor.background: #1f2f6f
*textEditor.foreground: white
*textEditor.width: 70
*textEditor.height: 12
!! cursor width and color
*textEditor.cursor: arrow
*textEditor.insertWidth: 5
*insertBackground: yellow
!! text viewer window options
*textViewer.width: 74
*textViewer.height: 12
!! forecast editor orientation: horizontal or vertical
*orientation: vertical
!! listbox font
*Listbox.font: 7x14
!! entry font
*Entry.font: 7x14
!! entry window colors
*Entry.background: white
*MessageBar*Entry.background: grey85
!! balloon message (popup) font
*Balloon.Label.font: 7x14
!! modifies behavior of dialogs
*transientDialogs: 1
!! confirmation on closing editor
*confirmClose: 0
!! confirmation on sending amendments/corrections
*confirmSend: 1
!! alert options: use colors for the first level that should result in
!! notification - pale green, yellow, orange, red, purple or none
*notifyDefconify: yellow
*notifyRaise: red
*notifyPlay: pale green
!! not implemented
!!*notifyTalk: none
*playFile: /awips/fxa/data/sounds/asterisk.au
!! blink on new notification
*blink: 1
!! warning/error level to disallow send:
!! 'always', 'warning', 'error', 'fatal'
*disallowSend: error
!! forecast editor options
!! use template, one of: template, merge, latest
*loadOrder: merge
!! ask for transmission time when sending routine forecasts
*asktime: 1
!! periodically save bulletins in a backup file
NOTES:

1. Lines starting with "!!" are comments. Resource names begin with a "*" and end with ":". Anything after a ":" is a resource value. Many of the allowable values are not well documented, or cryptic. For example, valid values for cursor shape in a text window

   *Text.cursor: hand1

are defined in file /usr/include/X11/cursorfont.h. You have to know that the leading XC in the following line:

   #define XC_hand1 58

must be stripped and 58 ignored. Valid color names are in /usr/lib/X11/rgb.txt, but if you don't find one you like, a hexadecimal number can be used instead, such as the following:

   *lifrColor: #6c1916

We suggest that you use the graphical resource editor available from in avnsetup before making changes to this file.
2. Climate applications introduced in AvnFPS 3.2 have their own resource files: XWindRose, XCigVisTrend and XCVMonthly. These are not user-specific.

### 9.5.2.5 Climatological Data

AvnFPS 3.2 introduces a new set of climatological GUIs requiring updated climatological files. The GUIs require fast, efficient access to voluminous amounts (often 30 years) of data. The existing NetCDF, delivered with AvnFPS 3.1 was determined too slow and inefficient when GUI response times needed to be in seconds. HDF5 is a file format that emphasizes storage and I/O efficiency to meet performance requirements. The AvnFPS team decided to use this format for its new climatology GUIs.

The climatological data files for all locations where the NWS prepares TAFs have been posted to a web page that is available to the World Wide Web. The Universal Resource Locator (URL) for the directory where they are posted is http://www.mdl.nws.noaa.gov/~avnfps/data/hdf5. File names are of the form CCCC.hd5, where CCCC is the location identifier where the data were observed. The files are rather large but HDF5 already stores the data in compressed format, so there is no post-processing step after downloading it. The files are ready to be used at this point by the new climatological GUIs.

Although some WFOs may find this hosting solution awkward, it seemed to be the safest way to protect the AWIPS WAN from the impacts of large data transfers.

The following procedures can be used to download the data files and make them available to AvnFPS.

#### To Download From an Internet Browser

1. On a host that can access the WWW, launch a browser application and direct it to [http://www.mdl.nws.noaa.gov/~avnfps/data/hdf5](http://www.mdl.nws.noaa.gov/~avnfps/data/hdf5). You will see a page that lists available stations along with file modification times and sizes. The link for each station points to a binary file.

2. Find the station whose data you want to download.

3. Use the appropriate browser features to download the file for the station your site needs. On most browsers, this involves performing a right click on the station name and a feature such as **Save Link Target As...** or **Save Link As**.

4. Move this file to AvnFPS climate directory /data/adapt/avnfps/climate. Files can be moved physically via removable media, transferred through the AWIPSPUB firewall, or transferred via LDAD.

5. Once the save process is completed you may proceed for other stations using the same procedure.
To download using wget command on a Linux host

1. Identify the URLs for each file you will download. They are of the form http://www.mdl.nws.noaa.gov/~avnfps/data/hdf5/CCCC.hd5, where CCCC is the Station ID.

2. Identify a directory on your local system where the files can reside temporarily.

3. Use the wget to download each file.
   
   ```bash
   wget --passive-ftp -nv dataURL -o output_dir/CCCC.hdf5
   ```
   
   where dataURL is the URL from the previous step, CCCC is the site ID you and output_dir is the directory where you want to save the files.

4. Move these files to AvnFPS data directory /data/adapt/avnfps/climate. This directory is visible from all hosts. Files can be moved physically via removable media, transferred through the AWIPSPUB firewall, or transferred via LDAD.

9.5.3 Starting and Stopping AvnFPS

All AvnFPS processes, servers, clients and utilities are started by the `avnstart.sh` shell script. This script is responsible for setting the appropriate environmental variables required by AvnFPS. The variables are actually set in the file `bin/avnenv.sh` which is sourced by `avnstart.sh`. The first argument is process name (a Python script residing in the directory /awips/adapt/avnfps/3.1/py without the extension .py). Remaining arguments, if any, are passed to the process without modification.

To start avnsetup from the command line, from your individual user account:

```bash
TYPE: /awips/adapt/avnfps/bin/avnstart.sh avnsetup
```

To start avnmenu from the command line, enter:

```bash
TYPE: /awips/adapt/avnfps/bin/avnstart.sh avnmenu
```

9.5.3.1 Starting AvnFPS Servers

AvnFPS requires four servers. They are:

- **avnserver** Which helps client processes find the services they need (name service) and notifies clients when data are available (event service).

- **avndis** The Data Ingest Server, copies data from various AWIPS sources into the AvnFPS directory tree, reformattting as needed.
avndrs  The Data Request Server, answers requests for data from client processes.

avnxs  The Forecast Transmission Server, manages transmit queues and the interface to the AWIPS software that transmits products.

The way these servers work together requires some rules.

- The order in which the servers start is important; avnserver must be started first.
- There can be only one instance of avnserver within WFO subnet; running it on both PX1 and PX2 will lead to unexpected results.
- The avndis depends on avndrs to process Low Level Wind Shear data.
- The simplest way to avoid these problems is to not launch these servers directly. The application avninit searches for active instances of all four AvnFPS servers and launches the ones that are not running. This application is modeled on the UNIX SYSV init and the AWIPS DataControllers. Like init and the DataControllers, avninit is a persistent process.

NOTES:

avninit will attempt to restart a failed server only 10 times within an hour. When the number of restarts exceeds 10 attempts, then avninit must be manually restarted.

avninit is invoked by the AWIPS startup scripts px2apps and px2apps together with other AWIPS applications. If you have to start the servers afterwards, use the remoteServers.sh shell script. This script accepts the following command line arguments:

```
/awips/adapt/avnfps/bin/remoteServers.sh [start|stop|restart]
```

This script has to be run as user fxa. It uses ssh to start/stop avninit on PX2. Therefore it can be run from any workstation. It is located in the directory /awips/adapt/avnfps/bin.

Example usage:

```
TYPE: /awips/adapt/avnfps/bin/remoteServers.sh start
```

To list avnfps processes use the ps command. As your individual user:

```
TYPE: ps -efw | grep avnpython
```

Using PX2 as an example, you should see output similar to the following:

```
fxa   11280  11280  0 1 Mar 01 07:00:00 avnpython /awips/adapt/avnfps/3.2/py/avninit.py px2f
fxa   11498 11280 11498 0 10 Mar 01 07:00:00 avnpython /awips/adapt/avnfps/3.2/py/avnserver.py -d -n px2f
fxa   11500 11280 11500 0 4 Mar 01 07:00:00 avnpython /awips/adapt/avnfps/3.2/py/avndis.py -d -n px2f
fxa   11508 11280 11508 0 10 Mar 01 07:00:00 avnpython /awips/adapt/avnfps/3.2/py/avndis.py -d -n px2f
fxa   11509 11280 11509 0 1 Mar 01 07:00:10 avnpython /awips/adapt/avnfps/3.2/py/avnxs.py -d -n px2f
```
To see all the threads in hierarchical format, add -L option to ps. Using PX2
as an example, as your individual user:

TYPE: ps -ef | grep avnpython

Using PX2 as an example, you should see output similar to the following:

```
fxa      11280     1 11280  0    0 Mar  01 ?        00:00:00 avnpython /awips/adapt/avnfps/3.2/py/avninit.py px2f
fxa      11498 11280 11498  0   10 Mar  01 ?        00:00:00 avnpython /awips/adapt/avnfps/3.2/py/avnserver.py -d -n px2f
fxa      11498 11280 11501  0  10 Mar  01 ?        00:00:00 avnpython /awips/adapt/avnfps/3.2/py/avnserver.py -d -n px2f
fxa      11498 11280 11502  0  10 Mar  01 ?        00:00:00 avnpython /awips/adapt/avnfps/3.2/py/avnserver.py -d -n px2f
fxa      11498 11280 11507  0  10 Mar  01 ?        00:00:03 avnpython /awips/adapt/avnfps/3.2/py/avnserver.py -d -n px2f
fxa      11498 11280 11515  0  10 Mar  01 ?        00:00:04 avnpython /awips/adapt/avnfps/3.2/py/avnserver.py -d -n px2f
fxa      11498 11280 11518  0  10 Mar  01 ?        00:00:10 avnpython /awips/adapt/avnfps/3.2/py/avnserver.py -d -n px2f
fxa      11498 11280 11518  0  10 Mar  01 ?        00:00:10 avnpython /awips/adapt/avnfps/3.2/py/avnserver.py -d -n px2f
fxa      11498 11280 11519  0  10 Mar  01 ?        00:00:00 avnpython /awips/adapt/avnfps/3.2/py/avnserver.py -d -n px2f
fxa      11498 11280 11521  0  10 Mar  01 ?        00:00:18 avnpython /awips/adapt/avnfps/3.2/py/avnserver.py -d -n px2f
fxa      11498 11280 11523  0  10 Mar  01 ?        00:00:00 avnpython /awips/adapt/avnfps/3.2/py/avnserver.py -d -n px2f
fxa      11500 11280 11500  0  4 Mar  01 ?        00:00:00 avnpython /awips/adapt/avnfps/3.2/py/avndrs.py -d -n px2f
fxa      11500 11280 11504  0  4 Mar  01 ?        00:00:00 avnpython /awips/adapt/avnfps/3.2/py/avndrs.py -d -n px2f
fxa      11500 11280 11505  0  4 Mar  01 ?        00:00:00 avnpython /awips/adapt/avnfps/3.2/py/avndrs.py -d -n px2f
fxa      11500 11280 11524  0  4 Mar  01 ?        00:00:01 avnpython /awips/adapt/avnfps/3.2/py/avndrs.py -d -n px2f
fxa      11508 11280 11508  0  10 Mar  01 ?        00:00:00 avnpython /awips/adapt/avnfps/3.2/py/avndis.py -d -n px2f
fxa      11508 11280 11511  0  10 Mar  01 ?        00:00:13 avnpython /awips/adapt/avnfps/3.2/py/avndis.py -d -n px2f
fxa      11508 11280 11512  0  10 Mar  01 ?        00:00:15 avnpython /awips/adapt/avnfps/3.2/py/avndis.py -d -n px2f
fxa      11508 11280 11513  0  10 Mar  01 ?        00:01:41 avnpython /awips/adapt/avnfps/3.2/py/avndis.py -d -n px2f
fxa      11508 11280 11516  0  10 Mar  01 ?        00:00:02 avnpython /awips/adapt/avnfps/3.2/py/avndis.py -d -n px2f
fxa      11508 11280 11517  0  10 Mar  01 ?        00:00:01 avnpython /awips/adapt/avnfps/3.2/py/avndis.py -d -n px2f
fxa      11508 11280 11520  0  10 Mar  01 ?        00:00:00 avnpython /awips/adapt/avnfps/3.2/py/avndis.py -d -n px2f
fxa      11508 11280 11526  0  10 Mar  01 ?        00:00:11 avnpython /awips/adapt/avnfps/3.2/py/avndis.py -d -n px2f
fxa      11508 11280 11527  0  10 Mar  01 ?        00:00:43 avnpython /awips/adapt/avnfps/3.2/py/avndis.py -d -n px2f
fxa      11508 11280 11532  0  10 Mar  01 ?        00:00:15 avnpython /awips/adapt/avnfps/3.2/py/avndis.py -d -n px2f
fxa      11509 11280 11509  0  1 Mar  01 ?        00:00:10 avnpython /awips/adapt/avnfps/3.2/py/avnxs.py -d -n px2f
```

NOTE: Red Hat Enterprise 4™ threads are listed differently by ps. The -H (process hierarchy) option cannot be used with the thread (m, l or T) request.

9.5.3.2 Stopping AvnFPS Servers

To stop the AvnFPS servers, as user fxa, run the following command from your workstation:

TYPE: /awips/adapt/avnfps/bin/remoteServers.sh stop

As an alternative option for stopping the AvnFPS servers, you can log on to the host(s) where the AvnFPS servers run and use the avnkill command located in /awips/adapt/avnfps/bin. The avnkill command first issues a SIGTERM signal and then waits 30 seconds for all threads to terminate gracefully. If the threads do not exit, avnkill issues a SIGKILL signal to those processes that continue to run.

Yet another option is to send a SIGTERM signal to avninit. For the sample listing above, as user fxa on PX2, the command would be: kill 30926

This should stop all the servers because avninit sends SIGTERM to all its child processes before quitting.

NOTE: When avninit starts, all the existing AvnFPS server processes are stopped.
Under special circumstances, (e.g., a hung server), you may want to stop and start a single AvnFPS server. Since avninit monitors them and starts those which are not running, you only need to kill the hung process. Bear in mind that the AvnFPS servers are threaded and only the top-level thread accepts signals.

For the ps listing above, the hierarchy is represented by order. For instance:

- **To restart the avndrs server, find the first occurrence of avndrs in the output above, and type:**
  
  ```
  TYPE: kill 11500
  ```

  **Result:** Process 11500 and its threads are terminated. Once the avndrs server exits, avninit will try to start a new avndrs server instance.

### 9.5.4 Avnsetup

The **avnsetup** is a GUI that can be used to configure most of the features of AvnFPS. Exhibit 9.5.4-1 shows the **avnsetup** main GUI.

![Exhibit 9.5.4-1. avnsetup Main GUI](image)

This section describes tasks that can be accomplished with this program.

**Important!** The AvnWatch GUI must be restarted in order for changes made using the following setup GUIs (except the Trigger Editor) to take effect. The reason for this is that the underlying files are read once into memory on startup.
9.5.4.1 Editing Monitoring Rules

From the avnsetup main GUI, select the Monitoring Rules button to launch the Monitoring Criteria Editor (Exhibit 9.5.4.1-1).

Exhibit 9.5.4.1-1. AvnFPS Monitoring Criteria Editor

The Monitoring Criteria Editor contains several pages, one for each monitored data source. Pages are selected via “recipe tabs” near the top of the menu. The tab labels refer to the following data sources:

- Rules for monitoring METAR observations (mtrs)
- Rules for monitoring lightning observations (ltg)
- Rules for monitoring lightning nowcasts (rltg)
- Rules for monitoring Collaborative Convective Forecast Product (ccfp)
- Rules for monitoring forecast grids (grids)
- Rules for monitoring low level wind shear (llws).
Each page has an identical layout. The top part lists currently defined rules, the bottom part contains a list of all available methods and the area where you can view, modify, add or delete a rule.

AvnFPS supports site-specific monitoring rules as well as a set of default rules. When the rule editor is started, the default rules are loaded. This is indicated by XXXX in the Site Id entry field.

- **To load the default set of rules**
  1. Enter ‘XXXX’ into the box labeled Site Id.
  2. Press Enter or select the Load button.

- **To load the specific rules for a site**
  1. Enter the Station ID into the box labeled Site Id.
  2. Press Enter or select the Load button.

- **To create a new set of site-specific rules**
  1. Modify the rules using the instructions below.
  2. Enter the applicable Station ID into the box labeled Site Id.
  3. Select the Save button.

- **To remove site-specific rules**
  1. Enter the Station ID into the box labeled Site Id.
  2. Select the “recipe tab” for the data source for which you want to remove the rules.
  3. Select the Delete button.

The lower half of the Monitoring Criteria Editor allows you to view and modify each of the monitoring rules. The current set of rules is listed by color and message in a selectable list. The Rule Editor at the bottom of the menu shows details of the rule that is currently selected. To the left to the Rule Editor is a list of Available methods. These methods are software techniques that compare the current forecast to the latest values received from the data source. Each method has its own set of arguments that configure its behavior.

- **To view the details of an existing rule**
  1. Select the rule from the list.

- **To modify an existing rule**
  1. Select the rule you want to modify from the list.
2. Use the Rule Editor to make the necessary changes.
3. Select Replace.
4. Select Save to write changes to the configuration file.

► To remove a rule
1. Select the rule you want to remove from the list.
2. Select Remove.
3. Select Save to write changes to the configuration file.

► To add a new rule
1. Select a method from the “Available Methods” list. The Rule Editor will update with the appropriate fields. You can see method details by pointing the cursor to the respective button on the list.
2. Use the Rule Editor to Modify arguments as needed. You must enter a message.
3. Select Add.
4. Select Save to write changes to the configuration file.

**NOTE:** Changes made in the Monitoring Criteria Editor will not affect the TAF Monitor until it is restarted.

9.5.4.2 Editing AvnFPS TAF Site Information

From the avnsetup main GUI, select the TAF SiteInfo button to display the TAF Site Info Editor, as shown in Exhibit 9.5.4.2-1. The fields in the editor are described in Table 9.5.4.2-1.
### Exhibit 9.5.4.2-1. AvnFPS TAF Site Info Editor

#### Table 9.5.4.2-1. AvnFPS TAF Site Info Editor Fields and Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TAF WMO</td>
<td>WMO header used to transmit forecast</td>
</tr>
<tr>
<td>TAF AFOS</td>
<td>PIL used during transmission</td>
</tr>
<tr>
<td>XTF WMO</td>
<td>WMO header used to transmit extended forecast (experimental)</td>
</tr>
<tr>
<td>XTF AFOS</td>
<td>PIL used during transmission</td>
</tr>
<tr>
<td>Visibility</td>
<td>A comma-separated list of visibilities in miles that define the categories used by the monitoring modules. Given the list $0 &lt; V_1 &lt; V_2 &lt; \ldots &lt; V_n$, the category $k$ consists of visibilities $V$ such that $V_{k-1} \leq V &lt; V_k$, $k = 1, \ldots, n+1$ and $V_0 = 0$, $V_{n+1} = \infty$.</td>
</tr>
<tr>
<td>Ceiling</td>
<td>A comma-separated list of ceilings, in feet, defining the categories used by the monitoring modules. These categories are defined in the same way as visibilities, shown above.</td>
</tr>
<tr>
<td>Radar Cutoff</td>
<td>A comma separated list of heights in meters defining lowest height to be used in computing LLWS using this radar. The length of the cutoff list must be equal to the number of radars listed in the 'Radar' text field below.</td>
</tr>
<tr>
<td>Profiler Cutoff</td>
<td>A comma separated list of heights in meters defining lowest height to be used in computing LLWS using this profiler. The length of list must be equal to the number of profilers listed in the 'Profiler' text field below.</td>
</tr>
<tr>
<td>Latitude</td>
<td>Site latitude, in degrees north</td>
</tr>
<tr>
<td>Longitude</td>
<td>Site longitude, in degrees east</td>
</tr>
<tr>
<td>Elevation</td>
<td>Site elevation, in meters</td>
</tr>
<tr>
<td>Runway(s)</td>
<td>A comma-separated list of runway directions in degrees. Used to calculate crosswind.</td>
</tr>
</tbody>
</table>
### Site Id

<table>
<thead>
<tr>
<th>Site Id</th>
<th>TAF Site ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>METAR</td>
<td>The METAR site ID associated with the TAF, most likely the same as TAF. If more than one site is needed, separate entries by a comma.</td>
</tr>
<tr>
<td>ETA</td>
<td>The closest site ID found in ETA BUFR message associated with the TAF.</td>
</tr>
<tr>
<td>NGMMOS</td>
<td>The closest site ID found in NGM MOS message associated with the TAF.</td>
</tr>
<tr>
<td>AVNMOS</td>
<td>The closest site ID found in AVN MOS message associated with the TAF.</td>
</tr>
<tr>
<td>ETAMOS</td>
<td>The closest site ID found in ETA MOS message associated with the TAF.</td>
</tr>
<tr>
<td>GFSLAMP</td>
<td>The closest site ID found in LAMP message associated with the TAF. This list is not yet available.</td>
</tr>
<tr>
<td>Radars</td>
<td>A comma-separated list of radars associated with the TAF. This list can be empty.</td>
</tr>
<tr>
<td>Profilers</td>
<td>A comma-separated list of profilers associated with the TAF. This list can be empty.</td>
</tr>
</tbody>
</table>

#### To add TAF site data

1. Enter the Station ID into the box labeled **Site Id**.
2. Select the **Update** button. Most of the fields will be filled in. Headers are read from the `afos2awips.txt` file. Latitude, longitude and elevation come from the `metarStationInfo.txt` file. Other values are built-in defaults.
3. Modify displayed entries as needed. Only the **METAR** entry is mandatory, all of the others are optional. A simple validation test is performed when you type. The program will not allow you to enter obviously incorrect values. Incomplete values are indicated by a pink background.
4. Select the **Save** button.
5. Create templates as described in the next section.

#### To modify TAF site data

1. Enter the Station ID into the box labeled **Site Id**.
2. Select the **Load** button.
3. Modify the displayed entries as needed.
4. Select the **Save** button.

#### 9.5.4.3 Editing AvnFPS TAF Site Templates

The bottom part of TAF Site Editor provides tools to create and edit template files that can be used to initialize forecasts. There is a separate template for each forecast issue time to allow users to enter issue specific phrases such as AMD NOT SKED AFT.…

#### To create TAF templates

1. Enter the Station ID into the box labeled **Site Id**.
2. Select the **Make** button. This will create all four templates containing Station ID, issue and valid times. For example, a 06Z template for KPIT would be KPIT DD0520Z DD0606 =
To modify TAF template

1. Enter the Station ID into the box labeled Site Id.
2. Select issue time from Issue menu and press the Edit to invoke a text editor. Do not press the Make button, as this will overwrite your current templates.

9.5.4.4 Editing AvnFPS TAF Product Definition

From the avnsetup main GUI, select TAF Products button. TAF Product Configuration Editor will display (see Exhibit 9.5.4.4-1).

Exhibit 9.5.4.4-1. AvnFPS TAF Product Configuration Editor

The existing products will be displayed in alphabetical order, with the default (if defined) on the top of the list. The Idents window displays list of all TAF sites in the product.

To add a product

1. Type the product name in the text window under the Products label and press Enter. An error message will be displayed that the product cannot be loaded, this is OK.
2. Enter Site ID in the window under the Idents label. Hit Enter after each site.
3. When finished, press Save. A new product definition file will be created.
4. Select the Verify button to check whether all necessary files exist. Site info and templates must exist for all Site IDs listed in the product definition.

To modify an existing product

1. Select the product on the Products list.
2. To add a new TAF site, type the ID in the window under the Idents label and hit Enter.

3. To delete a site, select it on the Idents list and press the Delete button.

4. Press Save to update product definition file.

▶ To delete a product
1. Select the product on the Products list.
2. Press Delete. A confirmation dialog will be displayed. Select OK; this will delete product definition file. TAF site files (info and templates) will not be removed.

▶ To mark a product as the default
1. Select the product on the Products list.
2. Press the Default. This will designate the selected product as default product.

9.5.4.5 Editing TWEB Route Information using AvnFPS

From the avnsetup main GUI, select TWEB Route Info button. An empty TWEB Route Info Editor will display (see Exhibit 9.5.4.5-1).

Exhibit 9.5.4.5-1. AvnFPS TWEB Route Info Editor

The TWEB Route Info Editor is an entry form for data specific for a particular route or synopsis. You must enter data for all routes before creating a product definition (that is, a “collective”). See Table 9.5.4.5-1.
Table 9.5.4.5-1. AvnFPS TWEB Route Info Editor Fields and Descriptions

<table>
<thead>
<tr>
<th>Route</th>
<th>TWEB Route Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>WMO</td>
<td>WMO header used to transmit forecast</td>
</tr>
<tr>
<td>AFOS</td>
<td>The PIL used during transmission</td>
</tr>
<tr>
<td>METARS</td>
<td>A comma-separated list of METAR site IDs that should be displayed in the viewer</td>
</tr>
<tr>
<td></td>
<td>window of the TWEB Editor.</td>
</tr>
<tr>
<td>TAFS</td>
<td>A comma-separated list of TAF site IDs that should be displayed in the viewer</td>
</tr>
<tr>
<td></td>
<td>window of the TWEB Editor.</td>
</tr>
</tbody>
</table>

To add TWEB route data

1. Enter the route number into the Route box.
2. Select the Update to initialize WMO and AFOS headers from the afos2awips.txt file.
3. Enter the METAR and TAF site IDs you want to see in the TWEB Editor in the Associated ids area. If you prepare TWEB products comprising more than one route, assure that the values you enter in those two fields are the same for each route listed in the product.
4. Press Save.
5. Select the Make button in the Templates area to create the basic template file.
6. If you want to customize the template, select Edit to invoke a text editor.

To modify TWEB route data

1. Enter the Route Number into the box labeled Route.
2. Modify displayed entries as needed.
3. Select the Save button.

9.5.4.6 Editing the AvnFPS TWEB Route Template

The bottom part of TAF Site Editor provides tools to create and edit template files that can be used to initialize forecasts.

To create a TWEB template using AvnFPS

1. Enter the Route Number into the box labeled Route.
2. Select the Make button to create a template file containing phrase:

   NNN TWEB DDHHMM . ALL HGTS AGL EXC TOPS.

   Where NNN is the route number.
To modify TWEB template
1. Enter the Route Number into the box labeled Route.
2. Select Edit to invoke a text editor. Do not press the Make button, as this will overwrite your current template.

9.5.4.7 Editing TWEB Product Definition using AvnFPS

From the avnsetup main GUI, select TWEB Products button to launch the TWB Product Configuration Editor (see Exhibit 9.5.4.7-1).

To add a product
1. Type the product name in the text window under the Products label and hit Enter. An error message will be displayed that the product cannot be loaded, this is ok.
2. Enter the site ID in the window under the Idents label. Hit Enter after each site.
3. When finished, press Save. A new product definition file will be created.
4. Select the Verify button to check whether all necessary files exist. Route info and the template must exist for all routes listed in the product definition.
Chapter 9  Background Applications

To modify an existing product
1. Select the product on the Products list.
2. Select the product on the list. To add a new TWEB route, type the route number in the window under the Idents label and hit Enter.
3. To delete a route, select it on the Idents list and select Delete.
4. Select Save to update product definition file.

To delete a product
1. Select the product on the Products list.
2. Press Delete. A confirmation dialog will be displayed.
3. Select OK; this will delete the product definition file. The TWEB route files (info and template) will not be removed.

To mark a product as the default
1. Select the product on the Products list.
2. Select Default. This will designate the selected product as default product.

9.5.4.8  Creating AvnFPS Database Triggers

AvnFPS obtains TAFs, TWEBs, and METAR observations from the fxatext database. As new versions of these products arrive, fxatext (actually, the underlying relational database engine, Postgres) can trigger the launch of an application program. AvnFPS uses this trigger feature to retrieve TAF, TWEB and METAR products for monitoring as soon as they arrive in the database. The Trigger Editor manages the configuration of the textdb triggers needed to support AvnFPS. The current version creates the trigger template to be processed by localization and simultaneously updates the watchwarn table in the fxatext database holding the Postgres trigger scripts. So you do not have to rerun localization after creating the template file. However the file is still necessary, as the localization process drops watchwarn and re-creates it from templates.

To start the Trigger Editor, select the Triggers button. The Trigger Editor will display.

On startup, the Trigger Editor (see Exhibit 9.5.4.8-1) searches the AvnFPS configuration files for all TAF sites, METAR sites, and TWEB routes that have been configured for the AWIPS site. (These are files stored in the etc/tafs, etc/mtrs, and etc/twbs directories.) The three columns in the Trigger Editor, TAF, METAR and TWB, list those sites and routes. If there is a template file, etc/triggerTemplate, its entries are used to fill the corresponding PIL boxes. If the trigger template does not have an entry for an ID, then the relevant box will remain blank with a pink background. The column XTF contains entries for extended (30 hour) TAFs. Unless your office issues such forecasts, it will be
empty. Column CCFP should always have entries as shown above. Those text
products are not site-specific.

Exhibit 9.5.4.8-1. AvnFPS Trigger Editor

To create AvnFPS triggers

1. Select the Update. The program will try to determine missing PILs for TAFs
   and TWEBs from the file afostawips.txt. METAR PILs are created from
   corresponding TAF PILs by replacing TAF by MTR.

   NOTE: The existing values will not be overwritten.

2. Correct the entries and fill in empty fields.

3. Press Make when finished. Empty (pink) entries will be skipped. If you have
   an invalid entry, an error message will display.

9.5.4.9 Using the AvnFPS Text Editor

Use the text editor in AvnFPS to edit the remaining configuration files: forecasters,
logging.cfg, server.cfg, gui.cfg, ids.cfg, flt_cat.cfg, and grp_tag.cfg. It is invoked by
selecting Text Edit.

To edit configuration files using the AvnFPS Setup Editor

1. From the avnsetup Main GUI, select Text Editor. The Setup Editor will
display. (See Exhibit 9.5.4.9-1.)
2. Select **Open**. A file selection dialog listing the contents of directory `/etc` will display. (See Exhibit 9.5.4.9-2.)

3. In this dialog, select a file to edit, then select **Open**.

4. Select **Save** when finished. If you want to create a file, simply type in its content in the text window, then select **Save As**.

You can also use the text editor to edit X resource configuration file. However it is suggested that you only modify the system-wide file that is used as an initial selection when a forecaster modifies his own resources through the graphical editor available in **AvnWatch**. Proceed as above, in the file selection dialog select first, directory **app-resources**, then the file **X**. You may edit files for individual forecasters (those with
names X, however any comments you enter will be stripped by the graphical editor each time the forecaster modifies the resources from **AvnWatch**.

**NOTE:** Font and color names may be cryptic. You may want to use the graphical editor in **AvnWatch** first to display color codes and available fonts.

### 9.5.5 Configuring IFPS to Send Gridded Data to AvnFPS

The GFE text formatter (IFPS2AvnFPS.py) was created to do the job of accessing IFPS gridded database and writing the information to the `/data/adapt/avnfps/grids` directory. **IFPS2AvnFPS.py** is run at 10 times an hour via cron on the px2f machine.

**NOTE:** The only requirement for the new text formatter to run is that all TAFs must have a GFE edit area defined in the ifpServer database. This restriction should be removed in a later build. Offices that have configured IFPS to send gridded data to AvnFPS are likely to have already satisfied this requirement.

### 9.5.6 Logging

AvnFPS has an extensive logging system; it is similar to collective logging in AWIPS. The amount of information written to log files and its format are configurable.

Each program has its own log file, one for each day of the week. File naming uses the scheme `proname_DayOfWeek`, so **avndis** will produce files named `avndis_Mon` through `avndis_Sun`. Files are broken (old ones are closed and new ones are opened) at the first write attempt of each new day.

The file `etc/logging.cfg` controls the location, format and log level for the log files. The format and log level is the same for all programs.

**NOTE:** The log directory is local to each server/workstation. The `/awips/adapt/avnfps/3.2/logs` directory points to `/data/logs/adapt/avnfps`, which resides on the local partition. This is consistent with the AWIPS standard.

For example:

- **TYPE:** 11 /awips/adapt/avnfps/3.2/logs
  - Your output should look similar to the following:

```
lwxrwxrwx 1 fxa   fxalpha  23 Oct 8 16:41 /awips/adapt/avnfps/3.2/logs -> /data/logs/adapt/avnfps
```

Here is an example listing of a log file for **avndis**:

Log rollover at 00:00:01
2006-03-01 00:00:01,280 DEBUG [10517:126217136] TriggerThread: Processing PHLMTRMDT
2006-03-01 00:00:01,647 INFO  [10517:126217136] TextThreadP: Processed KMDT from SAUS70 KMDT 080000
2006-03-01 00:00:01,649 DEBUG [10517:126217136] TriggerThread: Done PHLMTRMDT
2006-03-01 00:00:01,649 DEBUG [10517:126217136] TriggerThread: Processing PITMTRDUJ
2006-03-01 00:00:01,656 DEBUG [10517:27261872] DataIngestServ: Msg: {'src': 'mtrs', 'ident': 'KMDT'}
2006-03-01 00:00:01,754 INFO [10517:126217136] TextThreadP: Processed KDUJ from SAUS70 KDUJ 080000
2006-03-01 00:00:01,755 DEBUG [10517:126217136] TriggerThread: Done PITMTRDUJ
2006-03-01 00:00:01,756 DEBUG [10517:126217136] TriggerThread: Processing BUFMTRART
2006-03-01 00:00:01,981 DEBUG [10517:27261872] DataIngestServ: Msg: {'src': 'mtrs', 'ident': 'KDUJ'}
2006-03-01 00:00:01,989 INFO [10517:126217136] TextThreadP: Processed KART from SAUS70 KART 080000
2006-03-01 00:00:01,990 DEBUG [10517:126217136] TriggerThread: Done BUFTMRART
2006-03-01 00:00:01,993 DEBUG [10517:27261872] DataIngestServ: Msg: {'src': 'mtrs', 'ident': 'KART'}
2006-03-01 00:00:05,289 DEBUG [10517:126217136] TriggerThread: Processing HFOMTRHNY
2006-03-01 00:00:05,576 INFO [10517:101030832] LtgThread: Processing file /data/dfa/point/binLightning/netcdf/20060228_2300
2006-03-01 00:00:05,629 INFO [10517:126217136] TextThreadP: Processed PHNY from SAUS80 PHNY 080000
2006-03-01 00:00:05,631 DEBUG [10517:126217136] TriggerThread: Done HFOMTRHNY
2006-03-01 00:00:05,633 DEBUG [10517:126217136] TriggerThread: Processing BUFTMTRROC
2006-03-01 00:00:06,020 DEBUG [10517:27261872] DataIngestServ: Msg: {'src': 'mtrs', 'ident': 'PHNY'}

Each line contains the following fields:

- Date and time of the event
- Severity level (in caps)
- Process and thread number (in square brackets)
- Python module name which generated the message
- Actual message content.

Some typical errors may include:

2005-07-08 02:00:48,843 INFO [10517:84941744] MosData: Retrieved ngm data for BUF
2005-07-08 02:00:48,851 ERROR [10517:84941744] MosData: BVI not in /data/dfa/point/mos/NGM/netcdf/20050708_0000*
2005-07-08 02:00:48,882 INFO [10517:84941744] MosData: Retrieved ngm data for CHA
connected=1 type thread.error**
--- traceback of this exception follows:

```
File "/awips/adapt/avnfps/Python-2.4.1/lib/python2.4/site-packages/Pyro/protocol.py", line (922), in Daemon::connectionHandler
Source code:
  self.handleInvocation(conn)
File "/awips/adapt/avnfps/Python-2.4.1/lib/python2.4/site-packages/Pyro/core.py", line (695), in Daemon::handleInvocation
Source code:
```

---
Notes:
* This message indicates problem with TAF Site Info configuration file: data for KBVI is not distributed for NGM MOS product. This does not cause any harm, but one can either select another site which is close enough, or leave the NGM MOS site id empty.

** Some messages provide detailed traceback to point out to the exact location of the code where the error occurred. In this case, the error comes from Pyro code and it is relatively serious. The data ingest server hit the threshold of allowable memory consumption. avndis has this threshold set programmatically to avoid clogging the whole operating system. In a situation like this one should restart avndis to avoid data loss. If the problem reoccurs, further investigation is required. (Note: This example is contrived; the threshold was set too low during development.)

9.5.7 *AvnFPS Troubleshooting*

Techniques for troubleshooting various aspects of AvnFPS are addressed in the Government AvnFPS system manager’s manual. This section covers some of the more likely fault scenarios and troubleshooting techniques.

9.5.7.1 *Overall AvnFPS Server Health*

A number of server processes keep AvnFPS running. A series of status lights on the TAF Monitor GUI reports the status of most of these processes. “Keep Alive” messages are sent among the servers every 30 seconds to set the colors of these status lights. Section 9.5.1, “System Overview” contains a detailed description of each server and the host on which it should be running. Server hosting can vary between AWIPS releases.

9.5.7.1.1 *Diagnostic Tools*

► If all the server status lights are red, check for
   • A failed name/event server (avnserverserver)
   • A network failure.

► To list the AvnFPS server processes

The command ps command is used to list server processes and any component threads. In the example below, the process is run on PX2 and the operating system is Red Hat Enterprise 4. The indented text in the rightmost column indicates hierarchy.

**TYPE:**

```
ps -efwH | grep avnpython
```

Your output should be similar to the following:

```
fxa 18615     1  0 Mar06 ?  00:00:00  avnpython /awips/adapt/avnfps/3.2/py/avninit.py px2f
fxa 18713 18615  0 Mar06 ?  00:00:33  avnpython /awips/adapt/avnfps/3.2/py/avnserver.py -d -n px2f
fxa 18718 18615  0 Mar06 ?  00:00:01  avnpython /awips/adapt/avnfps/3.2/py/avndrs.py -d -n px2f
fxa 18730 18615  0 Mar06 ?  00:01:08  avnpython /awips/adapt/avnfps/3.2/py/avndis.py -d -n px2f
fxa 18735 18615  0 Mar06 ?  00:00:14  avnpython /awips/adapt/avnfps/3.2/py/avnxs.py -d -n px2f
ncfuser:853$ 
```
To list the threads spawned by AvnFPS server processes

**TYPE:**  
\texttt{ps -efL | grep avnpython}

**NOTES:** Some comments follow:

<table>
<thead>
<tr>
<th>PID</th>
<th>Comments</th>
<th>“Kill”-able?</th>
</tr>
</thead>
<tbody>
<tr>
<td>18615</td>
<td>avninit daemon</td>
<td>Yes</td>
</tr>
<tr>
<td>18713</td>
<td>avnserver process</td>
<td>Yes</td>
</tr>
<tr>
<td>18718</td>
<td>Data Request Server (avndrs) process</td>
<td>Yes</td>
</tr>
<tr>
<td>18730</td>
<td>Data Ingest (avndis) process</td>
<td>Yes</td>
</tr>
<tr>
<td>18735</td>
<td>Transmit Server (avnxs) process</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Threads of a process will not accept signals from a kill command. You must identify the process that owns the thread and kill that top-level process. Once a parent process terminates, all of its threads will terminate as well.

To diagnose connections among various AvnFPS servers

The command \texttt{netstat -a | grep 9090} can be used to diagnose connections among the various AvnFPS servers. The name/event server uses port 9090 to disseminate information about the various servers that are up and running.

On PX2

**TYPE:**  
\texttt{netstat -a | grep 9090}

- Your output should be similar to the following:

```
  tcp  0  0 px2f-wfo:9090         px2f-wfo:55812        ESTABLISHED
  tcp  0  0 px2f-wfo:9090         px2f-wfo:55819        ESTABLISHED
  tcp  0  0 px2f-wfo:55819        px2f-wfo:9090          ESTABLISHED
  tcp  0  0 px2f-wfo:55812        px2f-wfo:9090          ESTABLISHED
  tcp  0  0 px2f-wfo:9090         lx3-wfo:40638          ESTABLISHED
```

**NOTES:**

1. The columns are Protocol, Receive Queue, Send Queue, Local Address, Foreign Address, and State.

2. Six connections to and from the name/event server can be seen in the px2 listing.
   - The first four established connections show connections between the servers on px2 and the name/event server.
   - There is an additional connection to a process on lx3. Most likely, this is an instance of the AvnWatch GUI.

### 9.5.7.2 Log Files

Log files for AvnFPS3.2 server processes are stored in the directory tree
\texttt{/data/logs/adapt/avnfps}, local to the host computer. The names of the logs files are
formed by the name of the application and the current day of the week (e.g., avnmenu_Thu and avndis_Fri). All applications use collective logging which means that log file entries from different instances of an application can be found, interleaved, in a single log file. The following screen capture shows sample directory listings:

### GUI logs on a workstation

- avnclimate_Fri  avnmenu_Mon  avnmenu_Thu  avnqsstats_Tue  avnsetup_Thu  avnwatch_Mon  avnwatch_Thu
- avnclimate_Wed  avnmenu_Sat  avnmenu_Tue  avnsetup_Mon  avnsetup_Thu  avnwatch_Sat  avnwatch_Thu
- avnmenu_Fri  avnmenu_Sun  avnmenu_Wed  avnsetup_Sun  avnwatch_Fri  avnwatch_Sun  avnwatch_Wed

### Server logs on px1/px2 …

- avndrs_Fri  avndrs_Sun  avndrs_Wed  avninit_Sun  avninit_Wed  avnserver_Sat  avnserver_Tue
- avndrs_Mon  avndrs_Thu  avninit_Fri  avninit_Thu  avnserver_Fri  avnserver_Sun  avnserver_Wed
- avndrs_Sat  avndrs_Tue  avninit_Sat  avninit_Tue  avnserver_Mon  avnserver_Thu

It is possible for certain unexpected errors to go undetected in log files. One way to test against this possibility is to launch a GUI process from the command line and watch the standard error and standard output streams.

- **To launch the AvnFPS startup menu from the command line**
  TYPE: `/awips/adapt/avnfps/bin/avnstart.sh avnmenu`

### 9.5.7.3 Data Ingest

The AvnFPS Data Ingest Servers (instances of `avndis`) monitor events in AWIPS and decode various data as they become available. For some data types, the term “decode” means little more than copying a file. Once `avndis` has put the data into the AvnFPS directory tree, the AvnFPS Data Request Servers (instances of `avndrs`) serve the data to the various GUIs.

### 9.5.7.3.1 Data Ingest Troubleshooting

The log files for `avndrs` show data being delivered to GUIs as well as data access problems. See if the GUIs are connecting to `avndrs` successfully and receiving data.

Log files for `avndis` show data arriving and being decoded. Errors are rather easy to spot in these files.

### 9.5.7.3.2 Specific Hints for Specific Data Types

- **Text (TAFs, METAR observations)**: AvnFPS database triggers deliver products to the `/awips/adapt/avnfps/data/text` directory. As an alternative to triggers, the AWIPS acquisition server can be configured to copy arriving text products into `/awips/adapt/avnfps/data/raw` which can be processed by `avndis`.
Text (TAFs, METAR observations) As an alternative to triggers, the AWIPS acquisition server can be configured to copy arriving text products into /data/fxa/point/avnfps/raw which can be processed by avndis.

Lightning Observations The /data/fxa/point/binLightning/netcdf directory is monitored for new and updated files.

Lightning Probability: The /data/fxa/img/SBN/netCDF/LATLON/3hr/LTG directory is monitored for new and updated files.

Low-Level Wind Shear The following directories are monitored for new files: /data/fxa/point/profiler/netcdf /data/fxa/point/progression/netcdf /data/fxa/point/progression/progression, where CCCC is the radar ID

IFPS Grids IFPS controls the process that exports the data to /awips/adapt/avnfps/data/grid where it is monitored.

9.5.7.4 Product Transmission

Product transmission is covered in depth in Section 9.5.1.4, Transmission Server.” Check the log file for avnxs. If a forecast prepared by AvnFPS was written to the pending queue, /awips/adapt/avnfps/3.2/xmit/pending, and the transmission server attempted to access the file, then there should be a corresponding entry, starting with the word SUCCESS or FAILNNN, where NNN is the code returned by the system call to handleOUP.pl.

If avnxs indicates there was a handleOUP.pl failure, then investigate the handleOUP log file, /data/logs/fxa/<yyyymmdd>/handleOUP.

9.6 NOAA Weather Radio Editor

The NOAA Weather Radio (NWR) Editor (NWRE) provides the manual Voice-Ready Product (VRP) creation and dissemination functionality on AWIPS. It consists of four basic operations:

- VRP creation from values stored in the AWIPS fxatext and ifps_<xxx> databases, where <xxx> is the AWIPS site ID, all in lowercase (e.g., sto)
- VRP creation from a previously stored/distributed VRP stored on disk
- Creation/management of VRP default message attributes stored in the ifps_<xxx> database
- Dissemination of a properly formatted VRP from AWIPS to the NWR Console Replacement System (CRS).
Chapter 9

NOTE: Knowledge of proper VRP message attribute/synopsis creation and use is assumed at this point. This chapter is not meant to serve as a tutorial on proper Voice-Ready Product creation/use.

9.6.1 NWR Editor Setup and Configuration

In order to accomplish the NWRE’s four basic tasks, the FXA_TEXT and DB_ICWF system environment variables must be set at the sites.

Also, one configuration data file must exist for the NWRE to execute correctly. This file is used by the transferNWR process to determine the proper line configuration and CRS login needed for successful VRP dissemination. This file must exist on each workstation used for manual VRP creation and dissemination, along with the transferNWR script, the nwrEditor script, and the nwrEditorWish (TclTk interpreter); the sendToNWR process must exist on the Data Server.

Configuration Files

The NWRE requires that prior to startup the following configuration files exist on each workstation that will run the NWRE. All of the files reside on the workstations in /awips/xa/bin unless otherwise noted.

- **nwrEditor.** A TclTk script providing GUI access to a library of NWR-specific functionality.
- **nwrEditorWish.** A TclTk interpreter that has been extended to include NWR application-specific commands.
- **transferNWR.** An Expect script used to move voice ready products to the /data/xa/workFiles/nwr/ready directory.
- **nwr.cfg.** A configuration file containing AWIPS/NWRCRS line configuration, NWRCRS login, and a line of dummy text as placeholder (this will be the NWRCRS password until CRS Build 10 is released).

The *nwr.cfg* file is located in the SFXA_DATA/workFiles/nwr directory. It is needed by the transferNWR and sendToNWR processes, and therefore the NWRE will be installed with sample entries. The system administrator will need to populate the file with site-specific line configuration, CRS login, and placeholder text. Table 9.6.1-1 shows the file format.

<table>
<thead>
<tr>
<th>File Name</th>
<th>File Line #</th>
<th>File Format</th>
<th>BNF</th>
</tr>
</thead>
<tbody>
<tr>
<td>nwr.cfg</td>
<td>1. CRS login 2. &quot;placeholder text&quot; 3. AWIPS--CRS line</td>
<td>0 mp’n ’n’ XXXXLAN’n’n’</td>
<td>CRS_LOGIN=&lt;new line character&gt;&lt;PLACEHOLDER&gt;[new_line_character]&lt;EOF&gt; &lt;LINE_CONFIG&gt;&lt;new line character&gt;</td>
</tr>
</tbody>
</table>
Environment Variables

The environment variables used in discussing the NWRE are shown in Table 9.6.1-2.

<table>
<thead>
<tr>
<th>Environment Variable Name</th>
<th>Environment Variable Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>FXA_TEXT</td>
<td>The fxatext database</td>
</tr>
<tr>
<td>DISPLAY</td>
<td>Your workstation display (e.g., xt1-sto:0.0)</td>
</tr>
<tr>
<td>DB_ICWF</td>
<td>AWIPS site ID (e.g., sto)</td>
</tr>
<tr>
<td>LOG_PREF</td>
<td>/awips/fxa/data/ingestLogPref</td>
</tr>
</tbody>
</table>

Use the `echo` command to verify that the environment variables were set correctly during installation:

```bash
TYPE: echo $FXA_TEXT
TYPE: echo $DISPLAY
TYPE: echo $DB_ICWF
```

**NOTE:** The installation process should set these variables to the proper values for each workstation.

9.6.2  NWR Editor Event Logging

NWRE log files are generated and maintained in a log file directory on each workstation. Users can inspect pertinent log files for errors. On a workstation, log data are written to the `/data/logs/fxa/display/SDISPLAY/<yyyymmdd>` file, where `<yyyymmdd>` is the current date in year-month-day format and `SDISPLAY` is the display environment variable, as previously defined.

**NOTE:** Remember that the workstation logs are local to the machine from which NWRE is being run. For example, if you log in from lx5-sto to lx2-sto and set `SDISPLAY` to lx5-sto:0.0, you will find a directory on lx2-sto named `/data/logs/fxa/lx5-sto:0.0/<yyyymmdd>` (where `<yyyymmdd>` is the current date) if you use the NWRE to send a VRP to the CRS.
NOTE: The full display variable is also used for the X-terms (e.g., xt1-sto:0.0) for SDISPLAY and the log file pathname when the NWRE is run on an X-term. However, if you run the NWRE on a workstation, the path to the logs would be /data/logs/fixa/display:/0/<yyymddd> or /data/logs/fixa/display:/0.0/<yyymddd>.

All NWRE logs have the following naming convention:

```
nwrEditorWish<processId><server><hhmmss>
```

**Example:** nwrEditorWish186909lx2-nhda171016

NWRE log files contain minimal logging when the process runs successfully; normally only problems are logged. To see all logging from process startup to process shutdown, the $LOG_PREF file must contain an entry for the nwrEditorWish with debug turned on. Use of logStream’s logPreferences configuration file is outside the scope of this discussion. The standard FSL-developed “logStream” logging mechanism is used and the formats for the logs are as documented by the logStream foundation class.

Exhibit 9.6.2-1 shows a sample NWR Editor nwrEditorWish log file with $LOG_PREF debug=on for the NWRE process, which would constitute normal process startup.

```
LOG-STATUS: Log file opened on host bd1-napo at Wed May 12 17:10:17 2006
17:10:16.962 VRPMessageAttributes.C DEBUG: default VRPMessageAttributes constructor called
default VRPMessageAttributes set
17:10:17.029 VRPSynopsis.C DEBUG: default constructor VRPSynopsis() called
17:10:17.029 VRPSynopsis.C DEBUG: default _vrpSynopsis = "SYNOPSISTEST"
17:10:17.030 VoiceReadyProduct.C DEBUG: preparing to get voice ready product in four part structure
17:10:17.030 VRPMessageAttributes.C DEBUG: getting vrp message attributes
17:10:17.041 VRPMessageAttributes.C DEBUG: moving everything toupper() except eolac
17:10:17.042 VRPMessageAttributes.C DEBUG: returning vrp message attributes...
17:10:17.042 VoiceReadyProduct.C DEBUG: esca/msg attributes/ newline:
   [aT_ENGCCNNNNX040512171000405121710 AD c0405121710
17:10:17.042 VoiceReadyProduct.C DEBUG: synopsis/newline/escb/endl:
   [bSYNOPSISTEST
17:10:17.044 VoiceReadyProduct.C DEBUG: vrp data is: [aT_ENGCCNNNNX040512171000405121710
   AD c0405121710
```

**Exhibit 9.6.2-1. Sample NWR Editor nwrWish Log File Where debug=on**

As previously noted, the default is for the NWRE log file to record only problems with the NWRE process. In addition, a unique log file for each call to transferNWR can be
found in the same logging directory as the NWRE process in a file given a name with the following format:

```
logStreamExpect
```

**Example:** `logStreamExpect`

A more detailed log of the actual VRP transfer from AWIPS to the CRS via `sendToNWR` can be found on the PX1 in `$LOG_DIR/<yyyymmdd>` in a file given a name with the following format:

```
logStreamFxpect
```

During one NWRE session, a user will have:

- One log file `nwrEditorwish<processid><server><hhmmss>`,
- One `sendToNWR logStreamExpect` log file, and
- Perhaps many `transferNWR logStreamExpect` log files.

This ratio depends on the number of “send to CRS” sessions invoked by the user from the NWRE. Exhibit 9.6.2-2 displays a sample `sendToNWR logStreamExpect` log file.

```
LOG-STATUS: Log file opened on host pxl-nhda at Wed May 12 18:48:06 2006
18:48:06.177 TclLogStream.C EVENT: File CCCNNNXXX will be automatically transferred
18:48:06.586 TclLogStream.C EVENT: Successful ping to 0MP
18:48:06.809 TclLogStream.C EVENT: Successful ping to 5MP
18:48:09.265 TclLogStream.C EVENT: Master processor is 0MP
18:48:09.266 TclLogStream.C EVENT: 7 ss_ms processes running on OMP currently
18:48:11.012 TclLogStream.C EVENT: transfer complete to 0MP
18:48:11.228 TclLogStream.C EVENT: Successful transfer of CCCNNNXXX to OMP
```

Exhibit 9.6.2-2. Sample `sendToNWR logStreamExpect` Log File

### 9.6.3 The NWR Editor GUI

There are three ways that the NWRE process can be started:

- By selecting it from the AWIPS D2D System Control (root) menu
- The NWR Browser **Edit** button
- The Text Workstation Text Window Display **NWR Editor** button.

The Government-furnished Exhibit 9.6.3-1 shows these three ways. The NWRE is shut down by using the **File -> Close** menu selections on the **NWR Editor GUI**.
The NWRE GUI allows users to communicate with the `ifps_<xxx>` database to create and manage default message attributes. Using previously stored values in the `ifps_<xxx>` and `fxatext` databases, or by supplying the proper values, users can manually create a properly formatted VRP for public dissemination over the NWR Console Replacement System (NWRCRS).

The NWRE GUI was designed to emulate basic NWRCRS functionality and to include some additional features. Current NWRCRS capabilities rely on speech synthesis, tower transmission, and other features that are not present within the NWRE GUI. The NWRE is not meant to replace NWRCRS functionality; instead it is meant to allow an AWIPS user to access to specific NWRCRS functionality in an easy to understand and accessible manner.

To a large extent, users familiar with the NWRCRS can readily apply their working knowledge to navigate and interact with the NWRE. It is recommended that users who are unfamiliar with basic VRP generation and dissemination read the NWRCRS user’s manual section dealing with a VRP’s four-part structure, paying special attention to the creation of listening area codes and message reference descriptors. There are certain restrictions on the information that may be put into the NWRE’s text fields. For further information on the legal values for these fields, please refer to Government NWRCRS manuals.

The NWRE GUI provides access to VRP generation via three routes:
• The concatenation of default message attributes stored in the \texttt{ifps\_<xxx>} database and the synopsis created from text products stored in the \texttt{fxatext} database

• The retrieval of a previously formatted and disk-stored (flat file) VRP

• The provision of required values (message attributes and synopsis) for a new VRP.

The \textbf{NWRE GUI} provides the user with the following options:

• Default message attribute creation/management

• VRP dissemination

• VRP request from stored values

• Disk/database storage of part or whole VRPs.

The \textbf{NWRE GUI} is displayed in Exhibit 9.6.3-2.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{exhibit9_6_3_2.png}
\caption{The NWR Editor GUI}
\end{figure}

The upper-half of the GUI is used to configure the following message attributes:

• Text identifier - Loads a product from the database into the Editor
• NWR identifier - AFOS ID (CCCNNNXXX), where
  
  CCC is the AFOS node origination site.  
  NNN is the AFOS-based product category.  
  X ([X] | [XX]) is the AFOS-based specific product category.  

  Creation date and time  
  Effective date and time  
  Expiration date and time  
  Periodicity  
  Listening area codes (LAC)  
  Message reference descriptor (MRD)  
  Alert  
  Save  
  Format  
  Confirmation  
  Interrupt  
  Storage  

  The lower half of the NWRE GUI is concerned with the configuration of the message contents, also known as the Synopsis, and is a free text area.  

  The very bottom of the NWRE GUI displays quick buttons for product dissemination and saving to disk and a status bar to keep the user informed.  

9.6.4  NWR Editor Application Design Overview

  The NWRE GUI layer is a thin layer that sits on nwrEditorWish, a TclTk interpreter that has been extended to incorporate NWR application-specific commands. The GUI is a facade to the underlying NWR functionality. This was accomplished by decoupling the GUI from the interpreter. By creating an NWR-specific interpreter, applications needing application-specific commands are easily and efficiently expanded using the underlying C++ classes. This allows for the overhead incurred from scripts to be minimized to the small amount needed for display of the GUI. The GUI is responsible for sending events to the interpreter where, upon notification from the event handler, the interpreter processes the GUI’s request.  

  All requests to the underlying functionality are broken into two types of events, Get() and Set(), based on the facade and command patterns.  

  Exhibit 9.6.4-1 presents a Government-furnished high-level design with the NWR Editor (Tcl script) layer on top of the VRP Generator. C/C++ function calls are shown as thin arrows and Tcl command calls are shown as thick arrows.
The Tcl C library implements the interpreter and the core Tcl commands such as `set`, `while`, `proc`, `open`, and `socket`.

Application-specific Tcl commands are implemented in C++ and registered as commands in the interpreter.

The interface to all Tcl commands is the same.

The interpreter calls these command procedures when the script uses a Tcl command.

The application-specific command procedures are thin layers over existing functionality in extensions that implement suites of Tcl commands in the compiled code.

In the exhibit, the large arrow represents calls from the VRP Generator back to the MPG GUI script layer. The application can call out to the script layer at any point, even inside command procedures.

Exhibit 9.6.4-2 presents a Government-furnished system-level view of the application.
Exhibit 9.6.4-2. NWR Editor Application Structure – System View
Chapter 10
Asynchronous Product Scheduler
Chapter 10. Asynchronous Product Scheduler

Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.1 Configuration Files and Environment Variables</td>
<td>2</td>
</tr>
<tr>
<td>10.2 Processes</td>
<td>3</td>
</tr>
<tr>
<td>10.3 Additional Notes</td>
<td>6</td>
</tr>
</tbody>
</table>

List of Exhibits

Exhibit 10.2-1. Acquire and Ingest of Asynchronous Products 4
The Asynchronous Product Scheduler (APS) product distribution allows text products to be routed to/from external PCs configured on an AWIPS communications port. APS supports text product dissemination from the PC to the Console Replacement System (CRS) and the NOAA Weather Wire Service (NWWS) as well as storage of text products to the AWIPS text database.

APS runs on Linux pre-processors. It consists of eight executables that run together as a daemon to:

- Receive newly arrived text database products in real time
- Receive newly arrived text products in real time
- Ingest locally generated text products into AWIPS
- Send requests for text products
- Send requests for text database products.

The executables are located in /awips/ixa/bin, and are described as follows.

- The asyncScheduler spawns asyncLine, one per aps_line.tbl entry and asyncExec to route incoming products.
- The asyncLine is a persistent process that is spawned by asyncScheduler to handle serial port I/O.
- The asyncExec is spawned by asyncScheduler to route incoming products (to the Network Control Facility (NCF) and NWWS and/or CRS, depending on the product’s entry in aps_pil.tbl).
- The asyncPilMsgServer is a persistent process. It receives the notification messages from asyncPilNotify process.
- The asyncPilNotify is called by the asyncPollData.pl persistent process when a user-subscribed product arrives.
- The asyncPollData.pl is a persistent process. It constantly scouts the products arriving at the ${EDEX_FXA_HOME}/workFiles/asyncScheduler/incoming directory.
- The asyncWatchDog.sh is a persistent cron process. It monitors the existence of the asyncPollData.pl process every second. If it does not exist, it automatically starts it up.
- The asyncPilTrigger.sh is a process invoked by the asyncScheduler process to subscribe the user-defined products to the metadata database during asyncScheduler startup. It also instructs the EDEX textdb server where to deposit the incoming product.
10.1 Configuration Files and Environment Variables

Three configuration files tell the APS what products to process and how to process them. These files are located in $FXA_DATA/workFiles/asyncProdScheduler, which is NFS-mounted from the NAS.

- **aps_line.tbl.** This file is the comms line configuration file for asyncScheduler, which provides line configuration settings on initial startup. It has the following format:

```
0   |test1    |1     |/dev/ttyn1a|9600|8       |NONE  |1       |
0   |test2    |1     |/dev/ttyn1b|9600|8       |NONE  |1       |
# 0  |fema     |1     |/dev/ttyn1b|9600|7       |EVEN  |1       |
# 3  |ttyn1c   |1     |/dev/ttyn1c|9600|8       |NONE  |1       |NONE
# 4  |ttyn1d   |1     |/dev/ttyn1d|9600|8       |NONE  |1       |SW
# 5  |ttyn1e   |1     |/dev/ttyn1e|1200|8       |NONE  |1       |HW
# 6  |ttyn1f   |1     |/dev/ttyn1f|9600|8       |NONE  |1       |
# 7  |ttyn1g   |1     |/dev/ttyn1g|9600|8       |NONE  |1       |NONE
# 8  |        |1     |       |     |       |       |       |RESERVED FOR NWW S
```

# . first line ignored for backwards compatibility
# . delimit fields with "|
# . spaces and tabs are ignored

- A child process is spawned for each line entry.
- **Line Name:** The string of characters in the second field.
- **Status:** Legal values are "0" (not activated) and "1" (activated).
- **Device Name:** Use the full path. or the 8-port board from Digi, it will look something like /dev/ttyn1a
- **Baud:** Device speed, in bps. Use only well known values (2400, 4800, 9600, etc.).
- **Data Bits:** Character size. It must be 5, 6, 7, or 8.
- **Parity:** Must be "NONE", "ODD", or "EVEN"
- **Stop Bits:** 1 or 2
- **Flow Control:** Optional. Use "SW", "HW", or "NONE" (the default).
  
  SW = xon/xoff  
  HW = rts/cts

This file provides a data field to specify a device file associated with a port. Great care must be taken to ensure that the device file specified for each line is correct; otherwise, the results are undefined.
• **aps_pil.tbl.** This file is the asyncScheduler’s routing table. The asyncScheduler routes products in two directions:
  - From the text database to one or more async mux ports.
  - From one or more async mux ports to the text database. Additionally, products received in this way (i.e., from **commsLine** clients via the mux) are routed to the NCF, and may also be routed to either or both CRS and NWWS. CRS/NWWS routing is controlled by the 4th field, “Routing.”

Example:

<table>
<thead>
<tr>
<th>PID</th>
<th>Line #</th>
<th>Priority</th>
<th>Routing</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAABBBCCC</td>
<td>0</td>
<td>2</td>
<td>CRS, NWWS</td>
</tr>
<tr>
<td>AAABABCCC</td>
<td>0</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** ID, Line #, and Priority fields are required. The Routing field is optional.

- Line # should have a corresponding entry in **aps_line.tbl.** This is enforced at runtime.
- Priority can be 1 (highest), 2, or 3 (lowest)
- Legal Routing field entries are "CRS" and "NWWS"
  - When product AAABBBCCC enters the text database, it will be retrieved by the asyncScheduler and sent to line 0 (defined in **aps_line.tbl** with priority 2.
  - When product AAABABCCC enters the text database, it will be retrieved by the asyncScheduler and sent to line 0 with priority 3.
  - When product AAABBBCCC is received from the **commsLine** client connected to line 0, it is written to the text database, sent to the NCF, and then sent to CRS and NWWS.
  - When product AAABABCCC is received from the **commsLine** client connected to line 0, it is written to the text database and sent to the NCF. It is not sent to CRS or to NWWS because the fourth field is blank.

• **aps_route.tbl.** This file lists valid APS routing keywords (e.g., NWWS, CRS).

### 10.2 Processes

Exhibit 10.2-1 describes the Asynchronous Product Scheduler data flow.
Chapter 10

Asynchronous Product Scheduler

Config Files:

$FXA_DATA/workFile/asyncProdScheduler/aps_line.tbl
$FXA_DATA/workFiles/asyncProdScheduler/aps_pil.tbl
$FXA_DATA/workFiles/asyncProdScheduler/aps_route.tbl

Log Files:

$LOG_DIR/<yyyymmdd>/asyncScheduler<logid>

Data Files:

$FXA_DATA/workFiles/asyncProdScheduler/aps_pil.tbl
$FXA_DATA/workFiles/asyncProdScheduler/aps_route.tbl

Databases:

AWIPS Text Databases (dx1f)

Asynchronous Line (up to 8 lines)

NWWS

External Comms Port

CRS

PX1

Exhibit 10.2-1. Acquire and Ingest of Asynchronous Products
AsyncScheduler is the parent process, responsible for transferring products from the text database to commsLine clients, and from commsLine clients to the text database, the NCF, and possibly CRS and/or NWWS (depending on the product entry’s “route” field in \texttt{aps\_pil.tbl}). At startup, it subscribes user-defined products in \texttt{aps\_pil.tbl} to the subscription table in the metadata database during system startup. In the meantime, it spawns one persistent port-controlling child process for each entry in \texttt{aps\_line.tbl}. It then waits for input notification from either the asyncPilMsgServer or any one of up to eight child processes (one per \texttt{aps\_line.tbl} entry). Each notification is checked against the list of PILs in \texttt{aps\_pil.tbl} (\texttt{aps\_pil.tbl} entries are sorted in a Standard Template Library (STL) map, keyed by PIL for quick searches). With each match, a command line \texttt{textdb} call to EDEX textdb server is made for the product, which is then appropriately formatted and sent to the child commsLine-controlling process of each matched line.

When a product matches a user-subscribed product, the EDEX trigger will call asyncPilTrigger.sh to copy the incoming product PIL into \texttt{$(FXA\_DATA)/workFiles/asyncProdScheduler/incoming} directory.

When the \texttt{asyncPollData.pl} detects a product’s PIL has arrived at \texttt{$(FXA\_DATA)/workFiles/asyncProdScheduler/incoming}, it invokes the asyncPilNotify process to send the PIL to asyncPilMsgServer.

The asyncPilMsgServer sends the PIL to APSPilMsgClient, which then sends the PIL to asyncScheduler.

AsyncScheduler uses the 'textdb -r' command line to read the product from the fxatext database and routes the data to the user via AsyncScheduler's CommsLines/ports.

When a user sends a text product, via AsyncScheduler's CommsLines/ports, from an application to the AsyncScheduler for storage to the text database, AsyncScheduler writes the text products into a temporary filename in PIL\_timeStamp format to \texttt{$(EDEX\_FXA\_HOME)/workFiles/asyncProdScheduler} directory and calls 'textdb -w' command line to store it into the fxatext database. AsyncScheduler removes the temporary file after successful storage to the fxatext database.

The asyncLine is the commsLine controlling process, spawned by asyncScheduler. There will be one process running for each entry in \texttt{aps\_line.tbl}. The asyncLine is responsible for passing products from asyncScheduler to the comms port, and from the comms port to asyncScheduler.

\textbf{Note}: Even though this executable’s name is asyncLine, it is spawned by asyncScheduler with its \texttt{argv[0]} set to the second (“Line Name”) field of its entry in \texttt{aps\_line.tbl}. For example, if \texttt{aps\_line.tbl} contained the single entry

\begin{verbatim}
<table>
<thead>
<tr>
<th>#</th>
<th>Line Name</th>
<th>Status</th>
<th>Device Name</th>
<th>Baud</th>
<th>DataBits</th>
<th>Parity</th>
<th>StopBits</th>
<th>FlowControl</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>bubble</td>
<td>1</td>
<td>/dev/ttyn1a</td>
<td>1200</td>
<td>8</td>
<td>NONE</td>
<td>1</td>
<td>HI</td>
</tr>
</tbody>
</table>
\end{verbatim}

Then the asyncScheduler would spawn one child asyncLine process and name it “bubble.” The child would log to the file \texttt{bubble<pid><hostname><hhmmss>}. 
Products from **commsLine** clients (i.e., products received from the serial port) are truncated precisely on the “ZCZC”/“NNN” boundary before being sent to the parent (the parent only sees the complete products from **asyncLine**). Products received from the parent are put into a priority queue, as specified by that product’s priority setting in **aps_pil.tbl**. Products in the queue are then written out to the port as the port becomes available, highest priority first. Each line currently maintains 3 priority queues, “1”–“3”, “1” being the highest priority. An **aps_pil.tbl** entry with an out-of-range value set to “0” will be assigned the highest priority. All other out-of-range values will be reassigned the lowest priority, “3.” Each line enforces a maximum queue size, currently set to 1,000 products. If a product is received from the parent (**asyncScheduler**) when there are already 1,000 products queued, it will be dropped. No special attempt is made to re-order the queue according to priority (i.e., a newly received Priority One product does not bump a Priority Three product from a full queue – the Priority One product is dropped).

The **asyncExec** spawns **distributeProduct** for products received from **commsLine** clients. It is not persistent (i.e., it returns with **distributeProduct**’s return value).

### 10.3 Additional Notes

- AWIPS II AsyncScheduler still assumes the 'textdb' executable is placed under the $FXA_HOME/bin directory (it is wrapped to CLI/bin directory).
- The AsyncScheduler trigger log file is placed in the $EDEX_HOME/logs/apsPilTrigger.log on the primary EDEX server.
- The AsyncScheduler trigger script, asyncPilTrigger.sh, is placed in the $EDEX_HOME/bin.
- The current AsyncScheduler subscription command is implemented as: `textdb -ldad -a <pil> $EDEX_HOME/bin/asyncPilTrigger.sh`.
- Legacy logging for the asyncScheduler still exists; therefore, the logs for the process itself will go to /data/logs/fxa/YYYYMMDD on px1.
- The legacy startIngest.px1 changes the modifications of the device files for the dgnc. Because this is not run under AWIPS II at this time, ensure that the /dev/ttyn* are 666 modifications to allow proper functionality.
Chapter 11
Crons and Purging
Chapter 11. Crons and Purging

Table of Contents

11.1 Crons ........................................................................................................................................ 1
  11.1.1 Site Local Crons .................................................................................................................. 1
    11.1.1.1 How to Modify Site Standard Crons ........................................................................... 1
    11.1.1.2 How to Modify Site HA Crons ................................................................................. 2
  11.1.2 Workstation Crons ............................................................................................................. 4
  11.1.3 DX Crons ........................................................................................................................... 7
    11.1.3.1 WFO DX Standard Crons ......................................................................................... 9
    11.1.3.2 WFO dx1f HA Crons .................................................................................................. 11
    11.1.3.3 WFO dx2f HA Crons .................................................................................................. 13
  11.1.4 PX Crons ........................................................................................................................... 13
    11.1.4.1 WFO PX Standard Crons .......................................................................................... 13
    11.1.4.2 WFO px1f HA Crons .................................................................................................. 14
    11.1.4.3 WFO px2f HA Crons .................................................................................................. 16
  11.1.5 WFO LS Crons ................................................................................................................. 17
    11.1.5.1 WFO LS Standard Crons ......................................................................................... 17
    11.1.5.2 WFO LS HA Crons .................................................................................................. 17
  11.1.6 WFO CSPBN Crons .......................................................................................................... 18
  11.1.7 WFO AX Crons ................................................................................................................. 18
  11.1.8 Wes2Bridge Archiver Crons .............................................................................................. 19
  11.2 quartz .................................................................................................................................... 20
  11.3 AWIPS II Data Purging ........................................................................................................ 22
    11.3.1 Purging Processed Data Storage ...................................................................................... 22
    11.3.2 Purging Raw Data Storage ............................................................................................ 24
  11.4 EDEX Log File Purging ........................................................................................................ 24
  11.5 Locating a Script .................................................................................................................. 25
  11.6 Hydro Cron Control Application Tokens ............................................................................ 26
    11.6.1 Configuration .................................................................................................................. 26
    11.6.2 Site-Specific Configurations .......................................................................................... 27
    11.6.3 Logging ......................................................................................................................... 29

List of Tables

Table 11.1.1-1. Site Standard crons ................................................................................................ 2
Table 11.1.1-2. Site HA crons ................................................................. 2
Table 11.1.1.1-1. Modifying Site Standard crons ................................. 3
Table 11.1.2-1. WFO Workstation crons ................................................ 8
Table 11.1.3.1-1. WFO DX Standard crons ........................................... 10
Table 11.1.3.2-1. WFO dx1f HA crons .................................................... 11
Table 11.1.3.3-1. WFO dx2f HA crons .................................................... 13
Table 11.1.4.1-1. WFO PX Standard crons ............................................ 13
Table 11.1.4.2-1. WFO px1f HA crons .................................................... 15
Table 11.1.4.3-1. WFO px2f HA crons .................................................... 17
Table 11.1.5.1-1. WFO LS Standard Crons ............................................ 17
Table 11.1.5.2-1. WFO LS HA Crons ..................................................... 18
Table 11.1.6-1. WFO CPSBN crons ......................................................... 18
Table 11.1.7-1. WFO AX Crons ............................................................. 19
Table 11.2-1. Crons Jobs and Associated Schedules for AWIPS II ........... 21
Table 11.5-1. Cron Locations ............................................................ 25
11.1 Crons

Each hardware device (server, processor, and workstation) contains baseline crons that perform system checks, purges, and backups. Sites are also provided cron files for local use.

AWIPS crons are maintained and executed in one of 3 ways:

- **Standard Linux cron.** These crons are updated and executed according to `man 8 cron`, `man 1 crontab`, and `man 5 crontab`. Both workstations and servers have standard crons running.

- **Linux High Availability (HA) crons.** These crons are part of the failover heartbeat and are maintained in the `/etc/ha.d/cron.d/` directories on the px1/px2, dx1/dx2 and ls2/ls3 HA clusters. Editing the copied version in `/etc/cron.d/` without copying the changes to `/etc/ha.d/cron.d/` will result in the loss of your changes the next time the cron is activated from its maintained location.

- **Quartz.** These time-based jobs are compiled into EDEX running on dx3/dx4. Quartz jobs can execute outside of cron or they can be loaded into cron with a java call.

**Note:** AWIPS II makes minimal use of Linux cron jobs for data processing. Instead, operations such as data purging, SmartInit, and other periodic data processing operations are triggered using quartz timer events. The quartz timer is a time-based event generator that is part of the Camel SOA infrastructure underlying the EDEX processes. The quartz timer mechanism allows EDEX services to run at specified times. At the present time, the EDEX quartz timer events are configured internally to EDEX and are not configurable in the field.

AWIPS II uses heartbeat-managed cron jobs on the PX1/2 cluster and the DX1/2 cluster. The crons are staged in `/etc/ha.d/cron.d` and are installed into `/etc/cron.d` when activated. The file names of the AWIPS II crons begin with “a2” – `a2px1cron`, `a2px2cron`, `a2dx1cron`, and `a2dx2cron`.

11.1.1 Site Local Crons

Sites can configure local crons in two ways:

- Standard crons on a workstation or sever
- HA crons on a failover pair

Sites should be careful to avoid conflicts between local crons and existing baseline crons. Sites should not modify baseline cron files as this may have a negative impact on system performance, and baseline files may be overwritten with the next software upgrade.

Table 11.1.1-1 identifies site standard crons; Table 11.1.1-2 identifies HA crons.
Table 11.1.1-1. Site Standard crons

<table>
<thead>
<tr>
<th>Directory</th>
</tr>
</thead>
<tbody>
<tr>
<td>/var/spool/cron/&lt;username&gt;</td>
</tr>
<tr>
<td>/etc/cron.d/</td>
</tr>
<tr>
<td>/etc/cron.hourly/</td>
</tr>
<tr>
<td>/etc/cron.daily/</td>
</tr>
<tr>
<td>/etc/cron.weekly/</td>
</tr>
<tr>
<td>/etc/cron.monthly/</td>
</tr>
</tbody>
</table>

Table 11.1.1-2. Site HA crons

<table>
<thead>
<tr>
<th>Server</th>
<th>Site cron File</th>
<th>Directory</th>
</tr>
</thead>
<tbody>
<tr>
<td>dx1f</td>
<td>a2SITEdx1cron</td>
<td>/etc/ha.d/cron.d (and /etc.cron.d on the active server)</td>
</tr>
<tr>
<td>dx2f</td>
<td>a2SITEdx2cron</td>
<td>/etc/ha.d/cron.d (and /etc.cron.d on the active server)</td>
</tr>
<tr>
<td>px1f</td>
<td>a2SITEpx1cron</td>
<td>/etc/ha.d/cron.d (and /etc.cron.d on the active server)</td>
</tr>
<tr>
<td>px2f</td>
<td>a2SITEpx2cron</td>
<td>/etc/ha.d/cron.d (and /etc.cron.d on the active server)</td>
</tr>
<tr>
<td>ls</td>
<td>SITEls cron</td>
<td>/etc/ha.d/cron.d (and /etc.cron.d on the active server)</td>
</tr>
</tbody>
</table>

Sites that configure HA crons must keep the crons in sync across servers and between the /etc/cron.d and /etc/ha.d/cron.d directories on the active server.

NOTE: ALL cron files in the /etc/cron.d directory are active. Do not place ANY backup files (e.g., SITEpx1cron.bak) in the /etc/cron.d directory because crond does not discriminate and will run every job in every file in the directory. Save them in a directory reserved for SITE use that will not be overwritten by a software upgrade (e.g., /awips/dev/SITE_crons, /home/SAlocal/SITE_crons).

NOTE: Standard BOIVerify crons have been delivered as part of the local site a2SITEpx2cron. Sites should modify the existing a2SITEpx2cron for their BOIVerify and other related local app needs.

11.1.1.1 How to Modify Site Standard Crons

Sites can create cron files in directories managed by crond. These crons will be available when the machine is up; they are not part of the HA failover crons.

The cron daemon checks the /etc/crontab file, the /etc/cron.d/ directory, and the /var/spool/cron/ directory every minute for changes. Any changes found are loaded into memory. crontab does not need to be restarted. See Table 11.1.1.1-1.
Table 11.1.1-1. Modifying Site Standard crons

<table>
<thead>
<tr>
<th>Directory</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>/var/spool/cron/&lt;username&gt;</td>
<td>Cron files here have names that match usernames in /etc/passwd. These</td>
</tr>
<tr>
<td></td>
<td>files are executed using the permissions of the username. Users can</td>
</tr>
<tr>
<td></td>
<td>create files here as their account name using crontab -e, if permission</td>
</tr>
<tr>
<td></td>
<td>is granted in /etc/cron.allow and/or /etc/cron.deny. Entries are</td>
</tr>
<tr>
<td></td>
<td>similar to /etc/crontab:</td>
</tr>
<tr>
<td></td>
<td>minute hour day  month  dayofweek  command</td>
</tr>
<tr>
<td></td>
<td>field  allowed values</td>
</tr>
<tr>
<td></td>
<td>-----  --------------</td>
</tr>
<tr>
<td></td>
<td>Minute  0-59</td>
</tr>
<tr>
<td></td>
<td>hour    0-23</td>
</tr>
<tr>
<td></td>
<td>day of month    1-31</td>
</tr>
<tr>
<td></td>
<td>month   1-12</td>
</tr>
<tr>
<td></td>
<td>day of week  0-7 (0 or 7 is Sun)</td>
</tr>
<tr>
<td>/etc/cron.d/</td>
<td>The files in these directories are in the same format as /etc/crontab.</td>
</tr>
<tr>
<td>/etc/cron.hourly/</td>
<td>The files in these directories are in the same format as /etc/crontab.</td>
</tr>
<tr>
<td>inn-cron-ntpsend</td>
<td></td>
</tr>
<tr>
<td>inn-cron-rnews</td>
<td></td>
</tr>
<tr>
<td>/etc/cron.daily/</td>
<td></td>
</tr>
<tr>
<td>00webalizer</td>
<td></td>
</tr>
<tr>
<td>certwatch</td>
<td></td>
</tr>
<tr>
<td>logrotate</td>
<td></td>
</tr>
<tr>
<td>prelink</td>
<td></td>
</tr>
<tr>
<td>tmpwatch</td>
<td></td>
</tr>
<tr>
<td>0anacron</td>
<td></td>
</tr>
<tr>
<td>cups</td>
<td></td>
</tr>
<tr>
<td>makewhatis.cron rpm</td>
<td></td>
</tr>
<tr>
<td>0logwatch</td>
<td></td>
</tr>
<tr>
<td>inn-cron-expire</td>
<td></td>
</tr>
<tr>
<td>mlocate.cron</td>
<td></td>
</tr>
<tr>
<td>tetex.cron</td>
<td></td>
</tr>
<tr>
<td>/etc/cron.weekly/</td>
<td></td>
</tr>
<tr>
<td>0anacron</td>
<td></td>
</tr>
<tr>
<td>99-raid-check</td>
<td></td>
</tr>
<tr>
<td>makewhatis.cron</td>
<td></td>
</tr>
<tr>
<td>/etc/cron.monthly/</td>
<td></td>
</tr>
<tr>
<td>0anacron</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows a user-created cron in /var/spool/cron that runs with the permissions of the user. Note: Output files are created with permissions of the user, this is a consideration when using these type of crons operationally.

```
lx4-ntcd{rschup}118: crontab -e
  * * * * * echo "test" > ~rschup/test_cron
  :wq
  "/tmp/crontab.XXXXQ1St1T" 1L, 42C written
  crontab: installing new crontab
```
11.1.1.2 How to Modify Site HA Crons

Modification and activation of site HA crons requires the following steps:

1. Login to the active server for the HA package
2. Modify /etc/ha.d/cron.d/a2SITE<host>cron with local changes
3. Copy /etc/ha.d/cron.d/a2SITE<host>cron to the failover pair
4. Copy /etc/ha.d/cron.d/a2SITE<host>cron to the active server /etc/cron.d/

**NOTE:** Do not leave backup files in /etc/ha.d/cron.d. They could affect the swap process. Ensure that /etc/ha.d/cron.d contains only the files expected to be there.

**Modifying DX a2SITE Crons**

In the following example, the hb_stat command shows the dx1f package is running on dx1 and the HA cronfiles are a2dx1cron, a2SITEdx1cron. The local site a2SITEdx1cron will be copied from the site created backup directory /home/SAlocal/d.CRONS/ into the /etc/ha.d/cron.d/ directory on each failover pair and into /etc/cron.d/ on the active dx1f server. The crond daemon checks /etc/cron.d/ for changes every minute, so the running crond does not have to be restarted. The output from the cron is checked in /awips/dev/, then the test is removed from the system. Modifying the dx2f cron a2SITEdx2cron follows the same procedure.
root@dx1f's password:

[root@dx1-ntcb ~]# bb_stat
Heartbeat Status Monitor Mar 29 16:44:03

= Member Status =

<table>
<thead>
<tr>
<th>Member</th>
<th>Status</th>
<th>IP address</th>
</tr>
</thead>
<tbody>
<tr>
<td>dx1-ntcb</td>
<td>Up</td>
<td>165.92.107.131</td>
</tr>
<tr>
<td>dx2-ntcb</td>
<td>Up</td>
<td>165.92.107.132</td>
</tr>
</tbody>
</table>

= Service Status =

<table>
<thead>
<tr>
<th>Service</th>
<th>IPaddr</th>
<th>Cronfile</th>
<th>Owner</th>
<th>Start Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>a2dx1apps</td>
<td>165.92.107.195</td>
<td>a2dx1cron,a2SIT dx1-ntcb</td>
<td>dx1-ntcb</td>
<td>2011-03-23 21:32:03</td>
</tr>
<tr>
<td>a2dx2apps</td>
<td>165.92.107.196</td>
<td>a2dx2cron,a2SIT dx2-ntcb</td>
<td>dx2-ntcb</td>
<td>2011-03-14 16:11:12</td>
</tr>
</tbody>
</table>

[root@dx1-ntcb cron.d]# cd /etc/ha.d/cron.d
[root@dx1-ntcb cron.d]# ls -l a2SITE*
-rw-r--r-- 1 root root 0 Mar 14 19:14 a2SITEdx1cron
-rw-r--r-- 1 root root 0 Mar 14 16:01 a2SITEdx2cron

[root@dx1-ntcb cron.d]# cat /home/SAlocal/d.CRONS/a2SITEdx1cron
* * * * * fxa echo "test" >/awips/dev/test_cron

[root@dx1-ntcb cron.d]# cp /home/SAlocal/d.CRONS/a2SITEdx1cron .
cp: overwrite "/a2SITEdx1cron"? y

[root@dx1-ntcb cron.d]# ls -l a2SITE*
-rw-r--r-- 1 root root 50 Mar 29 16:55 a2SITEdx1cron
-rw-r--r-- 1 root root 0 Mar 14 16:01 a2SITEdx2cron

[root@dx1-ntcb cron.d]# scp -p a2SITEdx1cron dx2:/etc/ha.d/cron.d/
a2SITEdx1cron

100% 50 0.1KB/s 00:00

[root@dx1-ntcb cron.d]# cp a2SITEdx1cron /etc/cron.d
cp: overwrite "/etc/cron.d/a2SITEdx1cron"? y

[root@dx1-ntcb cron.d]# ls -l /awips/dev/test_cron
-rw-r--r-- 1 fxa fxalpha 5 Mar 29 18:33 /awips/dev/test_cron

[root@dx1-ntcb cron.d]# cat /dev/null > a2SITEdx1cron

[root@dx1-ntcb cron.d]# ls -l a2SITEdx1cron
-rw-r--r-- 1 root root 0 Mar 29 18:36 a2SITEdx1cron

[root@dx1-ntcb cron.d]# scp a2SITEdx1cron dx2:/etc/ha.d/cron.d/
a2SITEdx1cron 100% 0 0.0KB/s 00:00

[root@dx1-ntcb cron.d]# cp a2SITEdx1cron /etc/cron.d/
cp: overwrite `/etc/cron.d/a2SITEdx1cron'? Y

[root@dx1-ntcb cron.d]# rm /awips/dev/test_cron
rm: remove regular file `/awips/dev/test_cron'? y

Modifying PX a2SITE Crons

The following example shows the normal px1f/px2f configuration. Site crons are synchronized across px1/px2 and /etc/ha.d/cron.d. Modifying px1f/px2f a2SITE crons follows the same procedure as described for dx1f/dx2f

[root@px1-ntcb ~]# hb_stat
Heartbeat Status Monitor Mar 14 15:35:21

======================================== Member Status ==============================

Member        Status     IP address
--------------     ----------   ---------------
px1-ntcb       Up         165.92.107.137
px2-ntcb       Up         165.92.107.138

======================================== Service Status ==============================

Service             IPaddr                 Cronfile               Owner       Start Time
--------------        ---------------          ---------------           ----------      -------------------
a2px1apps      165.92.107.193  a2px1cron,a2SITE px1-ntcb   2011-02-22 19:44:12
a2px2apps      165.92.107.194  a2px2cron,a2SITE px2-ntcb   2011-03-01 21:38:50

[root@px1-ntcb ~]# ls -l /etc/cron.d/a2SITE*
-rw-r--r-- 1 root root 256 Feb 22 19:44 /etc/cron.d/a2SITEpx1cron

[root@px1-ntcb ~]# ls -l /etc/ha.d/cron.d/a2SITE*
-rw-r--r-- 1 root root 256 Feb 22 19:44 /etc/ha.d/cron.d/a2SITEpx1cron
-rw-r--r-- 1 root root 256 Feb 22 19:44 /etc/ha.d/cron.d/a2SITEpx2cron

[root@px1-ntcb ~]# ssh px2 "ls -l /etc/cron.d/a2SITE*"
-rw-r--r-- 1 root root 256 Mar  1 21:38 /etc/cron.d/a2SITEpx2cron

[root@dx1-ntcb cron.d]# ls -ld /etc/cron.d
drwxr-xr-x 2 root root 4096 Mar 29 17:05 /etc/cron.d

[root@dx1-ntcb cron.d]# touch /etc/cron.d

[root@dx1-ntcb cron.d]# ls -ld /etc/cron.d
drwxr-xr-x 2 root root 4096 Mar 29 17:05 /etc/cron.d

[root@px1-ntcb ~]# ssh px2 "ls -l /etc/ha.d/cron.d/a2SITE*"
-rw-r--r-- 1 root root 256 Feb 22 19:44 /etc/ha.d/cron.d/a2SITEpx1cron
-rw-r--r-- 1 root root 256 Feb 22 19:44 /etc/ha.d/cron.d/a2SITEpx2cron
Modifying LS SITEls Crons

The following example shows the normal LS configuration. Because only one LS is active at any time, there is only one site cron, “SITElscron.” The cron is synchronized between ls2/ls3 /etc/ha.d/cron.d/ and copied to /etc/cron.d on the active server.

[root@ls2-ntcb ~]# hb_stat

Heartbeat Status Monitor Mar 14 15:49:33

Member Status IP address
ls2-ntcb Up 192.168.1.11
ls3-ntcb Up 192.168.1.12

Resource Status

Resource Parameters Owner Status Start Time
IPaddr2 192.168.1.10/24/eth0 ls2-ntcb Running OK 2011-01-14 19:32:05
LS1IPSRCaddr ls2-ntcb OK 2011-01-14 19:32:05
apache ls2-ntcb Running OK 2011-01-14 19:32:07
csportd ls2-ntcb CSPORTD 2011-01-14 19:32:17
Xinetd wu_ftpd ls2-ntcb Running OK 2011-01-14 19:32:18
hylafax ls2-ntcb OK 2011-01-14 19:32:18
ldad ls2-ntcb Starting 2011-01-14 19:32:19
dserver ls2-ntcb OK 2011-01-14 19:32:51
LDM ls2-ntcb OK 2011-01-14 19:33:08
Crontab SITElscron ls2-ntcb OK 2011-01-14 19:33:18
Crontab lscron ls2-ntcb OK 2011-01-14 19:33:19
SITEprocesses ls2-ntcb OK 2011-01-14 19:33:20

HylaFAX Details:
HylaFAX client-server protocol server: hfaxd (pid 3620) is running...
HylaFAX queue manager process: faxq (pid 3617) is running...

[root@ls2-ntcb ~]# ls -l /etc/ha.d/cron.d/SITE*
-rw------- 1 root root 330 Nov  9  2007 /etc/ha.d/cron.d/SITElscron

[root@ls2-ntcb ~]# ls -l /etc/cron.d/SITE*
-rw------- 1 root root 330 Nov  9  2007 /etc/cron.d/SITElscron

[root@ls2-ntcb ~]# ssh ls3 "ls -l /etc/ha.d/cron.d/SITE*"
-rw------- 1 root root 330 Nov  9  2007 /etc/ha.d/cron.d/SITElscron
-rwxr-xr-x 1 root root 330 Nov  9  2007 /etc/ha.d/cron.d/SITElscron.200801221903

11.1.2 Workstation Crons

AWIPS LX and XT workstations run standard linux crons from /var/spool/cron/ and /etc/. See man 8 cron, man 1 crontab, and man 5 crontab for more information. Crons in /etc/daily, /etc/hourly, /etc/weekly, and /etc/monthly are identical on all
AWIPS Linux workstations and servers. The /var/spool/cron/root crontab is identical across all AWIPS workstations and servers. See Table 11.1.2-1.

### Table 11.1.2-1. WFO Workstation crons

<table>
<thead>
<tr>
<th>Cron File</th>
<th>Cron Entry</th>
<th>What It Does</th>
<th>Frequency</th>
<th>Cron Time (min, hour, day of month, month, day of week)</th>
</tr>
</thead>
<tbody>
<tr>
<td>/var/spool/cron/fxa</td>
<td>startCtrlCpu.sh</td>
<td>Starts CPU monitor if it is not running</td>
<td>Every 10 minutes</td>
<td>0,10,20,30,40,50 **</td>
</tr>
<tr>
<td></td>
<td>diskUsage.pl</td>
<td>Creates a data page on disk usage for the FX Data Monitor</td>
<td>Every 10 minutes</td>
<td>0,10,20,30,40,50 **</td>
</tr>
<tr>
<td></td>
<td>cd ${LOG_DIR}/display;</td>
<td>Cleans out any display logs that are more than one day old twice a day (at</td>
<td>2 times/day</td>
<td>47 0,12 * * *</td>
</tr>
<tr>
<td></td>
<td>find . -mtime +1</td>
<td>0047Z and 1247Z).</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>breakLogDisplay</td>
<td>Breaks display logs daily</td>
<td>1 time/day</td>
<td>0 0 * * *</td>
</tr>
<tr>
<td></td>
<td>breakLogCtrlCpu</td>
<td>Breaks ctrlCpu log daily</td>
<td>1 time/day</td>
<td>0 0 * * *</td>
</tr>
<tr>
<td>/var/spool/cron/ifps</td>
<td>clean_up.sh</td>
<td>Removes old IFPS logfiles and diagnostic output from programs</td>
<td>1 time/day</td>
<td>16 00 * * *</td>
</tr>
<tr>
<td></td>
<td>logAndRemoveCoreFile.k</td>
<td>Removes core files</td>
<td>1 time/day</td>
<td>0 10 * * *</td>
</tr>
<tr>
<td></td>
<td>/usr/bin/find /tmp ('{}\ -path /tmp/lost+found/**) -xdev (-mtime +30 -o -ctime +30 ) -exec rm {} ;</td>
<td>Does cleanup – first day of the month</td>
<td>1 time/month</td>
<td>01 5 1 * *</td>
</tr>
<tr>
<td></td>
<td>/usr/bin/find /var/tmp -xdev -type f(-mtime +30 -o -ctime +30 ) -exec rm {} ;</td>
<td>Does cleanup – first day of the month</td>
<td>1 time/month</td>
<td>02 5 1 * *</td>
</tr>
<tr>
<td></td>
<td>/usr/bin/find /var/tmp -xdev -type f -group fxalpha -mtime +1</td>
<td>Cleans up /var/tmp</td>
<td>1 time/day</td>
<td>20 7 * * *</td>
</tr>
<tr>
<td></td>
<td>rm -f</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>/usr/bin/find /tmp -xdev -type f -name &quot;FXA*.bundles&quot; -mtime +1</td>
<td>Cleans up /tmp FXA.* bundles</td>
<td>1 time/day</td>
<td>10 7 * * *</td>
</tr>
<tr>
<td></td>
<td>xargs rm -f</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>/usr/bin/find /tmp -xdev -type f -name &quot;*.PNG&quot; -mtime +1</td>
<td>Cleans up /tmp *.PNG files</td>
<td>1 time/day</td>
<td>15 7 * * *</td>
</tr>
<tr>
<td></td>
<td>xargs rm -f</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>boxktp</td>
<td>Gathers performance and system metrics</td>
<td>1 time/day</td>
<td>59 23 * * *</td>
</tr>
<tr>
<td></td>
<td>boxstats_cleanlogs</td>
<td>Cleans up data logs created from boxstats performance gathering scripts</td>
<td>1 time/day</td>
<td>00 0 * * *</td>
</tr>
<tr>
<td></td>
<td>/var/spool/cron/root</td>
<td>boxallstats</td>
<td>1 time/day</td>
<td>01 0 * * *</td>
</tr>
<tr>
<td></td>
<td>reset-faillog.sh</td>
<td>--</td>
<td>Every minute</td>
<td>* * * * *</td>
</tr>
</tbody>
</table>
### Cron File

<table>
<thead>
<tr>
<th>Cron Entry</th>
<th>What It Does</th>
<th>Frequency (Z time)</th>
<th>Cron Time (min, hour, day of month, month, day of week)</th>
</tr>
</thead>
<tbody>
<tr>
<td>/etc/cron.d/sysstat root /usr/lib/sa1 1</td>
<td>Runs system activity accounting tool</td>
<td>every 10 minutes</td>
<td>*/10 * * * *</td>
</tr>
<tr>
<td>/usr/lib/sa2 -A</td>
<td>Provides daily summary of process accounting</td>
<td>1 time/day 23:53</td>
<td>53 23 * * *</td>
</tr>
<tr>
<td>*/10 * * * *</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/etc/cron.daily</td>
<td>Parses system logs and creates output in /var/log</td>
<td>1 time/day 04:20</td>
<td>02 4 * * *</td>
</tr>
<tr>
<td>logrotate</td>
<td>Rotates log files</td>
<td>1 time/day 04:20</td>
<td>02 4 * * *</td>
</tr>
<tr>
<td>/var/log/wtmp /var/log/btmp</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>makewhatis.cron</td>
<td>Reads all man pages and updates the whatis database</td>
<td>1 time/day 04:20</td>
<td>02 4 * * *</td>
</tr>
<tr>
<td>mlocate.cron</td>
<td>Creates or updates a database of file names used by the locate command</td>
<td>1 time/day 04:20</td>
<td>02 4 * * *</td>
</tr>
<tr>
<td>prelink</td>
<td>Modifies ELF shared libraries and ELF dynamically linked binaries to reduce the time needed for the dynamic linker to perform relocations at startup</td>
<td>1 time/day 04:20</td>
<td>02 4 * * *</td>
</tr>
<tr>
<td>rpm</td>
<td>Creates list of rpm packages in /var/log/rpmpkgs</td>
<td>1 time/day 04:20</td>
<td>02 4 * * *</td>
</tr>
<tr>
<td>Tmpwatch</td>
<td>Recursively removes files that have not been accessed for a given number of hours.</td>
<td>1 time/day 04:20</td>
<td>02 4 * * *</td>
</tr>
<tr>
<td>/etc/cron.weekly</td>
<td>Reads all man pages and writes a line in the whatis database consisting of the name of the man page and a short description</td>
<td>1 time/week on Sunday at 04:22</td>
<td>22 4 * * 0</td>
</tr>
<tr>
<td>makewhatis.cron</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### 11.1.3 DX Crons

The DX servers run standard crons on all servers but only dx1 and dx2 run HA crons. The dx3 and dx4 are edex clusters (not failover clusters) that use quartz to run time based jobs compiled into edex.

#### 11.1.3.1 WFO DX Standard Crons

The DX standard crons run when the machine is up and are not part of the HA failover packages or the EDEX cluster quartz schedule. The dx1 is the NIS master and runs yppmake 2 times an hour to keep the /etc/passwd file on dx1 in synch with the NIS.
password file. This is the only entry in dx1 /var/spool/cron/root that is different from the AWIPS workstations and servers. See Table 11.1.3.1-1.

Table 11.1.3.1-1. WFO DX Standard crons

<table>
<thead>
<tr>
<th>Cron File</th>
<th>Cron Entry</th>
<th>What It Does</th>
<th>Frequency (Ztime)</th>
<th>Cron Time (min, hour, day of month, month, day of week)</th>
</tr>
</thead>
<tbody>
<tr>
<td>/var/spool/cron/root (DX1 only)</td>
<td>/var/yp/ypmake</td>
<td>Remakes NIS maps if needed and push to NIS client servers</td>
<td>2 times/day</td>
<td>08:00, 20:00</td>
</tr>
<tr>
<td></td>
<td>/bin/cp -f /var/yp/ypmake.log /var/yp/ypmake.log.old; /bin/cp -f /dev/null /var/yp/ypmake.log</td>
<td>Copies current ypmake.log to ypmake.old then zero out current ypmake.log</td>
<td>1 time/day</td>
<td>07:00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 7 * * *</td>
</tr>
<tr>
<td>/var/spool/cron/root (DX1 and DX2)</td>
<td>logAndRemoveCoreFile.ksh</td>
<td>Removes core files</td>
<td>1 time/day</td>
<td>10:00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 10 * * *</td>
</tr>
<tr>
<td></td>
<td>/usr/bin/find /tmp (! -path /tmp/lost+found/**) -xdev -type f (-mtime +30 -o -ctime +30) -exec rm {} ;</td>
<td>Does cleanup – first day of the month</td>
<td>1 time/month</td>
<td>1st day of the month 05:01</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>01 5 1 * *</td>
</tr>
<tr>
<td></td>
<td>/usr/bin/find /var/tmp -xdev -type f (-mtime +30 -o -ctime +30) -exec rm {} ;</td>
<td>Does cleanup – first day of the month</td>
<td>1 time/month</td>
<td>1st day of the month 05:02</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>02 5 1 * *</td>
</tr>
<tr>
<td></td>
<td>/usr/bin/find /var/tmp -xdev -type f -group fxalpha -mtime +1</td>
<td>Cleans up /var/tmp</td>
<td>1 time/day</td>
<td>07:20</td>
</tr>
<tr>
<td></td>
<td>-xargs rm -f</td>
<td></td>
<td></td>
<td>20 7 * * *</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>/usr/bin/find /tmp -xdev -type f -name &quot;FXA*&quot; -mtime +1</td>
<td>Cleans up /tmp FXA.* bundles</td>
<td>1 time/day</td>
<td>07:10</td>
</tr>
<tr>
<td></td>
<td>-xargs rm -f</td>
<td></td>
<td></td>
<td>10 7 * * *</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>/usr/bin/find /tmp -xdev -type f -name &quot;.PNG&quot; -mtime +1</td>
<td>Cleans up /tmp *.PNG files</td>
<td>1 time/day</td>
<td>07:15</td>
</tr>
<tr>
<td></td>
<td>-xargs rm -f</td>
<td></td>
<td></td>
<td>15 7 * * *</td>
</tr>
<tr>
<td>/var/spool/cron/root (DX1 and DX2)</td>
<td>boxktp</td>
<td>Gathers performance and system metrics</td>
<td>1 time/day</td>
<td>23:59</td>
</tr>
<tr>
<td></td>
<td>boxstats_cleanlogs</td>
<td>Cleans up data logs created from boxstats performance gathering scripts</td>
<td>1 time/day</td>
<td>00:00</td>
</tr>
<tr>
<td></td>
<td>boxallstats</td>
<td>Generates sar and top statistics</td>
<td>1 time/day</td>
<td>00:01</td>
</tr>
<tr>
<td></td>
<td>removeArchives</td>
<td>Compress and purge old logs: /awips/ops/logs -r7&gt; &gt; /awips/ops/logs/removeArchives.log 2&gt;&amp;1</td>
<td>29 mins past the hour</td>
<td>9 0 * * *</td>
</tr>
<tr>
<td>/etc/cron.d/sysstat/</td>
<td>root /usr/lib/sa/sa1 1 1</td>
<td>Runs system activity accounting tool</td>
<td>every 10 minutes</td>
<td>*/10 * * *</td>
</tr>
<tr>
<td></td>
<td>root /usr/lib/sa/sa2 -A</td>
<td>Generates daily summary of process accounting</td>
<td>1 time/day</td>
<td>23:53</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>53 23 * * *</td>
</tr>
</tbody>
</table>
### Chapter 11  Crons and Purging

<table>
<thead>
<tr>
<th>Cron File</th>
<th>Cron Entry</th>
<th>What It Does</th>
<th>Frequency (Z time)</th>
<th>Cron Time (min, hour, day of month, month, day of week)</th>
</tr>
</thead>
<tbody>
<tr>
<td>/etc/cron.daily</td>
<td>logwatch</td>
<td>Parses system logs and creates output in /var/log</td>
<td>1 time/day 04:20</td>
<td>02 4 * * *</td>
</tr>
<tr>
<td></td>
<td>logrotate</td>
<td>Rotates log files /var/log/wtmp /var/log/btmp</td>
<td>1 time/day 04:20</td>
<td>02 4 * * *</td>
</tr>
<tr>
<td></td>
<td>makewhatis.cron</td>
<td>Reads all man pages and updates the whatis database</td>
<td>1 time/day 04:20</td>
<td>02 4 * * *</td>
</tr>
<tr>
<td></td>
<td>mlocate.cron</td>
<td>Creates or updates a database of file names used by the locate command</td>
<td>1 time/day 04:20</td>
<td>02 4 * * *</td>
</tr>
<tr>
<td></td>
<td>prelink</td>
<td>Modifies ELF shared libraries and ELF dynamically linked binaries to reduce time needed for dynamic linker to perform relocations at startup</td>
<td>1 time/day 04:20</td>
<td>02 4 * * *</td>
</tr>
<tr>
<td></td>
<td>rpm</td>
<td>Creates list of rpm packages in /var/log/rpmpkgs</td>
<td>1 time/day 04:20</td>
<td>02 4 * * *</td>
</tr>
<tr>
<td></td>
<td>tmpwatch</td>
<td>Recursively removes files that have not been accessed for a given number of hours.</td>
<td>1 time/day 04:20</td>
<td>02 4 * * *</td>
</tr>
<tr>
<td>/etc/cron.weekly</td>
<td>makewhatis.cron</td>
<td>Reads all man pages and writes a line in the whatis database1 time/day 04:20 consisting of the name of the man page and a short description</td>
<td>1 time/week on Sunday at 04:22</td>
<td>22 4 * * 0</td>
</tr>
</tbody>
</table>

**11.1.3.2 WFO dx1f HA Crons**

Two cronfiles are active in /etc/cron.d/ when the dx2f a2dx2apps package is running. One cronfile – a2SITEdx2cron – is for site-specific crons; the other – a2dx2cron – is for the baseline-delivered crons. See Table 11.1.3.2-1.

**Table 11.1.3.2-1. WFO dx1f HA crons**

<table>
<thead>
<tr>
<th>Cron File</th>
<th>Cron Entry</th>
<th>What It Does</th>
<th>Frequency (Z time)</th>
<th>Cron Time (min, hour, day of month, month, day of week)</th>
</tr>
</thead>
<tbody>
<tr>
<td>a2SITEdx2cron</td>
<td>Site specific</td>
<td>Site specific</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>a2dx2cron (root entries)</td>
<td>backup_pgdb_a2 -a</td>
<td>Performs daily pg_dump backup of databases</td>
<td>1 time/day 22:50</td>
<td>05 22 * * *</td>
</tr>
<tr>
<td></td>
<td>backup_pgdb_a2 -d</td>
<td>More frequent backups of metadata</td>
<td>Every 4 hours, skipping 22z</td>
<td>10 1,5,9,13,17 * * *</td>
</tr>
<tr>
<td></td>
<td>nas_backup_daily</td>
<td>Makes full tar backups of certain files/directories</td>
<td>1 time/day 02:10</td>
<td>10 2 * * *</td>
</tr>
<tr>
<td>/usr/bin/find /home/<em>/.gnome-smproxy</em> -mtime +30 -exec rm -f {} ;</td>
<td></td>
<td>Removes user gnome files older than 30 days every Sunday</td>
<td>1 time/week Every Sunday at 02:35</td>
<td>35 2 * * 0</td>
</tr>
<tr>
<td>Cron File</td>
<td>Cron Entry</td>
<td>What It Does</td>
<td>Frequency (Z time)</td>
<td>Cron Time (min, hour, day of month, month, day of week)</td>
</tr>
<tr>
<td>-----------</td>
<td>------------</td>
<td>--------------</td>
<td>--------------------</td>
<td>------------------------------------------------------</td>
</tr>
<tr>
<td>/usr/bin/find home/*/ .sawfish/sessions -type f -mtime +30 -exec rm -f {} \;</td>
<td>Removes user sawfish files older than 30 days every Sunday</td>
<td>1 time/week Every Sunday at 02:40</td>
<td>40 2 * * 0</td>
<td></td>
</tr>
<tr>
<td>msg_cleanup</td>
<td>Performs MHS Purge/Cleanup</td>
<td>1 time/day 07:20</td>
<td>2 7 * * *</td>
<td></td>
</tr>
<tr>
<td>arch_cleanup_mhs</td>
<td>Cleans up archive MHS files</td>
<td>Every 30 minutes</td>
<td>30 * * * *</td>
<td></td>
</tr>
<tr>
<td>purge_old_files</td>
<td>Purges old files from /var/spool/mqueue</td>
<td>Every 30 minutes</td>
<td>30 * * * *</td>
<td></td>
</tr>
<tr>
<td>/awips/wdv/wdv --mode=control</td>
<td>Warning Dissemination Verification system monitors warning products disseminated from a site, and cross-references what is received from the SBN downlink within a reasonable period of time to ensure the product was uplinked.</td>
<td>Every 5 minutes</td>
<td>00,05,10,15,20,25,30,35,40,45,50,55 * * *</td>
<td></td>
</tr>
<tr>
<td>(fxa entries)</td>
<td>NWWSKeepAliveMsg</td>
<td>Tests uplink status</td>
<td>4 times/hour :13, :28, :43, :58</td>
<td>13,28,43,58 * * *</td>
</tr>
<tr>
<td>startScour</td>
<td>Scours daily to clean up log files and a few items not hit by master.purge</td>
<td>1 time/day 00:30</td>
<td>30 0 * * *</td>
<td></td>
</tr>
<tr>
<td>breakLogIngest</td>
<td>Breaks ingest logs daily</td>
<td>1 time/day 00:00</td>
<td>0 0 * * *</td>
<td></td>
</tr>
<tr>
<td>nwrWatchDog.sh</td>
<td>Restarts nwrTrans.pl if necessary</td>
<td>Every minute</td>
<td>* * * * *</td>
<td></td>
</tr>
<tr>
<td>NWRWAVESpurge.sh</td>
<td>Cleans DEBUG LOGS OUTPUT INPUT and TEST Directories</td>
<td>1 time/day 00:45</td>
<td>45 0 * * *</td>
<td></td>
</tr>
<tr>
<td>(fxa entries)</td>
<td>removeExpiredNWR.sh</td>
<td>Removes any expired or corrupt files from pending and sent side of NWRWAVESBrowser</td>
<td>2 times/hour :20, :40</td>
<td>20,40 * * * *</td>
</tr>
<tr>
<td>updateSUMMARY.csh</td>
<td>Updates the summary message four times an hour</td>
<td>4 times/hour :00, :15, :30, :45</td>
<td>00,15,30,45 * * * *</td>
<td></td>
</tr>
<tr>
<td>pendingDirCheck</td>
<td>Checks for old messages in the pending directory every ten minutes</td>
<td>Every 10 minutes</td>
<td>00,10,20,30,40,50 * * *</td>
<td></td>
</tr>
<tr>
<td>legalArchiver.sh</td>
<td>Archives Observations and Official User Products text files; purges obs product files older than 7 days, and OUP files older than 31 days</td>
<td>1 time/hour :55</td>
<td>55 * * * *</td>
<td></td>
</tr>
</tbody>
</table>
11.1.3.3 WFO dx2f HA Crons

Two cronfiles are active in /etc/cron.d/ when the dx2f a2dx2apps package is running: a2SITEdx2cron is for site-specific crons; and a2dx2cron is for the baseline-delivered crons. See Table 11.1.3.3-1.

Table 11.1.3.3-1. WFO dx2f HA crons

<table>
<thead>
<tr>
<th>Cron File</th>
<th>Cron Entry</th>
<th>What It Does</th>
<th>Frequency (Z time)</th>
<th>Cron Time (min, hour, day of month, month, day of week)</th>
</tr>
</thead>
<tbody>
<tr>
<td>a2SITEdx1cron</td>
<td>Site specific</td>
<td>Site specific</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

11.1.4 PX Crons

The PX servers run crons for standard Linux and for HA clusters. The px1f and px2f crons are maintained in /etc/ha.d/cron.d on both px1 and px2. When the HA packages start up, the appropriate crons are copied to /etc/cron.d and activated.

11.1.4.1 WFO PX Standard Crons

The PX standard crons run when the machine is up (running) and are not part of the HA failover packages. The PX standard crons are different from the workstations and the DX. See Table 11.1.4.1-1.

Table 11.1.4.1-1. WFO PX Standard crons

<table>
<thead>
<tr>
<th>Cron File</th>
<th>Cron Entry</th>
<th>What It Does</th>
<th>Frequency (Z time)</th>
<th>Cron Time (min, hour, day of month, month, day of week)</th>
</tr>
</thead>
<tbody>
<tr>
<td>/var/spool/cron/</td>
<td>logAndRemoveCoreFile.ksh</td>
<td>Removes core files</td>
<td>1 time/day 10:00</td>
<td>0 10 * * *</td>
</tr>
<tr>
<td>root</td>
<td>boxktp</td>
<td>Gathers performance and system metrics</td>
<td>1 time/day 23:59</td>
<td>59 23 * * *</td>
</tr>
<tr>
<td></td>
<td>boxstats_cleanlogs</td>
<td>Cleans up data logs created from boxstats performance gathering scripts</td>
<td>1 time/day 00:00</td>
<td>00 0 * * *</td>
</tr>
<tr>
<td></td>
<td>boxallstats</td>
<td>Generates sar and top statistics</td>
<td>1 time/day 00:01</td>
<td>1 0 * * *</td>
</tr>
<tr>
<td>Cron File</td>
<td>Cron Entry</td>
<td>What It Does</td>
<td>Frequency (Z time)</td>
<td>Cron Time (min, hour, day of month, month, day of week)</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>--------------------</td>
<td>---------------------------------------------------</td>
<td>--------------------</td>
<td>--------------------------------------------------------</td>
</tr>
<tr>
<td>/etc/cron.d/sysstat</td>
<td>root /usr/lib/sa1</td>
<td>Runs system activity accounting tool</td>
<td>Every 10 minutes</td>
<td>*/10 * * * *</td>
</tr>
<tr>
<td></td>
<td>root /usr/lib/sa2</td>
<td>Provides daily summary of process accounting</td>
<td>1 time/day 23:53</td>
<td>53 23 * * *</td>
</tr>
<tr>
<td></td>
<td>-A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/etc/cron.daily (run from entry in /etc/crontab)</td>
<td>logwatch</td>
<td>Parses system logs and creates output in /var/log</td>
<td>1 time/day 04:20</td>
<td>02 4 * * *</td>
</tr>
<tr>
<td></td>
<td>logrotate</td>
<td>Rotates log files</td>
<td>1 time/day 04:20</td>
<td>02 4 * * *</td>
</tr>
<tr>
<td></td>
<td>makewhatis.cron</td>
<td>Reads all man pages and updates the whatis database</td>
<td>1 time/day 04:20</td>
<td>02 4 * * *</td>
</tr>
<tr>
<td></td>
<td>mlocate.cron</td>
<td>Creates or updates a database of file names used by the locate command</td>
<td>1 time/day 04:20</td>
<td>02 4 * * *</td>
</tr>
<tr>
<td></td>
<td>prelink</td>
<td>Modifies ELF shared libraries and ELF dynamically linked binaries to reduce the time needed for the dynamic linker to perform relocations at startup</td>
<td>1 time/day 04:20</td>
<td>02 4 * * *</td>
</tr>
<tr>
<td>/etc/cron.daily (run from entry in /etc/crontab)</td>
<td>rpm</td>
<td>Creates list of rpm packages in /var/log/rpmpkgs</td>
<td>1 time/day 04:20</td>
<td>02 4 * * *</td>
</tr>
<tr>
<td></td>
<td>tmpwatch</td>
<td>Recursively removes files which haven’t been accessed for a given number of hours</td>
<td>1 time/day 04:20</td>
<td>02 4 * * *</td>
</tr>
<tr>
<td>/etc/cron.weekly (run from entry in /etc/crontab)</td>
<td>makewhatis.cron</td>
<td>Reads all man pages and writes a line in the whatis database consisting of the name of the man page and a short description</td>
<td>1 time/week on Sunday at 04:22</td>
<td>22 4 * * 0</td>
</tr>
</tbody>
</table>

### 11.1.4.2 WFO px1f HA Crons

Two cronfiles are active in /etc/cron.d/ when the px1f a2px1apps package is running: a2SITEpx1cron is for site-specific crons; and a2px1cron is for the baseline-delivered crons. See Table 11.1.4.2-1.
Table 11.1.4.2-1. WFO px1f HA crons

<table>
<thead>
<tr>
<th>Cron File</th>
<th>Cron Entry</th>
<th>What It Does</th>
<th>Frequency (Z time)</th>
<th>Cron Time (min, hour, day of month, month, day of week)</th>
</tr>
</thead>
<tbody>
<tr>
<td>a2SITEpx1cron</td>
<td>Site specific</td>
<td>Site specific</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>a2px1cron (fxa entries)</td>
<td>breakLogIngest</td>
<td>Breaks ingest logs daily</td>
<td>1 time/day 00:00</td>
<td>0 0 * * *</td>
</tr>
<tr>
<td></td>
<td>breakAnnouncementFiles</td>
<td>Breaks announcer files daily</td>
<td>1 time/day 00:00</td>
<td>0 0 * * *</td>
</tr>
<tr>
<td></td>
<td>master.purge</td>
<td>Performs master purger daily, to clean out old log directories</td>
<td>1 time/day 01:30</td>
<td>30 1 * * *</td>
</tr>
<tr>
<td></td>
<td>startScour</td>
<td>Scours daily to clean up log files and a few items not hit by master.purge</td>
<td>1 time/day 00:30</td>
<td>30 0 * * *</td>
</tr>
<tr>
<td></td>
<td>climate.sh auto am</td>
<td>Morning Climate (MDL)</td>
<td>1 time/day 12:25</td>
<td>25 12 * * *</td>
</tr>
<tr>
<td></td>
<td>climate.sh auto pm</td>
<td>Evening Climate (MDL)</td>
<td>1 time/day 22:25</td>
<td>25 22 * * *</td>
</tr>
<tr>
<td>a2px1cron (fxa entries)</td>
<td>launch_AEV.csh</td>
<td>Controls the MTR decoder feeding Climate (MDL)</td>
<td>Every hour at :07</td>
<td>7 * * *</td>
</tr>
<tr>
<td></td>
<td>clean_FSS_tables.sh</td>
<td>Purges the MTR decoder tables about once a week (MDL)</td>
<td>1 time/week</td>
<td>40 3 1,8,15,22 * *</td>
</tr>
<tr>
<td></td>
<td>hwrnwr -t</td>
<td>NOAA Weather Radio HWR</td>
<td>Every hour at :10</td>
<td>10 * * *</td>
</tr>
<tr>
<td></td>
<td>hwrnwws -t</td>
<td>NWWS HWR</td>
<td>Every hour at :10</td>
<td>10 * * *</td>
</tr>
<tr>
<td></td>
<td>awips2_radarIngest.pl</td>
<td>LAPS Ingest radar</td>
<td>Roughly every 5 minutes</td>
<td>05,10,15,19,25,30,35,40,45,50,55 * * *</td>
</tr>
<tr>
<td></td>
<td>awips2_satIngest.pl</td>
<td>LAPS Ingest satellite</td>
<td>Every 15 minutes</td>
<td>12,17,42,57 * * *</td>
</tr>
<tr>
<td></td>
<td>sched.pl</td>
<td>LAPS – runs LAPS analyses</td>
<td>Every hour at :20</td>
<td>20 * * *</td>
</tr>
<tr>
<td></td>
<td>startPurgeProcess</td>
<td>Keeps purgeProcess running</td>
<td>Every 10 minutes</td>
<td>*/10 * * *</td>
</tr>
<tr>
<td></td>
<td>WFOA_MSAS_Sfcmnc.run</td>
<td>MSAS Ingest the NCEP surface grids every 6 hours</td>
<td>Every 6 hours at :35</td>
<td>35 5,11,17,23 * *</td>
</tr>
<tr>
<td></td>
<td>WFOA_MSAS_Surface.run</td>
<td>MSAS Run the surface cycle 4 times an hour</td>
<td>4 times/hour at :11, :26, :41, :56</td>
<td>11,26,41,56 * * *</td>
</tr>
<tr>
<td></td>
<td>WFOA_MSAS_Asos.run</td>
<td>MSAS Compile the daily surface QC stats</td>
<td>1 time/day 00:05</td>
<td>5 0 * * *</td>
</tr>
<tr>
<td></td>
<td>WFOA_MSAS_QCstage1_2.run</td>
<td>QCMS stage 1 &amp; 2 QC on current hour's data</td>
<td>12 times/hour</td>
<td>1,6,11,16,21,26,31,36,41,46,51,56 * * *</td>
</tr>
<tr>
<td></td>
<td>WFOA_MSAS_QCstage1_2_late.run</td>
<td>QCMS stage 1 &amp; 2 QC on previous hour's data</td>
<td>12 times/hour</td>
<td>1,6,11,16,21,26,31,36,41,46,51,56 * * *</td>
</tr>
<tr>
<td></td>
<td>WFOA_MSAS_QCday.run</td>
<td>QCMS yesterday's QC stage 1, 2 &amp; 3 daily summaries</td>
<td>1 time/day at 01:15</td>
<td>15 1 * * *</td>
</tr>
<tr>
<td>Cron File</td>
<td>Cron Entry</td>
<td>What It Does</td>
<td>Frequency (Z time)</td>
<td>Cron Time (min, hour, day of month, month, day of week)</td>
</tr>
<tr>
<td>--------------------</td>
<td>----------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------------------------------</td>
<td>--------------------------------------------------------</td>
</tr>
<tr>
<td>(oper entries)</td>
<td>purge_files</td>
<td>Purges WHFS UNIX files</td>
<td>Every 4 hours at :01</td>
<td>01 1,5,9,13,17,21 * *</td>
</tr>
<tr>
<td></td>
<td>purge_mpe_files</td>
<td>Purges WHFS mpe files</td>
<td>1 time/day at 12:01</td>
<td>01 12 * * *</td>
</tr>
<tr>
<td></td>
<td>run_floodseq</td>
<td>Updates flood history for mpe files</td>
<td>1 time/day at 06:30</td>
<td>30 6 * * *</td>
</tr>
<tr>
<td></td>
<td>run_mpe_whfs</td>
<td>Set of programs that completes mpe analysis</td>
<td>1 time/hour at :24</td>
<td>24 * * *</td>
</tr>
<tr>
<td></td>
<td>process_qpe_mosaic</td>
<td>WFOs only - grib encodes the RFC QPE mosaics and sends them to awips for display</td>
<td>4 times/hour at :05,:20,:35,:50</td>
<td>05,20,35,50 * * *</td>
</tr>
<tr>
<td></td>
<td>purge_hpe_files</td>
<td>Purges HPE files according to retention period</td>
<td>Every 30 minutes</td>
<td>8,38 * * *</td>
</tr>
<tr>
<td></td>
<td>run_SSHP_HPN_preprocess</td>
<td>SSHP HPN Preprocessor</td>
<td>1 time/hour at :06</td>
<td>06 * * *</td>
</tr>
<tr>
<td></td>
<td>run_SSHP_var</td>
<td>Variational Data Assimilation for SAC-SMA model in SSHP [Note: Uncomment this cron entry line if your office is using VAR within SSHP]</td>
<td>1 time/hour at :30 (if uncommented)</td>
<td>#30 * * *</td>
</tr>
<tr>
<td></td>
<td>run_SSHP_SAC_state_update</td>
<td>Runs SSHP in background for RiverMonitor's SSHP fcst columns.</td>
<td>1 time/hour at :36 (if uncommented)</td>
<td>#36 * * *</td>
</tr>
<tr>
<td>(root entries)</td>
<td>VIRtest</td>
<td>Performs VIR test 3 times/day</td>
<td>3 times/day 02:15, 10:15, 18:15</td>
<td>15 2,10,18 * * *</td>
</tr>
<tr>
<td></td>
<td>startPurgeProcess</td>
<td>Starts purgeProcess if it is not running</td>
<td>Every 10 min</td>
<td>*/10 * * *</td>
</tr>
<tr>
<td></td>
<td>diskMon.sh</td>
<td>Updates disk monitor portion of AWIPS II System Monitor</td>
<td>Every 120 min</td>
<td>*/20 * * *</td>
</tr>
<tr>
<td></td>
<td>ldad_diskMon.sh</td>
<td>Updates disk monitor portion of AWIPS II System Monitor</td>
<td>Every 20 min</td>
<td>*/20 * * *</td>
</tr>
<tr>
<td></td>
<td>cpuMon.sh</td>
<td>Updates CPU monitor portion of AWIPS II System Monitor</td>
<td>Every 20 min</td>
<td>*/20 * * *</td>
</tr>
</tbody>
</table>

**11.1.4.3 WFO px2f HA Cron**

Two cronfiles are active in /etc/cron.d/ when the px2f a2px2apps package is running: a2SITEpx2cron is for site-specific cron; and a2px2cron is the baseline delivered cron. See Table 11.1.4.3-1.
Table 11.1.4.3-1. WFO px2f HA crons

<table>
<thead>
<tr>
<th>Cron File</th>
<th>Cron Entry</th>
<th>What It Does</th>
<th>Frequency (Z time)</th>
<th>Cron Time (min, hour, day of month, month, day of week)</th>
</tr>
</thead>
<tbody>
<tr>
<td>a2SITEpx2cron</td>
<td>BOIVerify and Site specific</td>
<td>BOIVerify and Site specific</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>a2px2cron (fxa entries)</td>
<td>startScour</td>
<td>Scours daily to clean up log files and a few items not hit by master.purge</td>
<td>1 time/day 00:30</td>
<td>30 0 ** **</td>
</tr>
<tr>
<td></td>
<td>breakLogIngest</td>
<td>Breaks ingest logs daily</td>
<td>1 time/day 00:00</td>
<td>0 0 ** **</td>
</tr>
<tr>
<td></td>
<td>breakLogLDAD.px2</td>
<td>Breaks ingest and decoder logs daily at 00Z</td>
<td>1 time/day 00:00</td>
<td>0 0 ** **</td>
</tr>
<tr>
<td></td>
<td>gatherLogs.pl</td>
<td>Collects system log files from several hosts</td>
<td>1 time/day 07:23</td>
<td>23 7 ** **</td>
</tr>
<tr>
<td></td>
<td>/usr/local/bin/scp -q ls1:/home/securityLogs/* /home/securityLogs</td>
<td>Copies the Security Logs from ls1 to px2</td>
<td>1 time/day 07:34</td>
<td>34 7 ** **</td>
</tr>
<tr>
<td></td>
<td>logAndRemoveCoreFile.Ksh</td>
<td>Removes core file from /home on an hourly basis at 10 past the hour</td>
<td>Every 10 minutes</td>
<td>10 ** ** **</td>
</tr>
</tbody>
</table>

11.1.5 WFO LS Crons

The LS servers run crons for both standard Linux and for the LS HA cluster. The LS crons are maintained in /etc/ha.d/cron.d on both ls2 and ls3. When the HA package is up, the appropriate crons are copied to /etc/cron.d and activated.

11.1.5.1 WFO LS Standard Crons

The LS standard crons run when the machine is up; they are not part of the HA failover package. See Table 11.1.5.1-1.

Table 11.1.5.1-1. WFO LS Standard Crons

<table>
<thead>
<tr>
<th>Cron File</th>
<th>Cron Entry</th>
<th>What It Does</th>
<th>Frequency (Z time)</th>
<th>Cron Time (min, hour, day of month, month, day of week)</th>
</tr>
</thead>
<tbody>
<tr>
<td>/var/spool/cron/root</td>
<td>No entries</td>
<td>Site Specific</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>/var/spool/cron/ldad</td>
<td>startScour.sh</td>
<td>Runs scour daily to clean up log files</td>
<td>1 time/day 00:30</td>
<td>30 0 ** **</td>
</tr>
<tr>
<td>/etc/cron.d/sysstat</td>
<td>root /usr/lib/sa/sa1 1 1</td>
<td>Runs system activity accounting tool</td>
<td>Every 10 minutes</td>
<td>*/10 ** **</td>
</tr>
<tr>
<td></td>
<td>root /usr/lib/sa/sa2 -A</td>
<td>Runs daily summary of process accounting</td>
<td>1 time/day 23:53</td>
<td>53 23 ** **</td>
</tr>
</tbody>
</table>

11.1.5.2 WFO LS HA Crons

Two cronfiles are active when the LS HA package is running: SITElscron is for site-specific crons; and lscron is the baseline delivered crons. See Table 11.1.5.2-1.
Table 11.1.5.2-1. WFO LS HA Crons

<table>
<thead>
<tr>
<th>Cron File</th>
<th>Cron Entry</th>
<th>What It Does</th>
<th>Frequency (Z time)</th>
<th>Cron Time (min, hour, day of month, day of week)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SITElscron</td>
<td>Site specific</td>
<td>Site specific</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>lscron (ldad entry)</td>
<td>breakLogLDAD.ls</td>
<td>Breaks LDAD logs daily at 00Z</td>
<td>1 time/day</td>
<td>0 0 * * *</td>
</tr>
<tr>
<td>(root entries)</td>
<td>rsync_cron.sh data_ldad</td>
<td>sync /data/ldad directory</td>
<td>1 time/hour at :10</td>
<td>10 * * *</td>
</tr>
<tr>
<td></td>
<td>rsync_cron.sh ldm</td>
<td>sync /usr/local/ldm/data/directory</td>
<td>2 times/day</td>
<td>20 6,12 * * *</td>
</tr>
<tr>
<td></td>
<td>rsync_cron.sh home</td>
<td>sync /home directory</td>
<td>1 time/day</td>
<td>01 23 * * *</td>
</tr>
</tbody>
</table>

11.1.6 WFO CSPBN Crons

The CPSBN servers run standard crons. See Table 11.1.6-1.

Table 11.1.6-1. WFO CPSBN crons

<table>
<thead>
<tr>
<th>Cron File</th>
<th>Cron Entry</th>
<th>What It Does</th>
<th>Frequency (Z time)</th>
<th>Cron Time (min, hour, day of month, day of week)</th>
</tr>
</thead>
<tbody>
<tr>
<td>/var/spool/cron/ root</td>
<td>removeArchives</td>
<td>Removes old monitor and control archives</td>
<td>1 time/day</td>
<td>9 0 * * *</td>
</tr>
<tr>
<td>/etc/cron.d/ sysstat</td>
<td>root /usr/lib/sa 1 1</td>
<td>Runs system activity accounting tool</td>
<td>Every 10 minutes</td>
<td>*/10 * * *</td>
</tr>
<tr>
<td></td>
<td>root /usr/lib/sa2 -A</td>
<td>Runs a daily summary of process accounting</td>
<td>1 time/day</td>
<td>53 23 * * *</td>
</tr>
<tr>
<td>a2cp1cron (ldm entries)</td>
<td>ldmadmin scour</td>
<td>Scours data directories</td>
<td>Every 3 hours</td>
<td>0 */3 * * *</td>
</tr>
<tr>
<td></td>
<td>ldmadmin newlog</td>
<td>Rotates log file</td>
<td>1 time/day</td>
<td>0 0 * * *</td>
</tr>
</tbody>
</table>

A qpid monitoring script takes snapshots of qpid status and other system-level statistics in order to monitor their health related to qpid in the event that issues arise related to qpid. The cron runs every 10 minutes on cp1f (it is part of a2cp1cron). The script collects qpid information and stores it in /data/fxa/qpid. It keeps one week’s worth of data.

*/10 * * * root sh -lc "/awips2/python/bin/monitor_qpid_host.sh"

11.1.7 WFO AX Crons

The AX server runs standard crons. See Table 11.1.7-1.
### Table 11.1.7-1. WFO AX Crons

<table>
<thead>
<tr>
<th>Cron File</th>
<th>Cron Entry</th>
<th>What It Does</th>
<th>Frequency</th>
<th>Cron Time (min, hour, day of month, month, day of week)</th>
</tr>
</thead>
<tbody>
<tr>
<td>/var/spool/cron/root</td>
<td>logAndRemoveCoreFile.ksh</td>
<td>Removes core files</td>
<td>1 time/day 10:00</td>
<td>0 10 * * *</td>
</tr>
<tr>
<td></td>
<td>boxkttp</td>
<td>Gathers performance and system metrics</td>
<td>1 time/day 23:59</td>
<td>59 23 * * *</td>
</tr>
<tr>
<td></td>
<td>boxstats_cleanlogs</td>
<td>Cleans up data logs created from boxstats performance gathering scripts</td>
<td>1 time/day 00:00</td>
<td>00 0 * * *</td>
</tr>
<tr>
<td></td>
<td>boxallstats</td>
<td>Generates sar and top statistics</td>
<td>1 time/day 00:01</td>
<td>1 0 * * *</td>
</tr>
<tr>
<td></td>
<td>nas_backup_weekly</td>
<td>Backs up all volumes on the NAS on a weekly basis</td>
<td>1 time/week</td>
<td>10 3 * * 1</td>
</tr>
<tr>
<td></td>
<td>startScour</td>
<td>Runs scour daily to clean up log files and a few items not hit by other purgers</td>
<td>1 time/day 00:30</td>
<td>30 0 * * *</td>
</tr>
<tr>
<td>/etc/cron.d/sysstat</td>
<td>root /usr/lib/sa/sa1 1 1</td>
<td>Runs system activity accounting tool</td>
<td>Every 10 minutes</td>
<td>0,10,20,30,40,50,50 * * * * * * * * * *</td>
</tr>
<tr>
<td></td>
<td>root /usr/lib/sa/sa2 -A</td>
<td>Runs daily summary of process accounting</td>
<td>1 time/day 23:53</td>
<td>53 23 * * *</td>
</tr>
</tbody>
</table>

#### 11.1.8 Wes2Bridge Archiver Crons

Archiver Retention is the management of data deletion (purging) from the file system. It is defined by configuration files stored in the EDEX Localization subsystem. The configurations can be edited through the Archive Retention Dialog, accessible through the menu CAVE > Archive Retention..., or manually (not recommended) by editing the XML configuration file through the CAVE Localization Perspective.

The Archive Purger is run by a quartz cron timer. This timer is a configurable property in the `/awips2/edex/conf/resources/com.raytheon.uf.edex.archive.cron.properties` file,

```
archive.purge.cron=0+5+0/3+*+*+?     
```

This is the delivered baseline value, which configures it to run once every three hours at 5 minutes after the hour, starting at midnight. The purger can be disabled by changing the value of the following from `true` to `false`:

```
archive.purge.enable=true   [change `true` to `false`]   
```

Moving data to the archive folder is quartz cron timer configurable property, 

```
archive.cron=0+40+*+*+*+*+?  
```
This is the delivered baseline value, which configures it to run once every hour at 40 minutes after the hour. Moving the data to the archive folder can be disabled by changing the property,

```
archive.enable=true
```

**Note:** EDEX will need to be restarted before any change in property values in the com.raytheon.uf.edex.archive.cron.properties file will take effect.

**Note:** The default archiver cron disabled the Localization and Processed entries. These should be on by default, and the times should be as follows:

- Localization: 1/day at 7z
- Processed: Every six hours after EDEX has updated the data in the archive directory.

### 11.2 quartz

EDEX essentially consists of a collection of components that execute in response to events. A “quartz”-driven event is one type of event available in EDEX. quartz provides a cron-like facility inside EDEX.

- quartz is part of EDEX, and is only used as part of EDEX. This means that quartz is only used on DX3/4, and that quartz-controlled jobs execute only while EDEX is running. EDEX does coordinate between DX3 and DX4, however. This means quartz-controlled jobs will run as long as EDEX is running on at least one of DX3 and DX4.
- quartz does not log to /var/log/cron. quartz errors will log to the appropriate EDEX process log. Jobs controlled by quartz may provide their own logging.
- quartz entries are compiled into EDEX; sites are not able to modify or view quartz entries.

Quartz jobs are scheduled to run when a given Trigger occurs. Triggers can be created with nearly any combination of the following directives:

- At a certain time of day (to the millisecond)
- On certain days of the week
- On certain days of the month
- On certain days of the year
- Not on certain days listed within a registered Calendar (such as business holidays)
- Repeated a specific number of times
- Repeated until a specific time/date
- Repeated indefinitely
- Repeated with a delay interval.
When quartz “fires,” an event is sent to the EDEX component. A quartz event can be configured to fire using the same basic expression as cron.

For example, EDEX uses a quartz trigger to run a periodic data purge. In the descriptor for the purge service, the Uri for the event source is

"quartz://purge/purge Scheduled/?cron=0+15+*+*+*+?".

This is equivalent to a cron expression of 15 * * * * — that is, quartz fires 15 minutes after every hour. AWIPS II EDEX contains a number of quartz event components that must perform periodic tasks like this.

As a result, many of the crons that are used in AWIPS I have become quartz event-driven EDEX components in AWIPS II.

A list of cron jobs and associated schedules for AWIPS II is provided in Table 11.2-1.

<table>
<thead>
<tr>
<th>Plug-In</th>
<th>Cron</th>
<th>Fires</th>
<th>Configuration</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.raytheon.edex.autobldsrv</td>
<td>0+<em>+</em>+<em>+</em>+?</td>
<td>Every minute</td>
<td>Subscription table in DB</td>
<td>Runs cron-based script runner</td>
</tr>
<tr>
<td>com.raytheon.edex.plugin.gfe</td>
<td>0+15+<em>+</em>+*+?</td>
<td>15 mins. after each hour</td>
<td>Normal localization</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td>0+10+<em>+</em>+*+?</td>
<td>10 mins. after each hour</td>
<td>Normal install/localization</td>
<td>Part of service backup</td>
</tr>
<tr>
<td>com.raytheon.edex.rpgenvdata</td>
<td>0+0+<em>+</em>+*+?</td>
<td>On the hour, every hour</td>
<td>Normal localization</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td>0+26,46+<em>+</em>+*+?</td>
<td>26 and 46 mins. after each hour</td>
<td>Normal localization</td>
<td>--</td>
</tr>
<tr>
<td>com.raytheon.uf.edex.distribution</td>
<td>0/5+<em>+</em>+<em>+</em>+?</td>
<td>Every 5 seconds</td>
<td>Distribution files</td>
<td>Refreshes data distribution rules</td>
</tr>
<tr>
<td>com.raytheon.uf.edex.metartohnmbsrv</td>
<td>0+14+<em>+</em>+*+?</td>
<td>At 14 mins. after each hour</td>
<td>Non configurable</td>
<td>Runs the HMDB purge</td>
</tr>
<tr>
<td>com.raytheon.uf.edex.ohd</td>
<td>0+7,17,27,37,47,57+<em>+</em>+*+?</td>
<td>7, 17, 27, 37, 47, and 57 mins. after each hour</td>
<td>Normal localization</td>
<td>Runs the WHFS alarms service</td>
</tr>
<tr>
<td></td>
<td>0+20+0,6,12,18+<em>+</em>+*+?</td>
<td>At 0:20, 6:20, 12:20 18:20 daily</td>
<td>Normal localization</td>
<td>Runs DQC preprocessor</td>
</tr>
<tr>
<td></td>
<td>0+30+6+<em>+</em>+*+?</td>
<td>At 6:30 each day</td>
<td>Normal localization</td>
<td>Runs flood sequence service</td>
</tr>
<tr>
<td></td>
<td>0+45+7+<em>+</em>+*+?</td>
<td>At 7:45 each day</td>
<td>Normal localization</td>
<td>Runs the IHFS database purge</td>
</tr>
<tr>
<td></td>
<td>0+1+1,5,9,13,17,21+<em>+</em>+*+?</td>
<td>At 1:01, 5:01, 9:01, 13:01, 17:01, 21:01 daily</td>
<td>Normal localization</td>
<td>Runs scheduled log file purger</td>
</tr>
<tr>
<td></td>
<td>0+25+<em>+</em>+<em>+</em>+?</td>
<td>25 minutes after each hour</td>
<td>Normal localization</td>
<td>Runs the hourly MPE</td>
</tr>
<tr>
<td></td>
<td>0+1+12+<em>+</em>+*+?</td>
<td>At 12:01 daily</td>
<td>Normal localization</td>
<td>Runs the HPE file purger</td>
</tr>
<tr>
<td></td>
<td>0+2,17,32,47+<em>+</em>+<em>+</em>+?</td>
<td>2, 17, 32, and 47 mins. after each hour</td>
<td>Normal localization</td>
<td>Runs the WHFS services</td>
</tr>
</tbody>
</table>
### 11.3 AWIPS II Data Purging

#### 11.3.1 Purging Processed Data Storage

Purging of data from Processed Data Storage is performed by an EDEX Ingest process running on the DX3/4 cluster. The purge is triggered by a quartz timer event that fires at 30 minutes after each hour.

EDEX logs Data Purge events into a separate log; the purge log is named `edex-ingest-purge-yyyyymmdd.log` where `yyyyymmdd` is the date stamp of the log. For example, the name of the purge log for June 10, 2011 is `edex-ingest-purge-20110610.log`. Among the purge events logged are the start and end of the purge. Since the EDEX Ingest process manages the purge, purge events may also log to the EDEX Ingest log.

EDEX uses a cluster locking mechanism that prevents both EDEX Ingest processes from performing the purge, although the purge may occur on either EDEX server. As a result, the EDEX Ingest logs on both DX3 and DX4 must be examined to verify purging and/or to identify purging problems.

Data purging is performed according to purging strategies defined for each data decoder in EDEX. A default, time-based purge strategy is provided for decoders that do not provide a purging strategy. Time-based purging is based on the reference time of the data. Determination of the reference time is decoder based. In the default purge strategy, all data more than 24 hours old at the time of the purge is deleted.

Purging rules are defined in XML files in the Localization Store on DX3/4. The purge rule files are located in the `/awips2/edex/data/utility/common_static` file tree and follow the base/site localization pattern. The base purge rule files are located in base/purge under common_static; the site purge rule files are located in site/XXX/purge under common_static, where XXX is the site identifier, e.g., OAX.

The order of precedence for the purge rules is site – base – default; that is, site rules override base rules, which override the default rule. When modifying a purge rule, the correct process is to copy the base file to the site purge rules folder and modify that copy. The preferred method of modifying purge rules, as with most files in the Localization Store, is through the Localization Perspective in CAVE. EDEX ingest rereads the purge rules before launching the purge service each hour, so there is no need to bounce processes when making changes to purge rules.

To verify the data purge, look at the size of the following directory structures:

- hdf5 data structure in `/awips2/edex/data/hdf5` (on dx2f); and
- The PostgreSQL table spaces in `/awips2/data` (dx1f).
One strategy for doing this is to set up a cron job to determine the space used in these two directory trees and write the results to a log file. This file can be examined periodically and should show generally stable disk usage over time. A simple script to perform this function follows.

```bash
#!/bin/bash
SPACE_DB=$(du /awips2/data -sh)
SPACE_HDF=$(du /awips2/edex/data/hdf5 -sh)
TIME_NOW=$(date "+%D %T")
echo "${TIME_NOW} DB: ${SPACE_DB} HDF5: ${SPACE_HDF}" >> /tmp/space_usage.log
```

This script can be run approximately every 10 minutes to provide a good picture of disk space usage for Processed Data Storage. Note that it must be run by the `awips` user on the server running the a2dx2apps (for hdf5) and a2dx1apps (for PostgreSQL) high-availability packages.

Here is an example for the purge rule xml file (Radar File) and the basics of these tags. [Note: For other examples, please look at the base files in common_static/base/purge directory.]

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<purgeRuleSet xmlns:ns2="group">
  <rule>
    <id>
      <pluginName>radar</pluginName>
      <key>default</key>
    </id>
    <period>01-00:00:00</period>
  </rule>
</purgeRuleSet>

<purgeRuleSet> or </purgeRuleSet>: There can be only one set per file.

<rule> or </rule>: Defines one rule. You can have multiple rules in a purgeRuleSet. Refer to satellitePurgeRules.xml

`pluginName`: Matches an EDEX plugin/metadata table/hdf5 directory.

`key`: Can be a specific column in the metadata table (icao=kgrr), which would be specific to the data that matches that key, or default, in which case it would match everything. [Note: After id, there could also be a period (.), which describes how long to keep the data.]

The above example is 1 day. However, if it were 00-06:00:00, it would purge anything older than 6 hours.

Note: There are also versionsToKeep, which will be an integer, and it is exactly that number of versions to keep. For example, if it is 12, then it will keep 12 versions of that data, regardless of how old it is.
If a plug-in’s xml is missing from base/site, then the defaultPurgeRules.xml file is used.

### 11.3.2 Purging Raw Data Storage

The Archive Purge is used to purge data from Raw Data Storage. The Archive Purge is controlled by the EDEX quartz process. The directories and retention times for Raw Data Storage are controlled by Archive Retention. Refer to Chapter 28.7 of the SMM for Archive Retention.

### 11.4 EDEX Log File Purging

Log purging is designed to reduce the space required for EDEX logs. In operation it will:

- Zip any log files that are more than 7 days old into a date stamped log file.
  - Only files that match the “standard” log naming pattern are zipped.
    - File name pattern is “^\d{4}\d{2}\d{2}(.*)\log$”.
    - Name contains an 8 digit number and ends in “.log”.
    - Although not required, the 8 digits represent YYYYMMDD.
  - The files that have been zipped are removed from the logs directory.

- Remove any zip files that are more than one month old.
  - File name pattern is “^\d{4}\d{2}\d{2}\zip$”.
  - Name is 8 digit followed by “.zip”.
    - Although not required, the 8 digits represent YYYYMMDD.

**Purge timing:**
The log purge fires at 00:30 each day.

**Configuration:**
The log purge can be rescheduled by modifying its `quartz` timer trigger value in `project.properties`. `project.properties` is located in `/awips2/edex/conf/spring`. The token value to change is “purge.logs.cron”. The default setting for this token is

```
purge.logs.cron=0+30+0+*+*+?
```

This specifies that the log purge will fire at 00:30 every day.

**Event logging:**
Log purging events are logged to both the EDEX Ingest and EDEX Ingest Purge logs.

**Purge events:**

```
PurgeLogger: EDEX: EDEX - ---------START LOG PURGE---------
PurgeLogger: EDEX: EDEX - Removed 0 old files
PurgeLogger: EDEX: EDEX - Archived 0 files
PurgeLogger: EDEX: EDEX - Skipped processing 0 files
PurgeLogger: EDEX: EDEX - ---------END LOG PURGE---------
```
Comments:
This is a somewhat generic zipper/purger and can handle logs other than the standard EDEX logs. The main catch is that the log name must contain an embedded 8-digit number and end “.log”. Files not matching this pattern are simply ignored.

The caveat: File name patterns and retention times are coded in and cannot be changed.

11.5 Locating a Script

In order to locate a script:

Log into DX3/DX4

Type: \texttt{cd /awips2}

Type: \texttt{find . | grep -i <filename>}

An example follows:

Type: \texttt{find . | grep -i purge_files}

\texttt{./edex/data/share/hydroapps/whfs/bin/purge_files}

\texttt{./edex/data/share/hydroapps/whfs/local/data/log/misc/purge_files.log}

Table 11.5-1 illustrates the various CRONS and their respective locations.

<table>
<thead>
<tr>
<th>CRON Name</th>
<th>Description</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>logfilepurger.cron</td>
<td>Runs the purge_files script</td>
<td>/awips2/edex/data/share/hydroapps/whfs/bin/purge_files</td>
</tr>
<tr>
<td>ihfsdbpurge.cron</td>
<td>Runs the run_db_purge script to purge the IHFS database</td>
<td>/awips2/edex/data/share/hydroapps/whfs/bin/run_db_purge</td>
</tr>
<tr>
<td>floodarchiver.cron</td>
<td>Runs the run_floodseq script</td>
<td>/awips2/edex/data/share/hydroapps/whfs/bin/run_floodseq</td>
</tr>
<tr>
<td>mpehpefilepurge.cron</td>
<td>Runs the purge_mpe_files and purge_hpe_files scripts</td>
<td>/awips2/edex/data/share/hydroapps/precip_proc/bin/purge_mpe_files &amp;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/awips2/edex/data/share/hydroapps/precip_proc/bin/purge_hpe_files</td>
</tr>
<tr>
<td>mpefieldgen.cron</td>
<td>Starts the MPE fieldGen process. This is the runHourlyMpe method in the</td>
<td>/awips2/edex/data/share/hydroapps/precip_proc/bin/ run_mpe_fieldgen</td>
</tr>
<tr>
<td></td>
<td>class.com.raytheon.uf.edex.ohd.pproc.MpeFieldGenSrv</td>
<td></td>
</tr>
<tr>
<td>pointdataretrieval.cron</td>
<td>Runs the run_pdc_pp script</td>
<td>/awips2/edex/data/share/hydroapps/whfs/bin/run_pdc_pp</td>
</tr>
<tr>
<td>alarmwhfs.cron</td>
<td>Runs the run_alarm_whfs script</td>
<td>/awips2/edex/data/share/hydroapps/whfs/bin/run_alarm_whfs</td>
</tr>
</tbody>
</table>
11.6 Hydro Cron Control Application Tokens

Tokens in the Apps_Defaults configuration allow for execution control of Hydro Cron tasks. Site-specific configuration files allow sites to override the base-deployed configuration with their own configuration.

11.6.1 Configuration

Two files (not including the site override files) control the Hydro Cron applications: a configuration file for the quartz cron timers; and a configuration file to set the application ON/OFF tokens.

**Token Configuration:** The Hydro Cron tokens are defined in the base Apps_Defaults file, which is managed by EDEX Localization. This file is accessible through the localization perspective of CAVE. It is not recommended that this file be edited because base updates will write over this file. See section 11.6.2, **Site-Specific Configurations**, for site customization. Please note that they default to ON.

```
/awips2/edex/data/utility/common_static/base/hydro/Apps_defaults
```

The delivered Hydro Cron tokens in the Apps_Defaults file are:

```
#============== Apps/Script Execution Tokens ====================
WhfsSrv                                        : ON
WhfsSrv.purge_files                            : ON
WhfsSrv.run_db_purge                           : ON
WhfsSrv.run_floodseq                           : ON
PprocSrv                                       : ON
PprocSrv.purge_mpe_files                       : ON
PprocSrv.purge_hpe_file                        : ON
MpeFieldGenSrv.run_mpe_fieldgen                : ON
WhfsSrv.run_pdc_pp                             : ON
WhfsSrv.run_alarm_whfs                         : ON
WhfsSrv.run_alarm_whfs.run_roc_checker         : ON
WhfsSrv.run_alarm_whfs.run_report_alarm        : ON
WhfsSrv.run_alarm_whfs.run_report_alarm.textdb : ON
ArealQpeGenSrv                                 : ON
DqcPreProcSrv                                  : ON
DqcPreProcSrv.run_dqc_preprocessor             : ON
MpeRUCFreezingLevel                            : ON
MpelightningSrv                                : ON
```

**Timer Configuration:** The cron timer properties are found in the base com.raytheon.uf.ohd.properties file. It is not recommended that this file be edited because base updates will write over this file. In addition, any changes made to this file will not take effect until EDEX is restarted. See section 11.6.2, **Site-Specific Configurations**, for site customization.

```
/awips2/edex/conf/resources/com.raytheon.uf.edex.ohd.properties
```
Cron Timer Properties: The cron timing properties look like this:

- logfilepurger.cron=0+1+1,5,9,13,17,21+*+*?
- ihfsdbpurge.cron=0+45+7+*+*?
- floodarchiver.cron=0+30+6+*+*?
- mpehpefilepurge.cron=0+1+12+*+*?
- mpefieldgen.cron=0+25+*+*+*?
- pointdataretrieval.cron=0+2,17,32,47+*+*+*?
- alarmwhfs.cron=0+7,17,27,37,47,57+*+*+*?
- arealgpgegen.cron=0+10,25,40,55+*+*+*?
- dqcpreprocessor.cron=0+20+0,6,12,18+*+*?
- mpleightningsrv.cron=0+0/30+*+*+*+?
- freezingLevel.cron=0+5+2,8,14,20+*+*+?

11.6.2 Site-Specific Configurations

Site Token Configuration: The site-specific tokens can be set in an Apps_Defaults file in the site/*** subdirectory (where *** is the site; for example, OAX). If a token is not defined in the site file, but is defined in the base file, then the base token will be used. This means that only those tokens that a site needs to change from the base token values need to be in the site file. This file will not be overwritten by base updates. This file should be created and edited through the CAVE Localization perspective.

```
/awips2/edex/data/utility/common_static/site/OAX/hydro/Apps_defaults
```

**NOTE:** The token structure of the form A.b, such as WhfsSrv.purge_files, works when set within the Apps_defaults files. However, in the korn and bash shells, command line overrides, which allow temporary token value assignment via something like “export WhfsSrv=OFF,” will not work for any of the above tokens where “.” occurs, as neither of those shells allows a simple variable assignment of the form a.b=x. The get_apps_defaults system looks for tokens defined as env’t vars as overrides. However, for purposes of these tokens, to control cron executions, the need for a command line override, e.g., within a script, is unlikely.

Site Timer Configuration: Site-specific cron timing properties can be defined in the site subdirectory, site/***. These properties are not managed by localization. These properties will override the base properties defined above. If a property is not defined in the site file, but is defined in the base file, then the base property will be used. This means that only those properties that a site needs to change from the base properties need to be in the site file. These properties will be preserved when updates are deployed. For changes to this file to take effect, EDEX must be restarted. The following example is for an EDEX running with OAX site settings.

```
/awips2/edex/conf/resources/site/OAX/com.raytheon.uf.edex.chd.properties
```

If the site directory path does not exist, you will need to create it. This is how it works:
Hydro Cron application control works by using a property, APP_CONTEXT, to contain the application context for the currently running Java code or shell script.

The property is managed by utility methods in AppsDefaults.java for Java code and the utility script check_app_context for scripts.

In Java, it is a system property; in scripts, it is an environment variable. The environment variable is passed to any scripts that the java code starts.

Java code started by the crons set and check APP_CONTEXT by calling AppDefaults.setAppContext(Object). If the method returns true, the code continues; otherwise it exits. The AppDefaults methods log will reveal whether the code was executed or not.

MainMethod.java sets up the ProcessBuilder environment for scripts started by java code. MainMethod uses AppsDefaults.setAppContext(processBuilder) to set the APP_CONTEXT environment variable for the script. Scripts source the utility script, check_app_context, to set and check the environment variable APP_CONTEXT.

The utility script, check_app_context, will default to true if no APP_CONTEXT variable or token is found, so scripts that call this method can still be used from the command line.

A step-by-step example follows. For this example, the Java code, WhfsSrv, executes the script run_alarms_whfs, which executes the script run_roc_checker.

The Site App tokens used in this example are:

| WhfsSrv                                    | ON |
| WhfsSrv.run_alarms_whfs                   | ON |
| WhfsSrv.run_alarms_whfs.run_roc_checker  | OFF|

1. When WhfsSrv is run, it calls AppsDefaults.setAppContext(Object).

2. There will not be a value set for the APP_CONTEXT at this time. It will set the APP_CONTEXT to “WhfsSrv”. Then this value is used to retrieve the App token. In the example App tokens above, “WhfsSrv” is set to ON. AppsDefaults.setAppContext(Object) will return true.

3. WhfsSrv will continue execution and run the script run_alarms_whfs.

4. run_alarms_whfs will source check_app_context, which will add run_alarms_whfs to the APP_CONTEXT (which will then be “WhfsSrv.run_alarms_whfs”) and use that value to check for the App token. In the example App tokens above, “WhfsSrv.run_alarms_whfs” is set to ON. check_app_context will log this, and run_alarms_whfs will continue execution and run the script run_roc_checker.
run_roc_checker will source check_app_context, which will add run_roc_checker to the APP_CONTEXT (which will then be “WhfsSrv.run_alarms_whfs.run_roc_checker”) and use that value to check for the App token. In the preceding example App tokens, “WhfsSrv.run_alarms_whfs.run_roc_checker” is set to OFF. check_app_context will log this and exit the script run_roc_checker. run_alarms_whfs will then continue from the point after it ran run_roc_checker.

### 11.6.3 Logging

Hydro Cron logging is in the EDEX ingest log:

```
/awips2/edex/logs/edex-ingest-YYYYMMDD.log
```

where YYYY is the year, MM the month, and DD the day of the month.

Note that, even when scripts are turned off, these messages will still be logged.

- For Java output, look for:
  
  **App Execution Token for App Context** `<Java context>` is `<true|false>`

- For script output, look for:
  
  **App Execution Token for script** `<script name>` with App Context `<script context>` is `<ON|OFF>`

Scripts that are not executed also log this message

```
Script <script name> will exit and not run.
```
Chapter 12
Database Management
# Chapter 12. Database Management

## Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.1 AWIPS II Database Management</td>
<td>1</td>
</tr>
<tr>
<td>12.1.1 Rehosted Databases</td>
<td>1</td>
</tr>
<tr>
<td>12.2 Text DB Overview</td>
<td>2</td>
</tr>
<tr>
<td>12.3 Other PostgreSQL Databases</td>
<td>5</td>
</tr>
<tr>
<td>12.4 PostgreSQL Database Environment Variables</td>
<td>5</td>
</tr>
<tr>
<td>12.5 PostgreSQL Database Archiving</td>
<td>5</td>
</tr>
<tr>
<td>12.6 PostgreSQL Database Sparing</td>
<td>6</td>
</tr>
<tr>
<td>12.7 PostgreSQL Logs</td>
<td>6</td>
</tr>
<tr>
<td>12.8 Checking System Processes</td>
<td>6</td>
</tr>
<tr>
<td>12.9 Monitoring PostgreSQL-Dependent Applications</td>
<td>7</td>
</tr>
<tr>
<td>12.10 PostgreSQL Cron</td>
<td>8</td>
</tr>
<tr>
<td>12.11 Stopping/Starting Database Engine Dependent Applications</td>
<td>8</td>
</tr>
<tr>
<td>12.12 PostgreSQL Vacuum Operations</td>
<td>9</td>
</tr>
<tr>
<td>12.13 Verifying AWIPS II Data Purging</td>
<td>9</td>
</tr>
<tr>
<td>12.13.1 Verify Purging of Data from the AWIPS II Raw Data Store</td>
<td>10</td>
</tr>
<tr>
<td>12.13.2 Verify Execution of the EDEX Purge Service on DX3 and DX4</td>
<td>11</td>
</tr>
<tr>
<td>12.13.3 Verify Purging of Data from the AWIPS II HDF5 Processed Data Store</td>
<td>14</td>
</tr>
<tr>
<td>12.14 Triggers</td>
<td>15</td>
</tr>
<tr>
<td>12.14.1 Environmental Variables</td>
<td>16</td>
</tr>
<tr>
<td>12.14.1.1 Logging</td>
<td>17</td>
</tr>
<tr>
<td>12.14.1.2 Migrating AWIPS I to AWIPS II</td>
<td>17</td>
</tr>
<tr>
<td>12.14.1.3 Viewing Current Triggers in AWIPS II</td>
<td>17</td>
</tr>
<tr>
<td>12.14.1.4 Updating Triggers in AWIPS II</td>
<td>17</td>
</tr>
<tr>
<td>12.15 Text Triggers</td>
<td>18</td>
</tr>
<tr>
<td>12.15.1 Configuring Text Triggers</td>
<td>19</td>
</tr>
<tr>
<td>12.15.1.1 ADAPT Text Triggers</td>
<td>19</td>
</tr>
<tr>
<td>12.15.1.2 LDAD Text Triggers</td>
<td>21</td>
</tr>
<tr>
<td>12.15.1.3 Installing a New ldad Trigger</td>
<td>23</td>
</tr>
<tr>
<td>12.15.1.4 Facsimile (fax) Text Triggers</td>
<td>24</td>
</tr>
<tr>
<td>12.15.1.5 Local Site Text Triggers</td>
<td>24</td>
</tr>
</tbody>
</table>
List of Tables

Table 12.1.1-1. Databases Reused from AWIPS I ................................................................. 1
Table 12.4-1. PostgreSQL Environmental Variables ............................................................ 5
Table 12.9-1. WFO/RFC PostgreSQL-Dependent Databases .................................................. 7
Table 12.9-2. Linux Commands for Monitoring Processes that Impact the PostgreSQL
    Database .......................................................................................................................... 7
Table 12.15.1-1. Text Trigger Template Files ........................................................................ 19
Table 12.15.1.2-1. LDAD Trigger Template File Directives ................................................... 22
PostgreSQL is the official database for AWIPS. For AWIPS II, the PostgreSQL database is hosted on the database cluster (DX 1/2). AWIPS II utilizes several of the existing AWIPS databases, although they are re-hosted into a new PostgreSQL instance. AWIPS II also adds additional databases to support the new ingest and display paradigms used by AWIPS II.

12.1 **AWIPS II Database Management**

AWIPS II uses the Dam Catalog (DAMCAT), Text (FXATEXT), hydrology (HYDRO), and climatology (HMDB) databases from AWIPS I. FXATEXT has some changes, the most visible change being that it is not using the watchwarn table. (Triggers have been moved into the subscription.subscriptions table in the metadata database.)

AWIPS II has introduced two databases to the mix: maps and metadata. maps contains definitions of shapes for the various maps used in CAVE and is generally static. metadata contains all other AWIPS II-specific data used by CAVE and EDEX. This data is generally dynamic, but some tables are static. In metadata, the subscription schema replaces the watchwarn table in the FXA TEXT database. The awips schema includes tables containing product metadata, some of which are static. It also includes the VTEC active table and the practice active table, which are used by WarnGen and GFE. Both of these schemas and their tables are created when the database is installed.

AWIPS II handles database (and data store) purges via a quartz-triggered component of EDEX; as a result, there are no cron jobs related to AWIPS II data purging.

In AWIPS II, PostgreSQL is started and stopped using a script called edex_postgres, which is located in /etc/init.d on the database server.

There are two dx1f crons for postgres, backup_pgdb_a2 –a and backup_pgdb_a2 -d metadata. They are used for backing up the database (metadata gets backed up more often to have a more recent snapshot of ingested data).

12.1.1 **Rehosted Databases**

AWIPS II reuses several databases from AWIPS I; they are listed in Table 12.1.1-1.

<table>
<thead>
<tr>
<th>Table</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>hd_ob92xxx</td>
<td>hydrology database, replace xxx with site ID</td>
</tr>
<tr>
<td>dc_ob7xxx</td>
<td>Dam Catalog, replace xxx with site ID</td>
</tr>
<tr>
<td>hmdb</td>
<td>historical meteorological database</td>
</tr>
<tr>
<td>lsrdata</td>
<td>LSR data</td>
</tr>
<tr>
<td>ob7_histdata</td>
<td>(RFC only) Historical river data</td>
</tr>
</tbody>
</table>

All databases that AWIPS II reuses are imported into the AWIPS II database at initial system installation.
AWIPS II uses a modified version of the Text database (fxatext). See section 12.2 for additional details on the fxatext database.

### 12.2 Text DB Overview

Decoded text products are stored in a PostgreSQL database on the Direct Attached Storage (DAS). The partition where the database resides is mounted to dx1f. The database works on a circular buffer basis storing the newest version of each product over the oldest. The number of versions of each product or category of products is specified in the `versionstokeep` field of the `textproductinfo` table.

The fxatext database consists of seven tables:

1. afoslookup
2. afos_to_awips
3. autofaxrecord
4. statematch
5. stdtextproducts
6. practicestdtexxproducts
7. textproductinfo
8. watchwarn

#### afoslookup

This table contains a mapping of AFOS CCC codes to WMO sites. It has two columns: `ccc` and `origin`. An example follows.

<table>
<thead>
<tr>
<th>ccc</th>
<th>origin</th>
</tr>
</thead>
<tbody>
<tr>
<td>WBC</td>
<td>K7VM</td>
</tr>
<tr>
<td>BHM</td>
<td>KAAF</td>
</tr>
<tr>
<td>PHL</td>
<td>KABE</td>
</tr>
<tr>
<td>LBB</td>
<td>KABI</td>
</tr>
<tr>
<td>ABQ</td>
<td>KABQ</td>
</tr>
<tr>
<td>FSD</td>
<td>KABR</td>
</tr>
<tr>
<td>ABQ</td>
<td>KABX</td>
</tr>
<tr>
<td>FTW</td>
<td>KACT</td>
</tr>
<tr>
<td>PHL</td>
<td>KACY</td>
</tr>
<tr>
<td>ARB</td>
<td>KADG</td>
</tr>
</tbody>
</table>

#### afos_to_awips

This table stores the WMO identifier and WMO ttaii code for each AFOS PIL (afosid). It has three columns: `afosid`, `wmocccc`, and `wmottaii`. An example follows.

<table>
<thead>
<tr>
<th>afosid</th>
<th>wmocccc</th>
<th>wmottaii</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABQABVABQ</td>
<td>KABQ</td>
<td>UFUS45</td>
</tr>
<tr>
<td>ABQADAABQ</td>
<td>KABQ</td>
<td>NOUS75</td>
</tr>
</tbody>
</table>
This table contains a listing of all xxx and ccc ID combinations for a state. It has three columns: state, xxx, and ccc. An example follows.

<table>
<thead>
<tr>
<th>state</th>
<th>ccc</th>
<th>xxx</th>
</tr>
</thead>
<tbody>
<tr>
<td>MO</td>
<td>UNO</td>
<td>STL</td>
</tr>
<tr>
<td>MO</td>
<td>VIH</td>
<td>STL</td>
</tr>
<tr>
<td>MS</td>
<td>137</td>
<td>JAN</td>
</tr>
<tr>
<td>MS</td>
<td>138</td>
<td>JAN</td>
</tr>
<tr>
<td>MS</td>
<td>139</td>
<td>JAN</td>
</tr>
<tr>
<td>MS</td>
<td>140</td>
<td>JAN</td>
</tr>
<tr>
<td>MS</td>
<td>BIX</td>
<td>JAN</td>
</tr>
<tr>
<td>MS</td>
<td>BKS</td>
<td>JAN</td>
</tr>
<tr>
<td>MS</td>
<td>CBM</td>
<td>JAN</td>
</tr>
<tr>
<td>MS</td>
<td>GLH</td>
<td>JAN</td>
</tr>
<tr>
<td>MS</td>
<td>GPT</td>
<td>JAN</td>
</tr>
<tr>
<td>MS</td>
<td>GTR</td>
<td>JAN</td>
</tr>
<tr>
<td>MS</td>
<td>GWO</td>
<td>JAN</td>
</tr>
<tr>
<td>MS</td>
<td>HBG</td>
<td>JAN</td>
</tr>
</tbody>
</table>

This table stores standard text products such as TAFs and METAR observations. It includes information on the version and creation time for standard products stored in the database. The product is also displayed. The table has 10 columns: cccid, datacrc, hdrtine, nnnid, site, wmoid, xxxid, bbbid, createtime, and product. Two examples follow.

<table>
<thead>
<tr>
<th>Cccid</th>
<th>Datacrc</th>
<th>Hdrtine</th>
<th>Nnnid</th>
<th>Site</th>
<th>Wmoid</th>
<th>Xxxid</th>
<th>Bbbid</th>
<th>Createtime</th>
<th>Product</th>
</tr>
</thead>
<tbody>
<tr>
<td>FTW</td>
<td>3901128814</td>
<td>050800</td>
<td>MTR</td>
<td>KPRX</td>
<td>SAUS70</td>
<td>PRX</td>
<td>RRN</td>
<td>1280994984898</td>
<td>SAUS70 KPRX 050800 RRN KPRX 050755Z 22005KT 10SM CLR 27/23 A2989 RMK 01</td>
</tr>
</tbody>
</table>
practicestdtextproducts

This table stores text products such as TAFs and METAR observations issued when CAVE/Text WS is in practice mode. It has the same structure as the stdtextproducts table.

textproductinfo

This table stores version information for each product in the database. It has four columns: cccid, nnnid, xxxid, and versionstokeep. An example follows.

<table>
<thead>
<tr>
<th>Cccid</th>
<th>Nnnid</th>
<th>Xxxid</th>
<th>versionstokeep</th>
</tr>
</thead>
<tbody>
<tr>
<td>WSR</td>
<td>UAM</td>
<td>BBX</td>
<td>2</td>
</tr>
<tr>
<td>LBB</td>
<td>MTR</td>
<td>ELP</td>
<td>6</td>
</tr>
<tr>
<td>MSP</td>
<td>RR2</td>
<td>MSR</td>
<td>2</td>
</tr>
<tr>
<td>NYC</td>
<td>MTR</td>
<td>JFK</td>
<td>6</td>
</tr>
<tr>
<td>PHL</td>
<td>MTR</td>
<td>ABE</td>
<td>6</td>
</tr>
<tr>
<td>PWM</td>
<td>MTR</td>
<td>MWN</td>
<td>6</td>
</tr>
<tr>
<td>BIS</td>
<td>RRZ</td>
<td>LDA</td>
<td>2</td>
</tr>
<tr>
<td>BOI</td>
<td>MTR</td>
<td>SNT</td>
<td>6</td>
</tr>
<tr>
<td>PWM</td>
<td>MTR</td>
<td>CON</td>
<td>6</td>
</tr>
<tr>
<td>PWM</td>
<td>MTR</td>
<td>PWM</td>
<td>6</td>
</tr>
<tr>
<td>NCF</td>
<td>WTS</td>
<td>NCF</td>
<td>2</td>
</tr>
<tr>
<td>ARB</td>
<td>MTR</td>
<td>P58</td>
<td>6</td>
</tr>
<tr>
<td>MKE</td>
<td>MTR</td>
<td>MKE</td>
<td>6</td>
</tr>
<tr>
<td>BOS</td>
<td>MTR</td>
<td>PVD</td>
<td>6</td>
</tr>
</tbody>
</table>

watchwarn

The watchwarn table is no longer in use and should be empty. It has been replaced by the tables in the subscription schema of the metadata database.
12.3 Other PostgreSQL Databases

The PostgreSQL database engine is also utilized by hydrology applications and interactive forecast applications. Descriptions of their PostgreSQL use may be found in the Government’s application-specific documentation.

12.4 PostgreSQL Database Environment Variables

The PostgreSQL database engine runs on the Linux data server where the dx1f floating heartbeat address resides (either DX1 or DX2). The following chart lists the environment variables set for DX1 and DX2. These variables (DXs) need to be established (using the `setenv` or `export` commands) before executing the PostgreSQL commands listed on the following pages. Table 12.4-1 lists the environment variables.

NOTE: AWIPS II uses a different paradigm for setting the runtime environment required by AWIPS II applications. In this paradigm, the environment is set in the application startup script. As a result, these environment variables are not required for running AWIPS II applications. They are retained here for compatibility with AWIPS I scripts.

Table 12.4-1. PostgreSQL Environmental Variables

<table>
<thead>
<tr>
<th>Environment Variables</th>
<th>Description</th>
<th>DX1 and DX2</th>
</tr>
</thead>
<tbody>
<tr>
<td>PGDATA</td>
<td>Directory (or partition) for cluster data and default database data</td>
<td>/data/db/pgdata</td>
</tr>
<tr>
<td>PGDATA_IHFS</td>
<td>Partition for IHFS and DAMCREST</td>
<td>/data/db/ihfs</td>
</tr>
<tr>
<td>PGDATA_FXATEXT</td>
<td>Partition for FXATEXT</td>
<td>/data/db/fxatext</td>
</tr>
<tr>
<td>PGDATA_HMDB</td>
<td>Partition for HMDB</td>
<td>/data/db/hmdb</td>
</tr>
<tr>
<td>PGDATA_IFPS</td>
<td>Partition for IFPS and WWA</td>
<td>/data/db/ifps</td>
</tr>
<tr>
<td>PGHOST</td>
<td>PostgreSQL host device</td>
<td>dx1f</td>
</tr>
<tr>
<td>PG_INSTALL</td>
<td>Installation directory</td>
<td>/usr</td>
</tr>
<tr>
<td>PGDATA_TOP</td>
<td>Top of the data directory tree</td>
<td>/data/db</td>
</tr>
<tr>
<td>PGPORT</td>
<td>Port definition</td>
<td>5432</td>
</tr>
<tr>
<td>PGUSER</td>
<td>Non-administrative user</td>
<td>postgres</td>
</tr>
<tr>
<td>PATH</td>
<td>Path to PostgreSQL</td>
<td>includes /usr/include/pgsql</td>
</tr>
<tr>
<td>TERM</td>
<td>Terminal setting</td>
<td>vt100</td>
</tr>
<tr>
<td>PGDATA_LOCAL</td>
<td>Partition for local RFC databases</td>
<td>/data/db/dblocalrfc</td>
</tr>
</tbody>
</table>

Note: For the PXs and LXs, `SPGDATA` is defined as `/var/lib/pgsql/data`.

12.5 PostgreSQL Database Archiving

The entire PostgreSQL database is backed up daily at about 2205Z by running a cron that executes a backup of each individual database. The log files created from this backup are stored using the following naming convention:
backup_<db>_<date>

Where <db> is the database - in AWIPS it is pgdb (PostgreSQL database), and <date> is the month and day in MMDD format.

Example for 27 Oct 2011: backup_pgdb_1027. This file is located in /data/logs/afx on dx1.

An example of the backup file for backup_pgdb_1027 is as follows:

Oct 27 01:10:02 BEGIN pg_dump metadata
Oct 27 01:14:51 END pg_dump metadata EXIT_CODE=0
Oct 27 05:10:01 BEGIN pg_dump metadata
Oct 27 05:14:33 END pg_dump metadata EXIT_CODE=0
Oct 27 09:10:01 BEGIN pg_dump metadata
Oct 27 09:14:48 END pg_dump metadata EXIT_CODE=0
Oct 27 13:10:02 BEGIN pg_dump metadata
Oct 27 13:15:25 END pg_dump metadata EXIT_CODE=0

Note: For manual backup, the cron entry is
05 22 * * * root /awips/ops/bin/backup_pgdb_a2 –

The log file is /data/logs/afx/backup_pgdb_<MMDD>

Note: The database backups are in /data/afx/DAILY_BACKUP/postgres/<Day> dc_ob70ax, fxatext, globals, hd_ob830ax, hd_ob92eax, hb_ob92ntc, hmdb, ifps_eax, lsrdata, maps, metadata, postgres

12.6 PostgreSQL Database Sparing

If the primary Linux Data Server (DX1) fails, PostgreSQL operations move to the DX2. The database itself is on the Network Appliance (NetApp) FAS2020C DAS, attached to the DX so it is not affected if the DX1 fails.

12.7 PostgreSQL Logs

PostgreSQL logs its actions in /awips2/data/pg_log. It also has logs for its database backups (see section 12.5) and vacuum operations (see section 12.12). These logs reside in the /data/logs/afx directory on nas1-<siteID>.

12.8 Checking System Processes

To check the system processes on the PostgreSQL Database:

```
TYPE: ps -wef | grep postgres

You should see output similar to the following:
```

postgres 19792 1 0 Mar21 ? 00:00:15 /usr/bin/postgres -D /data/db/pgdata
postgres 20002 19792 0 Mar21 ? 00:00:18 postgres: writer process
Chapter 12 Database Management

12.9 Monitoring PostgreSQL-Dependent Applications

AWIPS II PostgreSQL-dependent databases are listed in Table 12.9-1.

Table 12.9-1. WFO/RFC PostgreSQL-Dependent Databases

<table>
<thead>
<tr>
<th>WFO</th>
<th>RFC</th>
</tr>
</thead>
<tbody>
<tr>
<td>fxatext</td>
<td>fxatext</td>
</tr>
<tr>
<td>dc_obx&lt;site&gt;</td>
<td>hd_obx&lt;site&gt;</td>
</tr>
<tr>
<td>hd_obx&lt;site&gt;</td>
<td>dc_obx&lt;site&gt;</td>
</tr>
<tr>
<td>ifps_&lt;site&gt;</td>
<td>ifps_&lt;site&gt;</td>
</tr>
<tr>
<td>wwa_&lt;site&gt;</td>
<td>wwa_&lt;site&gt;</td>
</tr>
<tr>
<td>hmd5</td>
<td>hmd5</td>
</tr>
<tr>
<td>lsrdata</td>
<td>lsrdata</td>
</tr>
</tbody>
</table>

NOTE: The ifps_<site>, wwa_<site> and lsrdata are not used by AWIPS II software.

To Check Processes Executing on the Active DX

The following process must be executing at all times on the active Linux data server:

- postgres <database> as root or postgres

To check the processes executing on the active data server, use the `grep` command for the process you want to monitor (see Table 12.9-2).

Table 12.9-2. Linux Commands for Monitoring Processes that Impact the PostgreSQL Database

<table>
<thead>
<tr>
<th>Use</th>
<th>UNIX Command</th>
<th>User</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monitor whether PostgreSQL is running</td>
<td>ps -wef</td>
<td>grep postgres</td>
</tr>
</tbody>
</table>

To Check Interaction between the PostgreSQL Database and the TextDB Server

Log in remotely to either server in the EDEX cluster (DX3 or DX4). As user fxa or your individual user account, execute any of the following commands for checking the fxatext database:
NOTE: AFOS commands are case sensitive.

**TYPE:** `textdb -r <AFOS_PIL>`

- Performs a standard AFOS read, for example
  `textdb -r PITAFDPIT`

**TYPE:** `textdb -w <productID> or <AFOS_PIL>`

- Writes the product to the database, for example
  `textdb -w PITAFDPIT`

**TYPE:** `textdb -t <productID> or <AFOS_PIL>`

- Retrieves create time for products, for example
  `textdb -t PITAFDPIT`

**TYPE:** `textdb -A <productID> or <AFOS_PIL>`

- Retrieves all times for a product, for example
  `textdb -A BOSMTRBOS`

### 12.10 PostgreSQL Cron

The PostgreSQL `vacuum/analyze cron` executes frequently to maintain a fairly level steady-state usage of disk space. PostgreSQL’s `vacuum` command must be run on a regular basis for several reasons:

- To recover disk space occupied by updated or deleted rows. This disk space must be reclaimed for reuse by new rows to avoid infinite growth of disk space requirements.
- To update data statistics used by the PostgreSQL query planner to run better database queries. This information is gathered via the `analyze` command, which can be run as an option by `vacuum` and is set up to run as part of the cron.
- To protect against the loss of very old data due to transaction ID wraparounds that can occur if more than two billion database transactions occur before the `vacuum` command is run for that database.

### 12.11 Stopping/Starting Database Engine Dependent Applications

If the PostgreSQL database engine must be stopped and restarted manually, the **EDEX processes on DX3 and DX4** must also be stopped and restarted manually in a specific sequence.

As user **root** on both DX3 and DX4, perform the following steps.

1. On DX3, stop the EDEX processes
   
   **TYPE:** `service edex_camel stop`

2. On DX4, stop the EDEX processes
   
   **TYPE:** `service edex_camel stop`
3. On DX3, verify these processes have all stopped. As any user
   TYPE: service edex_camel status

4. On DX4, verify these processes have all stopped. As any user
   TYPE: service edex_camel status

When the database engine is backed up, you will need to restart all of these processes. First, verify that they are down by rerunning the status command (steps 3 and 4 above), and then restart the processes if they are down.

As user root on DX3, restart the EDEX processes.
   TYPE: service edex_camel start

Still as user root, but on DX4, restart the EDEX processes
   TYPE: service edex_camel start

12.12 PostgreSQL Vacuum Operations

PostgreSQL vacuum operations occur every 4 hours for the hdob<version><site> database, twice a day for hmdb and fxatext, and once a day for the other databases. The log files created from this operation are located in /data/logs/fxa/.

Database logging for AWIPS applications continues as previously done. Log files for some AWIPS applications are still found in the /data/logs/fxa directory (e.g., ingProcMon.log), and the logs for the other AWIPS applications still reside in the /data/logs/fxa/<YYYYMMDD> directory (e.g., /data/logs/fxa/20111017 for 17 October 2011).

12.13 Verifying AWIPS II Data Purging

Data purging is the process of deleting outdated data from the data storage systems. In AWIPS II, data purging involves the following: purging data from the AWIPS II Raw Data Store; verifying the execution of the EDEX purge service on DX3 and DX4; and purging data from the AWIPS II HDF5 Processed Data Store.

Validation of purging will be somewhat different for each area, but the general strategy is:

1. Determine the retention strategy used for the purge.
2. Verify that the purge is scheduled.
3. Examine the appropriate process logs to determine if the purge has taken place.
4. Examine the data to verify that the purge has taken place.

The strategy for verifying that data has been purged is somewhat different for a database than for a file system, but, essentially, you compare a count of “all” data against the
count of “old” data. The count of old data should be zero or significantly less than the count of all data. Note that counting data by age can be rather time consuming.

The AWIPS II Processed Data Storage purge is managed by the EDEX Ingest processes running on the DX3/4 cluster. Purge events are logged to the edex-ingest-purge-yyyyymmdd.log file in /awips2/edex/logs. This file is located on both DX3 and DX4, although EDEX uses a locking mechanism between DX3 and DX4 to ensure that only one purge executes at any one time. The purge log lists the number of items purged for each purge rule every time a purge is run. Because the log is maintained on both EDEX servers, both servers need to be checked to verify purge execution.

Descriptions of the three areas of AWIPS II data purging identified in this section follow.

12.13.1 Verify Purging of Data from the AWIPS II Raw Data Store

A quartz cron job on the EDEX ingest server manages purging of the AWIPS II Raw Data Store. The Raw Data Store is purged using a time-based retention strategy, which is configured using Archive Retention.

**Note**: Archiver purging of raw data is only for WFOs. The ldm raw data purge continues for NCEP and RFC platforms.

The steps follow.

1. Open a terminal window on your workstation.
2. Log into the EDEX ingest server.
   
   **Enter:** `ssh awips@dx3`
3. Change directory to the EDEX logs.
   
   **Enter:** `cd /awips2/edex/logs`
4. Verify that the Archive purge is enabled and that it has a valid quartz cron entry.
   
   **Enter:**
   ```
   grep archive.purge
   com_raytheon_uf_edex_archive_cron_propertiesarchive
   .purge.enable=true
   archive.purge.cron=0+5+0/3+**+?n
   ```
5. Change directory to the EDEX log directory.
   
   **Enter:** `cd /awips2/edex/logs`
6. Verify that the Archive Purge is firing.
7. Exit the shell.

12.13.2 Verify Execution of the EDEX Purge Service on DX3 and DX4

The EDEX purge service runs as part of the EDEX Ingest process on DX3 and DX4. Purge events are logged to the Ingest Purge log, edex-ingest-purge-yyyy-mm-dd.log, located in /awips2/edex/logs on both DX3 and DX4. In normal operation, you should see a “START PURGE” message at 30 minutes after each hour on either DX3 or DX4. Problems with the purge will also be in this log file.

The steps follow.

1. Open a terminal window on your workstation.

2. Log into DX3 as the root user.
   
   **Enter:** ssh root@dx3

3. Switch to the bash shell.
   
   **Enter:** bash

4. Change directory to the EDEX log directory.
   
   **Enter:** cd /awips2/edex/logs

5. List the current purge log.
   
   **Enter:** ls edex-ingest-purge-$\{\text{date} +\%Y\%m\%d\}.log

   **Note:** Because the purge can occur on either DX3 or DX4, it is possible that there will be no purge log on DX3. In that case you may skip the next step.
6. Display “START PURGE” and “END PURGE” events.
   Enter: `egrep -A2 "(START|END) PURGE" edex-ingest-purge-$(date +%Y%m%d).log`
   
   Note: Because the purge can occur on either DX3 or DX4, it is possible that no
   purge events will be logged on DX3.

7. Terminate the bash shell.
   Enter: `exit`

8. Log off DX3.
   Enter: `exit`

9. Log into DX4 as the root user.
   Enter: `ssh root@dx4`

10. Switch to the bash shell.
    Enter: `bash`

11. Change directory to the EDEX log directory.
    Enter: `cd /awips2/edex/logs`

12. List the current purge log.
    Enter: `ls edex-ingest-purge-$(date +%Y%m%d).log`
    
    Note: Because the purge can occur on either DX3 or DX4, it is possible that there
    will be no purge log on DX4. In that case, you may skip the next step.

13. Display “START PURGE” and “END PURGE” events.
    Enter: `egrep -A2 "(START|END) PURGE" edex-ingest-purge-$(date +%Y%m%d).log`
    
    Note: Because the purge can occur on either DX3 or DX4, it is possible that no
    purge events will be logged on DX4.
PurgeLogger: EDEX - PURGE LOGS:--------START LOG PURGE--------

INFO 2013-11-04 00:30:00,023 [Default Quartz Scheduler- 
INFO 2013-11-04 00:30:00,023 [Default Quartz Scheduler- 
purgeLogsContext_Worker-5] PurgeLogger: EDEX - PURGE LOGS: Skipped unknown file: edex-ingestGrib- 
20131104.log.lck
INFO 2013-11-04 00:30:00,023 [Default Quartz Scheduler- 
INFO 2013-11-04 00:30:00,023 [Default Quartz Scheduler- 
purgeLogsContext_Worker-5] PurgeLogger: EDEX - PURGE LOGS: Skipped file with invalid file name: 
apsPilTrigger.log
INFO 2013-11-04 00:30:00,024 [Default Quartz Scheduler- 
purgeLogsContext_Worker-5] PurgeLogger: EDEX - PURGE LOGS: Skipped unknown file: edex-ingestDat- 
20131104.log.lck
INFO 2013-11-04 00:30:00,024 [Default Quartz Scheduler- 
INFO 2013-11-04 00:30:00,024 [Default Quartz Scheduler- 
INFO 2013-11-04 00:30:00,024 [Default Quartz Scheduler- 
purgeLogsContext_Worker-5] PurgeLogger: EDEX - PURGE LOGS: Skipped unknown file: edex-ingestDat- 
20131104.log.lck
INFO 2013-11-04 00:30:00,025 [Default Quartz Scheduler- 
clusteredpurgeContext_Worker-2] PurgeLogger: EDEX - PURGE OUTGOING:
INFO 2013-11-04 00:30:00,025 [Default Quartz Scheduler- 
clusteredpurgeContext_Worker-2] PurgeLogger: EDEX - PURGE OUTGOING:--------START OUTGOING PURGE--------
INFO 2013-11-04 00:30:00,038 [Default Quartz Scheduler- 
clusteredpurgeContext_Worker-2] PurgeLogger: EDEX - PURGE OUTGOING: Skipped file with unexpected file name: 
447_test.doc
INFO 2013-11-04 00:30:00,038 [Default Quartz Scheduler- 
clusteredpurgeContext_Worker-2] PurgeLogger: EDEX - PURGE OUTGOING: Removed 0 files
INFO 2013-11-04 00:30:00,038 [Default Quartz Scheduler- 
clusteredpurgeContext_Worker-2] PurgeLogger: EDEX - PURGE OUTGOING:--------END OUTGOING PURGE--------

14. Terminate the bash shell.
   **Enter: exit**

15. Log off DX4.
Enter:  exit

16. Expected results: If the EDEX Purge service is firing correctly, you should see paired “START PURGE” and “END PURGE” events at about 30 minutes after each hour. **Note:** If this procedure is executed shortly after 30 minutes after the hour, the final “START PURGE” event will not have a paired “END PURGE” event. Compare the results of the *egrep* commands in steps (f) and (m); there should be log entries covering all hours of the current day.

17. Error results: Any missing “START PURGE” messages should be investigated. EDEX uses a database table (*awips.cluster_task* in the *metadata* database) to manage process locking between DX3 and DX4. As a result, any database-related errors in the purge log should be investigated. An example follows.

```plaintext
INFO  2013-02-12 00:49:00,138 [Purge-TAF-Thread] PurgeLogger: EDEX - TAF::Purging expired data...
FATAL 2013-02-12 00:49:02,565 [DefaultQuartzScheduler_Worker-2] PurgeLogger: EDEX - TEXT::Purger running time has exceeded timeout duration of 20 minutes. Current running time: 21 minutes
ERROR 2013-02-12 00:49:02,586 [DefaultQuartzScheduler_Worker-2] PurgeLogger: EDEX - TEXT::Stack trace for Purge Job
Thread:
Thread ID: 16619  Thread state: RUNNABLE
java.net.SocketInputStream.socketRead0(Native Method)
java.net.SocketInputStream.read(SocketInputStream.java:129)
org.postgresql.core.VisibleBufferedInputStream.readMore(VisibleBufferedInputStream.java:135)
org.postgresql.core.VisibleBufferedInputStream.ensureBytes(VisibleBufferedInputStream.java:104)
org.postgresql.core.VisibleBufferedInputStream.read(VisibleBufferedInputStream.java:73)
```

**12.13.3 Verify Purging of Data from the AWIPS II HDF5 Processed Data Store**

The HDF5 Data Store is located in /awips2/exed/data/hdf5 on dx2f. A “quick and dirty” method of determining data purge is to count the number of files of specific ages in this directory structure. Unlike the LDM-managed AWIPS II Raw Data Store, the AWIPS II HDF5 Processed Data Store is managed primarily via version-based purging. That is, the number of products of a specific data type, rather than the age of the data, is the primary purge mechanism. As a result, there is no way to determine the age of the oldest HDF5 file that we should expect to find. As a quick check, we can find the number of files of various ages; that count should decrease as the age increased.

The steps follow.
1. Open a terminal window on your workstation.
2. Log into dx2f as the root user.
   
   Enter: `ssh root@dx2f`
3. Change directory to the AWIPS II HDF5 Data Store.
   
   Enter: `cd /awips2/edex/data/hdf5`
4. Determine the number of files in the AWIPS II HDF5 Data Store.
   
   Enter: `find . -type f | wc -l`

   Note that this command may take several minutes to execute.

   37474

5. Determine the number of files in the AWIPS II HDF5 Data Store that are more than one, two, three, and four days old.

   Enter: `find . -mtime +1 -type f | wc -l`

   8930

   Enter: `find . -mtime +2 -type f | wc -l`

   8591

   Enter: `find . -mtime +3 -type f | wc -l`

   8124

   Enter: `find . -mtime +4 -type f | wc -l`

   7686

   Note: Each command may take several minutes to execute.


   Enter: `exit`

7. Expected results: If the purge of the AWIPS II HDF5 Data Store is set up and executing, each `find | wc -l` command should show fewer files.

12.14 Triggers

Triggers are maintained in AWIPS II basically in the same way they are maintained in AWIPS I – through the `/data/fxa/siteConfig/textApps/siteTrigger.template` file. The structure of this file has not changed from AWIPS I.
1. Triggers run as user **awips** from within edex on dx3 and dx4.

2. The product files are written to /awips2/edex/data/fxa/trigger; therefore, most triggers scripts will have to be changed for this new directory. This directory is only available on dx3, dx4, px1, and px2.

3. Use the config_awips2.sh script to invoke changes to the siteTrigger.template file. Refer to the *AWIPS II Site Migration Guide* for instructions on running config_awips2.sh.

4. siteTrigger.template file will likely be a symbolic link to an AWIPS I/II-specific file. Make sure you are changing the appropriate file for AWIPS I/II use.

5. The location of your trigger script on the system has changed from AWIPS I. All local application programs and scripts are recommended to be in /localapps/runtime. This is a long-term goal. At present, it is not a hard requirement when initially switching to running AWIPS II operationally. All AWIPS I directories such as /data/local and /awips/dev remain unchanged when running AWIPS II.

6. When porting AWIPS I trigger scripts, it is like that there will be no need to set environment variable FXA_HOME or source readenv.(c)sh. These should be removed from the scripts.

Triggers in AWIPS II are run as edex **Subscriptions** and not as postgres database triggers. The subscription command should not be used to add or delete triggers. Use siteTrigger.template and config_awips2.sh.

### 12.14.1 Environmental Variables

Trigger scripts will likely inherit a limited environment; therefore, trigger scripts must explicitly set what the script will need. Most of the environment settings that are needed to run AWIPS-specific programs are defined in /etc/profile.d. You can examine them to determine what your application needs are and source accordingly. Otherwise, to get the default environment for the user the script is running as, add the "-l" (lower case L) option to the shell invocation on the first line of the script:

```
#!/bin/sh -l
```

or for csh:

```
#!/bin/csh -l
```

This forces reading system wide settings in /etc/profile.d and user settings in $HOME/.bashrc, .cshrc, .login, etc.

**Note:** For user fxa, the default shell is csh, so a user who is running a bash script will not get .cshrc settings in /awips/fxa/.cshrc. The user might have to create a .bashrc file to further set up the environment.
Use the -l option only as a last resort if you cannot find the specific environment variables you need.

For cron or trigger scripts that run perl, python, tcl, etc. directly, you might have to wrap in a shell script and have the shell script set up the environment:

```sh
#!/bin/sh -l
export TZ=EST5EDT
exec /localapps/runtime/bin/myApp.tcl > myLogFile 2>&1
```

Baseline programs such as txtdb, gfeclient, runProcedure, etc. should set up the necessary environment they need to run. If you have problems running these from cron or triggers due to environment problems, open a ticket. Note that the path setting to these programs may still need to be set:

```sh
txtdb -r PIL - might not work because of PATH setting

/awips/fxa/bin/ txtdb -r PIL - should always work.
```

### 12.14.1.1 Logging

To send output of a trigger script to a log file, use a wrapper script like the one in the example above. This will capture errors in the script as well as any normal output.

It is recommended that sh (bash) scripts be used instead of csh. If using an sh script, you can redirect stdout and stderr to a log file directly within the script using the “exec” statement.

### 12.14.1.2 Migrating AWIPS I to AWIPS II

Follow the steps in section 5, Automation Tool (config_awips2), in the AWIPS II Site Migration Guide.

### 12.14.1.3 Viewing Current Triggers in AWIPS II

You can view your current AWIPS II triggers/subscriptions via the following command:

```
/awips2/fxa/bin/subscription -o read -t pil
```

**Note:** The subscription does have other commands as well.

### 12.14.1.4 Updating Triggers in AWIPS II

As root, ssh to DX1

- Update dx1:/data/fxa/siteConfig/textApps/siteTrigger.template
  - Remove any triggers that are no longer relevant.
— Update any script locations which may have changed.

- cd /data/fxa/sdc (as root on DX1)
- ./config_awips2.sh -f triggers XXX (where XXX is your 3-letter site ID in CAPS)

**Note:** The -f will purge and repopulate the triggers with national triggers as well as your siteTrigger.

- Once the script is completed, you will need to bounce the asyncScheduler on px1 to append the asyncScheduler triggers. To do this:
  - As user fxa on PX1f
    - stopAsyncScheduler
    - startAsyncScheduler
- When done, an output copy of the triggers is located in dx1:/var/tmp/awipsTriggersSetup.txt

### 12.15 Text Triggers

The Text Triggers utility monitors the text data stream and matches a set of predefined product PILs based on the old AFOS PILs, against the data being inserted into the fxadatabase. A template contains the PILs to be detected; it is refined using configuration files that contain text-product strings. Text Triggers allow processing of the data by processes that are not monitoring the raw data input stream and that may desire locally generated text products (which may or may not be distributed outside of the local office) in addition to products from the Satellite Broadcast Network.

Text Triggers can be invoked for processing by the following AWIPS functional areas:

- LDAD (for the site, and for site backup)
- ADAPT
- Fax
- Local triggers

The LDAD triggers are generated using a site-specific configuration file in conjunction with a template file.

Fax triggers are set by the text workstation user, and local-products triggers are created from site-generated files.

In all cases, the PIL is passed as the last argument to the script.

The following environmental variables and their corresponding definitions are used in this section.

```bash
$FXA_DATA   /data/fxa
$FXA_HOME   /awips/fxa
```
Text Trigger templates provide a list of products and formats for sites to follow and build upon in setting up their own sites’ triggers. These templates are usually found in the $FXA_DATA/nationalData directory, with the notable exceptions of the facsimile (fax) and local site triggers. Table 12.15.1-1 lists some of these template types and the paths to their respective files. These template types are discussed in the following subsections.

**Note:** The Hydro Triggers in AWIPS I has been re-engineered into AWIPS II.

### Table 12.15.1-1. Text Trigger Template Files

<table>
<thead>
<tr>
<th>Trigger Template Type</th>
<th>Trigger Template Full Path and File Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADAPT</td>
<td>$FXA_DATA/nationalData/adaptTrigger.template, $FXA_DATA/nationalData/adaptTriggerAll.template</td>
</tr>
<tr>
<td>LDAD</td>
<td>$FXA_DATA/nationalData/ldadTrigger.template</td>
</tr>
<tr>
<td>LDAD Site Backup</td>
<td>$FXA_DATA/nationalData/ldadSiteBackupTrigger.template</td>
</tr>
<tr>
<td>Facsimile (Fax)</td>
<td>$FXA_DATA/workFiles/fax/&lt;LLL&gt;-faxTrigger.template</td>
</tr>
<tr>
<td>Local Site Text Triggers</td>
<td>$FXA_DATA/siteConfig/textApps/siteTrigger.template</td>
</tr>
</tbody>
</table>

#### 12.15.1.1 ADAPT Text Triggers

The Meteorological Development Laboratory uses Text Triggers for ADAPT. ADAPT ingests all the SAW and SEV products stored in the database for display within WWA and the ZFP, DSM, and CCF products for designated WFOs.

Messages to be detected are defined in the file $FXA_HOME/data/localization/<LLL>/adaptSiteConfig.txt, where <LLL> is the site’s localization ID, usually the same as the site ID. Each distinct entry is a single line with the following format:

```
www <old_AFOS_ID>
```

Where

- **www** is a literal string, and
- **<old_AFOS_ID>** is a variable three-letter string that identifies the old AFOS ID of the WFO whose messages are to be detected.

You can edit the <LLL>-adaptSiteConfig.txt file to add as many entries as you like. The following is an example of an <LLL>-adaptSiteConfig.txt file with four distinct entries:

```
www WBC
www PHL
```
The Text Triggers utility reads `<LLL>-adaptSiteConfig.txt` and automatically constructs the AFOS PILs to be detected.

These products are brought in via `fxaTextTriggerActions` entries copied from the file `/data/fxa/nationalData/adaptTriggerAll.template`:

```
CCCSAWXXX | | | /awips/adapt/ifps/bin/linux/ingest_saw-lx.bat |
CCCSEVXXX | | | /awips/adapt/ifps/bin/linux/ingest_sev-lx.bat |
CCCWCLXXX | | | /awips/adapt/ifps/bin/linux/ingest_wcl-lx.bat |
CCCWOUXXX | | | /awips/adapt/ifps/bin/linux/ingest_wou-lx.bat |
CCCTCVXXX | | | /awips/adapt/ifps/bin/linux/ingest_tpc-lx.bat |
```

ZFP products are used to create the Surface Area Forecast (SAF) read on the NOAA Weather Radio, and the local ASOS summary messages are used by the Climate program; all of these products are brought in via the `~nationalData/adaptTrigger.template` file, which contains the following entries:

```
wwwDSMXXX | | | /awips/adapt/climate/bin/LINUX/ASOS_smDecoder.sh |
wwwMSMXXX | | | /awips/adapt/climate/bin/LINUX/ASOS_smDecoder.sh |
```

where the `www` directive(s) are set in the `<LLL>-adaptSiteConfig.txt` file and `<LLL>` represents the localization ID, which is usually the same as the site ID. Both CCC and XXX are wildcards.

The contents of the `$FXA_LOCAL_SITE-wsoTrigger.template` file, if one exists, are also added to the `$FXA_HOME/postgres/fxatextTriggerActions.txt` file. This addition provides access to non-AWIPS NWS sites’ (Alaska Region) products in the WWA monitor. An example of the added products follows.

```
JNUWSWANN | | | /awips/adapt/ifps/bin/linux/ingest_wso_lx.bat |
JNUWSWYAK | | | /awips/adapt/ifps/bin/linux/ingest_wso_lx.bat |
ANCWSWADQ | | | /awips/adapt/ifps/bin/linux/ingest_wso_lx.bat |
ANCWSWAKN | | | /awips/adapt/ifps/bin/linux/ingest_wso_lx.bat |
FAIWSWOME | | | /awips/adapt/ifps/bin/linux/ingest_wso_lx.bat |
FAIWSWOTZ | | | /awips/adapt/ifps/bin/linux/ingest_wso_lx.bat |
JNUNPWANN | | | /awips/adapt/ifps/bin/linux/ingest_wso_lx.bat |
JNUNPWYAK | | | /awips/adapt/ifps/bin/linux/ingest_wso_lx.bat |
ANCNPWBET | | | /awips/adapt/ifps/bin/linux/ingest_wso_lx.bat |
ANCNPWCDB | | | /awips/adapt/ifps/bin/linux/ingest_wso_lx.bat |
ANCSPSADQ | | | /awips/adapt/ifps/bin/linux/ingest_wso_lx.bat |
```

www CRW
www PIT
12.15.1.2 LDAD Text Triggers

The **SFXA_HOME/postgres/fxatextTrigger.sh** script sets up the LDAD Text Triggers in the text database to send specified text data to the LDAD system. The script reads a data file containing the required product IDs (AFOS PILs) and associated actions. This file is created by the localization routine **fxatextTriggerConfig.sh** and is found in **SFXA_HOME/postgres/fxatextTriggerActions.txt**.

An example of the **SFXA_HOME/postgres/fxatextTriggerActions.txt** output is

```
SFORVISTO | | | /awips/fxa/ldad/bin/textdbNotify.pl statement |
SFOSHPSTO | | | /awips/fxa/ldad/bin/textdbNotify.pl report |
SFOSHISTO | | | /awips/fxa/ldad/bin/textdbNotify.pl report |
SFOSMASTO | | | /awips/fxa/ldad/bin/textdbNotify.pl warning |
SFOSPFSTO | | | /awips/fxa/ldad/bin/textdbNotify.pl forecast |
SFOSWOSTO | | | /awips/fxa/ldad/bin/textdbNotify.pl outlook |
```

The **SFXA_DATA/nationalData/ldadTrigger.template** file contains the LDAD actions, and includes entries like

```
wwwSPSww1 | | | /awips/fxa/ldad/bin/textdbNotify.pl statement |
wwwTORww1 | | | /awips/fxa/ldad/bin/textdbNotify.pl warning |
wwwSVRww1 | | | /awips/fxa/ldad/bin/textdbNotify.pl warning |
wwwFFAww1 | | | /awips/fxa/ldad/bin/textdbNotify.pl watch |
wwwSTPsss | | | /awips/fxa/ldad/bin/textdbNotify.pl report |
rrrFFGsss | | | /awips/fxa/ldad/bin/textdbNotify.pl outlook |
wwwSMAww1 | | | /awips/fxa/ldad/bin/textdbNotify.pl forecast |
```

The **www**, **ww1**, **sss**, and **rrr** placeholders included in the template are replaced by site-specific locations defined in the **SFXA_HOME/data/localization/<LLL>/<LLL>-ldadSiteConfig.txt** file, where `<LLL>` represents the localization ID, usually the same as the site ID. The file consists of at least four lines, defining each of the directives, as shown in Table 12.15.1.2-1.
Table 12.15.1.2-1. LDAD Trigger Template File Directives

<table>
<thead>
<tr>
<th>Term 1</th>
<th>Term 2</th>
<th>Description of Terms</th>
</tr>
</thead>
<tbody>
<tr>
<td>www</td>
<td>wfo</td>
<td>www is a literal string and &lt;wfo&gt; is a variable three-letter string that identifies the CCC of an AFOS PIL (i.e., the AFOS node, like SFO for site STO).</td>
</tr>
<tr>
<td>ww1</td>
<td>wfo</td>
<td>ww1 is a literal string and &lt;wfo&gt; is a variable three-letter string that identifies the XXX of an AFOS PIL (i.e., the AWIPS site ID, like STO). It is possible to have the same string as the www or to have a different site identifier for ww1*.</td>
</tr>
<tr>
<td>sss</td>
<td>state ID</td>
<td>sss is a literal string and &lt;state ID&gt; is a variable two-letter string that identifies a state (i.e., the state’s 2-letter USPS designation, like CA for California).</td>
</tr>
<tr>
<td>rrr</td>
<td>rfc</td>
<td>rrr is a literal string and &lt;rfc&gt; is a variable three-letter string that identifies a River Forecast Center.</td>
</tr>
</tbody>
</table>

*The default <LLL>-ldadSiteConfig.txt files do not include a ww1 directive. One is added during triggers processing using the value of the FXA_LOCAL_SITE environment variable. If a different ww1 is needed, it can be included in a copy of the <LLL>-ldadSiteConfig.txt file placed in $FXA_CUSTOM_FILES.*

Some sites need to consider products from only one WFO, state, and RFC while others require multiple states and RFCs. Two examples are shown below. One for Tucson [TWC] and one for Pittsburg [PBZ]

TWC-ldadSiteConfig.txt
www PHX
ww1 TWC
sss AZ
rrr SLC

PBZ-ldadSiteConfig.txt
www PIT
ww1 PBZ
sss PA
sss OH
sss MD
sss WV
rrr PIT
rrr CRW

It is acceptable for more than one WFO to appear in the ldadSiteConfig.txt file, although none of the default files has more than one.

The LDAD site backup trigger capability is identical to the main LDAD trigger configuration except that the backup configuration file is named
<LLL>-ldadSiteBackupConfig.txt and the template is $FXA_DATA/nationalData/ldadSiteBackupTrigger.template.

Actions associated with the backup files are included in the TriggerActions file. Even if the current site backup template lacks RFC information, an entry for rrr must be present.

**Note:** Individual site backup configuration files are not delivered as part of the AWIPS installation and must be created locally by each site.

### 12.15.1.3 Installing a New ldad Trigger

The following steps are needed to install a new ldadTrigger

1. Install the trigger in the ldadTrigger.template file. On DX1 as user root,

```
TYPE: cd /data/fxa/sdc
```

```
TYPE ./config_awips2.sh -f triggers LLL
```

**Note:** <LLL> is the localization identifier and XXX is your 3-letter site ID in caps. Both of these IDs are usually the same as the site’s 3-letter AWIPS ID.

2. Verify DB installation. On DX1, as user root,

```
TYPE: psql -U awips metadata
```

```
TYPE: select trigger, filepath from subscription.subscriptions;
```

3. The results should look similar to the following:

```
productid | script
-----------+--------------------------------------------------
ANCHWOAFC | /awips/fxa/ldad/bin/textdbNotify.pl statement
ANCSVSAFC | /awips/fxa/ldad/bin/textdbNotify.pl statement
ANCTORAFC | /awips/fxa/ldad/bin/textdbNotify.pl warning
ANCSVRAFC | /awips/fxa/ldad/bin/textdbNotify.pl warning
ANCFFAAFC | /awips/fxa/ldad/bin/textdbNotify.pl watch
ANCFFWAFC | /awips/fxa/ldad/bin/textdbNotify.pl warning
ANCFFSAFC | /awips/fxa/ldad/bin/textdbNotify.pl statement
ANCRFWAFC | /awips/fxa/ldad/bin/textdbNotify.pl warning
ANCFLSAFC | /awips/fxa/ldad/bin/textdbNotify.pl statement
ANCFLWAFC | /awips/fxa/ldad/bin/textdbNotify.pl warning
ANCPSWAFC | /awips/fxa/ldad/bin/textdbNotify.pl warning
ANCNPWAFC | /awips/fxa/ldad/bin/textdbNotify.pl forecast
ANCLFPAFC | /awips/fxa/ldad/bin/textdbNotify.pl forecast
```
4. Exit the database.

   **TYPE:** /q
   
   **Result:** The user should see a prompt similar to the following that lets the user know that he/she exited the database:
   
   [root@dx1-tbw4 bin]$ 

5. Look in the `/awips/fxa/postgres/fxatextTriggerActions.txt` file for the HWO product line, which should look similar to the following:

   WBCHWOLWX | | /awips/fxa/ldad/bin/textdbNotify.pl statement| 

6. Using the product that you see in the `fxatextTriggerActions.txt` file, create an entry in the `fxatext` database. The following is an example:

   **TYPE:** textdb -w WBCHWOLWX
   **TYPE:** This is a test
   **TYPE:** [Ctrl-d]

7. To check that the product has been triggered, grep for the product in the edex-ingest-<date>.log file on all EDEX servers

   grep WBCHWOLWX /awips2/edex/logs/edex-ingest-<date>.log

   You should see entries with “LdadMicroEngine:” indicating that EDEX has spawned the script for that trigger.

### 12.15.1.4 Facsimile (fax) Text Triggers

Text Triggers can be configured to transmit a facsimile automatically when an AFOS PIL is detected. Using the **Configure Auto Fax** interface on the text workstation, a user can set up a list of products that are to be sent by fax upon receipt. This list of products is stored in `$FXA_DATA/workFiles/fax/<LLL>-faxTrigger.template` and is inserted directly in the `fxatextTriggerActions.txt` file after being modified to fit the `TriggerActions` file’s format.

### 12.15.1.5 Local Site Text Triggers

Local site text triggers should be placed in the file `SFXA_DATA/siteConfig/textApps/siteTrigger.template`. These triggers are handled in the same way as the fax triggers – simply expanded to fit the `TriggerActions` format. Each entry consists of a single line listing the product PIL and the full path name of an
executable to be invoked when the product is stored. You can create
**siteTrigger.template** and add as many entries as you would like.

The following is an example of the **siteTrigger.template** file:

```
NMCRRMWY /data/fxa/siteConfig/textApps/snotel/decodeSNOTEL.csh
DENFWCTAD /data/fxa/siteConfig/textApps/tempfcst/tempfcst.csh
DENAFDDEN /data/fxa/siteConfig/textApps/collector/collectafd.csh
DENALGDEN /data/fxa/siteConfig/textApps/collector/collectalg.csh
DENTAPDEN /data/fxa/siteConfig/textApps/collector/collecttap.csh
DENRR3DEN /home/andersen/scripts/decodeRR3s.pl DENRR3DEN
DENRR3GJT /home/andersen/scripts/decodeRR3s.pl DENRR3GJT
DENRR3PUB /home/andersen/scripts/decodeRR3s.pl DENRR3PUB
DENRR1GJT /home/andersen/scripts/decodeRR3s.pl DENRR1GJT
DENRR2DEN /home/andersen/scripts/decodeRR3s.pl DENRR2DEN
DENRR2ALS /home/andersen/scripts/decodeRR3s.pl DENRR2ALS
TOPRR3GLD /home/andersen/scripts/decodeRR3s.pl TOPRR3GLD
DENSTPCO /home/andersen/scripts/decodeRR3s.pl DENSTPCO
SLCSTPUT /home/andersen/scripts/decodeRR3s.pl SLCSTPCO
DENSPSDEN /data/fxa/siteConfig/textApps/sigwx/sigwx.pl
DENWRKRR1 /home/andersen/scripts/decodeMtnstates.pl
```

**Note:** Any script that is created for the local triggers should take the AFOS PIL of the product as the final argument.
Chapter 13
Event Notification
Chapter 13. Event Notification

Table of Contents

13.1 Description of Text Alarm/Alert Capability .................................................................1

List of Exhibits

Exhibit 13-1. Text Workstation User Interface .................................................................1
Exhibit 13.1-1. default.xml Alarm/Alert and Proximity Alarm Products GUI ..................2
Exhibit 13.1-2. default.xml Alarm/Alert and Proximity Alarm Product GUI Error Message ....3
The text workstation can be configured to monitor the `fxatext` database for arrival of certain products, for specific content of selected products, or for products whose valid area intersects a defined area of interest. The Text Alarm/Alert capability is accessed through the text workstation user interface, as shown in Exhibit 13-1 (refer to Section 5.4 of the *AWIPS CAVE-D2D User’s Manual* for guidance on working with Alarm/Alert features).

![Text Workstation User Interface](image)

**Exhibit 13-1. Text Workstation User Interface**

The Text Alarm/Alert capability allows the user to select text products to be alarmed (sound) or alerted (no sound). A flashing bell at the upper left corner of the screen appears when a product of interest is received, optionally accompanied by a sound. A click on the bell silences the alarm (if on) and opens a dialog containing the product name(s). The user can display the product by clicking on it with the mouse.

### 13.1 Description of Text Alarm/Alert Capability

The list of text products to monitor is built from four files; two of them are controlled by the system manager and the other two can be defined and edited by any AWIPS user. In each case, one file is for the Alarm/Alert function, and the other is for the new Proximity Alarm capability. The list of Text Alarm/Alert products and Proximity Alarm products is presented to the user in a combined user interface (UI), as shown in Exhibit 13.1-1.

To display the Alarm/Alert and Proximity Alarm Products GUI, as shown in Exhibit 13.1-1:

1. Click on the Alarm Alert button in the Text Workstation GUI.
2. Click on the Product List button in the Current Alarm Queue GUI.
In Exhibit 13.1-1, the products available in the default.xml window (the upper window of the UI), above the line that reads “--items above are uneditable--,” are defined by the system manager and cannot be modified. [Note: The alarm files should be maintained through the Localization Perspective, where a DefaultSiteAlarms.xml file can be established on a site, per-workstation, or per-user basis.]

Any products listed in the default.xml window of the UI below the line that reads “--items above are uneditable--” are user-defined and user-editable using the Text Alarm/Alert UI. This product list resides in the /awips2/edex/data/utility/common_static/user/uname>/alarms/default.xml file, where <uname> is the user’s login name. Note that the file is maintained on the EDEX cluster (DX 3/4) and is copied to /home/<uname>/common/user/<uname>/alarms.

In Exhibit 13.1-1, the products available in the Alarm/Alert and Proximity Alarm Products window displayed (the lower window of the UI) above the line that reads “--items above are uneditable--,” are defined by the system manager and cannot be
modified via the Text Alarm/Alert UI. **Note:** The alarm files should be maintained through the Localization Perspective, where a `DefaultSiteAlarms.xml` file can be established on a site, per-workstation, or per-user basis.

- If any products are listed in the **Proximity Alarm Products** window of the UI below the line that reads “—items above are uneditable—,” these products are user-defined and user-editable using the Text Alarm/Alert UI. This product list resides in the `/awips2/edex/data/utility/common_static/user/<uname>/alarms/default.xml` file, where `<uname>` is the user’s login name. Note that the file is maintained on the EDEX cluster (DX 3/4) and is copied to `/home/<uname>/caveData/common/user/<uname>/alarms`.

If the user attempts to select any of the entries in the system manager-controlled list, an error message stating that it is not allowed is displayed at the bottom of the window, as shown in Exhibit 13.1-2.

Exhibit 13.1-2. default.xml Alarm/Alert and Proximity Alarm Product GUI Error Message
Users can elect to save their list of user-controlled Alarm/Alert products for later recall; the file is saved on the EDEX cluster in

```bash
SEDEX_HOME/data/utility/common_static/user/<uname>/alarms/default.xml
```

The user-defined file name will appear in the window title bar (i.e., referring to Exhibits 13.1-1 and 13.1-2); “init” in the window title bar would be replaced by the user-defined file name, if any. This way, users and system managers know what file is controlling the alarm/alert function.
Chapter 14
Message Handling System
Chapter 14. Message Handling System

Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>14.1 SBN Retransmission</td>
<td>1</td>
</tr>
<tr>
<td>14.2 Administrative Message Handling System</td>
<td>2</td>
</tr>
<tr>
<td>14.2.1 MHS Environment Variables, Directories, and Configuration Files</td>
<td>2</td>
</tr>
<tr>
<td>14.2.1.1 MHS Environment Variables and Values (on all DX1 as user fxa)</td>
<td>2</td>
</tr>
<tr>
<td>14.2.1.2 MHS Administrative Directories</td>
<td>3</td>
</tr>
<tr>
<td>14.2.1.3 MHS Configuration Files</td>
<td>3</td>
</tr>
<tr>
<td>14.3 MHS Receiver Processing</td>
<td>3</td>
</tr>
<tr>
<td>14.4 Address Expansion</td>
<td>5</td>
</tr>
<tr>
<td>14.4.1 NCF Administrative Messages</td>
<td>6</td>
</tr>
<tr>
<td>14.4.1.1 Troubleshooting and Monitoring Commands</td>
<td>7</td>
</tr>
<tr>
<td>14.4.1.1.1 Troubleshooting</td>
<td>7</td>
</tr>
<tr>
<td>14.4.1.1.2 Monitoring Commands</td>
<td>8</td>
</tr>
<tr>
<td>14.5 Site Level Illustration</td>
<td>8</td>
</tr>
</tbody>
</table>

List of Exhibits

Exhibit 14.5-1. Receive Distributed Product and Send Acknowledgment...... 9
Exhibit 14.5-2. Distribute Product and Receive Acknowledgement................ 10

List of Tables

Table 14.3-1. Message Handlers................................................................... 4
Table 14.3-2. Key Words in Handling Specifications.................................... 4
The deployment of an SMTP protocol enables users to distribute text, grid, and radar imagery products efficiently and accurately among the AWIPS sites, via the terrestrial WAN.

To verify that the SMTP Message Handling System (MHS) processes are running, on DX1:

```
TYPE: ps -wef|grep smtp
You should see output similar to the following:

root     29944     1  0 Jul11 ?        00:00:06 mdp_daemon -k 7789 -f
/awips/ops/data/mhs/mdp-smtp.cfg
awipsmhs 29945 29944  0 Jul11 ?        00:18:26 smtp_send -eb -cz -k7789
-n0 -p/tmp/smtp0_send_fifo
awipsmhs 29971     1  0 Jul11 ?        00:00:10 smtp_recv -n0 -m0 -
o/data/mhs/smtp_recv/output - b/var/spool/mail/awipsmhs
```

### 14.1 SBN Retransmission

SBN Retransmission provides the capability for a site to automatically request retransmission of missed NWS Telecommunications Gateway (NWSTG) products from the Network Control Facility (NCF). Retransmission is intended to handle short-term data loss situations; it is not intended to provide data replenishment for long-term outages.

SBN Retransmission is enabled by default. In addition, CPSBN reboots re-enable retransmission. **Note:** All channels except GOES support automatic retransmission. GOES does not because the NCF software does not support it.

When the site’s CPSBN1 determines that an interruption in NWSTG product receipt has occurred, a retransmission request is automatically transmitted over the WAN to the NCF. The CPSBN determines if an interruption has occurred by detecting gaps in the NCF-assigned product sequence numbers. For example, if product number 870 follows 865, a retransmission request for products 866, 867, 868, and 869 is automatically issued.

Once the NCF receives the retransmission request, the requested range of missing NWSTG data will be retransmitted over the SBN to every site. The site responsible for the retransmission request ingests the products as “RETRANS.” Sites that already have these products will discard them as duplicates. There will be occasions, however, when products will arrive at a nonrequesting site and be processed as first-time products. This duplicate processing may occur if a communications processor (CP) has recently been restored or spared over. To avoid excessive WAN traffic, retransmission requests from a single site are sent at a minimum interval of 30 seconds.

The number of products retransmitted by the NCF depends on two items: the number of products requested; and the number of product requests already in the site’s outstanding product retransmit request table. Currently, the default is 10,000 products for a single request and at most 15,000 products queued for a site. As requested products are
retransmitted, the associated outstanding product request is deleted from the request table. This default configuration can be increased manually to accommodate increased product volumes during the anticipated 15- to 20-minute solar outage periods (i.e., 20,000 products for a single request and at most 20,000 products queued). Site requests that exceed these maximum values are normally logged at the NCF but discarded. An alarm is generated to notify NCF operators that site replenishment may be required.

**NOTE:** Data collected thus far indicate that sites typically request fewer than 2,000 products, so large requests (over 5,000 products) do not occur frequently. Retransmission requests are not issued for problems such as power failures or reboots.

### 14.2 Administrative Message Handling System

The AWIPS WAN provides the full connectivity AWIPS sites required to exchange data.

Use of the text workstation to produce, send, and display received administrative messages is described in the AWIPS CAVE-D2D User’s Manual.

All products sent to the default NCF address may be disseminated over the Satellite Broadcast Network (SBN), the National Digital Forecast Database (NDFD), NWSTG, the National Weather Wire Service (NWWS), ALL (via WAN), and/or the Radar Multicast, depending on the product. The default address table at the NCF determines what products are forwarded where, based on the WMO header. Currently, all warning products from the NWSTG are transmitted on both the SBN and the WAN.

In the event of an SBN failure, a selected set of products, defined by NWS Headquarters, will be transmitted via the WAN until the SBN is back up. The current set of products is provided in Appendix H.

### 14.2.1 MHS Environment Variables, Directories, and Configuration Files

The MHS data is configured and installed as part of the AWIPS install (or upgrade) procedure.

#### 14.2.1.1 MHS Environment Variables and Values (on all DX1 as user fxa)

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$PROJECT</td>
<td>/awips/ops</td>
</tr>
<tr>
<td>$PATH</td>
<td>/awips/ops/bin:$PATH</td>
</tr>
<tr>
<td>$SHLIB_PATH</td>
<td>/awips/ops/sharedlib</td>
</tr>
<tr>
<td>$FXA_DATA</td>
<td>/data/fxa</td>
</tr>
</tbody>
</table>
14.2.1.2 MHS Administrative Directories

`$FXA_DATA/mhs`
`$FXA_DATA/mhs/enclosures/outfiles`

The **outfiles** directory is the only directory actively used, and it is the location for user-created attachments (enclosures) for outgoing administrative messages. Users may either create attachments or copy existing attachments here in this directory.

The **enclosures** directory is scoured once a day by the fxa crontab removing products that are more than 30 days old.

14.2.1.3 MHS Configuration Files

Several configuration files dictate how a received message will be processed. These files are located in `/awips/ops/data/mhs`, on the server that hosts MHS (dx1f for field sites, mh[1-6] at ANCF and BNCF). These files are:

- `ack_handler.tbl`
- `addr_handler.tbl`
- `all_sites.data`
- `default_addr.data`
- `dfltaddr.*`
- `nack_handler.tbl`
- `rcv_handler.tbl`

14.3 MHS Receiver Processing

When a message is received by MHS, the first action taken is to attempt to resolve the destination. For messages sent to the AWIPS field sites, the specified destination address will always match the site ID of the recipient. This is rarely the case for messages received by the NCF, so additional steps must be taken to determine how to process these messages. If the destination is the same as the current site ID, then no additional resolution is needed and the message can be processed using the receive handler table (`/awips/ops/data/mhs/rcv_handler.tbl`). This file tells MHS how to process an incoming message based on the message code provided by the sender. The message code is specified using the “-c” option to message send. With the Java API, it is provided either as a parameter to the MhsMessage class constructor, or by using the setActionCode method. It is important to note that the sender must provide all message details as defined in the receive handler table, or the message may be discarded or processed incorrectly by recipients.

For each message code, the receive handler table defines the destination directory for any attachments, a handler, and the handler specification. If DEFAULT is specified as the queue directory, enclosures will be stored in the `/data/x400/msg/inbox/%MSGTYPE`
directory, where %MSGTYPE is the sender-passed message type string (“routine” by default). The four valid handlers and their descriptions are described in Table 14.3-1.

Table 14.3-1. Message Handlers

<table>
<thead>
<tr>
<th>Handler</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEFAULT</td>
<td>Save attachment(s) in the specified queue directory. Any file handling specification is ignored.</td>
</tr>
<tr>
<td>NULL</td>
<td>Discard attachments and ignore any handling specification.</td>
</tr>
<tr>
<td>SYS</td>
<td>Save attachment(s) in the specified queue directory and execute a system call as specified in the handling specification.</td>
</tr>
<tr>
<td>PIPE (pipename)</td>
<td>Save attachment(s) in the specified queue directory and write a message using the handling specification to the specified named pipe.</td>
</tr>
</tbody>
</table>

The handling specification can contain the following key words prefixed with the ‘%’ character that are taken from the MHS message as shown in Table 14.3-2.

Table 14.3-2. Key Words in Handling Specifications

<table>
<thead>
<tr>
<th>Key Word</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSGID</td>
<td>The MHS message ID</td>
</tr>
<tr>
<td>MSGTYPE</td>
<td>Sender-assigned message type (msg_send -t option)</td>
</tr>
<tr>
<td>MSGCODE</td>
<td>Sender-assigned message code (msg_send -c option)</td>
</tr>
<tr>
<td>SUBJECT</td>
<td>Sender-assigned message subject (msg_send -s option)</td>
</tr>
<tr>
<td>SENDERS</td>
<td>Sending site ID code (e.g. - Miami’s site code is MFL)</td>
</tr>
<tr>
<td>PRIORITY</td>
<td>Sender-assigned message priority (msg_send -p option)</td>
</tr>
<tr>
<td>BODY</td>
<td>Path to a file containing the contents of the message body (msg_send -b option)</td>
</tr>
<tr>
<td>ENCLIST</td>
<td>Comma-separated list of file paths for message enclosures (msg_send -e option)</td>
</tr>
<tr>
<td>ENCLOSE(N)</td>
<td>Path to file containing the Nth enclosure</td>
</tr>
<tr>
<td>TOTENC</td>
<td>Total number of enclosures</td>
</tr>
<tr>
<td>X400DOC</td>
<td>Path to X.400 document file</td>
</tr>
</tbody>
</table>

The following is an example entry in the receive handler table.

```
42 /data/fxa/workFiles/nwws/rcvq
PIPE(/data/fxa/workFiles/nwws/nwws_mhs_pipe) \
%SUBJECT %ENCLOSE(1)
```

This line is for processing a message assigned code 42, which is for weather wire (NWWS) products at field sites that have NWWS uplink systems.

MHS will save attachments to /data/fxa/workFiles/nwws/rcvq. The attachments will have a file name matching the MHS message ID assigned by the originating site when it was submitted and an extension of a three-digit numeric string representing the attachment number in the order specified. So the first attachment for message ABQ-123456 would be named ABQ-123456.001 and would be stored in /data/fxa/workFiles/nwws/rcvq. The PIPE handler indicates that a message will be written to the named pipe.
Chapter 14 Message Handling System

/data/fxa/workFiles/nwws/nwws_mhs_pipe containing the string that was passed as a subject and the path to the first enclosure. For this to function properly, a separate application must be reading from the specified pipe; if it is not, the operation will fail. The user can also see that a message subject and a single attachment must be provided by the sender.

14.4 Address Expansion

When the specified destination address does not match the current site ID, MHS will attempt to perform address expansion to translate the address, or it will forward the entire message to a different location using the following rules. First, MHS tries to match the specified address with a special address listed in the address handler table (addr_handler.tbl). For each special address, this file specifies a site ID, a queue directory, a handler, and a handling specification. The handler and the handling specification are the same as described in section 14.3 for the receive handler table. The site ID replaces the special address. The queue directory is a relative path under /data/mhs. Some of the configuration lines in the address handler table are complex and require additional explanation. The special address DEFAULTNCF has the following specification:

```
DEFAULTNCF    ANCF    DEFAULT    FORWARD (DEFAULT)
```

This means that a message addressed to DEFAULTNCF will have its address temporarily changed to ANCF. The message and any attachments will be written to the default directory for the message type (/data/mhs/inbox) and named using the original message ID. The “FORWARD (DEFAULT)” specification indicates that the address expansion will be performed with the default address table (default_addr.data) using the product ID assigned by the sender. MHS compares every line in the file with the product ID. Address expansion is cumulative. For instance, a product ID of WOUS99 matches the following expressions in the default address table:

```
W* | ALL
*  | NWSTG, SBN
```

The result is that the destination address is expanded to “ALL,NWSTG,SBN”.

A product ID of SDUS75 matches:

```
*  | NWSTG, SBN
SDUS? * | INET, !NWSTG
```

A ‘!’ in front of a destination removes it. The !NWSTG address in the second line negates the NWSTG in the first. The resulting expanded address becomes “SBN,INET”.

The addresses like ALL, SBN, NWSTG, and INET are resolved by passing each expanded address through the address handler table a second time. The special addresses SBN, NWSTG, INET, and NDFD all result in message data getting written to special named pipes for final processing. Each of these pipes is monitored by a separate instance.
of the dissem_dist daemon, which will disseminate the message according to rules in its associated configuration files (dfltaddr.*).

The NWWS special address results in the program “process_nwws” being executed to send the product to the weather wire interface.

ALL is a special symbol that is remapped to ALLNCF for resolution in the address handler table. It represents a distribution list defined in the file all_sites.data and results in the message being transmitted by the message request daemon (msgreq_svr) to every site in the list except for the originating site, which is removed from the list automatically.

One unusually complex rule to note is the one used for service backup. These messages are addressed to DEFAULTNCF with message code 21 and have a product ID of SVCBKPIFP. Using the rules already defined in this section, the default address table rule for DEFAULTNCF forces address expansion of SVCBKPIFP, resulting in a final address of ANCF. Because the address matches the site ID, the message code 21 specification in the receive handler table is used to process the message.

14.4.1 NCF Administrative Messages

The NCF sends administrative messages to notify sites about outages, late model runs, or other important information. Sites should have at least one workstation alarmed for these messages. AWIPS administrative messages may be sent as either urgent messages or routine messages. Messages may be set to alarm using the old AFOS PIL.

- **ADA**: URGENT administrative messages have the following WMO header and message ID:
  - WMO header: NOUS71 KNCF
  - Message ID for Alarm: NCFADANCF

- **ADM**: Routine administrative messages have the following WMO header and message ID:
  - WMO header: NOUS72 KNCF
  - Message ID for Alarm: NCFADMNCF

Whenever possible, the NCF will advise all or specific sites, NWSTG, and NCEP in advance of activities that might affect service or communications. When the NCF becomes aware of problems that could have an adverse impact on data receipt or site operations, it will transmit advisories to affected sites, including NWS NRS sites, as well as commercial and private entities that have NRS-like capability.

The NCF may send an AWIPS administrative message for outages like the following:

- Circuits to or from the NCF
- MGS
- GINI
14.4.1.1 Troubleshooting and Monitoring Commands

14.4.1.1.1 Troubleshooting

Log files are generally the first things to check when you want to see if the system is functioning properly. The log files are useful for two reasons:

- You can tell if the MHS processes are currently sending and receiving messages. The processes all write information to their log files about the messages they handle. If the log files have been updated recently, then you know that the system is still running.

- You can tell if any of the processes are having problems, because the MHS processes will write error messages to their log files.

Please check the `msgreq_svr.log` (located at awips/ops/logs/dx1-<site>/msgreq_svr.log) and the `msgrcv_svr.log` (located at awips/ops/logs/dx1-<site>/msgrcv_svr.log) with the new transactions and the MhsServerRequest log located in /data/logs/fxa/<current day> folder for updates. E.g.: MhsRequestServer2032dx1-tbw3000509

Examples for the above log files follow.

**msgrcv_svr.log**

```
TBW3-5026: NTXX98 KTBW 221013 (/data/mhs/smtp_recv/output/m000006.doc)
Wed Sep 22 12:28:54 2010 <msgrcv_svr: 9658> Discard OK
TBW3-5024: Routine to=TBW3, from=TBW3, pri=0 code=42 enc=1 len=69 dly=(0+9952+0)s
TBW3-5024: NTXX98 KTBW 220943 (/data/mhs/smtp_recv/output/m000007.doc)
Wed Sep 22 12:28:54 2010 <msgrcv_svr: 9658> Discard OK
TBW3-5023: Routine to=TBW3, from=TBW3, pri=0 code=42 enc=1 len=69 dly=(0+10853+0)s
```

**msgreq_svr.log**

```
Wed Sep 22 03:43:02 2010 <msgreq_svr:11236> OK request
TBW3-4992: Routine to=TBW3, from=TBW3, pri=0 code=42 enc=1 len=69
TBW3-4992: NTXX98 KTBW 220343 (/data/mhs/msgtbl/TBW3-4992.msg)
Wed Sep 22 03:52:18 2010 <msgreq_svr:11253> OK request
TBW3-4993: Retransmit Request to=RETRANS, from=TBW3, pri=0 code=0 enc=1 len=373
TBW3-4993: <no_prodid> (/data/mhs/msgtbl/TBW3-4993.msg)
```
14.4.1.1.2 Monitoring Commands

- **msg_stats**

  TYPE: cd /awips/ops/bin

  TYPE: msg_stats

This program displays general information on the status of the processes, and updates the display every few seconds.

The display is divided into three sections: Message Item Information; Process Information; and detailed information on the first and last messages in each of the three queues.

<table>
<thead>
<tr>
<th>MHS Process</th>
<th>PID</th>
<th>QLen</th>
<th>State</th>
<th>Last MsgID</th>
<th>Count</th>
<th>Start_Up</th>
<th>Last_Msg</th>
</tr>
</thead>
<tbody>
<tr>
<td>msgreq_svr</td>
<td>11233</td>
<td>0</td>
<td>WAIT MSG</td>
<td>TBW3-5063</td>
<td>332</td>
<td>21:12:42</td>
<td>17:58:02</td>
</tr>
<tr>
<td>mdp_daemon</td>
<td>11229</td>
<td>0 I</td>
<td>WAIT MSG</td>
<td>TBW3-5063</td>
<td>332</td>
<td>21:12:42</td>
<td>17:58:02</td>
</tr>
<tr>
<td>smtp_send</td>
<td>11230</td>
<td>0 I</td>
<td>WAIT MSG</td>
<td>TBW3-5063</td>
<td>332</td>
<td>21:12:42</td>
<td>17:58:02</td>
</tr>
<tr>
<td>sendmail</td>
<td>19581</td>
<td>-1 I UP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```
$ msg_stats
```

14.5 Site Level Illustration

Site Level data flow diagrams for MHS products are presented in Exhibits 14.5-1 and 14.5-2. Descriptions of the various steps in the flows follow each exhibit.
Exhibit 14.5-1. Receive Distributed Product and Send Acknowledgment

As shown in Exhibit 14.5-1,

1. An SMTP session completes between sendmail instances at a remote location and the local site. A mail message is written to the awipsmhs mailbox file in /var/spool/mail.

2. The SMTP Receive Daemon (smtp_recv) retrieves the message from the awipsmhs mailbox, strips off the SMTP envelope, and extracts the message.

3. The SMTP Receive Daemon stores the message to the local disk.

4. The SMTP Receive Daemon places the message handle and type of message in the Message Priority Queue Shared Memory.

5. The Message Receive Dispatcher (mhd_daemon) retrieves the message handle and message type from the Message Priority Queue Shared Memory.

6. The Message Receive Dispatcher determines that it is a product message and sends a transaction to the Message Receive Server (msgrecv_svr) to process the product message.

7. The Message Receive Server creates a Message Receive Service to handle the message.

8. The Message Receive Service reads the message header information from the Message Priority Queue Share Memory to determine the message type, filename(s), sending site, and whether an acknowledgment is required.

9. The Message Receive Service performs the action specified in the receive handler table (rcv_handler.tbl) by the message code.

10. If requested by the sender, the Message Receive Service queues an acknowledgment to the Message Priority Queue Shared Memory for processing.
11. The Message Send Dispatcher (*mdp_daemon*) dequeues a message handler from the Message Priority Queue Shared Memory for processing.

12. The Message Send Dispatcher sends the message handle to the SMTP Submission Application (*smtp_send*).

13. The SMTP Submission Application uses the message handle to access the message header information in the Message Priority Queue Shared Memory.

14. The SMTP Submission Application submits the SMTP message to *sendmail*.

15. The *sendmail* application establishes an SMTP session with all required remote mail servers.

16. The *sendmail* application transmits the message across the WAN.

---

Exhibit 14.5-2. Distribute Product and Receive Acknowledgement

As shown in Exhibit 14.5-2,

1. The user selects a message to be sent over MHS via D2D through the Text Window Tool user interface from an XT or LX workstation.

2. The Text Window Tool queues the message for the MHS Server (*MhsServer*).
3. The MHS Server forks and executes a `msg_send` command with all required parameters to queue the message for MHS.
4. The `msg_send` application retrieves the product data from any specified disk files.
5. The `msg_send` application accepts the product distribution request parameters from the user and sends them to the Message Request Server (`msgreq_svr`) as a transaction.
6. The Message Request Server creates a Message Request Service to handle the request.
7. The `msg_send` application writes the product data across the socket to the Message Request Service.
8. The Message Request Service stores the message data into disk files in a predefined directory.
9. The Message Request Service writes the request handle to the Message Priority Queue Shared Memory, along with message header information such as the message type.
10. The Message Send Dispatcher (`mdp_daemon`) determines the message handle of the next message to be dispatched from the Message Priority Queue Shared Memory.
11. The Message Send Dispatcher sends the message handle to the SMTP submission application (`smtp_send`).
12. The SMTP Submission Application uses the message handle to obtain the message header information from the Message Priority Queue Shared Memory.
13. The SMTP Submission Application retrieves the message data contained in the local disk files and builds an SMTP mail message.
14. The SMTP Submission Application submits the SMTP message to sendmail.
15. The local sendmail application establishes remote SMTP sessions to route the message across the WAN.
16. When the SMTP message is successfully delivered to the remote site, an SMTP message containing the acknowledgment is returned to the local `sendmail` and is written to the `awipsmhs` user's mailbox in `/var/spool/mail`.
17. The SMTP Receive Daemon (`smtp_recv`) reads the message from the `awipsmhs` mailbox, strips off the envelope, and extracts the acknowledgment message.
18. The SMTP Receive Daemon stores the acknowledgment message to disk.
19. The SMTP Receive Daemon places the message request handle and its type in the Message Priority Queue Shared Memory.
20. The Message Receive Dispatcher (`mhd_daemon`) retrieves the message request handle and message type of the next incoming message to be processed from the Message Priority Queue Shared Memory.
21. The Message Receive Dispatcher sends a transaction to the Message Receive Server (`msgrcv_svr`) to process the newly arrived message.
22. The Message Receive Server creates a Message Receive Service to handle the message.

23. The Message Receive Service reads the message header information from the Message Priority Queue Shared Memory to determine which message is being acknowledged. The message being acknowledged is marked as having been acknowledged by the site that sent the acknowledgment. If all sites required to acknowledge the message have acknowledged, then the message entry is deleted from the Message Priority Queue Shared Memory.

24. The Message Receive Service moves the message files to the acknowledgment message inbox directory. If an acknowledgment for the message has been received, the message data files that were written by the Message Receive Service are removed.
Chapter 15
Localization
Chapter 15. Localization

Table of Contents

15.1 Conventions Applied to This Chapter ................................................................. 2
  15.1.1 General Conventions .................................................................................. 2
  15.1.2 Variable Conventions ............................................................................... 2
15.2 Radar Localization............................................................................................... 3
  15.2.1 Changing Radar Menus .............................................................................. 3
  15.2.2 Adding or Removing an RPG (Access Configuration Files) ....................... 6
    15.2.2.1 RCM config.xml File ........................................................................ 6
15.3 WarnGen Localization.......................................................................................... 10
  15.3.1 Modifying Default WarnGen GUI Configuration .................................... 10
  15.3.2 Product Templates for WarnGen .......................................................... 13
  15.3.3 WarnGen Templates ............................................................................... 15
    15.3.3.1 Auxiliary WarnGen Template Files ........................................ 17
  15.3.4 Configuration .......................................................................................... 18
    15.3.4.1 Configuration Files (.xml) ............................................................ 18
    15.3.4.1.1 Velocity Templates ............................................................... 26
15.4 Other Menu Localizations .................................................................................. 30
  15.4.1 Upper Air Menus ...................................................................................... 30
15.5 Modifying Shapefiles and Area Files .............................................................. 31
15.6 How to Download Shapefile Data ..................................................................... 34
15.7 Load and Activate the New Data on AWIPS ................................................... 36
15.8 Downloading Local Shapefiles .......................................................................... 36

List of Exhibits

Exhibit 15-1. Localization in AWIPS II .................................................................. 1
Exhibit 15.3-1. WarnGen Flow Chart...................................................................... 10

1 This chapter is a work in progress. For now, please refer to the AWIPS II Site Migration Guide for additional guidance on Localization.
### List of Tables

Table 15.1.2-1. Document Variable Conventions ................................................................. 3  
Table 15.2.1-1. Sections of the radarsInUse.txt File .............................................................. 5  
Table 15.2.2-1. RPG Access Configuration Files ................................................................. 6  
Table 15.2.2.1-1. Elements of the config.xml File ................................................................. 6  
Table 15.3.1-1. Localization Fields ...................................................................................... 12  
Table 15.3.3-1. WarnGen Severe Weather Product Templates ........................................... 15  
Table 15.3.3-2. WarnGen Marine Weather Product Templates .......................................... 15  
Table 15.3.3-3. WarnGen Hydrologic Product Templates ................................................. 15  
Table 15.3.3-4. WarnGen Non-Warning Product Templates .............................................. 16  
Table 15.3.4.1-1. Field Values ............................................................................................ 18  
Table 15.5.1. Shapefiles ........................................................................................................ 32  
Table 15.5-2. Zone Attributes ............................................................................................. 34  
Table 15.6-1. Shapefile Download Commands .................................................................... 35
Localization adapts (i.e., configures) the AWIPS national baseline software to the unique data and display requirements of the site. AWIPS II performs localization dynamically at system startup using data from the localization data environment.

This chapter serves as a guide to changing specific files in the localization environment to configure an EDEX installation to a specific site, and as a guide to adding localization files needed for launching and running client applications such as CAVE and AlertViz. It focuses on a SERVER-CLIENT localization paradigm, much like that which exists on AWIPS I. The steps taken should be sufficient to guide a person familiar with AWIPS I systems through the creation of a usable AWIPS II localization for an EDEX and CAVE instance.

Localization in AWIPS II is done by the EDEX process itself, as opposed to a script (mainScript.csh) from AWIPS I. Based on specific configuration files, the EDEX process produces output files, which are transferred, or downloaded, to the CAVE client when CAVE starts and whenever a change to a file has been detected. Exhibit15-1 describes the flow of configuration on the left, the localization engine (EDEX) in yellow (center), the output of that localization in orange (top right), and the usage of those localization files via download to a user’s caveData directory on a workstation in red (bottom right).

Exhibit 15-1. Localization in AWIPS II
This chapter focuses on creating the configuration files necessary for EDEX to correctly create the output files needed for proper baseline operations.

**General Note:** Never edit files that are in a base directory tree. Although there are a few exceptions as of the date of this document’s creation, anything beneath a directory named base in the configuration tree should never be edited. This is analogous to the nationalData directory in AWIPS I.

### 15.1 Conventions Applied to This Chapter

#### 15.1.1 General Conventions

When specific commands are being displayed to type, the following will be used as the convention. The user should type on the screen everything following the `#` symbol.

For example:

```bash
# ssh dx1
```

This indicates the user should secure shell into dx1 (`ssh dx1`).

At points in this chapter, it is necessary to reference a command that could be used for multiple iterations to achieve the same effect. The double bracket will enclose a description of what the user is to input in place of the notated reference.

For example:

```
[[ database name ]]
```

This indicates the user should run the command for as many iterations of database name as needed to complete the desired configuration.

When a substitution needs to be made in a command that is to be interpreted by the user, the following syntax will be used:

```
<< Site ID >>
```

This indicates that the user should interpret the correct site identifier to put into the command.

When specific paths are shown, they will be given a background shade (as in the example that follows):

```
/awips2/edex
```

#### 15.1.2 Variable Conventions

Table 15.1.2-1 defines a list of variables referenced in this chapter, provides a description of each variable, and identifies the current location of each variable as of the date of this publication. Wherever a variable occurs in this chapter, the actual value should be substituted when typing the commands on an AWIPS system. *These are not environment
variables at this point in time, so manual substitution should be made when typing commands that contain references to these variables.

Table 15.1.2-1. Document Variable Conventions

<table>
<thead>
<tr>
<th>Environment Variable</th>
<th>Description</th>
<th>Default Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>$CAVE</td>
<td>The hostname of a CAVE workstation.</td>
<td>lx1, lx2, lx3, etc....</td>
</tr>
<tr>
<td>$DB_SERVER</td>
<td>The hostname of the server running the Postgres database engine.</td>
<td>dx1f</td>
</tr>
<tr>
<td>$EDEX1</td>
<td>The primary EDEX servers.</td>
<td>dx3</td>
</tr>
<tr>
<td>$EDEX_CONFIG</td>
<td>The location in which the configuration files on the EDEX servers reside.</td>
<td>/awips2/edex/data/utility</td>
</tr>
<tr>
<td>$EDEX_DB_HOME</td>
<td>The location of the database files that control the functionality of the PostgreSQL engine, as well as the physical storage for the data held within the tables.</td>
<td>/awips2/data [DX1 and DX2]</td>
</tr>
<tr>
<td>$EDEX_HOME</td>
<td>The home location for the EDEX JVM.</td>
<td>/awips2/edex</td>
</tr>
<tr>
<td>$LDM_DOWNSTREAM</td>
<td>The hostname of the server running the downstream LDM client.</td>
<td>dx2f</td>
</tr>
<tr>
<td>$LDM_HOST</td>
<td>The hostname of the server running the upstream/feeder LDM host.</td>
<td>cpsbn1/cpsbn2</td>
</tr>
<tr>
<td>$RCM_INSTALL_DIR</td>
<td>The location in which the AWIPS II RadarServer is installed.</td>
<td>/awips2/rcm</td>
</tr>
<tr>
<td>$RCM_SERVER</td>
<td>The hostname of the server running the AWIPS II RadarServer.</td>
<td>dx1f</td>
</tr>
</tbody>
</table>

**Note:** As of the date this chapter was produced, there is no plan to have these be system environment variables on an AWIPS or ADAM system. The location should be known to the user following the steps outlined here. Normally the value under the expected location is that to which an AWIPS site or ADAM platform would adhere, with the exception of the server and cave platforms. Those, for an ADAM platform only, would all be adam1.

### 15.2 Radar Localization

EDEX dynamically creates the files needed for the CAVE Radar, from specific input files in the common utility configuration directory tree.

#### 15.2.1 Changing Radar Menus

During site migration, the config_awips2.sh took information from AWIPS I input files and created a file that used by EDEX to localize the radar menus in CAVE. This file is named `radarsInUse.txt` and it resides in the following location:

`/awips2/edex/data/utility/common_static/site/LLL/radar`

(where LLL is the AWIPS ID of your localization site)

This location is accessible on your EDEX servers (dx3, dx4 for WFOs, dx3, dx4, dx5, and dx6 for NCEP).
An example of the file follows:

    # DO NOT EDIT LINES BEGINNING WITH '##'
    # LOCAL_RADARS (including terminal) - MUST HAVE THIS LINE
    kfdr
    ktlx
    kvnx
    tokc
    koun

    # DIAL_RADARS - MUST HAVE THIS LINE
    kinx
    kddc
    kict
    ksrx
    kfws
    kdyx
    klbb
    kama
    xcyr
    xrsr
    xlwe
    xsao

    # ASR_RADARS - MUST HAVE THIS LINE

    # ARSR_RADARS - MUST HAVE THIS LINE

    # MOSAIC_RADARS - MUST HAVE THIS LINE
    ktlx
    kvnx
    kfdr
    kddc
    kict
    kinx
    kdyx
    kfws
    kama
As noted, the lines beginning with '#' are necessary, as the code uses these to parse through the file.

Table 15.2.1-1 describes the sections of the radarsInUse.txt file.

### Table 15.2.1-1. Sections of the radarsInUse.txt File

<table>
<thead>
<tr>
<th>Section Divider</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOCAL_RADARS (including terminal) -</td>
<td>For each radar this divider finds after this line, and before the next section divider, it creates a contribution to the CAVE menu which shows that radar on the main CAVE menu across the top of the D2D perspective.</td>
</tr>
<tr>
<td>MUST HAVE THIS LINE</td>
<td></td>
</tr>
<tr>
<td>DIAL_RADARS - MUST HAVE THIS LINE</td>
<td>For each radar this divider finds after this line, and before the next section divider, it creates a contribution to the CAVE menu under the Radar --&gt; Dial Radars submenu structure. There will be a submenu for each radar.</td>
</tr>
<tr>
<td>ASR_RADARS - MUST HAVE THIS LINE</td>
<td>For each radar this divider finds after this line, and before the next section divider, it creates a contribution to the CAVE menu under the Radar --&gt; ASR-11 Radar submenu structure. There will be a submenu for each radar.</td>
</tr>
<tr>
<td>ARSR_RADARS - MUST HAVE THIS LINE</td>
<td>For each radar this divider finds after this line, and before the next section divider, it creates a contribution to the CAVE menu under the Radar --&gt; ARSR-4 Radar submenu structure. There will be a submenu for each radar.</td>
</tr>
<tr>
<td>MOSAIC_RADARS - MUST HAVE THIS LINE</td>
<td>For each radar this divider finds after this line, and before the end of the file, it will include the radar in the default mosaic when selecting a product under the ---- Mosaic ---- selections in the Radar menu of the D2D perspective in CAVE.</td>
</tr>
</tbody>
</table>

This file is not available via the localization perspective and must be edited from an EDEX server. This means that the maintainer must have root access in order to gain access to user awips.

To make a change to the radar menus, perform the following steps:

1. Log in to an EDEX server (dx3 or dx4 at a WFO/RFC, dx3-dx6 at NCEP) as user root.
   
   ```
   ssh root@dx3
   ```

2. Switch to user awips.
   
   ```
   su - awips
   ```

3. Navigate to the site-level radar directory for the CAVE localization you are attempting to affect. In the example below, it is noted as LLL. Replace LLL with the appropriate AWIPS ID when typing.
   
   ```
   cd /awips2/edex/data/utility/common_static/LLL/radar
   ```

4. Add or remove radars to the appropriate section using Table 15.2.1-1 as a guide as to how to affect the menus.
   
   ```
   vi radarsInUse.txt
   ```

5. Save and close the file.
   
   ```
   :wq!
   ```
6. Launch a new CAVE.

15.2.2 Adding or Removing an RPG (Access Configuration Files)

Configuration files that control which RPGs are usable by the active site are now in /awips2/rcm on dx1 and dx2 where the RPG Configuration Manager (RCM) JVM will run along with the a2dx1apps package. When adding data to these files, the site should make sure that the Radar Operations Center (ROC) is aware of these additions so that there is no RPG contention, especially during severe weather. When deleting from these files, please be aware that you may be completely disabling that radar from your ingest (SBN, dial, and dedicated).

See Table 15.2.2-1 for brief descriptions of the of the RPG access configuration files.

Table 15.2.2-1. RPG Access Configuration Files

<table>
<thead>
<tr>
<th>Location</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/awips2/rcm/data/config/drop-ins</td>
<td>Directory for RPS lists, both configured and baseline rps-RPGOP* files. Also holds prodList.txt, and wmoSiteInfo.txt</td>
</tr>
<tr>
<td>/awips2/rcm/data/config/persist</td>
<td>Directory for configuration files that affect how RCM runs. These include ActiveMultipleRequests.xml and config.xml</td>
</tr>
</tbody>
</table>

15.2.2.1 RCM config.xml File

The AWIPS II RCM config.xml file is the main configuration file for how the RCM process runs. Initially, it is created with the importAwips1 script, which parses the multiple AWIPS I files and creates the one config.xml for AWIPS II. There should be no need to edit this file manually. However, if there is need, a text editor (like vi) must be used. Table 15.2.2.1.1 describes portions of the config.xml for information purposes only.

Table 15.2.2.1-1. Elements of the config.xml File

<table>
<thead>
<tr>
<th>Element</th>
<th>Sub-Elements</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;config&gt;</td>
<td>&lt;pupID&gt;</td>
<td>Main configuration element in the xml file.</td>
</tr>
<tr>
<td></td>
<td>&lt;regionCode&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;wmoSiteID&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;collectionEnabled&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;tdwrCollectionLimited&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;decompressProducts&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;edexEndPoint&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;endPointConfig&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;radars&gt;</td>
<td></td>
</tr>
<tr>
<td>Element</td>
<td>Sub-Elements</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-----------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>&lt;pupID&gt;</td>
<td></td>
<td>Valid pupID assigned by the ROC and should not be randomly assigned or changed.</td>
</tr>
<tr>
<td>&lt;regionCode&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;wmoSiteID&gt;</td>
<td></td>
<td>Valid AWIPS site ID for WMO header on radar products sent via WAN for SBN uplink.</td>
</tr>
<tr>
<td>&lt;collectionEnabled&gt;</td>
<td></td>
<td>Master enable token that indicates whether this site is a national reporter for any configured radar and the products should be sent out over the WAN. Also affects related behavior like adding the national RPS list to the local RPS list.</td>
</tr>
<tr>
<td>&lt;tdwrCollectionLimited&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;decompressProducts&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;edexEndpoint&gt;</td>
<td></td>
<td>This tag is no longer utilized.</td>
</tr>
<tr>
<td>&lt;endPointConfig&gt;</td>
<td>&lt;archiveRoot&gt;</td>
<td>Configured end point for data delivery within the AWIPS II system architecture.</td>
</tr>
<tr>
<td></td>
<td>&lt;connectionURL&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;topic&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;archiveRoot&gt;</td>
<td></td>
<td>Directory where products are written after processing from RPG. Default: /data_store/radar.</td>
</tr>
<tr>
<td>&lt;connectionURL&gt;</td>
<td></td>
<td>Connection URL for qpidd running on the site. Should not be changed.</td>
</tr>
<tr>
<td>&lt;topic&gt;</td>
<td></td>
<td>QPID topic to populate once raw file is written to disk. Default: radarserver.dropbox</td>
</tr>
<tr>
<td>&lt;radars&gt;</td>
<td>&lt;radar&gt;</td>
<td>High level element which holds individual &lt;radar&gt; elements that configure specific connections.</td>
</tr>
<tr>
<td>&lt;radar&gt;</td>
<td>&lt;radarID&gt;</td>
<td>Beginning of element which defines a specific RPG connection using the sub-tags defined in the previous column.</td>
</tr>
<tr>
<td></td>
<td>&lt;nexradID&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;enabled&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;dedicated&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;collectionEnabled&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;sendEnvironmentalData&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;linkType&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;links&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;productAvailabilityFieldUsable&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;radarID&gt;</td>
<td></td>
<td>Four letter alpha character identifier for the radar (e.g., klwx).</td>
</tr>
</tbody>
</table>
### Chapter 15 Localization

#### Element | Sub-Elements | Description
---|---|---
<nexradID> |  | The unique nexrad ID of the radar. Assigned by the ROC.
<enabled> |  | Whether the link is enabled or disabled. Values are **true** or **false**.
<dedicated> |  | Is this a dedicated connection, or used just for OTR? Values are **true** or **false**.
<collectionEnabled> |  | Indicates whether products collected from this radar ID should be sent out over the WAN. Also affects related behavior like adding the national RPS list to the local RPS list.
<sendEnvironmentalData> |  | Whether or not to send environmental data to the RPG.
<linkType> |  | Should always be TCP_WAN.
<links> | <link> | Starts a set of connections for his radar.
<link> | <dedicated> | True or false. Dedicated or Dial.
<linkAddress> |  | IP address including port for the link, e.g., 165.92.109.75:4489.
<linkIndex> |  | Link index for the connection, e.g., 25.
<linkType> |  | Should always be TCP_WAN.
<maxRpsListSize> |  | Maximum number of products to send in an RPS list to the radar.
<portPassword> |  | Password for the link.
<tcmPassword> |  | TCM access password for the link.
<userPassword> |  | Class 2 user password.
<productAvailabilityFieldUsable> |  | **True** or **False**. Configures whether or not the RCM is to pay attention to the status flag sent by the RPG as to product availability. Should always be false for non-live RPG feeds.
**Example config.xml**

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<config>
  <pupID>999</pupID>
  <regionCode>3</regionCode>
  <wmoSiteID>KXXX</wmoSiteID>
  <collectionEnabled>true</collectionEnabled>
  <tdwrCollectionLimited>true</tdwrCollectionLimited>
  <decompressProducts>true</decompressProducts>
  <edexEndpoint>/awips2/edex/data/sbn/radar</edexEndpoint>
  <endpointConfig>
    <archiveRoot>/data_store/radar</archiveRoot>
    <connectionURL>amqp://guest:guest@/edex?brokerlist='tcp://cp1f:5672'</connectionURL>
    <topic>radarserver.dropbox</topic>
  </endpointConfig>
  <radars>
    <radar>
      <radarID>kxxx</radarID>
      <nexradID>519</nexradID>
      <enabled>true</enabled>
      <dedicated>true</dedicated>
      <collectionEnabled>true</collectionEnabled>
      <sendEnvironmentalData>true</sendEnvironmentalData>
      <linkType>TCP_WAN</linkType>
      <links>
        <link>
          <dedicated>true</dedicated>
          <linkAddress>165.92.999.99:4489</linkAddress>
          <linkIndex>25</linkIndex>
          <linkType>TCP_WAN</linkType>
          <maxRpsListSize>150</maxRpsListSize>
          <portPassword>TCPG</portPassword>
          <tcmPassword>passwd</tcmPassword>
          <userPassword>XXXWIP</userPassword>
        </link>
      </links>
    </radar>
  </radars>
</config>
```
15.3 WarnGen Localization

This section provides information relating to WarnGen localization activities. Like other areas of the software, WarnGen is localized by using configuration files on the EDEX server. These files affect the look and functionality of the application when run through CAVE. Exhibit 15.3-1 illustrates the Flow Chart for WarnGen.

Exhibit 15.3-1. WarnGen Flow Chart

15.3.1 Modifying Default WarnGen GUI Configuration

The default WarnGen configuration file was originally configured using the config_awips2.sh wg option during site migration. It took values from the wwaConfig.txt in AWIPS I to create the entries in the AWIPS II config.xml file. It now can be modified...
from the Localization Perspective within CAVE. Users who should have permission to modify this file should be granted the proper permissions in the roles.xml file, which resides in /awips2/edex/data/utility/edex_static/site/LLL/roles

```xml
<role
  roleID="com.raytheon.localization.site/common_static/warngen/config.xml">
  <user id="kevinj"/>
</role>
```

With this role setup, those users with permissions can open the SITE-level config.xml in the following location in the localization perspective.

**D2D → Warngen → config.xml**

![Image of config.xml file]

From this point, locate the file config.xml and expand it by clicking in the ▶

Double click on the SITE (LLL) entry to enter the editor.
Use Table 15.3.1-1 to find the field that needs to be localized and change the value.

Table 15.3.1-1. Localization Fields

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;warngenOfficeShort&gt;</td>
<td>The office description included in the heading of the warning.</td>
</tr>
<tr>
<td>&lt;warngenOfficeLoc&gt;</td>
<td>The short office description included in the heading of the warning.</td>
</tr>
<tr>
<td>&lt;backupCWAs&gt;</td>
<td>A comma-separated list of sites that appear in the Backup pull-down menu in</td>
</tr>
<tr>
<td></td>
<td>the WarnGen GUI, including a description of the site to be included in the</td>
</tr>
<tr>
<td></td>
<td>template:</td>
</tr>
<tr>
<td></td>
<td>Format: XXX/DESCRIPTION</td>
</tr>
<tr>
<td></td>
<td>Each entry for a site must include both the XXX and the DESCRIPTION,</td>
</tr>
<tr>
<td></td>
<td>where XXX is the AWIPS Site ID.</td>
</tr>
<tr>
<td>&lt;sideNode&gt;</td>
<td>Provides a default site node if one cannot be found in the afos2awips table</td>
</tr>
<tr>
<td></td>
<td>within the metadata database. This field is unused if an entry is found in</td>
</tr>
<tr>
<td></td>
<td>the table.</td>
</tr>
<tr>
<td>&lt;defaultTemplate&gt;</td>
<td>The WarnGen template identifier of the template whose radio button is</td>
</tr>
<tr>
<td></td>
<td>selected by default when the WarnGen GUI is launched. This template must</td>
</tr>
<tr>
<td></td>
<td>be listed in &lt;mainWarngenProducts&gt; tag also.</td>
</tr>
<tr>
<td>&lt;mainWarngenProducts&gt;</td>
<td>A comma-separated list of templates that appear under 'Product Type' on the</td>
</tr>
<tr>
<td></td>
<td>WarnGen GUI and are able to be selected via radio buttons</td>
</tr>
<tr>
<td></td>
<td>Format: template Title/ warngen template file name</td>
</tr>
<tr>
<td>&lt;otherWarngenProducts&gt;</td>
<td>A comma-separated list of templates that appear under the Other drop-down</td>
</tr>
<tr>
<td></td>
<td>selector in the WarnGen GUI within the 'Product Type' section.</td>
</tr>
<tr>
<td></td>
<td>Format: template Title/ warngen template file name</td>
</tr>
</tbody>
</table>
Example File:

```xml
<?xml version="1.0" encoding="ISO-8859-1" standalone="no"?>

<!--
This_software_was_developed_and_or_modified_by_Raytheon_Company,

U.S._EXPORT_CONTROLLED_TECHNICAL_DATA
This_software_product_contains_export-restricted_data_whose
export/transfer/disclosure_is_restricted_by_U.S._law._Dissemination
to_non-U.S._persons WHETHER IN THE United_States or_abroad requires
an_export_license_or_other_authorization.

Contractor_Name:________Raytheon_Company
Contractor_Address:_____6825_Pine_Street,_Suite_340
________________________Mail_Stop_B8
________________________Omaha,_NE_68106
________________________402.291.0100

See_the_AWIPS_II_Master_Rights_File_("Master_Rights_File.pdf")_for
further_licensing_information.
-->

<configuration>
<warngenOfficeShort>GRAND RAPIDS MI</warngenOfficeShort>
<warngenOfficeLoc>GRAND RAPIDS </warngenOfficeLoc>
<backupCWAs>GRR/GRAND RAPIDS,DTX/DETROIT/PONTIAC,IWX/NORTHERN
INDIA</backupCWAs>
<siteNode>ARB</siteNode>
<defaultTemplate>severeThunderstormWarning</defaultTemplate>
<mainWarngenProducts>Tornado/tornadoWarning,Severe
Thunderstorm/severeThunderstormWarning,Severe Weather
Statement/severeWeatherStatement,Significant Weather
Advisory/significantWeatherAdvisory,Flash Flood
Warning/flashFloodWarning</mainWarngenProducts>
<otherWarngenProducts>Flash Flood Statement/flashFloodWarningFollowup,Non-
Convective FFW (incl. Dam Break)/nonConvectiveFlashFloodWarning,Non-Convective
Flash Flood Statement/nonConvectiveFlashFloodWarningFollowup,Areal Flood
Warning/arealFloodWarning,Areal Flood Warning
Followup/arealFloodWarningFollowup,Areal Flood
Advisory/arealFloodAdvisory,Areal Flood Advisory
Followup/arealFloodAdvisoryFollowup,Special Weather
Statement/specialWeatherStatement,Short Term
Forecast/shortTermForecast</otherWarngenProducts>
</configuration>

15.3.2 Product Templates for WarnGen

Templates for WarnGen are housed on the EDEX servers in the following location:
The baseline templates should be sufficient to send properly formatted products using WarnGen. If the need arises to change, it can be modified by promoting a BASE level file to the SITE level file, and making the appropriate changes.

Users who should have permission to modify this file should be granted the proper permissions in the roles.xml file which resides in

/awips2/edex/data/utility/edex_static/site/LLL/roles

```xml
<role
  roleId="com.raytheon.localization.site/common_static/warngen/config.xml">
  <user id="kevinj"/>
</role>
```

With this role setup, those users with permissions can promote BASE templates to the SITE level in the following location in the localization perspective:

**D2D ➔ WarnGen**

From this point, locate the template file and expand it by clicking in the ▶. Then right click and select Copy To ➔ SITE.
### 15.3.3 WarnGen Templates

Tables 15.3.3-1 – 15.3.3-4 provide lists of WarnGen templates for AWIPS II. Equivalents for AWIPS I are also included in the tables for reference.

#### Table 15.3.3-1. WarnGen Severe Weather Product Templates

<table>
<thead>
<tr>
<th>Warning</th>
<th>AWIPS I</th>
<th>AWIPS II</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tornado Warning - PIL: TOR, e.g., STLTOREAX</td>
<td>wwa_tor.preWWA</td>
<td>tornadoWarning.vm and tornadoWarning.xml</td>
</tr>
<tr>
<td>Severe Thunderstorm Warning - PIL: SVR, e.g., STLSVREAX</td>
<td>wwa_svr.preWWA</td>
<td>severeThunderstormWarning.vm and severeThunderstormWarning.xml</td>
</tr>
<tr>
<td>Severe Weather Statement, follow-up to TOR or SVR - PIL: SVS, e.g., STLSVSEAX</td>
<td>wwa_svrwx_sta_county.preWWA</td>
<td>severeWeatherStatement.vm and severeWeatherStatement.xml</td>
</tr>
<tr>
<td>Extreme Wind Warning - PIL: EWW, e.g., MIAEWWMFL</td>
<td>wwa_eww.preWWA</td>
<td>extremeWindWarning.vm and extremeWindWarning.xml</td>
</tr>
<tr>
<td>Severe Weather Statement, follow-up to EWW - PIL: SVS, e.g., MIASVSMFL</td>
<td>wwa_eww_svs.preWWA</td>
<td>extremeWindWarningFollowup.vm and extremeWindWarningFollowup.xml</td>
</tr>
</tbody>
</table>

#### Table 15.3.3-2. WarnGen Marine Weather Product Templates

<table>
<thead>
<tr>
<th>Warning</th>
<th>AWIPS I</th>
<th>AWIPS II</th>
</tr>
</thead>
<tbody>
<tr>
<td>Special Marine Warning - PIL: SMW, e.g., BOSSMWBBOX</td>
<td>wwa_specmarine.preWWA</td>
<td>specialMarineWarning.vm and specialMarineWarning.xml</td>
</tr>
<tr>
<td>Marine Weather Statement, follow-up to SMW - PIL: MWS, e.g., BOSMWSBOX</td>
<td>AWIPS I template: wwa_mar_wx_sta.preWWA</td>
<td>specialMarineWarningFollowup.vm and specialMarineWarningFollowup.xml</td>
</tr>
</tbody>
</table>

#### Table 15.3.3-3. WarnGen Hydrologic Product Templates

<table>
<thead>
<tr>
<th>Warning</th>
<th>AWIPS I</th>
<th>AWIPS II</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flash Flood Warning (convective) - PIL: FFW, e.g., STLFFWSGF</td>
<td>wwa_ffw.preWWA</td>
<td>flashFloodWarning.vm and flashFloodWarning.xml</td>
</tr>
<tr>
<td>ALASKA: flashFloodWarning_Zones.vm and flashFloodWarning_Zones.xml</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Flash Flood Statement (follow-up to convective FFW) - PIL: FFS, e.g., STLFFSSGF</td>
<td>wwa_flflood_sta_county.preWWA</td>
<td>flashFloodWarningFollowup.vm and flashFloodWarningFollowup.xml</td>
</tr>
<tr>
<td>ALASKA: flashFloodWarningFollowup_Zones.vm and flashFloodWarningFollowup_Zones.xml</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 15.3.3-4. WarnGen Non-Warning Product Templates*

<table>
<thead>
<tr>
<th>Warning</th>
<th>AWIPS I</th>
<th>AWIPS II</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-Convective Flash Flood Warning (incl. Dam Break) - PIL: FFW, e.g., OMAFFWOAX</td>
<td><code>wwa_dam_break.preWWA</code></td>
<td><code>nonConvectiveFlashFloodWarning.vm</code> and <code>nonConvectiveFlashFloodWarning.xml</code> ALASKA: <code>nonConvectiveFlashFloodWarning_Zones.vm</code> and <code>nonConvectiveFlashFloodWarning_Zones.xml</code></td>
</tr>
<tr>
<td>Non-Convective Flash Flood Statement (incl. Dam Break) - PIL: FFS, e.g., OMAFFSOAX</td>
<td><code>wwa_fflood_sta.preWWA</code></td>
<td><code>nonConvectiveFlashFloodWarningFollowup.vm</code> and <code>nonConvectiveFlashFloodWarningFollowup.xml</code> ALASKA: <code>nonConvectiveFlashFloodWarningFollowup_Zones.vm</code> and <code>nonConvectiveFlashFloodWarningFollowup_Zones.xml</code></td>
</tr>
<tr>
<td>Areal Flood Warning - PIL: FLW, e.g., WBCFLWLWX</td>
<td><code>wwa_flood_wm.preWWA</code></td>
<td><code>arealFloodWarning.vm</code> and <code>arealFloodWarning.xml</code> ALASKA: <code>arealFloodWarning_Zones.vm</code> and <code>arealFloodWarning_Zones.xml</code></td>
</tr>
<tr>
<td>Areal Flood Statement (follow-up to FLW) - PIL: FLS, e.g., WBCFLSLWX</td>
<td><code>wwa_flood_sta.preWWA</code></td>
<td><code>arealFloodWarningFollowup.vm</code> and <code>arealFloodWarningFollowup.xml</code> ALASKA: <code>arealFloodWarningFollowup_Zones.vm</code> and <code>arealFloodWarningFollowup_Zones.xml</code></td>
</tr>
<tr>
<td>Areal Flood Advisory (events below severe criteria) - PIL: FLS, e.g., STLFLSSGF</td>
<td><code>wwa_flood_adv.preWWA</code></td>
<td><code>arealFloodAdvisory.vm</code> and <code>arealFloodAdvisory.xml</code> ALASKA: <code>arealFloodAdvisory_Zones.vm</code> and <code>arealFloodAdvisory_Zones.xml</code></td>
</tr>
<tr>
<td>Areal Flood Advisory Statement (follow-up to areal flood advisory) - PIL: FLS, e.g., STLFLSSGF</td>
<td><code>wwa_flood_adv_sta.preWWA</code></td>
<td><code>arealFloodAdvisoryFollowup.vm</code> and <code>arealFloodAdvisoryFollowup.xml</code> ALASKA: <code>arealFloodAdvisoryFollowup_Zones.vm</code> and <code>arealFloodAdvisoryFollowup_Zones.xml</code></td>
</tr>
</tbody>
</table>

*These are non-warning products, which can be used; however, WFOs are not mandated to use them.*

---

**Warning**

[72x746]Chapter 15 Localization

– Hard copies uncontrolled. Verify effective date prior to use. –
15.3.3.1 Auxiliary WarnGen Template Files

A list of the Auxiliary WarnGen Template files follows. You will likely note additional files in the AWIPS II WarnGen directory, some of which may be edited. Others are part of the baseline.

The baseline files (found in $EDEX_HOME/data/utility/common_static/base/warngen) are:

- **countyTypes.txt.** A translational table to create proper county/zone/independent city wording.
- **states.txt.** A translational table of state abbreviations.
- **immediateCauses.txt.** A translational table of hydro VTEC immediate causes.
- **VM_global_library.** A Velocity template repository houses functions that are used repeatedly through the WarnGen templates. If you find a #functionname(args) like syntax and cannot figure out what it does, chances are that the referenced function is contained here.

The editable files (found in $EDEX_HOME/data/utility/common_static/site/XXX/warngen) are:

- **dupCounties.vm.** An import file to assist those sites that have same-named counties across two states. Similar to a patch placed in AWIPS I to produce correct text output.
- **damInfo.vm.** Created to assist users in creating local dam information for the non-convective Flash Flood Warning/Statement?.
- **damInfoBullet.xml.** Same as damInfo.vm, except that it is used to create bullets that are displayed in the WarnGen GUI through the accompanying non-convective Flash Flood Warning/Statement? XML files.
- **damInfoBulletName.xml.** Same as damInfoBullet.xml, but used to create bullets that are displayed in the WarnGen GUI through the accompanying non-convective Flash Flood Warning/Statement? XML files.
- **mileMarkers.xml.** An XML file that creates WarnGen objects from the mile markers tables in the AWIPS II database. The WarnGen documentation describes the utilization of the importMarkersInfo.sh script to get AWIPS I mile markers into AWIPS II. This XML file contains instructions and an example for making the mile marker information available for use in the templates. This XML file is referenced via an "include" statement in each WarnGen template's XML configuration file. Simply uncomment the include section that has already been inserted by the developers.
- **mileMarkers.vm.** The "guts" that actually generate the mile marker or road output. Here you will reference the mile marker objects created in the XML file, and create "lead-ins" that will create proper phrasing for your mile marker output. This auxiliary vm file is then referenced via a "parse" statement in each WarnGen template’s Velocity .vm file, instructing WarnGen to generate the mile marker output. Simply
uncomment the appropriate section in each Warngen template's vm file that has already been inserted by the developers.

- **forecasterName.vm.** vm code that will help assign a warning forecaster name to warnings automatically. Must be customized to create local user's desired moniker (name, initials, number, etc.). This auxiliary vm file is then referenced (if desired) via a "parse" statement added to each WarnGen template's Velocity .vm file.

### 15.3.4 Configuration

#### 15.3.4.1 Configuration Files (.xml)

Each warning Velocity template has a complementary configuration file (.xml). This configuration will set different variables that can dynamically change how WarnGen acts depending on the warning type. It is important to note that each configuration must start and end with a `<warngenConfig>` and `</warngenConfig>` tag. Also, comments can be maintained with `<!--comment-->`.

**How to configure warnings to be county, zone, or marine zone based**

Each warning references an area source or table in the database: County; Zone; or MarineZones. Table 15.3.4.1-1 shows what values to set for the fields in order to configure a warning to the correct area source properly. The ugcline referenced in the .vm files will automatically be in the correct format (county or zone).

#### Table 15.3.4.1-1. Field Values

<table>
<thead>
<tr>
<th></th>
<th>areaConfig</th>
<th>areaConfig</th>
<th>pathcastConfig</th>
<th>geospatialConfig</th>
<th>geospatialConfig</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>areaField</td>
<td>fipsField</td>
<td>areaField</td>
<td>areaSource</td>
<td>pointSource</td>
</tr>
<tr>
<td>County</td>
<td>COUNTYNAME</td>
<td>FIPS</td>
<td>COUNTYNAME</td>
<td>COUNTY</td>
<td>CITY</td>
</tr>
<tr>
<td>Zone</td>
<td>NAME</td>
<td>STATE_ZONE</td>
<td>NAME</td>
<td>ZONE</td>
<td>CITY</td>
</tr>
<tr>
<td>Marine Zones</td>
<td>NAME</td>
<td>ID</td>
<td>NAME</td>
<td>MARINEZONES</td>
<td>MARINESITES</td>
</tr>
</tbody>
</table>

- **Units**: `<unitDistance>` & `<unitSpeed>`. Unit distance is the unit the distance result will be in and unit speed is the unit the speed result will be in.
  
  `<unitDistance>mi</unitDistance>`
  `<unitSpeed>mph</unitSpeed>`

- **Maps**: `<maps>`. Maps contained within `<map></map>` will get loaded with this template. Refer to 'Maps' menu in CAVE. The various menu items are also the different maps that can be loaded with each template.

- **Followups**: `<followups>`. Each associated follow-up should be contained within `<followup></followup>`. Each follow-up will become available when the appropriate time range permits.

- **Phen Sigs**: `<phensigs>`. Each associated phen sig should be contained within `<phensig></phensig>`.

_AWP.MAN.SMM.A2_ 15-18

– Hard copies uncontrolled. Verify effective date prior to use. –
• **Enable Restart:** `<enableRestart>`.

  Enables/disables user from selecting the Restart button on the GUI. This is usually set to false for follow up configuration files. Because this variable is usually set to false for follow-ups, this also controls if the 'Drag me to storm' text will become available.

• **Enable Dam Break Threat:** `<enableDamBreakThreat>`.

  Enables/disables the 'Dam Break Threat Area' button. This is mainly used for dam break templates.

• **Auto Lock Text:** `<autoLockText>`.

  Enables/disables the system to lock text based on various patterns. Refer to Locking Text section for more information.

• **Watches:** `<includedWatches>`.

  Each watch to be included should be contained within `<includedWatch></includedWatch>`. The only two possible values are 'torWatches' and 'svrWatches'. Refer to includeWatchAreaBuffer in the Area Configuration for more information.

• **Duration:** `<durations>`.

  Each available duration should be contained within `<duration></duration>`. The default duration should be contained with `<defaultDuration></defaultDuration>`. However, default duration should not be included within `<durations></durations>`.

• **Locked Groups on Followup:** `<lockedGroupsOnFollowup>`.

  On a follow-up, there are certain groups that cannot be unselected, such as a dam or ic. However, there may be another group defined by the user that he or she would want to have locked down, such as advType. The following is an example:

  ```xml
  <lockedGroupsOnFollowup>dam,ic</lockedGroupsOnFollowup>
  ```

• **Bullet Action Groups:** `<bulletActionGroups>`.

  The bullet action groups maintain all bulletActionGroups and should be contained with `<bulletActionGroups></bulletActionGroups>`. Within the bulletActionGroup is a `<bulletActionGroup>` that contains all the available bullets for the associated action, phen, and sig.

  Example:

  ```xml
  <bulletActionGroups>
  <bulletActionGroup action="NEW" phen="TO" sig="W">
  <bullets>
    <!-- List of bullets for a NEW, Tornado Warning -->
  </bullets>
  </bulletActionGroup>
  <bulletActionGroup action="COR" phen="TO" sig="W">
  <bullets>
    <!-- List of bullets for a COR, Tornado Warning -->
  </bullets>
  </bulletActionGroup>
  </bulletActionGroups>
  ```

• **Bullets:** `<bullets>`.

  The bullets populate the selection list in the WarnGen dialog. Each bullet should be contained in a `<bullet>` tag. The following are the different parameters available to a bullet:
- **bulletName**: An id that is passed to the template when a bullet is selected. This name should be unique as it can be used as an identifier with the velocity files. However, 'siteimminent' and 'sitefailed' are reserved bullet names for dam breaks.
- **bulletText**: The text presented to the user in the selection list.
- **bulletGroup**: only one bullet can be selected per bulletGroup.
- **bulletType**: 'Title' is unselectable.
- **showString**: If the string MATCHES a phrase in the follow-up, then the bullet will be available; if this value is not set, then the bullet will automatically be available.
- **parseString**: If this string MATCHES a phrase in the follow-up warning, then the associated bullet will be automatically highlighted in the selection list for a follow-up.

If there are multiple strings that need to be tested, the escape character '"' needs to be placed before and after the phrase to be matched. For example, parseString=""LINE OF SEVERE THUNDERSTORMS"", parseString=""SQUALL"". If the warning text contains “LINE OF SEVERE THUNDERSTORMS” and “SQUALL”, then the associated bullet will be highlighted.

Furthermore, if a phrase begins with '-' and is not in the warning text, then the associated bullet will always be highlighted. For example, parseString=""-SQUALL""
- **loadMap**: Refer to 'Maps' menu in CAVE. The various menu items are also the different maps that can be loaded with each template.
- **floodSeverity**: Used to highlight the correct severity on a correction.

- **Dam Info Bullets**: <damInfoBullets>. The bullets populate the selection list in the WarnGen dialog. Each bullet should be contained in a <damInfoBullet> tag. The following are the different parameters available to a bullet:
  - **bulletName**: An id that is passed to the template when a bullet is selected. This name should be unique as it can be used as an identifier with the velocity files.
  - **bulletText**: The text presented to the user in the selection list.
  - **bulletGroup**: Should use either 'dam','ic', 'scenario'. Using 'dam' or 'ic' will lock the bullet on a follow-up.
  - **bulletType**: 'Title' is unselectable.
  - **coords**: The lat/lon coordinates in the format usually found at the bottom of the warnings (i.e., LAT...LON 3981 7868 3980 7861 3972 7862 3972 7874).

Example:

```xml
<damInfoBullets>
  <damInfoBullet bulletName="branchedOakDam" bulletText="Branched Oak dam (Lancaster county)"
                 bulletGroup="dam" parseString="BRANCHED OAK DAM"
                 coords="LAT...LON 4096 9686 4094 9680 4089 9679 4089 9681 4087 9681 4086 9679 4081 9678 4082 9674 4077 9672 4082 9671 4084 9668 4085 9670 4084 9676 4093 9676 4100 9679 4096 9679 4098 9685"/>
</damInfoBullets>
```
For a further example, reference /site/OAX/warngen/dambreak.vm.

- **Include**: `<include file="filename.txt"/>`. Allows an external file to be inserted in the configuration file. This functionality is useful when a user wants to list dam info bullets in a separate file. The format of the include tag is `<include file="filename.txt"/>` where filename is the name of the file including the extension of the file (i.e. .txt). The file to be included should follow the rules of localization: user, site, then base.

  **Note**: In order to comment out the include statement, it must be commented out in the following manner <!-- include file="filename.txt"/> -->

- **Area Configuration**: `<areaConfig>`. Specifies how the area portion of the warning is generated. This configuration is mainly used for determining the impacted counties in the 1st bullet. The following are the different parameters available to the areaConfig:

  - **inclusionPercent**: If an area greater than this percentage of area is covered, include it in the warning.
- **inclusionAndOr**: AND – both inclusionPercent and inclusionArea must pass in order to be included. OR – either inclusionPercent or inclusionArea can pass to be included.

- **inclusionArea**: If an area greater than this is covered, include it in the warning (square kilometers).

- **areaField**: The column name of the areaSource that contains the name of the area (i.e. COUNTYNAME, NAME).

- **parentAreaField**: The column name of the parentSource that contains the name of the parent (i.e., NAME).

- **areaNotationField**: The column in the areaSource that contains the key used to map to the correct area notation. Look at countTypes.txt.

- **areaNotationTranslationFile**: Translation file to look up an area notation. File located in /edex/data/utility/common_static/base/warngen.

- **fipsField**: The name of the attribute that retrieves the fips.

- **pointField**: The column name in the pointSource that contains the name of the point.

- **sortBy**: Can sort by 'name', 'parent', 'size', partOfArea, partOfParent, areaNotation, fips.

- **pointFilter**: Controls which point to include based on the WARNGENLEV of the point. WARNGENLEV is a value from 1 to 3 where 1 is the highest priority.
  - Key WARNGENLEV (column name in the pointSource).
  - constraintValue 1, 2, and/or 3.
  - constraintType: EQUALS, IN, LIKE, NOT EQUALS, etc.

- **includeWatchAreaBuffer**: The number of MILES the warning polygon will be increased to include nearby watches.

Example:

```xml
<areaConfig>
  <inclusionPercent>0.00</inclusionPercent>
  <inclusionAndOr>AND</inclusionAndOr>
  <inclusionArea>10</inclusionArea>
  <areaField>COUNTYNAME</areaField>
  <parentAreaField>NAME</parentAreaField>
  <areaNotationField>STATE</areaNotationField>
  <areaNotationTranslationFile>countyTypes.txt</areaNotationTranslationFile>
  <fipsField>FIPS</fipsField>
  <pointField>NAME</pointField>
  <sortBy>
    <sort>name</sort>
  </sortBy>
  <pointFilter>
    <mapping key="WARNGENLEV">
      <constraint constraintValue="1" constraintType="EQUALS"/>
    </mapping>
  </pointFilter>
</areaConfig>
```
• **Point Source Configuration:** `<pointSource>`. This configuration allows various point source results to be generated. The following are the different parameters:
  - `variable`: The name of point source results to be referenced in the .vm.
  - `searchMode`: TRACK – search for points along the entire track. POINTS – search for every center point.
  - `withinPolygon`: If true, only return points that are within the warning polygon; otherwise, false will allow points within or beyond the warning polygon.
  - `pointField`: The field out of the pointSource that is used for naming the point.
  - `pointFilter`: Controls which point to include based on the WARNGENLEVEL of the point. WARNGENLEVEL is a value from 1 to 3 where 1 is the highest priority:
    - `key WARNGENLEVEL (column name in the pointSource)`
    - `constraintValue 1, 2, and/or 3`
    - `constraintType: EQUALS, IN, LIKE, NOT EQUALS, etc.`
  - `distanceThreshold`: The allowable distance to allow a point to be included from the track or event center.
  - `maxResults`: The max points allowed to be returned.
  - `sortBy`: Can sort by 'distance', 'name', 'warngenlev', 'population', 'lat', 'lon', 'area', 'parentArea'.

Example:

```xml
<pointSource variable="closestPoints">
  <pointField>NAME</pointField>
  <searchMethod>POINTS</searchMethod>
  <filter>
    <mapping key="WARNGENLEVEL">
      <constraint constraintValue="1,2" constraintType="IN"/>
    </mapping>
  </filter>
  <maxResults>1</maxResults>
  <distanceThreshold>50</distanceThreshold>
  <sortBy>
    <sort>distance</sort>
    <sort>warngenlev</sort>
  </sortBy>
</pointSource>
```

• **Path cast Configuration:** `<pathcastConfig>`. This configuration indicates a track product is generated to determine the areas the storm will pass under. This is mainly for the fourth bullet and used if type 1 is selected. Refer to the macros contained in the VM_global_library.vm. The following are the different parameters:
- **withinPolygon**: If true, only return points that are within the warning polygon; otherwise, false will allow points within or beyond the warning polygon.

- **defaultDirection**: Default direction (0 degrees is South) to generate the storm track if a recent track was not issued or if there is not STI data.

- **nearThreshold**: Specifies a distance in miles that indicates how close a storm can be to a location to be included in the path cast.

- **interval**: The interval to round the time of each path cast to.

- **maxCount**: The max number of points to include in each path cast.

- **delta**: Offsets for the first pathcast entry.

- **maxGroup**: The max number of time groups to use in the pathcast.

- **pointFilter**: Controls which point to include based on the WARNGENLEV of the point. WARNGENLEV is a value from 1 to 3 where 1 is the highest priority:
  - Key WARNGENLEV (column name in the pointSource)
  - Value 1, 2, or 3 (can only be a single value), 0 does nothing
  - Type: INCLUSIVE – include only points that match the value
    EXCLUSIVE-exclude only points that match the value
  
  For example, INCLUSIVE 1, will only include points that are WARNGENLEV 1, EXCLUSIVE 3, will only include points that are WARNGENLEV 1 and 2.

- **areaField**: The column name in the areaSource table that contains the name of the area.

- **parentAreaField**: The column name in the areaSource that contains the parent of the area (i.e., State).

- **areaNotationField**: The column name in the areaSource that contains the key used to map to the correct areaNotation. Look at countyTypes.txt.

- **areaNotationTranslationFile**: Translation file to look up an area notation. File located in /edex/data/utility/common_static/base/warngen.


Example:

```xml
<pathcastConfig>
  <withinPolygon>true</withinPolygon>
  <defaultDirection>45</defaultDirection>
  <nearThreshold>8.0</nearThreshold>
  <interval>5</interval>
  <delta>5</delta>
  <maxCount>4</maxCount>
  <maxGroup>8</maxGroup>
  <pointField>Name</pointField>
  <pointFilter>
    <key>WARNGENLEV</key>
    <value>0</value>
  </pointFilter>
</pathcastConfig>
```
• **Geospatial Configuration:** `<geospacialConfig>`. This configuration maintains global variables (i.e., `pointSource`, `areaSource`, `parentAreaSource`) used by area configuration, closest points configuration, and path cast configuration. This configuration also initializes WarnGen to gather all the areas within the CWA. The following are the different parameters:

- **pointSource:** The name of the table that reads the points from (i.e., City, MarineSites).
- **areaSource:** The name of the table that reads the areas from (i.e., County, MarineZones).
- **parentAreaSource:** The name of the table that reads the parent areas from (i.e., States).
- **timezoneSource:** The name of the time zone table (i.e., TIMEZONES).
- **timezoneField:** The name of the column in the timezoneSource that contains the time zones.

Example:

```xml
<geospacialConfig>
  <pointSource>City</pointSource>
  <areaSource>County</areaSource>
  <parentAreaSource>States</parentAreaSource>
  <timezoneSource>TIMEZONES</timezoneSource>
  <timezoneField>TIME_ZONE</timezoneField>
</geospacialConfig>
```

• **River Basin Configuration:** `<riverBasinConfig>`. This configuration uses the same format as the point source configuration and uses the variable “riverdrainages” and derives the points from the `ffmp_basins` table.

Example:

```xml
<pointSource variable="riverdrainages">
  <pointSource>ffmp_basins</pointSource>
  <pointField>streamname</pointField>
  <filter>
    <mapping key="cwa">
      <constraint constraintValue="SwarngenCWAFilter"
```
15.3.4.1.1 Velocity Templates

WarnGen passes various Java objects to the velocity templates that can contain various pieces of information or be a utility object which can be used in generating a warning. Some Java objects have their API available on the Internet, which can provide more information on the available methods the objects provide. It is important to understand that some objects derive information for tables that are set in the associated configuration file.

- **Header.** It is very important that the first two lines of your template contain the WMO Header and the PIL information as this information will be parsed by the text workstation and placed in the header block of a text editor.

  ```
  ${WMOId} ${vtecOffice} 000000 ${BBBId}
  TOR${siteId}
  ```

- **VM_global_library.vm.** VM_global_library.vm contains various macros that can be referenced by all templates. Each macro performs common functions that can be used in calculating and generating text. The function name is referenced by the first parameter of the macro while the following parameters are the required parameters. For example, the macro `#macro(direction $d)` is called by `direction(90)`. The appropriate macro will be recognized and produce the appropriate text based on the parameters. However, a user can create a site level version of the VM_global_library.vm See `#parse` macro for further uses.

- **#parse.** The `#parse` macro is function Velocity supported function that can be used to import a file into a .vm file. The important thing to note is that if a site level .vm file implements the `#parse` macro, then the imported file and a site-level version of VM_global_library must be present in the same site level directory.

- **Variables action.** The action the warning is performing: NEW, COR, EXT, CON, CAN, and EXP. CANCON is an action indicating there were canceled areas and continued areas. This action is mainly used to help rendering both.
  - `areaPoly`: An array of Coordinates of the warning polygon.
  - `areas`: An array of AffectedAreas object. This object is usually used in the 1st bullet. An AffectedAreas object has the following members:
    - `name`: The name of the area affected. Refer to areaConfig.areaField
    - `partOfArea`: NORTH, NORTHEAST, EAST, etc.
    - `areaNotation`: The notation of the area affected (COUNTY, PARISH, etc.). Refer to countyTypes.txt
    - `partOfParentRegion`: NORTH, NORTHEAST, EAST, etc.
- parentRegion: The name of the parent region that is affected. Refer to areaConfig.parentAreaSource
- fips: fips county number
- stateabbr. State abbreviation (i.e., NE for Nebraska)
- points. If applicable, a list of the points that are affected
- timezone
- size. Size of the area.

  - areaSource: The area source set in the geospatial config.
  - backupSite: The backup site selected in the 'Backup' selection area.
  - BBBId: A String that keeps track of the number of corrections performed on a warning (i.e., CCA).
  - bullets: An array of Strings that contain all the bullets that were selected.
  - cancelareas: An ArrayList of AffectedAreas objects that are canceled from the original warning. This is commonly used in follow up velocity files.
  - closestPoints: An array of ClosestPoint objects closest to the event. A ClosestPoint object has the following members:
    - Name: Name of the point, such as the city name or marine site name
    - Area: The name of the area, such as the county name
    - parentArea: The name of the parent area, such as the state
    - Point. Coordinate of the point
    - Distance: Distance from point to the event
    - roundedDistance: Distance as a rounded integer without any decimals
    - Azimuth: The angle of point from the event. In other words, the point is relative to the event. Because the event is the reference, the azimuth needs 180 subtracted in order for the city to become the reference for the event.
    - roundedAzimuth. Aazimuth to the closest 45 degree intervals
    - oppositeAzimuth. 180 degrees from the azimuth
    - oppositeRoundedAzimuth. 180 degrees from the roundedAzimuth
    - population. Population in the city (this variable is not available for marine sites or locally added cities or markers)
    - Warnngenlev. Priority level where 1 is the highest.
  - dateUtil: A DateUtil object can format a Date object (i.e., start) into a particular display string based on the passed timeFormat. Refer to timeFormat to see the all the possible types. The most useful method to use is format (date, timeFormat, interval, timezone).
  - etn: Event Tracking Number.
  - event: Date object of the time of occurrence of the warning/last frame.
  - eventLocation: The Coordinate of the event. This is commonly used in the 3rd bullet.
  - expire: Date object of the expire time of the warning.
- **floodseverity**: This is usually used in follow up hydro products. The value is derived from the HTEC of the initial warning. The flood severity is set in the velocity template using conditional logic based on the bullets selected in the dialog. Minor flood = 1, moderate flood = 2, major flood = 3.

- **floodic**: The immediate cause from the initial warning. This is commonly used in follow-ups.

- **ic**: The immediate cause used in the HTEC. Common values are ER, DR, GO, IJ, RS, SM, DM.

- **includedWatches**: An array of the included watches mentioned in the config.

- **list – ListTool object**: Refer to API for org.apache.velocity.tools.generic.ListTool for more information.

- **localtimezone**: Time zone derived when calculating the areas. Commonly used when using the secondBullet macro. This is a required field. Refer to the secondBullet macro for more info.

- **mathUtil**: A WarnGen Math Tool object that extends MathTools (org.apache.velocity.tools.generic.MathTool) is used for rounding to the nearest 5 or 10, for example, of a specific value. For instance, wind speeds in the third bullet should be rounded to the nearest 5 MPH so they should call this method with the arguments (windSpeed, 5). There are two available methods for this object aside from the methods in MathTools:
  - `roundTo1Int(value, multiple)`
  - `roundTo5(value)`: Calls `roundTo1Int(value,5)`.

- **movementDirection**: The direction of the storm track in degrees where 0 is South.

- **movementDirectionRounded**: The rounded direction of the storm track in degree where 0 is south. The possible values are 0, 90, 135, 180, 225, 270, 315. This variable is commonly used with the macro direction to give a general description if the storm is moving NORTH, NORTHEAST, WEST, etc.

- **movementInKts**: The movement speed in Knots. This is mainly used when generating the TML line of the warning.

- **movementSpeed**: The speed of the storm based on the track. The speed units can be defined in the closestPointsConfig.unitSpeed configuration.

- **now**: Date object of the current time.

- **officeShort**: The office short name 'warngenOfficeShort' set in config.xml. Refer to Config.xml section for more information.

- **officeLoc**: The office location name 'warngenLocation' set in config.xml. Refer to Config.xml section for more information.

- **otherPoints**: An array of ClosestPoint objects. This array is used when type 2 or 3 is passed to the fourth Bullet. Refer to areaConfig.otherPoints to sort the array. A ClosestPoint object has the following members: name, population, warngenlev, distance.

- **Pathcast**: An array of PathCast objects. Each PathCast object is grouped by 'time' in ascending order. This array is only used when type 1 is passed to fourthBullet. The following are members:
Chapter 15 Localization

- Points – all the points (i.e. Cities) that are listed for the path cast in the form of an array of ClosestPoint objects.
- Area – refer to pathCastConfg.areaField
- areaNotation – can be either COUNTY, PARISH, ZONE. Refer to countyTypes.txt
- parentArea – refer to pathCastConfig.parentAreaField
- timeZone
- Time – time of the path cast

- phenomena: Two-letter phenomena of the warning
- pointComparator: Allows two closest point array objects to be combined and sorted. The key method to use is to combine. The first two parameters are the two closest point object arrays to be combined. The following parameters can be an N number of Strings that the newly combined array will be sorted by ('distance', 'name', 'warnGenlev', 'population', 'lat', 'lon', 'area', 'parentArea'). For example, ${pointComparator.combine(${otherPoints},${i80mm},"warnGenlev")}.
- productClass: The single letter indicating if the warning in operational mode (O) or test mode (T).
- riverdrainages: An array of the names of the basins that are included in the warning polygon. Refer to River Basin Configuration for more info.
- secondtimezone: Time zone derived when calculating the path cast. Commonly used when using the secondBullet macro. This will only become available if the warning area crosses an additional timezone.
- siteId: The 3-letter site Id used for creating the PIL. Refer to the 'Header' section for more information.
- specialCorText: The raw message of the record to be corrected without the WMO header line and the PIL line. The VTEC in the raw message is updated to have a COR action. This variable is made available if there is no change in the storm track.
- start: Date object of the start time of the warning.
- stationary: True or false if there are no time points (i.e., only a single frame).
- stormType: Used to indicate if the warning is for a single storm or a line of storms. If the value is “line” then it is a line of storms or “single” if it is a single storm.
- timeFormat: Used along with the dateUtil object. The following are all the possible variables. Refer to the Java API SimpleDateFormat for more information:
  - Header: hmm a z EEE MM d yyyy
  - Plain: hmm a z EEEE
  - Clock: hmm a z EEEE
  - ymdthmz: yyMMdd'T'HHmm'Z'
  - ddhhmm: ddHHmm
  - Time: HHmm
- **ugcline**: A list of the county or zones of each affected area. This variable depends on the affected areas (i.e. county, zone, or marine zones).
- **ugclinesean**: A list of the ugc or zones of each county or zone that was canceled.
- **user**: Username.
- **vtpecOffice**: The 4-letter CWA that generated the warning (i.e. “KOAX”); Usually, used in the VTEC.
- **watches**: A WatchUtil object that can gather tornado and/or severe thunderstorm watches within the watch area. The areaConfig.includedWatchAreaBuffer is used to determine whether to include the watch or not. The WatchUtil has two useful methods getTorWatches and getSvrWatches. These methods return an array list of WeatherAdvisoryWatch objects that have the available variables: phensig, endTime, parentRegion, partOfParentRegion. It is important to note that the endTime is set to the latest time in the array list.
- **WMOId**: WarnGen automatically sets this variable to TTAAii. When the text is sent to text workstation, text workstation replaces it with the appropriate WMO header retrieved from the afos_to_awips table.

### 15.4 Other Menu Localizations

Other menus are localized in the same manner as the radar menus listed in section 15.2. Here are some details on how some other menus are localized from the baseline.

#### 15.4.1 Upper Air Menus

During site migration, the config_awips2.sh tool created the input file used by EDEX to create the SITE level Upper Air Menu file. The file that controls this menu creation is named `raobSitesInUse.txt` and it resides in the following location:

```
/awips2/edex/data/utility/common_static/site/LLL/upperair
```

(Where LLL is the AWIPS ID of your localized site)

This location is accessible on your EDEX servers (dx3, dx4 for WFOs, dx3, dx4, dx5 and dx6 for NCEP). An example of the file follows.

```
# DO NOT EDIT LINES BEGINNING WITH '#'
# FULLY DEFINES ALL SITES USED FOR THE LOCALIZED UPPER AIR MENU
IN CAVE
# Format is as follows:
# icao city name, state
# KXXX SiteID Anywhere, US

# UPPER_AIR menu
KOAX 72558 Omaha, NE

# LOCAL_UPPER_AIR submenu
KABR 72659 Aberdeen, SD
KBIS 72764 Bismark, ND
```
The file is initially created in a fashion that is similar to the way the AWIPS I raobLocalMenus.txt is created. It finds the raobMenus.txt file, which is a baseline nationalData file, and reads in every line. It determines, based on the site for which localization is being configured, how far away from the site the RAOB is located by using the common_obs_spatial database table in the metadata database. If the RAOB is within 200km of the site, it adds it under UPPER_AIR menu as seen in the example. If it is within 200km – 600km of the site, then it adds it into LOCAL_UPPER_AIR submenu (as also seen in the example).

When EDEX is started, it reads this file. If it has not created its output files, or the file raobSitesInUse.txt has been modified more recently than the output files have been modified or created, it will create its output files in

```
/awips2/edex/data/utility/cave_static/configured/LLL/menus/upperair
```

These files are eligible for override by promoting them to the SITE level using the localization perspective in the following path:

```
CAVE --> Menus --> upperair
```

### 15.5 Modifying Shapefiles and Area Files

A shapefile is made up of three components:

- `.dbf` file
- `.shp` file
- `.shx` file

Baseline shapefiles will be updated using the shp option to config_awips2.sh. They can be maintained as described in this section.

A site will have the option of renaming the files from the downloaded name itself, or it can allow the config_awips2.sh (which is also aware of the AGMG naming convention) to perform the renaming, so the files do not have to be renamed for AWIPS II purposes. The script itself will rename the files to their proper AWIPS I names in

```
data/fxa/nationalData
```

For example, the public forecast zones can be named c11-zone.shp OR z_mmddyy.shp. Both will be recognized by config_awips2.sh. If the shapefile is left as z_mmddyy.shp, the script will rename it to c11-zone.shp when it imports the file.

The tool works in this way: When the script is run with the shp argument, it will check for updated files, which are newer than those at the import time of the corresponding table in the `/awips2/edex/data/utility/edex_static/base/shapefiles` (directorymaps
database). If any are found, it will copy them into the AWIPS II directory in /awips2/edex/data/utility/edex_static/base/shapefiles, and import them into the database automatically. It also makes a backup of the old shapefiles in the following form:

YYYMMDD-filename.archive

Where filename is the name of the shapefile, for example, usa_lake.shp.

**IMPORTANT NOTE:**

AWIPS II relies on accurate shapefiles. If you notice a problem with an application using a shapefile that has been updated, do not try and re-import the same shapefile. Call the NCF, who has procedures on how to import the archived shapefile. Every time you re-run config_awips2.sh with the shp option, it will remove the archived shape files so you may lose the “good” ones.

All of the shapefiles used in the localization process can be found in the following directory.

**SFXA_DATA/nationalData**

In addition, **SFXA_DATA/nationalData** is NFS-mounted from the NAS on DX1 with links to it on the hosts. Shapefiles are the primary repository for the national configuration data set provided by NWS headquarters. Shapefiles contain data sets for the entire United States, and, where applicable, its territories. The shapefiles and their primary data content are shown in Table 15.5-1.

<table>
<thead>
<tr>
<th>Shapefile</th>
<th>Primary Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>c11-zone</td>
<td>Public Forecast Zones</td>
</tr>
<tr>
<td>usa_cwa</td>
<td>County Warning Area (CWA) boundaries</td>
</tr>
<tr>
<td>usa_cwa_total</td>
<td>Partial service backup</td>
</tr>
<tr>
<td>marine_zones</td>
<td>Marine zone boundaries</td>
</tr>
<tr>
<td>coastal_zones</td>
<td>Coastal zones</td>
</tr>
<tr>
<td>offshore_zones</td>
<td>Offshore marine zones</td>
</tr>
<tr>
<td>high_seas_zones</td>
<td>High seas marine zones</td>
</tr>
<tr>
<td>fs1-w88d</td>
<td>WSR-88D locations</td>
</tr>
<tr>
<td>basins</td>
<td>River basin boundaries</td>
</tr>
<tr>
<td>usa_rfc</td>
<td>RFC area boundaries</td>
</tr>
<tr>
<td>usa_lake</td>
<td>Lakes and other water boundaries</td>
</tr>
<tr>
<td>usa_wsfo</td>
<td>Redefining areas</td>
</tr>
<tr>
<td>timezones</td>
<td>Time zone boundaries</td>
</tr>
<tr>
<td>firewx</td>
<td>Fire Weather Zone</td>
</tr>
<tr>
<td>urban_bounds</td>
<td>Urban boundaries</td>
</tr>
<tr>
<td>marine_total</td>
<td>Partial marine zones backup</td>
</tr>
<tr>
<td>routes</td>
<td>Interstate, US, State highways</td>
</tr>
</tbody>
</table>

Shapefiles provide a complete non-topological description of two-dimensional geometric shapes, in three files:
filename.shp     shape description
filename.shx     index (shape to attributes)
filename.dbf     attributes

The shape of each geometric (spatial) object is a point, line, or polygon, and it is described completely and in sequential order by the *.shp file. Attributes for each shape are stored sequentially in the same order by the file *.dbf, using the dBASE III+ Database File Structure. An index of shapes to attributes is maintained in the *.shx file. The shapefile format is documented online at http://www.esri.com. By convention, a shapefile is associated by one prefix file name. The contents of the *.dbf file can be changed without damaging the index relationship to *.shp as long as dBASE records are not added, deleted, or reordered. Shapefiles can be renamed, provided that all three components are given the same file name and the extensions are not changed.

Each entry provides an image of the shapefile. When the geographic area is not obvious, the shapefile is shown as darker points or lines on a lighter background. Space is conserved using the “zip” compression technique. Public domain software to extract *.zip files is available for most operating systems and hardware environments from ftp://ftp.uu.net/pub/archiving/zip/. The unzip utility (unz520x.tar.Z) is mirrored here for NWS users. Make sure that the *.zip file has been downloaded using a binary transfer method.

filename.zip     compressed (zip)
filename.gif     Graphics Interchange Format
filename.txt     flat text listing of filename.dbf

Descriptions of the catalog entries follow.

- **Files Prefix**: Shows the extensions of the three component files of the shapefile.
- **Naming Convention**: A description of the name of the shapefile. Most data in the catalog have a two-letter prefix for the type of data and the date in ddmmyy format.
- **Location**: Directory in Catalog (URL)
- **Shapefile Type**: Type of shape {point, polyline, polygon}
- **Source**: Origin of shape and attribute information
- **Last Modified**: Last date shapes or attributes modified
- **Attributes**
  - **Field**: dBASE field name (1 to 10 characters)
  - **Type**: Field type {character, numeric, logical, date}
  - **Width, dec**: Field type specifications: {character: width = # characters} {logical: width = 1} {date: width = 8} {numeric: width = # digits, dec = # decimal places}
  - **Description**: Brief statement of field usage
An example follows.

**Files Prefix:** c11-zone.shp.Z/.shx/.dbf - where .shp, .shx, .dbf are the extensions of the three component files of the shapefile

**Naming Convention:** z_ddmmyy .shp/.shx/.dbf

**Location:** NOAA1 ftp server

**Shapefile Type:** polygon

**Source:** NWS Office of Climate, Water and Weather Services (OCWWS): NWS 10-507

**Last Modified:** 27 March 2009

**Attributes:** Refer to Table 15.5-2.

### Table 15.5-2. Zone Attributes

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>Width, decimal</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>STATE</td>
<td>Character</td>
<td>2</td>
<td>[ss] State abbreviation (U.S. Postal Standard)</td>
</tr>
<tr>
<td>ZONE</td>
<td>Character</td>
<td>3</td>
<td>[ZZZ] Zone number, from WSOM C-11</td>
</tr>
<tr>
<td>CWA</td>
<td>Character</td>
<td>3</td>
<td>Zone name, from WSOM C-47</td>
</tr>
<tr>
<td>NAME</td>
<td>Character</td>
<td>254</td>
<td>Zone name, from WSOM C-11</td>
</tr>
<tr>
<td>STATE_ZONE</td>
<td>Character</td>
<td>5</td>
<td>[sszzz] state+ (&quot;00&quot;+zone.trim).right(3)</td>
</tr>
<tr>
<td>TIME_ZONE</td>
<td>Character</td>
<td>2</td>
<td>[tt] Time zone, 2 chars if split</td>
</tr>
<tr>
<td>FE_AREA</td>
<td>Character</td>
<td>2</td>
<td>[aa] Cardinal area of state (occasional)</td>
</tr>
<tr>
<td>LON</td>
<td>Numeric</td>
<td>10,5</td>
<td>Longitude of centroid (decimal degrees)</td>
</tr>
<tr>
<td>LAT</td>
<td>Numeric</td>
<td>9,5</td>
<td>Latitude of centroid (decimal degrees)</td>
</tr>
<tr>
<td>SHORTNAME</td>
<td>Character</td>
<td>32</td>
<td>Shortened zone name</td>
</tr>
</tbody>
</table>

### 15.6 How to Download Shapefile Data

Follow these steps to download shapefile data.

1. At one workstation, if any D2D applications or text display applications are running, exit the D2D and text display applications.

2. Open a Terminal window at a workstation and log in as the **root** user.

3. Change user accounts to **fxa**.

   **TYPE:** su - fxa

4. Change to the **nationalData** directory.

   **TYPE:** cd $FXA_DATA/nationalData

5. Save the current files by entering the following commands, which use zone forecast area boundary files as examples:

   **TYPE:** mv c11-zone.dbf c11-zone.dbf.orig

   **TYPE:** mv c11-zone.shp.Z c11-zone.shp.Z.orig

   **TYPE:** mv c11-zone.shx c11-zone.shx.orig
For other types of shapefiles, repeat the same procedure and save the downloaded files in $SFXA/nationalData giving them new file name.

6. SFTP to the NOAA1 ftp server by entering the following command:
   
   TYPE:  
   
   sftp ftpawips@165.92.25.137
   
   Once you are connected. You will be prompted for the ftpawips password which is !SAWIPS4

7. Prepare to download the updated files by entering the following commands:
   
   TYPE:  
   
   cd pub/maps/currentBaseline
   
8. For Zone Forecast Area Boundaries, download the file using the following command:
   
   TYPE:  
   
   mget z_ddmmyy.*
   
   Where dd is the two digit date, mm is the two character month, and yy is the two digit year. Three files will be returned with the extensions dbf, shp.Z, and shx.

9. For other shapefiles substitute the correct file name in Step 8, as shown in Table 15.6-1.

<table>
<thead>
<tr>
<th>Shapefile Format</th>
<th>FTP Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>CWA</td>
<td>mget w_ddmmyy.*</td>
</tr>
<tr>
<td>Partial Service Backup</td>
<td>mget wtddmmyy.*</td>
</tr>
<tr>
<td>Coastal Marine Zone Area Forecast</td>
<td>mget mzddmmyy.*</td>
</tr>
<tr>
<td>Offshore Marine Zone Area Forecast</td>
<td>mget ozddmmyy.*</td>
</tr>
<tr>
<td>High Seas Marine Zone Area Forecast</td>
<td>mget hzddmmyy.*</td>
</tr>
<tr>
<td>WSR-88D Locations</td>
<td>mget w8ddmmyyy*</td>
</tr>
<tr>
<td>River Basin Boundaries</td>
<td>mget baddmmyy.*</td>
</tr>
<tr>
<td>RFC Area Boundaries</td>
<td>mget rfdmmyy.*</td>
</tr>
<tr>
<td>For Lakes and other Water Bodies</td>
<td>mget lkddmmyy.*</td>
</tr>
<tr>
<td>Interstate Highways</td>
<td>mget indmmyy.*</td>
</tr>
<tr>
<td>Redefining Area</td>
<td>mget usa_wsfoddmmyyy.*</td>
</tr>
<tr>
<td>Time Zones</td>
<td>mget tzddmmyy.*</td>
</tr>
<tr>
<td>Urban Boundaries</td>
<td>mget ubddmmyy.*</td>
</tr>
<tr>
<td>Marine Zones Partial Backup</td>
<td>mget mtddmmyy.*</td>
</tr>
<tr>
<td>Cities</td>
<td>mget cidmmyy.*</td>
</tr>
<tr>
<td>States</td>
<td>mget s_ddmmyy.*</td>
</tr>
<tr>
<td>Routes</td>
<td>mget rddmmyy.*</td>
</tr>
<tr>
<td>Fire Weather</td>
<td>mget fzddmmyy.*</td>
</tr>
</tbody>
</table>

10. Exit the NOAA1 Server.

   TYPE:  
   
   bye
15.7 Load and Activate the New Data on AWIPS

1. Change to the scripts directory. As user root,
   \[\text{TYPE: } \text{cd /data/fxa/sdc}\]
2. Run the configuration script for shapefiles.
   \[\text{TYPE: } \text{./config_awips2.sh shp LLL}\]
3. After the shapefile load completes, verify that all the changes were made properly. Start CAVE on a workstation and verify that the new map backgrounds are properly displayed by calling up various map scales. Also, run WarnGen and create test WarnGen messages to ensure that the proper zones are used.
4. Once you are satisfied that the update maps look correct, delete the .orig files.

15.8 Downloading Local Shapefiles

Local shapefiles can be imported into the maps database using the automation tool. Files should be staged in the following location:

\[/awips2/edex/data/utility/edex_static/site/LLL/shapefiles\]

The shapefiles should be added in a manner similar to the following:

\[\text{submenu/shape_desc/shapefile.(dbf|shp|shx)}\]

It is important to note a few things about this structure. First, the name of the shape file is determined by its parent directory and NOT by the name of the shapefile. Also, all underscore characters “_” will be translated to spaces in the CAVE menu.

Second, the directory name of SUBMENU is optional, but if desired this will create a submenu on your Maps menu in CAVE. This means that you can have a shapefile staged in this manner:

\[/awips2/edex/data/utility/edex_static/site/LLL/shapefiles/submenu/shape_desc/shapefile.shp\]

It will also create a menu entry in CAVE under Maps named shape_desc.

Any shapefiles staged within more than one directory will be listed under a submenu in the Maps pulldown in CAVE, which is named the same as the directory itself. To clarify, whatever text you put in for submenu above will be the text shown as the submenu on CAVE.

For example, the following staging:

\[/awips2/edex/data/utility/edex_static/site/LLL/shapefiles/local_roads/dirt_roads/dirtroad.shp\]
\[/awips2/edex/data/utility/edex_static/site/LLL/shapefiles/local_roads/grass_roads/grassroad.shp\]
will create the following structure in the menu of CAVE:

Maps → local roads → dirt roads

→ grass roads

Finally, it is important to note that only ONE shp, dbf, and shx file can reside in the final directory for staging shapefiles. This is NOT valid staging:

/awips2/edex/data/utility/edex_static/site/LLL SHAPEFILES/CITY_PARKS/PLAYGROUNDS.SHP

/awips2/edex/data/utility/edex_static/site/LLL SHAPEFILES/CITY_PARKS/BIGFIELDSA.SHP

Because there are two .shp files under local_roads, which would be used to create the “city parks” menu. In this case you probably want to create two subdirectories under city_parks – one named playgrounds/ and another bigfields/.

The last important thing to note is that the directory name of shape_desc will determine the table name into which the shapefiles will be imported. It also determines the name of the map bundle created. The name of the map bundle is case sensitive, while the table name will always be all lower case. All underscore characters (‘_’) in shape_desc will be converted to spaces when creating the Maps menu name.

For example, the following shapefiles will create mapdata.oax_county schema in the maps database and create a new entry OAX County directly on the main Maps menu in CAVE:

/awips2/edex/data/utility/edex_static/siote/OAX SHAPEFILES/OAX_County/OAX_County.dbf

/awips2/edex/data/utility/edex_static/siote/OAX SHAPEFILES/OAX_County/OAX_County.shp

/awips2/edex/data/utility/edex_static/siote/OAX SHAPEFILES/OAX_County/OAX_County.shx

Note: The staged directory name does not have to match the file name of the shape file.

In this example, the shapefiles will create mapdata.grr_cwa_roads schema in the maps database, and will create a Roads submenu in the Maps pulldown within CAVE. Under the Roads submenu will be a selection GRR CWA Roads.

/awips2/edex/data/utility/edex_static/site/GRR SHAPEFILES/ROADS/GRR_CWA_Roads/grr_cwa_roads.dbf

/awips2/edex/data/utility/edex_static/site/GRR SHAPEFILES/ROADS/GRR_CWA_Roads/grr_cwa_roads.shp

/awips2/edex/data/utility/edex_static/site/GRR SHAPEFILES/ROADS/GRR_CWA_Roads/grr_cwa_roads.shx
The name of the staging directory is used to create the database table name, as well as the map bundle XML file name. The bundle file name will be created in /awips2/edex/data/utility/cave_static/site/LLL/bundles/maps. For example, the first example above would create OAX_County.xml in this directory.

The label on the Maps menu in CAVE will match the name of the directory, except that every ‘_’ in the directory name will be turned into a space.

The second example above would create GRR_CWA_Roads.xml in /awips2/edex/data/utility/cave_static/site/LLL/bundles/maps/Roads

**Note:** The bundle will only be written by `config_awips2.sh` if the following is true: 1. It does not exist for 2. It exists and was created by `config_awips2.sh`. This means that if a bundle has been edited by the localization perspective since its original creation by `config_awips2`, it will **not** be overwritten.

Once everything is staged correctly, run the following to import the shapefiles staged into the database:

```
./config_awips2.sh shp LLL
```

When the force option (-f) is added to this command, it will also call the `config_ffmp_shapefiles` script to load the FFMP shapefiles.

To remove a shapefile from the system, follow these steps. Note that, for these steps, the example is removing a shape file named My_Shape_File:

1. Remove the shapefile from the system. Identify the shapefile staging directory, and remove the directory

   ```
   cd /awips2/edex/data/utility/edex_static/site/LLL/shapefiles
   rm -rf My_Shape_File
   ```

2. Remove the map bundle XML file, which is also created, and which also has the same name as the staging directory.

   ```
   cd /awips2/edex/data/utility/cave_static/site/LLL/bundles/maps
   rm My_Shape_File.xml
   ```

3. Connect to the database, and drop the table.

   ```
   su - awips
   psql maps
   DROP TABLE mapdata.my_shape_file;
   DELETE FROM mapdata.map_version where table_name='my_shape_file';
   \q
   ```

**Note:** If you are uncomfortable with any of these steps, call the NCF for assistance at 301.713.9344.
Chapter 16
Customization
Chapter 16. Customization

Table of Contents

16.1 New CAVE User Setup and User Preference Setup ........................................................... 1
  16.1.1 New CAVE User Setup .......................................................................................... 1
  16.1.2 User Preference Setup ......................................................................................... 4
16.2 CAVE Menu Customization ........................................................................................... 4
16.3 Shapefiles ..................................................................................................................... 5
  16.3.1 Local Shapefiles .................................................................................................. 5
  16.3.2 Baseline Shapefiles ............................................................................................. 6
16.4 Radar Customization ...................................................................................................... 6
  16.4.1 How to Change the Default RPS Lists ................................................................. 6
  16.4.2 Creating Radar-Specific Pre-localization RPS Lists ............................................. 7
  16.4.3 Adding Radar Four-Panel Displays ..................................................................... 7
  16.4.4 Changing Default Color Enhancements for Radar Imagery ............................... 8
  16.4.5 Modifying the Default Radar Color Curves ....................................................... 8
16.5 Data Monitor: Looking in the Wrong Place for Data .................................................... 9
16.6 Text Customization ..................................................................................................... 10
  16.6.1 Add Versions to Text Database ......................................................................... 10
  16.6.2 Updating the afos2awips.txt file ......................................................................... 11
  16.6.3 Adding Products to the Text Browser ................................................................. 12

List of Exhibits

Exhibit 16.1.1-1. Initial CAVE Startup Dialog ................................................................. 2
Exhibit 16.1.1-2. Valid Connectivity Preferences Selected ............................................... 3
Exhibit 16.1.1-3. CAVE Startup Screen ........................................................................... 3

List of Tables

Table 16.2-1. Baseline Contributions for CAVE-D2D ....................................................... 5
Customization refers to all changes made to the system that are not required for site operation.

**NOTE:** This chapter continues to be a work in progress. Three new sections were added to the last edition of this *System Manager’s Manual* (issued with OB13.5.3), and additional new material will be added to the OB14.2.1 edition, bringing this chapter of the AWIPS II SMM more closely in line with the customization contents of the AWIPS I SMM.

For more information on Customization is this manual, refer to Appendix V, Adding D2D Model Data into GFE, and Appendix X, Configuring metarToShefFiler.xml.

**DISCLAIMER:** If, after any customization, site problems are reported to the NCF, the NCF reserves the right to return your site to the baseliner configuration.

### 16.1 New CAVE User Setup and User Preference Setup

In AWIPS II, the user interface is provided through the Common AWIPS Visualization Environment (CAVE), which replaces the Display Two Dimensions (D2D) tool used in AWIPS, I. The D2D configuration that is done through GUIs will continue to be done in the same way they are done with AWIPS I; therefore, current AWIPS I documentation (e.g., the *User’s Manual*) still applies.

#### 16.1.1 New CAVE User Setup

AWIPS II does not generally distinguish CAVE users from computer users. Simply put, a CAVE user is a computer user who has access to the AWIPS II CAVE application.

Setting up a new CAVE user involves three steps:

1. A standard Linux account is established for the user.
2. CAVE and supporting AWIPS II components are installed for the new user. There are two ways this can be done.
   a. The CAVE package may be installed onto the user’s computer; or
   b. The CAVE package can be installed on a shared drive that is accessible from the user’s computer.
3. The user runs CAVE for the first time.

The first time a user runs CAVE, two events occur.

- First, a directory named *cave Data* is created in the user’s home directory. CAVE uses the *cave Data* directory tree to store local copies of the various configuration files generated by CAVE.
• Second, the user is prompted for the information CAVE needs to connect to its EDEX server and begin operation. The dialog used is the Connectivity Preferences dialog. It is shown in Exhibit 16.1.1-1.

![Exhibit 16.1.1-1. Initial CAVE Startup Dialog](image)

The Initial CAVE Startup Dialog will only be displayed the first time that you use CAVE. This dialog is requesting two critical items of information: 1) the connection address of the EDEX localization server (highlighted in red); and 2) the localization site. The user will need to change localhost (in red highlight) to the IP address or DNS name of the EDEX localization server. After Site, enter the three character designator of the WFO.

Clicking Validate attempts to verify that the user-provided localization server is correct. If so, the red highlight is removed. Note that the message does not change. Also note that the Site is not validated.

**NOTE:** The text never changes from “**Error: Unable to connect…**”, even after a correct server location is entered and Validate is clicked.

**NOTE:** This screen can be bypassed without entering a value for the site; however, CAVE will not localize correctly!

Exhibit 16.1.1-2 shows the Connectivity Preferences with a valid localization server and site entered.
Clicking OK continues the CAVE startup.

This will bring up the CAVE Startup screen, shown in Exhibit 16.1.1-3.

Exhibit 16.1.1-3. CAVE Startup Screen
The user does not have to validate the localization server before clicking OK; however, if
the localization server is invalid, a Connectivity Error dialog is displayed. A valid
localization server must be specified before CAVE will start.

**Note:** It is also suggested that a valid localization site be entered; if the site is not valid,
CAVE started in a default mode.

Clicking *Quit* exits CAVE without completing the startup.

**Note:** Once the initial startup of CAVE is complete, CAVE stores user preferences and
other configurations in the caveData directory tree. In most situations there will be
no need for the user to modify caveData directly.

16.1.2 **User Preference Setup**

Most user preferences (the main exception being the localization information discussed
above) are pre-set when CAVE is first installed. Other preferences are set automatically
by CAVE as the user interacts with the CAVE GUI. For example, CAVE originally
opens with the 5-panel D2D perspective selected, with a default window that is
somewhat smaller than a typical computer monitor. If the user resizes the CAVE
window, opens another perspective (such as the Hydro perspective), switches back to the
D2D perspective, and then exits CAVE, when CAVE restarts, it will be the same size as
before and will display the same perspectives as before. (Data displayed in CAVE will
not be reloaded, however.) This “state memory” is handled automatically by the Eclipse
RCP framework used in CAVE, and no user interaction is required.

The individual CAVE user has the option of changing many of the default preferences as
needed. Most changes are made using dialogs that are similar to the corresponding
application dialogs in AWIPS I. For example, CAVE defaults the image backgrounds in
the D2D perspective to black. If the user desires, the background color can be changed
using *Options-&gt;Set Background Color...* in the D2D perspective menu.

16.2 **CAVE Menu Customization**

The CAVE menu is designed to be highly customizable through XML files. The menu
system allows considerable flexibility in structuring menus. It is possible to structure
menus in as few or as many files as desired to improve manageability. The baseline
CAVE-D2D menu structure is defined in standard Eclipse RCP plugin.xml files in the
following plug-ins:

```
com.raytheon.viz.ui.personalities.awips/plugin.xml
com.raytheon.uf.viz.d2d.ui/plugin.xml
```

Individual CAVE-D2D plug-ins builds on these menus by making their own baseline
contributions. The baseline contributions for CAVE-D2D are shown in Table 16.2-1.
### 16.3 Shapefiles

There are two types of shape files: Local shapefiles and Custom shapefiles

#### 16.3.1 Local Shapefiles

Local shapefiles can be imported into the maps database using the automation tool. Files should be staged in the following location:

```
/awips2/edex/data/utility/edex_static/site/LLL/shapefiles
```

The shapefiles should be added in a manner similar to the following:

```
shape_desc/shapefile.(dbf|shp|shx)
```

The directory name of shape_desc will determine the table name into which the shapefiles will be imported. For example, the following shapefiles will create mapdata.oax_county schema in the maps database:

```
/awips2/edex/data/utility/edex_static/siote/OAX/shapefiles/OAX_County/OAX_County.dbf
```

```
/awips2/edex/data/utility/edex_static/siote/OAX/shapefiles/OAX_County/OAX_County.shp
```

```
/awips2/edex/data/utility/edex_static/siote/OAX/shapefiles/OAX_County/OAX_County.shx
```

**Note:** The staged directory name does not have to match the file name of the shape file.

The name of the staging directory is used to create the database table name, as well as the map bundle XML file name. The bundle file name will be created in

```
/awips2/edex/data/utility/cave_static/site/LLL/bundles/maps
```

 For example, the above would create OAX_County.xml in this directory.

The label on the Maps menu in CAVE will match the name of the directory, except that every ‘\_’ in the directory name will be turned into a space.
Once everything is staged correctly and you are ready to import the shapefiles staged into the database, refer to the *AWIPS II Site Migration Guide*.

### 16.3.2 Baseline Shapefiles

Baseline shapefiles will also be updated using the shp option to config_awips2.sh. They can be maintained in the same manner as AWIPS I, downloaded from the NOAA1 server and staged in nationalData, with the same names as AWIPS I. However, the config_awips2.sh is also aware of the AWIPS GIS Map Group (AGMG) naming convention so the files do not have to be renamed. For example, the public forecast zones can be named c11-zone.shp OR z_public forecast zones can be named c11-zone.shp OR z_mmddyy.shp.

When the above command is run, it will check for updated files, which are newer than those in the /awips2/edex/data/utility/edex_static/base/shapefiles directory. If any are found, it will copy them into the AWIPS II directory, and import them into the database automatically. Please refer to Chapter 15 and AWIPS II Site Migration Guide for more information.

### 16.4 Radar Customization

#### 16.4.1 How to Change the Default RPS Lists

The primary function of the Routine Product Selection (RPS) List Editor is to create or edit a list of up to 150 radar products and send the list to a WSR-88D (Open) Radar Product Generator (ORPG/RPG) for display in the next radar Volume Scan(s). *Note: In the RPS application, each RPS list uses one VCP only.*

Follow these steps to modify the Default Storm Mode and/or the Clear-Air Mode RPS lists.

1. On any workstation, open a Terminal window.
2. As user **awips on DX1**, change to the **rps-lists** directory.
   
   TYPE:  
   ```bash
   cd /data/fxa/rps-lists
   ```

3. Create a new RPS list by using the RPS List Editor and save the newly created file (e.g., `<rps-new>.VCP<xx>.rps`).
   
   Note: This file needs to be saved in /data/fxa/rps-lists

4. TYPE:  
   ```bash
   ls -ltr
   ```
   - Displays the RPS list files included in the directory from earliest to latest `<rps-new>.VCP<xx>.rps` should be listed

5. Copy the new RPS list into the storm mode and clear-air mode files.
Type all on one line:

**TYPE:**
```bash
  cp <rps-new>.VCP<xx>.rps /awips2/rcm/data/config/drop-ins/<RPG_ID>.storm.VCP<xx>
```

**TYPE:**
```bash
  scp <rps-new>.VCP<xx>.rps dx2:/awips2/rcm/data/config/drop-ins/<RPG_ID>.storm.VCP<xx>
```

**TYPE:**
```bash
  cp <rps-new>.VCP<xx>.rps /awips2/rcm/data/config/drop-ins/<RPG_ID>.clear-air.VCP<xx>
```

**TYPE:**
```bash
  scp <rps-new>.VCP<xx>.rps dx2:/awips2/rcm/data/config/drop-ins/<RPG_ID>.clear-air.VCP<xx>
```

**Note:** The above files must be copied to DX2 as well.

6. If desired, go to the RPS List Editor and send the new default.

**Note:** The default RPS list will be sent to the radar whenever the radar’s scan mode changes or the line from AWIPS to the radar is down. If the radar is in clear-air mode, the clear-air mode list will be sent. If in storm mode, the storm mode file will be sent.

### 16.4.2 Creating Radar-Specific Pre-localization RPS Lists

The creation of radar-specific pre-localization RPS applies only to AWIPS I. There is no localization process that copies RPS files in AWIPS II.

### 16.4.3 Adding Radar Four-Panel Displays

There are up to two types of files that must be edited to add a radar 4-panel display to the D-2D menu: Menu files and bundle files. Both types of files can be edited in the CAVE Localization perspective. To make the new display available for all users, create SITE-level override versions of these files. It is possible to create WORKSTATION- or USER-level overrides as well. If an override already exists, make a backup before editing it.

Be aware that, for WSR-88D displays, there are often dual-pol and non-dual-pol versions of a menu file. The dual-pol versions are in a subdirectory named dualPol.

The steps for adding radar four-panel displays follow.

1. Start CAVE on a workstation and open the Localization perspective.

2. Navigate to the **CAVE -> Menus -> radar** folder in the file browser. Determine where in the menu you want the new display to be made available. For example, existing four-panel displays are in menu files names like `baseRadarFourPanel.xml` and `baseRadarFourPanelBestRes.xml`. If you want the display to be in a different menu, trace through radar menus starting with `radarindex.xml` to find the correct file.
3. Edit the menu file determined in the previous step. If there is an existing bundle file that provides the kind of display you want, you can copy and paste an existing `<contribute>` element, changing the product code or elevation parameters as desired. Change the menuText attribute to set the name that will be displayed on the menu item. Change the id attribute to a value that is not used by other menu items.

4. If no existing bundle file is appropriate, create or copy an existing bundle file in the **CAVE -> Bundles** folder. Reference this bundle in the menu entry added in the previous step.

5. Restart CAVE (or start a new instance of CAVE) to test the new menus. **Note:** When testing your new four-panels, be sure the radar is in the proper VCP mode to supply the slices on the four-panel. Also be sure that you have sent an RPS list from AWIPS to the radar which contains the products on your four-panel display.

### 16.4.4 Changing Default Color Enhancements for Radar Imagery

Follow these steps to change the default color enhancements for radar imagery.

1. Start CAVE on a workstation and open the Localization perspective.

2. Determine the color curve you want to use. These can be found in the Localization perspective file browser under **CAVE -> Color Maps -> Radar**.

3. For each radar product that you want to assign a new color map to, determine the product code. These can be found by looking at the radar menu or bundle files. On the dx3 or dx4, the file `/awips2/edex/data/utility/common_static/base/radarInfo.txt` lists products with their associated codes.

4. Create a SITE-, WORKSTATION- or USER- level override of **CAVE -> Style Rules -> radarImageryStyleRules.xml**. If the override already exists, create a backup before editing it.

5. Look for the `<styleRule>` element containing `<parameter>\n</parameter>` where `n` is the product code. Change the value of the `<defaultColormap>` element to reference the color curve you want to use. Note that color maps are organized in subfolders and the `'/'-separated path must be used here, for example, "Radar/OSF/SRM Radial Velocity".

   If the `<styleRule>` element contains multiple `<parameter>` elements, all the referenced products will be affected. If this is not desired, make a copy of the original `<styleRule>` element, edit the copy's list of parameters so that it contains only relevant products, and then remove those parameters from the original rule.

6. Restart CAVE (or start a new instance of CAVE) to test the new style rules.

### 16.4.5 Modifying the Default Radar Color Curves

Follow these steps to modify the default radar color curves.

1. Navigate to **CAVE -> Color Maps -> Radar** in the file browser.
2. Double click on the color map you want to edit. This will open the color map editor for the file. XML and plain text editors are available by clicking on the relevant tab at the bottom of the editor.

3. Edit the file and save it.

4. Restart CAVE (or start a new instance of CAVE) to test the new color maps.

### 16.5 Data Monitor: Looking in the Wrong Place for Data

If the data (available within AWIPS II) is missing in the Data Monitor, then the Data Monitor is looking for the data in the wrong place.

Follow these steps to view the data monitor files.

1. Log in to PX1. As user root (or any user in fxalpha group), change to the directory where the Data Monitor files reside.

   TYPE: `cd /data/fxa/A2_SysMonitor/config`

2. Find all the data files ending in `.xml` that are used by the Data Monitor (e.g., `satellite.xml`, `grid.xml`, `radar.xml`, etc.).

   TYPE: `ls *.xml`

   Find all the data files ending in `.xml` that are used by the Data Monitor (e.g., `satellite.xml`, `grid.xml`, `radar.xml`, etc.).

3. Edit the file you need using a text editor. The `radar.xml` sample is shown as an example below.

   ```xml
   <radar>
   <data>
   <productname>KLWX 8-Bit Base Reflectivity (Z) elev0_5</productname>
   <sqltxt>SELECT reftime FROM radar WHERE productcode = '94' AND location_rda_id = 'KLWX' AND primaryelevationangle = '0.5' ORDER BY reftime DESC LIMIT 1</sqltxt>
   <warnthresh>720</warnthresh>
   <errorThresh>1440</errorThresh>
   </data>
   </radar>
   
   • The text inside the `<productname>` tag will be the text displayed on the data monitor page.
   
   • The text in the `<sqltxt>` tag is the SQL query to be executed against the metadata database. This returns the latest reftime of the product. This is what is used to compare to the threshold values.
   
   • The text in the `<warnthresh>` tag is the time, in seconds, where the data is considered late since its last arrival. This will be displayed with a yellow caution symbol.
   
   • The text in the `<errorthresh>` tag is the time, in seconds, where the data is considered very late since its last arrival. This will be indicated by a red X symbol in the data monitor.
4. If the data is missing in the Data Monitor but not in AWIPS, the Data Monitor may be using the wrong SQL query to look for times in the database. You should double check the SQL command, possibly even using psql or pgadmin to verify that it works.

5. Save the modified .xml file and observe the next time that the Data Monitor is refreshed in the browser.

16.5.1 How to Add or Delete a Product from the Data Monitor

To add or delete a product from the Data Monitor, begin by completing steps 1 through 4 in section 16.5 Data Monitor: Looking in the Wrong Place for Data. Then do the following.

To add a product

1. To add a product, add another set of XML tags between, and including the <data> </data> tags. Modify the necessary XML elements as desired.

2. Save the modified .xml file and observe the next time that the Data Monitor is refreshed in the browser.

To delete a product

1. To delete a product, simply remove the lines between, and including, the <data> </data> tags.

2. Save the modified .xml file and observe the next time that the Data Monitor is refreshed in the browser.

16.6 Text Customization

16.6.1 Add Versions to Text Database

The textdb -v CCCNNNXXX n command can be used to change the number of versions stored “on the fly” to n versions (where n is an integer).

For example, to change the number of versions of the LAXFLSSAN products from 2 to 10, as user fxax or using your individual user account, enter the command:

```
TYPE:   textdb -v LAXFLSSAN 10
```

This command can be used to increase or decrease the number of versions stored.

If a site needs to change all the FLS products to n versions, rather than individually issuing this command for each product, use the following procedure as user postgres on dx1,

```
TYPE:   psql fxatext << !
TYPE:   update textproductinfo
```
By changing the value of `versionstokeep` and `nnnid`, this procedure can be adapted to change the number of versions stored for other product categories.

The versions of each product are stored in a circular queue. An individual product will need to wait until its queue pointer for that product is set to the last element in the queue before the number of versions change.

You can list the FLS products in the `fxatext` database by typing the following sequence of commands:

```
set versionstokeep = 10
where nnnid = 'FLS';
```

```
psql fxatext << !
select * from textproductinfo
where nnnid = 'FLS';
```

16.6.2 Updating the `afos2awips.txt` file

The `afos2awips.txt` file maps AFOS PILs to WMO headers. When an AWIPS site transmits a locally created product via the AWIPS WAN, the locally created product must be transmitted by its WMO header. The `afos2awips.txt` file is the configuration file that translates the site’s local AFOS PILs into WMO headers. If a product is missing from the `afos2awips.txt` file, that product will not be transmitted via the AWIPS WAN to the NCF for distribution via the SBN.

The `afos2awips.txt` file is a “living” document that needs to be updated in a timely manner to meet AWIPS WAN transmission requirements. As a “living” document, AWIPS sites will need to occasionally get and install updates of the `afos2awips.txt` file from the NOAA1 server. AWIPS sites should verify that all of their locally created products are included in the `afos2awips.txt` file. To expedite corrections to the `afos2awips.txt` file, changes should be sent to the NWS Data Managers at nws.dm@noaa.gov. The `afos2awips.txt` file is updated weekly. The `afos2awips.txt` file should always be checked to ensure that site products are correct.

**Reviewing the updated `afos2awips.txt` file**

Complete these steps to review the updated `afos2awips.txt` file:
1. At an AWIPS workstation, open a Terminal window. Sites can temporarily store the updated `afos2awips.txt` file in the `/awips/fxa/awipsusr` directory for review. As user fxa,

   TYPE: `cd /awips/fxa/awipsusr`

2. SFTP into NOAA1 ftp server

   TYPE: `sftp ftpawips@165.92.25.137`

3. Once you are connected you will be prompted for the `ftpawips` password, which is !SAWIPS4

   At the `sftp>` prompt:
   TYPE: `binary`
   TYPE: `hash`
   TYPE: `cd /awips/fxa/data`
   TYPE: `get afos2awips.txt`
   TYPE: `bye`

The downloaded `afos2awips.txt` file is now stored on the AWIPS workstation in the `/awips/fxa/awipsusr` directory. Review the downloaded file to ensure it includes all site locally created products.

**Updating the afos2awips.txt file**

On DX3, copy the `afos2awips.txt` file to `/awips2/edex/data/ndm`. This will update the appropriate database table. To confirm the changes, after the file is taken out of `/awips2/edex/data/ndm` check in both the dx3 and dx4 edex-ingest logs by typing: `grep afos2awips /awips2/edex/logs/edex-ingest-<yyyymmdd>.log` to verify that it was processed. From the fxatext database, the user must type: `select * from afos_to_awips where afosid = '<CCCNNNXXX>'` where CCC is the node NNN is the product type and XXX is the site id.

16.6.3 Adding Products to the Text Browser

Complete the following steps to add products to the text browser. [Note: These steps are written as though your AWIPS site is localized to MTR. When carrying out these steps, replace MTR with your own AWIPS three-letter localization ID.]

1. Exit all D2D and text workstation sessions on the workstation.

2. Open a Terminal window on DX3, and as user fxa, change to the localization directory for your site.

   TYPE: `cd /awips/fxa/data/localization/MTR`

3. Type, all on one line:
4. If the product you wish to add to the text browser is not in MTR-afosMasterPIL.txt, edit MTR-afosMasterPIL.txt to add it (add in alphabetical order).

5. Copy the edited file back to the nationalData directory. Type, all on one line:

   TYPE: cp MTR-afosMasterPIL.txt /data/fxa/nationalData/afosMasterPIL.txt

6. Type, all on one line:

   TYPE: cp /awips/fxa/data/textCategoryClass.txt MTR-textCategoryClass.txt

7. Edit MTR-textCategoryClass.txt to add the product and class in which it is to be available in the text browser.

   A sample line would look like **AFD 1 2 15 16**

   Where **AFD** = Area Forecast Discussion

   The **numbers** that follow AFD indicate which classes the data is in. So the AFD would be available in the public (1), aviation (2), national obs. (15), and narr/disc./codes (16) class pulldown of the text browser.

8. Add products that are not in MTR-textCategoryClass.txt alphabetically.

9. Type, all on one line:

   TYPE: cp MTR-textCategoryClass.txt /awips/fxa/data/textCategoryClass.txt

10. Change to the scripts directory to rerun localization. Still as user fxa,

    TYPE: cd /data/fxa/sdc

    TYPE: ./config_awips2.sh ndm MTR

11. Close the Terminal window. Then select afosMasterPIL.txt and textCategoryClass.txt and copy to other workstations.

12. Start D2D and text workstations and bring up text browser to test whether your changes are evident.
Chapter 17
AWIPS System Monitor
## Chapter 17. AWIPS System Monitor

### Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>17.1</td>
<td>System Monitor Overview</td>
</tr>
<tr>
<td>17.2</td>
<td>SCAN and FFMP Data Monitoring System</td>
</tr>
<tr>
<td>17.3</td>
<td>Pages and Templates</td>
</tr>
<tr>
<td>17.4</td>
<td>Configuration Files</td>
</tr>
<tr>
<td>17.5</td>
<td>Data Display Tools</td>
</tr>
<tr>
<td>17.5.1</td>
<td>PG Admin III</td>
</tr>
<tr>
<td>17.5.2</td>
<td>HDFView</td>
</tr>
<tr>
<td>17.5.3</td>
<td>CAVE Product Browser</td>
</tr>
</tbody>
</table>

### List of Exhibits

- Exhibit 17.1-1. Data Set Status Symbols | 1
- Exhibit 17.1-2. AWIPS Start Menu – Firefox Web Browser | 1
- Exhibit 17.1-3. AWIPS System Monitor Default Main Page | 2
- Exhibit 17.1-4. AWIPS System Monitor Main Page (LDAD Monitoring) | 3
- Exhibit 17.1-5. AWIPS System Monitor Main Page (SCAN Data Monitoring) | 4
- Exhibit 17.1-6. AWIPS System Monitor Main Page (FFMP Monitoring) | 5
- Exhibit 17.1-7. AWIPS System Monitor DATA: Satellite Products Status | 6
- Exhibit 17.1-8. AWIPS System Monitor DATA: Radar Products Status | 7
- Exhibit 17.1-9. AWIPS System Monitor DATA: Point Data Products Status | 8
- Exhibit 17.1-10. AWIPS System Monitor DATA: Grid Products Status | 9
- Exhibit 17.1-11. AWIPS System Monitor DATA: Redbook Products Status | 10
- Exhibit 17.1-12. AWIPS System Monitor DATA: Disk Usage | 11
- Exhibit 17.1-13. AWIPS System Monitor DATA: CPU Utilization | 12
- Exhibit 17.1-14. Sample CPU Utilization Display | 13
- Exhibit 17.1-15. AWIPS System Monitor LDAD: LDAD Processes | 14
- Exhibit 17.1-16. AWIPS System Monitor LDAD: LDAD Acquisition Data | 15
- Exhibit 17.1-17. AWIPS System Monitor LDAD: LDAD Dissemination Data | 16
- Exhibit 17.1-18. EXTRAS: Online Documentation and Other Sites | 17
- Exhibit 17.2-1. SCAN Data Monitoring System | 18
- Exhibit 17.2-2. FFMP Data Monitoring System | 19
- Exhibit 17.5.1-1. PG Admin III Main Window | 22
- Exhibit 17.5.1-2. PG Admin III New Server Registration Dialog | 23
Exhibit 17.5.1-3. PG Admin III Edit Data Window ................................................................. 24
Exhibit 17.5.1-4. PG Admin III SQL Query Window .............................................................. 24
Exhibit 17.5.2-1. HDFView Main Window ........................................................................... 25
Exhibit 17.5.2-2. HDFView with File Loaded and Expanded ...................................................... 26
Exhibit 17.5.2-3. HDFView Dataset Selection Dialog ............................................................. 26
Exhibit 17.5.2-4. HDFView Tableview (Spreadsheet) of Data ................................................... 27
Exhibit 17.5.2-5. HDFView Image View of Data ................................................................ 27
Exhibit 17.5.3-1. CAVE with Product Browser Displayed ....................................................... 28
Exhibit 17.5.3-2. Satellite Product Loaded from Product Browser ........................................... 29

List of Tables

Table 17.5-1. Data Display Tools .......................................................................................... 22
17.1 System Monitor Overview

The AWIPS System Monitor is a software-based system used to monitor resources and performance (DATA, LDAD, SCAN, and FFMP). The monitoring software is provided separately from the other AWIPS re-host code. This allows for flexible installation options. The AWIPS System Monitor can be installed on any platform running a stand-alone or clustered instance of the AWIPS II software.

When left running, the AWIPS System Monitor automatically updates the displayed AWIPS information, indicating data set status with iconic symbols, as shown in Exhibit 17.1-1.

Exhibit 17.1-1. Data Set Status Symbols

On-time and complete data sets are flagged by a green icon on the monitor display. Late and/or incomplete data sets are flagged by a yellow or red icon, and notice is sent to the Network Control Facility (NCF). Efforts by site staff to track down and correct data delays must be closely coordinated with the NCF.

The AWIPS System Monitor uses a set of scripts to check on the timeliness of various data sets. It produces a display that can be used to track data ingest. The AWIPS System Monitor is accessed via the Internet, which is invoked from the AWIPS Start menu by selecting Firefox Web Browser as shown in Exhibit 17.1-2.

Exhibit 17.1-2. AWIPS Start Menu – Firefox Web Browser

The AWIPS System Monitor Default main page is shown in Exhibit 17.1-3.
Exhibit 17.1-3. AWIPS System Monitor Default Main Page

The system monitor consists of two distinct components:

- **Client**
- **Server**

The client component provides the information in a Web-based format. The server component is responsible for displaying that information. The server component is the component associated with the display portion of the data monitor. It is called a server component because the files associated with it need to be installed on the machine that runs the Mozilla server, which is accessed via the Firefox Web Browser (refer to Exhibit 17.1-2). The client component comprises the files necessary to monitor the data. These files reside on each machine that the user wants to monitor.

The structure of the System Monitor is as follows:

- **System Monitor Main Page.** The System Monitor Main Page, as shown in Exhibit 17.1-3, lists the top-level statuses of the products or processes of the
DATA monitor selection. This is the default System Monitor page that initially opens when the System Monitor URL is entered.

- **LDAD Monitor Selection**: The LDAD monitor selection lists the top-level statuses of the products or processes associated with LDAD, as shown in Exhibit 17.1-4.

![LDAD Monitor Selection](image)

*Exhibit 17.1-4. AWIPS System Monitor Main Page (LDAD Monitoring)*

- **Other Monitor Selections**: The other monitor selections are SCAN (shown in Exhibit 17.1-5) and FFMP (shown in Exhibit 17.1-6).
Exhibit 17.1-5. AWIPS System Monitor Main Page (SCAN Data Monitoring)
Individual Data Set Pages. Individual pages are opened by clicking on any of the links located on the collapsed main page (DATA and LDAD). There is a set of links for each data type for each machine being monitored. Individual pages contain detailed listings of products and processes within each data set and their respective state.
• **DATA Products Status.** Exhibits 17.1-7 through 17.1-11 illustrate the individual pages for monitoring DATA products, Disk Usage, and CPU Utilization.

![AWIPS System Monitor DATA: Satellite Products Status](image)

**Exhibit 17.1-7. AWIPS System Monitor DATA: Satellite Products Status**
Exhibit 17.1-8. AWIPS System Monitor DATA: Radar Products Status
Exhibit 17.1-9. AWIPS System Monitor DATA: Point Data Products Status
Exhibit 17.1-10. AWIPS System Monitor DATA: Grid Products Status
Exhibit 17.1-11. AWIPS System Monitor DATA: Redbook Products Status

Disk Usage Monitoring and CPU Utilization Monitoring is updated via cron every 20 minutes. These two options, which are shown in Exhibits 17.1-12 and 17.1-13, display the usage and performance of the monitored workstation. Both are accessible from either the DATA or LDAD monitoring screens. In order to display the CPU utilization chart on the Data or LDAD Monitor pages under the menu CPU Utilization Monitoring, click the link:

“Click here to display CPU Usage for the system”

This will execute the script /awips/fxa/bin/showCPU.sh and redirect the user to a new page. While executing, it will display the following message “Attempting to launch the CPU monitor on $ipAddress” where $ipAddress is the IP address of the machine launching the script. A sample CPU utilization display is shown in Exhibit 17.1-14.
NOTE: The showCPU.sh script can only be executed on a valid LX or XT workstation on the same subnet as the local LAN.

Exhibit 17.1-12. AWIPS System Monitor DATA: Disk Usage
Exhibit 17.1-13. AWIPS System Monitor DATA: CPU Utilization
Chapter 17  AWIPS System Monitor

Exhibit 17.1-14. Sample CPU Utilization Display

- **LDAD Processes and Products Status.** Exhibits 17.1-15 – 17.1-17 illustrate the individual pages for monitoring LDAD Processes and Products.
Exhibit 17.1-15. AWIPS System Monitor LDAD: LDAD Processes
Exhibit 17.1-16. AWIPS System Monitor LDAD: LDAD Acquisition Data
Exhibit 17.1-17. AWIPS System Monitor LDAD: LDAD Dissemination Data

- **EXTRAS:** Provides link for Online Documentation for the User’s Guide and AWIPS System Manager’s Manual is available as shown in Exhibit 17.1-18.
17.2 **SCAN and FFMP Data Monitoring System**

The SCAN_DMS and FFMP_DMS have been incorporated into the AWIPS II System Monitor, as shown in Exhibits 17.1-5 and 17.1-6. Detailed information for these two monitored systems follows.

The information loaded on the page occurs on page-load or refresh. Depending on the amount of data being kept in the data store and the number of SCAN radars in use, the page load can vary between 5 and 15 seconds. The legacy SCAN_DMS loads blank and requires the user to click on a button to run an external script.

The default `/var/www/html/config/scan_radar.xml` ships with an Omaha localization. Replace the `<rda>` tags with the RDA name of the SCAN-enabled radars. An identical configuration exists for the FFMP DMS page named `ffmp_radar.xml`.

```
<radar>
  <data>
    <rda>KOAX</rda>
    <errorthresh>600</errorthresh>
  </data>

  <data>
    <rda>KUEX</rda>
    <errorthresh>600</errorthresh>
  </data>
</radar>
```
The FFMP Data Monitoring System (DMS) contains a second configuration file for the RFC grids named `ffmp_grids.xml`.

**Note:** The `ffmpPurgeRules.xml` should not allow USER or Workstation overrides.

The default installation ships with all the RFC grids being monitored and can be trimmed as needed by each site.

The grid monitoring on the SCAN DMS page is controlled inside the `SCAN.php` template. As on the Data Monitor, simple SQL queries are executed against the metadata database. Lightning strikes are obtained by executing a uEngine script named `bltool.py`, which queries the HDF5 and sums up the past 15 minutes’ worth of lightning data. See Exhibit 17.2-1.

![Exhibit 17.2-1. SCAN Data Monitoring System](image-url)
The FFMP monitor also displays the status of the ffmp_plugin (below the DAT logo). It monitors only the primary EDEX server as defined in the **params.xml** file. See Exhibit 17.2-2.

![FFMP Data Monitoring System](image)

**Exhibit 17.2-2. FFMP Data Monitoring System**

### 17.3 Pages and Templates

The Apache directory is `/etc/httpd/conf`. Any pre-existing configuration inside this location on the px1 is not baseline. The existing `httpd.conf` will be saved before installation and renamed `AWIPS1-httpd.conf`. This is the apache server configuration, which rarely needs to be deviated from the national baseline.

The installation directory for the AWIPS II System Monitor is: `/var/www/html` and all configuration, scripts, and page templates are installed here.

The main php page templates are in the top-level directory and are:

- **index.php**: default page loaded when browsing [http://px1f](http://px1f)
- **LDAD.php**: LDAD System Monitor loaded when browsing [http://px1f/LDAD.php](http://px1f/LDAD.php)
SCAN.php  SCAN DMS loaded when browsing http://px1f/SCAN.php
FFMP.php  FFMP DMS loaded when browsing http://px1f/FFMP.php

All actions performed on page load, such as SQL queries to the metadata database, are contained inside these page template files.

There is no persistent process running and sending information to the AWIPS II System Monitor. All database queries and scripts are run on page load, refresh, or after refresh threshold is lapsed. The exceptions are the CPU and disk monitoring sections, which run via cronjob.

There are 4 root cronjobs that run to monitor the disk and cpu statistics. These crons are in /etc/ha.d/cron.d/a2px1cron and run on the server running the a2px1apps package, default PX1. The crons populate the Disk Monitor and CPU Monitor sections of the System and LDAD Monitor pages.

```bash
# cron to update disk monitor portion of awips II system monitor
*/20 * * * * /var/www/html/scripts/diskMon.sh >> /dev/null
*/20 * * * * /var/www/html/scripts/ldad_diskMon.sh >> /dev/null

# cron to update CPU monitor portion of awips II system monitor
*/20 * * * * /var/www/html/scripts/cpuMon.sh >> /dev/null
*/20 * * * * /var/www/html/scripts/ldad_cpuMon.sh >> /dev/null
```

**Note:** The AWIPS II rehost script will configure the heartbeat and cron appropriately. If the rehost script is not run, a manual insert of the above 4 cronjobs is required. In order to make the AWIPS II System Monitor highly available without running the rehost script, simply add the httpd (apache service) to /etc/ha.d/resource.d/a2px1apps.

### 17.4 Configuration Files

The configuration files are contained in /var/www/html/config.

The environment configuration is contained in params.xml
The values entered to the left are default and as shipped. Should the baseline system architecture change, these values may need to be changed as well to reflect such.

The <refresh> options are for the PHP template pages and are in seconds.

The configuration for the various products monitored are contained in this directory.

<table>
<thead>
<tr>
<th>XML File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>satellite.xml</td>
<td>The SQL commands execute on page load or refresh and are contained in these files and under the tag &lt;sqltext&gt;. The warning and error threshold for each data type is also contained therein, defined by &lt;warnthresh&gt; and &lt;errorthresh&gt;. The name to display in the monitor is defined by the &lt;productname&gt; tag.</td>
</tr>
<tr>
<td>radar.xml</td>
<td>The ffmp_grids.xml contains the configuration for the RFC grids while the ffmp_radar.xml contains the configuration for the radars.</td>
</tr>
<tr>
<td>point.xml</td>
<td>scan_radar.xml contains the radars for which SCAN is applicable.</td>
</tr>
<tr>
<td>grid.xml</td>
<td>To add products or radars to any of these configuration files, simply copy a pre-existing XML tag and change the appropriate variables, e.g., &lt;rda&gt;.</td>
</tr>
<tr>
<td>graphic.xml</td>
<td></td>
</tr>
<tr>
<td>ldad.xml</td>
<td></td>
</tr>
<tr>
<td>ffmp_grids.xml</td>
<td></td>
</tr>
<tr>
<td>ffmp_radar.xml</td>
<td></td>
</tr>
</tbody>
</table>

17.5 Data Display Tools

Several tools are available for data display/system monitoring. This section introduces three useful tools: PG Admin III; HDFView; and CAVE Product Browser. Table 17.5-1 provides a brief description of each tool.
Table 17.5-1. Data Display Tools

<table>
<thead>
<tr>
<th>Tool</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PG Admin III</td>
<td>PG Admin III is a PostgreSQL-specific database administration tool. PG Admin III provides a graphical user interface (GUI)-based interface to the database and allows the user to view data and run SQL commands. PG Admin III is part of the existing AWIPS toolset.</td>
</tr>
<tr>
<td>HDFView</td>
<td>HDFView is a GUI-based tool that provides an Explorer-style interface into the contents of an HDF5 file. HDFView is part of the AWIPS toolset.</td>
</tr>
<tr>
<td>CAVE Product Browser</td>
<td>The CAVE Product Browser is part of CAVE, the AWIPS II visualization environment. Used in conjunction with the CAVE D2D perspective, it provides access to displayable data and other products available in AWIPS II.</td>
</tr>
</tbody>
</table>

17.5.1 PG Admin III

PG Admin III is part of the existing AWIPS toolset and is available for use after AWIPS II has been installed. In the current architecture, it is installed in /usr/local/pgadmin3 and is accessible from AWIPS LX and XT workstations. PG Admin III provides a GUI interface to a PostgreSQL database.

To run PG Admin II, you will need to:

1. Log on to an LX or XT workstation.
2. Open a terminal window.
3. In the terminal window, change directory to /usr/local/pgadmin3/bin.
4. In the terminal window, enter: ./pgadmin3 &.

This will display PG Admin III’s main window, shown in Exhibit 17.5.1-1.

![PG Admin III Main Window](image)
The first time PG Admin is used with the AWIPS II databases, you will need to set up a connection to the database. The connection is established using the **New Server Registration** dialog (shown in Exhibit 17.5.1-2). The **New Server Registration** dialog is accessed via the “connection to server” button ( ) on the PG Admin III toolbar.

![Exhibit 17.5.1-2. PG Admin III New Server Registration Dialog](image)

When you register a new server, you need to provide four values:

1. The name of the database. This can be anything you want but it should be something meaningful such as “AWIPS II Database.”
2. The name of the host system. For AWIPS II, the host system is `dx1f`.
3. The username. For AWIPS II, the correct user name is `awips`.
4. The password.

To use PG Admin to view data, locate the desired table in the tree view on the left side of the main window and click the ‘view data’ button ( ) on the toolbar. The table will display in the Edit Data window (see Exhibit 17.5.1-3).

**Note:** The Edit Data window opens and displays a database cursor; any changes made to the data in this window will propagate to the underlying database.

PG Admin III can also be used to execute SQL against the database. In order to execute SQL, you need to open the PG Admin II SQL Query window. To open this window, first find the desired database in the tree view on the left side of the main window and then click the ‘execute SQL’ button ( ) in the toolbar. This will display the SQL Query window, shown in Exhibit 17.5.1-4.

**Note:** Use extreme care when executing SQL statements against the database. SQL is an extremely powerful database manipulation language and can be used to alter the structure and/or contents of the database. A change made to the database generally cannot be reversed.
Chapter 17

AWIPS System Monitor

Exhibit 17.5.1-3. PG Admin III Edit Data Window

Exhibit 17.5.1-4. PG Admin III SQL Query Window
17.5.2 **HDFView**

HDFView is a GUI-based tool that allows you to view the data in an HDF5 file. It provides an Explorer-style interface for locating the data and various options for viewing the data. HDFView is most effective when used to view raster data such as satellite imagery. HDFView is installed in `/awips2/tools/bin/HDFView` on DX1 and DX2.

To run HDFView, you will need to do the following:

1. Log on to an LX or XT workstation.
2. Open a terminal window.
3. Remote log into dx1f.
4. Change directory to `/awips2/tools/bin/hdfview2.7`.
5. Enter: `./hdfview`

This will display the main HDFView window (see Exhibit 17.5.2-1).

![HDFView Main Window](image)

*Exhibit 17.5.2-1. HDFView Main Window*

To locate and load an HDF5 file, select ‘File ➔ Open Read-Only’ from the HDFView menu. This will display a file open dialog. Browse for the desired file and click ‘Open’. This will display the file contents in a tree structure on the left side of the main window (see Exhibit 17.5.2-2).
To view the product:

1. Expand the tree view to find and click on the data for the desired product.
2. Right click on the data and select ‘Open As’.

This opens the Dataset Selection dialog, shown in Exhibit 17.5.2-3.

From this dialog, you can display the data as a spreadsheet or as an image. Use care when displaying data in the spreadsheet (table) view because the data can be modified in this view. Exhibit 17.5.2-4 illustrates the spreadsheet view; the image view is shown in Exhibit 17.5.2-5.
Exhibit 17.5.2-4. HDFView Tableview (Spreadsheet) of Data

Exhibit 17.5.2-5. HDFView Image View of Data
17.5.3 CAVE Product Browser

The CAVE Product Browser provides an Explorer-style tree view of data and other products that may be viewed in CAVE. It may be added to the selected CAVE perspective by selecting ‘CAVE ➔ Data Browsers ➔ Product Browser’ from the CAVE menu system.

Note: Although the CAVE Product Browser may be added to any perspective, this discussion demonstrates adding it to the D2D perspective.

To use the CAVE Product Browser, first launch CAVE on an AWIPS LX or XT workstation. Once CAVE is running, select ‘CAVE ➔ Data Browsers ➔ Product Browser’ from the CAVE menu system. The Product Browser will be added to the right side of the active perspective. (See Exhibit 17.5.3-1.)

Once the Product Browser has been loaded, expand the desired product type node to find the desired product; then right click on the product and select ‘Load Product’. After a short pause, the selected product will load (see Exhibit 17.5.3-2). [Note: For GRIB]
products, right click on the product and select ‘Load As → Contour’ or ‘Load As → Image’.

Exhibit 17.5.3-2. Satellite Product Loaded from Product Browser

Once the product has been loaded, it can be manipulated the same as any product loaded into D2D. Note in particular that time matching applies to products loaded by this method. As a result, attempting to load multiple products may result in errors being reported in AlertViz.
Chapter 18
Failover Management Procedures
Chapter 18. Failover Management Procedures

Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>18.1 Communications Processors</td>
<td>1</td>
</tr>
<tr>
<td>18.1.1 SBN Communications Processors</td>
<td>1</td>
</tr>
<tr>
<td>18.1.1.1 SBN Communications Processors Failover Procedures</td>
<td>2</td>
</tr>
<tr>
<td>18.1.1.2 Monitoring Data Acquisition on SBN Communications Processor</td>
<td>3</td>
</tr>
<tr>
<td>18.2 Linux Servers</td>
<td>5</td>
</tr>
<tr>
<td>18.2.1 Red Hat Cluster Manager</td>
<td>6</td>
</tr>
<tr>
<td>18.2.1.1 Linux Data Server Failover Scenarios</td>
<td>6</td>
</tr>
<tr>
<td>18.2.1.2 Linux Preprocessor Server Failover Scenarios</td>
<td>8</td>
</tr>
<tr>
<td>18.2.2 Post-Failover Events</td>
<td>11</td>
</tr>
<tr>
<td>18.2.3 NCF Post-Failover Actions</td>
<td>11</td>
</tr>
<tr>
<td>18.3 River Ensemble Processors (REP)</td>
<td>12</td>
</tr>
<tr>
<td>18.3.1 REP Server High-Availability Software</td>
<td>12</td>
</tr>
<tr>
<td>18.3.2 REP Server Failover Scenarios</td>
<td>13</td>
</tr>
<tr>
<td>18.3.3 Managing the Heartbeat Cluster Service</td>
<td>15</td>
</tr>
<tr>
<td>18.3.4 EDEX/IPVS, LDM Failovers</td>
<td>15</td>
</tr>
<tr>
<td>18.4 AWIPS II to AWIPS I Fallback</td>
<td>15</td>
</tr>
<tr>
<td>18.4.1 AWIPS I/AWIPS II Co-existence</td>
<td>16</td>
</tr>
<tr>
<td>18.4.2 Swapping Procedures</td>
<td>17</td>
</tr>
<tr>
<td>18.4.3 Limitations on System Swapping</td>
<td>18</td>
</tr>
</tbody>
</table>

List of Exhibits

<table>
<thead>
<tr>
<th>Exhibit</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>18.2.1.1-1</td>
<td>Cluster Status Monitor</td>
<td>8</td>
</tr>
<tr>
<td>18.2.1.2-1</td>
<td>Sample Output</td>
<td>10</td>
</tr>
<tr>
<td>18.2.1.2-2</td>
<td>Sample Verbose Output from Linux Cluster Status Command</td>
<td>10</td>
</tr>
</tbody>
</table>
Chapter 18 Failover Management Procedures

This chapter describes AWIPS failover management procedures. It contains three sections. Section 18.1 describes the communications processors. Refer to Section 18.2 for the Linux data servers (DX) failover management procedures. Section 18.3 describes the River Ensemble Processors (REP).

18.1 Communications Processors

The Communications Processors (CP) are the front-end processors that allow data to be acquired from sources external to the site. These CPs do not execute hydrometeorological applications. Their primary function is to provide protocol support to the external communication lines.

The SBN CPs support ingest of NESDIS and NWSTG/NWSTG2 data from the Network Control Facility. Failover procedures for each are discussed in this section.

18.1.1 SBN Communications Processors

The Satellite Broadcast Network (SBN) is the network that distributes weather data from the NCF to receiving sites throughout the United States. The network consists of the uplink Satellite Broadcast Processor (SBP) at the Master Ground Station (MGS) or backup MGS, the uplink MGS antenna in Hauppauge, New York; the backup MGS antenna at the NASA IV&V Facility in Fairmont, West Virginia; the SES-1 satellite; the downlink antenna at the locations receiving data; and the downlink SBP or NOAAPORT Receive Processor (NRP).

There are two components to the ingest software on the CPSBNs in AWIPS II:

1. **LDM NOAAport receiver software**. Ingest from DVB to CPSBN.

2. **UniData LDM software**. Routing/ingest from CPSBN acting as an LDM host (running on cp1f in normal configuration).

In AWIPS II the **LDM NOAAport receiver software** will be used for ingesting all four NOAAPORT Channels.

At each AWIPS site, there are two CPSBNs responsible for receiving GOES products and a combination of NCEP products from the NCF. No site has more than two CPSBNs unless it has an NRS. CPSBN1 and CPSBN2 function as a heartbeat cluster; failover is managed automatically or manually. Note that only a single high-availability package is currently used on this cluster. The package name is a2cp1apps. In failover mode, package execution shifts to the backup server. The a2cp1apps package manages the cp1f floating server name. Note that floating names are available only when the high-availability packages are running; they cannot be used to start or stop the high-availability packages.

First, check for the running of the LDM, as user ldm:

```
TYPE: ps -U ldm -u ldm u
```
To check whether all DVB-S channels are running on cpsbn1 in AWIPS II:

```
TYPE: config_dvb
```

Found all DVB channels NMC GOES NMC2 NOAAPORT_OPT NMC3
Checking file and shmem status for cpsbn1-tbw3

*************************************************************************
Link  SBN                        ---config_file--- -------- current_status -------
Channel                    type        rd_enable    reader           dh_proc   MCAST(DVB)
0      NMC                        MCAST     ON        DOWN           DOWN          n/a
1      GOES                       MCAST     ON        UNKNOWN  UNKNOWN  n/a
2      NMC2                      MCAST     ON        UNKNOWN  UNKNOWN  n/a
3      NOAAPORT_OPT  MCAST     ON        UNKNOWN  UNKNOWN  n/a
4      NMC3                      MCAST     ON        UNKNOWN  UNKNOWN  n/a

**NOTE:** Ensure that the rd_enable column reads ON. UNKNOWN status is nominal for the dh_proc and MCAST(DVB) columns when running in AWIPS II as these are AWIPS I-specific statistics.

In AWIPS II the **UniData LDM** software is used as a replacement for the CP ingest software.

- The file `pqact.conf` located at `/usr/local/ldm/etc` in the LDM installation contains the configuration information that tailors the data flow to AWIPS II.
- The file `pqact.conf` serves the same purpose in AWIPS II as `acq_patterns.txt` in AWIPS I.

### 18.1.1.1 SBN Communications Processors Failover Procedures

All CPSBN problems that you encounter should be reported immediately to the NCF.

If a CPSBN should fail,
• **Call the NCF.** The CPSBNs for your site can be swapped from the NCF by way of the Simple Network Management Protocol (SNMP). Refer to Exhibit 2.2.1-1 in Chapter 2 for rack configuration.

**NOTE:** During a DX failover, the CPSBNs will remain up and running because they are designed to send data to the active DX.

### 18.1.1.2 Monitoring Data Acquisition on SBN Communications Processor

**NOTE:** To monitor data acquisition you will need root access to the system.

Because the DVB has replaced the demods, both CPSBNs receive the complete data stream. At any one time, however, only one CP is actively acquiring and routing the data from the DVBs (default configuration has CPSBN1 active-active, CPSBN2 active-standby).

Connect to the active-active CPSBN1. Then, as user **root**, run the following command to monitor the state of network connections associated with LDM:

```
TYPE: netstat -ta | grep cpsbn
```

```
tcp        0      0 0 cpsbn1-tbdw.er.awips.no:ldm:*:*
  LISTEN
tcp        0      0 0 cpsbn1-tbdw.er.:itm-mcell-s nas2-tbdw:nfs
  ESTABLISHED
tcp        0 508 cpsbn1-tbdw.er.awips:omserv netapp1-tbdw:nfs
  ESTABLISHED
tcp        0 0 cpsbnf-tbdw:60301 cpsbnf-tbdw:amq
  ESTABLISHED
   p
tcp        0 0 cpsbnf-tbdw:60300 cpsbnf-tbdw:amq
  ESTABLISHED
   p
getnameinfo failed
getnameinfo failed
tcp        0 0 cpsbn1-tbdw.er.awips.no:ssh dx1-tbdw.er.awips.
s.noa:42869 ESTABLISHED
getnameinfo failed
getnameinfo failed
getnameinfo failed
getnameinfo failed
getnameinfo failed
getnameinfo failed
getnameinfo failed
tcp        0 0 ::ffff:165.92.29.190:amqp cpsbn2-tbdw.er.a
  wips.:45437 ESTABLISHED
tcp        0 0 ::ffff:165.92.29.190:amqp cpsbn2-tbdw.er.a
  wips.:45439 ESTABLISHED
tcp        0 0 ::ffff:165.92.29.190:amqp cpsbn2-tbdw.er.a
  wips.:45438 ESTABLISHED
tcp        0 0 ::ffff:165.92.29.190:amqp cpsbn2-tbdw.er.a
  wips.:45435 ESTABLISHED
tcp        0 0 ::ffff:165.92.29.190:amqp cpsbn2-tbdw.er.a
  wips.:45434 ESTABLISHED
```
Log files for SBN ingest on the active-active CPSBN are located under \texttt{/data/logs/ldm} and controlled via the syslog daemon (\texttt{syslogd}), which is started by the \texttt{init} process in runlevels 2-3-4-5. The syslog also spawns a kernel logging daemon, \texttt{klogd}, which does not concern itself with user-space logging. To check to see if the \texttt{syslogd} process is running run, as user \texttt{root}:

\begin{verbatim}
TYPE: service syslog status

syslogd (pid 32306) is running...
klogd (pid 32309) is running...
\end{verbatim}

The process ID (PID) will be returned along with the status of running if the process is currently active.

\begin{center}
\textbf{NOTE:} While the \texttt{service} command may be executed by any user, it requires access to the PID files in \texttt{/var/run} for \texttt{syslogd} and \texttt{klogd}, which are 600 permissions. Output when the service command is run by a non-root user will appear to show the process (in this case \texttt{syslogd}) as “dead,” which is inaccurate.
\end{center}

The configuration files for the \texttt{syslogd} are located in \texttt{/etc} and \texttt{/etc/sysconfig}. They are:

\begin{verbatim}
/etc/syslog.conf
/etc/sysconfig/syslog
\end{verbatim}

The configuration file \texttt{/etc/sysconfig/syslog} controls the daemon logging options, while \texttt{/etc/sysconfig} controls the user-level process logging and in this case the LDM logging configuration. You should not have to be concerned about \texttt{/etc/sysconfig/syslog}.

Inside \texttt{/etc/syslog.conf} are entries for the logging of LDM:

\begin{verbatim}
# UCAR Unidata LDM Logging
local0.debug /data/ldm/logs/ldmd.log
local3.debug /data/ldm/logs/nwstg.log
local4.debug /data/ldm/logs/goes.log
local5.debug /data/ldm/logs/nwstg2.log
local6.debug /data/ldm/logs/oconus.log
\end{verbatim}

These entries dictate the location for the logging facility of the various LDM NOAAport receivers that acquire the SBN data feed from the DVBs. The first entry in \texttt{/etc/syslog.conf} is also pertinent to LDM because it “excludes” the logging of the NOAAport receivers from the facility \texttt{messages}. The logging facility for the NOAAport receivers is defined in the LDM source.

To watch the receipt of data from the DVBs to the LDM, the above logfiles may be monitored or the \texttt{ldmadmin} utility used to view the feed in real-time. As user \texttt{ldm} on the active-active CPSBN:

\begin{verbatim}
TYPE: ldmadmin watch
\end{verbatim}
Chapter 18 Failover Management Procedures

18.2 Linux Servers

The four DX servers have been paired into two, two-server clusters. DX1 and DX2 are managed by a third-party software product that maintains a continuous heartbeat between the two members of the failover pair and accomplishes an automatic processing switch to the secondary processor should the primary processor fail. DX3 and DX4 do not form a primary/backup server pair. Rather, the EDEX processes run on both servers. EDEX uses QPID and the IP Virtual Server (IPVS), running on the CP1/2 cluster, to provide both load balancing and request balancing. IPVS is used to load-balance the connections from clients (AlertViz, CAVE, etc.) to the EDEX cluster (DX3/4). It is configured so that, when an incoming connection request is made to the IPVS virtual server, it looks to see which server, DX3 or DX4, has the smaller number of connections. It then routes the
connection to that server. If DX3 and DX4 have the same number of active connections, it routes to DX4 by default because it is configured with DX4 weighted higher than DX3.

18.2.1 Red Hat Cluster Manager

The Red Hat Cluster Manager provides automatic failover when either of the active PXs or DXs in the AWIPS configuration is lost. The AWIPS functionality hosted on that PX/DX server will automatically swap to the remaining corresponding server. An automatic failover action causes a configuration alarm event to be forwarded to the NCF via an SNMP trap.

The Red Hat Cluster Manager uses the concept of a cluster made up of nodes within its managed environment. The Linux Data Servers are paired as two, two-server clusters (DX1/DX2, DX3/DX4), and the Linux Preprocessor Servers (PX1 and PX2) are the members of the AWIPS Red Hat Cluster Manager cluster configuration.

The Red Hat Cluster Manager also uses the concept of “services” to transfer responsibilities from one network node to another. For the purposes of AWIPS configuration, the DXs (two pairs) make up two distinct package groups and the PXs make up two distinct package groups.

18.2.1.1 Linux Data Server Failover Scenarios

Key Components

- **Node.** Linux Data Server (DX).
- **Cluster.** Networked group of managed nodes (DX1 and DX2, DX3 and DX4).
- **Package.** The Red Hat Cluster Manager refers to “services,” which are equivalent to MC/ServiceGuard “packages.” The Red Hat Cluster Manager uses redundant hardware, shared disk storage, power management, and robust cluster communication and application failover mechanisms. The DX service/package names are:
  - a2dx1apps
  - a2dx2apps
  - a2dx3apps
  - a2dx4apps.

**NOTE:** In AWIPS II, the package-naming concept is preserved, but with an “a2” preceding the name. For example, in AWIPS I includes a package named dx1apps. In AWIPS II this package is modified and renamed **a2dx1apps**. The same naming convention is preserved for ingest. AWIPS I has an ingest script named startIngest.dx1. In AWIPS II this script is modified and renamed **startA2Ingest.dx1**. Similarly, the HA cronfiles also exist; they are named **a2dx1cron** and **a2SITEDx1cron**.
When issuing an `hb_stat` on a device that is not currently configured to have a package managed (e.g., DX3/DX4) you will see:

*dx3-tbw3 is not part of an heartbeat cluster*

**NOTE:** The device name will change depending on the server on which the command is issued.

The heartbeat services are stopped on those nodes that do not manage any packages or resources. However, the `hb_stat` will still execute because the command performs a sanity check not on whether the heartbeat process is running, but rather on the existence of the heartbeat RPM in the rpm-database.

- **Floating IP Address:** Relocatable IP address that can move between like nodes in a cluster as opposed to a stationary IP address (e.g., dx1f-<siteID>, dx2f-<siteID>, dx3f-siteID>, and dx4f-<siteID>, instead of dx1-<siteID>, dx2-<siteID>, dx3-<siteID>, and dx4-<siteID>).

**Red Hat Cluster Manager Failovers**

AWIPS supports two types of Red Hat Cluster Manager failovers – automatic and manual.

- **Automatic Failover.** If a DX fails, its associated package will be transferred automatically to the other node/server (backup server) in the package group (DX1 to DX2 and vice versa; DX3 to DX4 and vice versa). Failure scenarios, triggering automatic failover, include the following:
  - Loss of SCSI connectivity
  - Loss of Ethernet connectivity
  - CPU hang or panic
  - CPU failure
  - System disk failure
  - Power failure.

- **Manual Failover.** There may be times when you will want to have a manual failover initiated (for example, if the software on the primary node fails to operate properly or if the disk on the primary node becomes corrupted). As the system manager, you need to communicate the request for a manually initiated failover to the NCF; it is the NCF’s responsibility to initiate all manual failovers using Red Hat Cluster Manager.

**Failover Time**

The DX failover time, under normal conditions, is about 2 to 3 minutes.

**Package Switching**

When a DX failure occurs, the resident DX package is switched to the other DX because each is capable of handling the full load of applications as a backup node. Once the failed
node has been revived, the package that moved should be transferred back to its original node (server). The transition back to the original node should be scheduled with the NCF (which is responsible for the transition) in advance to ensure it occurs at an appropriate time (i.e., not during a severe weather condition).

To verify that swap packages are enabled for the swap package (**a2dx1apps**), as user **root**,

**TYPE:** \texttt{hb_stat}  
\textbf{Result:} Look for output similar to Exhibit 18.2.1.1-1.

<table>
<thead>
<tr>
<th>Heartbeat Status Monitor</th>
<th>Nov 02 18:15:29</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textbf{Member Status}</td>
<td>------------------</td>
</tr>
<tr>
<td>\textbf{Member}</td>
<td>\textbf{Status}</td>
</tr>
<tr>
<td>dx1-tbdw</td>
<td>Up</td>
</tr>
<tr>
<td>dx2-tbdw</td>
<td>Up</td>
</tr>
</tbody>
</table>

\textbf{Service Status}

<table>
<thead>
<tr>
<th>Service</th>
<th>IPaddr</th>
<th>Cronfile</th>
<th>Owner</th>
<th>Start Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>a2dx1apps</td>
<td>165.92.29.195</td>
<td>a2dx1cron,a2SIT dx1-tbdw</td>
<td>dx1-tbdw</td>
<td>2011-07-11 20:21:50</td>
</tr>
<tr>
<td>a2dx2apps</td>
<td>165.92.29.196</td>
<td>a2dx2cron,a2SIT dx2-tbdw</td>
<td>dx2-tbdw</td>
<td>2011-10-21 17:05:18</td>
</tr>
</tbody>
</table>

\textbf{Exhibit 18.2.1.1-1. Cluster Status Monitor}

\textit{18.2.1.2 Linux Preprocessor Server Failover Scenarios}

\textbf{Key Components}

- **Node.** Linux Preprocessor Server (PX).
- **Cluster.** Networked group of managed nodes (PX1 and PX2).
- **Package.** The Red Hat Cluster Manager refers to “services,” which are equivalent to MC/ServiceGuard “packages.” The Red Hat Cluster Manager uses redundant hardware, shared disk storage, power management, and robust cluster communication and application failover mechanisms. PX service/package names:
  - **a2px1apps** (usually runs on px1-<siteID>, may run on px2)
  - **a2px2apps** (usually runs on px2-<siteID>, may run on px1).
- **Floating IP Address.** Relocatable IP address that can move between like nodes in a cluster as opposed to a stationary IP address (e.g., px1f-<siteID> and px2f-<siteID> instead of px1-<siteID> or px2-<siteID>).
Red Hat Cluster Manager Failovers

AWIPS supports two types of Red Hat Cluster Manager failovers – automatic and manual.

- **Automatic Failover.** If a PX fails, its associated package will be transferred automatically to the other node/server (backup server) in the package group. Failure scenarios, triggering automatic failover, include the following:
  - Loss of SCSI connectivity
  - Loss of Ethernet connectivity
  - CPU hang or panic
  - CPU failure
  - System disk failure
  - Power failure.

- **Manual Failover.** At times you may want to have a manual failover initiated (for example, if the software on the primary node fails to operate properly or if the disk on the primary node becomes corrupted). As the system manager, you need to communicate the request for a manually initiated failover to the NCF; it is the NCF’s responsibility to initiate all manual failovers using Red Hat Cluster Manager.

Failover Time

The PX failover time, under normal conditions, is about 2 to 3 minutes.

Package Switching

When a PX failure occurs, the resident PX package is switched to the other PX because each is capable of handling the full load of applications as a backup node. Once the failed node has been revived, the package that moved should be transitioned back to its original node (server). The NCF is responsible for this transition.

The transition back to the original node should be scheduled with the NCF in advance so that it can occur at an appropriate time (i.e., not during a severe weather condition).

To verify that swap packages have been enabled for each swap package (px1apps and px2apps) as user root,

```
TYPE: hb_stat

Result: Look for output similar to Exhibit 18.2.1.2-1.
Exhibit 18.2.1.2-2 shows output of a verbose version of this command (hb_stat -v).
```
### Exhibit 18.2.1.2-1. Sample Output

<table>
<thead>
<tr>
<th>Member</th>
<th>Status</th>
<th>IP address</th>
</tr>
</thead>
<tbody>
<tr>
<td>px1-tbw3</td>
<td>Up</td>
<td>165.92.24.7</td>
</tr>
<tr>
<td>px2-tbw3</td>
<td>Up</td>
<td>165.92.24.8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Service</th>
<th>IPaddr</th>
<th>Cronfile</th>
<th>Owner</th>
<th>Start Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>a2px1apps</td>
<td>165.92.24.63</td>
<td>a2px1cron,a2SIT px1-tbw3</td>
<td>px1-tbw3</td>
<td>2012-12-06 19:52:33</td>
</tr>
<tr>
<td>a2px2apps</td>
<td>165.92.24.64</td>
<td>a2px2cron,a2SIT px2-tbw3</td>
<td>px2-tbw3</td>
<td>2012-12-10 14:15:03</td>
</tr>
</tbody>
</table>

### Exhibit 18.2.1.2-2. Sample Verbose Output from Linux Cluster Status Command

<table>
<thead>
<tr>
<th>Node</th>
<th>Type</th>
<th>Devices</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>px1-tbw3</td>
<td>ucast</td>
<td>bond0 165.92.24.8</td>
<td>Up</td>
</tr>
<tr>
<td>px1-tbw3</td>
<td>ucast</td>
<td>eth1 10.0.4.2</td>
<td>Up</td>
</tr>
<tr>
<td>px1-tbw3</td>
<td>ping</td>
<td>router-tbw3</td>
<td>Up</td>
</tr>
<tr>
<td>px2-tbw3</td>
<td>ucast</td>
<td>bond0 165.92.24.7</td>
<td>Up</td>
</tr>
<tr>
<td>px2-tbw3</td>
<td>ping</td>
<td>router-tbw3</td>
<td>Up</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Service</th>
<th>IPaddr</th>
<th>Cronfile</th>
<th>Owner</th>
<th>Start Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>a2px1apps</td>
<td>165.92.24.63</td>
<td>a2px1cron,a2SIT px1-tbw3</td>
<td>px1-tbw3</td>
<td>2012-12-06 19:52:33</td>
</tr>
<tr>
<td>a2px2apps</td>
<td>165.92.24.64</td>
<td>a2px2cron,a2SIT px2-tbw3</td>
<td>px2-tbw3</td>
<td>2012-12-10 14:15:03</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Resource</th>
<th>Parameters</th>
<th>Owner</th>
<th>Status</th>
<th>Start Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>caveData</td>
<td>Unknown host</td>
<td>px1-tbw3</td>
<td>unknown</td>
<td></td>
</tr>
<tr>
<td>ifconfig</td>
<td>`--help' gives usage information.</td>
<td>px1-tbw3</td>
<td>OK</td>
<td>2012-12-06 19:52:33</td>
</tr>
<tr>
<td></td>
<td>IPaddr 165.92.24.63</td>
<td>px1-tbw3</td>
<td>OK</td>
<td>2012-12-06 19:52:33</td>
</tr>
<tr>
<td>a2px1apps</td>
<td>a2px1cron</td>
<td>px1-tbw3</td>
<td>OK</td>
<td>2012-12-06 19:52:33</td>
</tr>
<tr>
<td>a2px2apps</td>
<td>a2px2cron</td>
<td>px2-tbw3</td>
<td>OK</td>
<td>2012-12-10 14:15:03</td>
</tr>
<tr>
<td>CronTab</td>
<td>a2SITEpx1cron</td>
<td>px1-tbw3</td>
<td>OK</td>
<td>2012-12-06 19:52:33</td>
</tr>
<tr>
<td>CronTab</td>
<td>a2SITEpx2cron</td>
<td>px2-tbw3</td>
<td>OK</td>
<td>2012-12-10 14:15:03</td>
</tr>
<tr>
<td>CronTab</td>
<td>archiver</td>
<td>px2-tbw3</td>
<td>OK</td>
<td>2012-12-10 14:15:03</td>
</tr>
</tbody>
</table>
In AWIPS II the **a2px2apps** also controls the following:

- **LDM edexBridge.** This serves up the data from the LDM (cp1f) to the EDEX cluster for processing. It is stopped/started along with the LDM software.

- **Apache daemon.** This is the httpd process that controls the AWIPS II System Monitor webpage. The resource configuration stops/starts the httpd process.

**NOTE:** All the above processes (LDM edexBridge and Apache daemon) are controlled via failover management and require no site or user interaction to configure or control outside of the cluster management suite.

### 18.2.2 Post-Failover Events

Events that take place after an automatic or manually initiated failover are as follows:

- The Red Hat Cluster Manager service will be swapped to the failover node.

- The NCF will be notified, by way of a pop-up dialog or an audible alarm from within HP OpenView Operations (OVO), of the swap completion and of the failover node in question. The node outage is alarmed (red) within OVO. The pop-up alarm is triggered by way of an SNMP trap sent to OVO at the NCF by Red Hat Cluster Manager at the site.

### 18.2.3 NCF Post-Failover Actions

NCF post-failover actions are provided for site information. NCF operators follow procedures contained in the Trouble Ticket software’s knowledge base, i.e.:

- Contact the site to confirm failover.

- Perform analysis of the new system to determine if any problems have occurred during failover and, if so, take the following appropriate steps to correct them:
  - View status of node and package
  - Examine package log file
  - Examine system for known problems.

- Execute various manual procedures to complete additional operations not included as a normal swap function.

- Recontact the site to determine if the site is continuing to acquire and process data (desktop data display applications are receiving and able to display the appropriate product data).
18.3 **River Ensemble Processors (REP)**

Two River Ensemble Processors (REP) are located at each RFC AWIPS site. The REP suite, part of the NWS’ Advanced Hydrologic Prediction Service (AHPS) program, was designed as a high-performance “blank slate” on which the RFCs can load new or site-modified applications in an environment that is very similar to AWIPS. It includes an availability infrastructure that allows failing-over of processes and crons if desired. In the future, this hardware will host “baseline” OH applications implemented via AWIPS or AHPS, as appropriate. The REP provides the RFCs with a platform designed for easy expansion based on mission requirements. The REP suite was delivered as a self-contained unit that includes a rack, a NAS device, two commodity servers, a GbE LAN, and a tape backup device. The servers are installed with Red Hat Linux 7.2. The file system structure of the servers and NAS is consistent with current operations and is ready to support installation of hydrology software.

18.3.1 **REP Server High-Availability Software**

The REP servers use Heartbeat for high-availability (see `heartbeat(8)` (type: `man heartbeat`) and [http://linux-ha.org/HeartbeatProgram/](http://linux-ha.org/HeartbeatProgram/) for more information). Documentation resides in the `/usr/share/doc/heartbeat-1.2.3` directory.

**Terminology**

- **Cluster.** A group of computers that provide access to resources.
- **Node.** A single computer in the cluster. A node may have multiple central processing units (CPUs) and/or LAN interfaces. Each node may provide access to any and all cluster resources.
- **Resource.** A service or device that is managed by the cluster (e.g., an IP address, a daemon, a file system, or a set of cron jobs).
- **Resource Group.** A set of resources that are always swapped together as a unit. This is equivalent to the notion of “package” for HP MC/ServiceGuard or “service” for the Linux Red Hat Cluster Manager.

**Configuration**

The cluster configuration file `/etc/ha.d/ha.cf` defines nodes, heartbeat channels, and various failover parameters. Most parameters have been left at their default values because there has been no reason to change them. Descriptions of some parameters of note follow.

- **Nice failback on.** This setting prevents resources from automatically failing over to the default host when it becomes available. The rationale behind this setting is that the failback should occur in a controlled manner. Automatic failback could cause unnecessary interruptions to applications that do not know or care whether a particular resource is on the default or backup node. Furthermore, the cause of the
original failure may require investigation and repairs may be required before the default node is brought back into service. This option is required in order to allow network interface failures to induce a failover (see ipfail plug-in below).

- **Logfacility local0.** Use syslog for high-availability software log messages. The syslog configuration is configured to log these files to /var/log/ha-log and logrotate is configured to maintain a history of these log files.

- **bcast eth1.** Broadcast heartbeat messages on the eth1 interface. Eth1 interfaces to the private subnet for the cluster.

- **unicast eth0 <peer_ip>.** Unicast heartbeat messages to peer on the eth0 interface. Eth0 interfaces to the site LAN. Having two cluster heartbeat interfaces ensures cluster integrity even if one of the interfaces goes down.

- **watchdog /dev/watchdog.** Reboot if we do not receive our own heartbeat messages within a minute. This option ensures that a node that has been cut off from the cluster due to a heartbeat network interface failure to relinquish its resources before another node takes them over.

- **node rp1-xxx rp2-xxx.** The list of nodes that comprise the cluster. Additional nodes may be added as required. Each of these nodes must be capable of providing access to any and all cluster resources.

- **ping router-xxx.** Treat router-xxx as a pseudo-cluster member. If a node cannot access router-xxx, it will query the other nodes to determine whether the ping node is down or the node’s network interface has failed.

- **respawn hacluster /usr/lib/heartbeat/ipfail.** Run the ipfail plug-in. This plug-in will fail over any held resources when a node determines that its network interface has failed (due to the failure to reach a ping host). Note that ipfail requires the nice_failback option.

The resource configuration file /etc/ha.d/haresources defines resource groups and preferred nodes. This configuration file must be identical on all nodes. A detailed description of the syntax for this file is contained in the header of the file.

### 18.3.2 REP Server Failover Scenarios

Cluster status monitoring is accomplished via the /awips/ops/bin/hb_stat utility script. With no arguments, the script reports the status of the heartbeat cluster daemons on each node and the owner/status of each resource group. In the example of sample output that follows, the heartbeat cluster daemons are up on both nodes, Resource Group 1 is running on rp2-tbdr, and Resource Group 2 is running on rp1-tbdr.
The cluster daemons log information and error messages to `/var/log/cluster`. This log file is managed by `syslog` and is rotated weekly.

**Automatic Failover**

A node will automatically take over a resource group if it cannot contact the owner of the resource group via either heartbeat channel within `deadtime` seconds (30 seconds). A node will automatically give up a resource group to a peer if it cannot ping the router within `deadtime` seconds. If a node does not receive its own heartbeat message within a minute, it will reboot itself, thereby giving up any resource groups it may have been managing. Failover of resource groups based on the failure of a managed process is not implemented at this time; no managed processes have yet been defined.

**Manual Failover**

Manual failover is provided via the `/awips/ops/bin/hb_swap` script. This script requests a resource group swap from the heartbeat cluster daemons. The first argument must be a resource unique to the resource group to be swapped as it appears in the `haresources` file (including any arguments). The entire resource group will be swapped, not just the resource specified. If that is not the desired result, the cluster is not configured correctly. If the node option is used, it must match the node name specified in the `ha.cf` and `haresources` files.

As an example, given the `haresources` file entries on site `tbdr` (a testbed),

```
rp1-tbdr IPaddr::165.92.29.77 REPsvc1 crontab::REPcron1
rp2-tbdr IPaddr::165.92.29.78 REPsvc2 crontab::REPcron2
```

To swap the first resource group to node `rp2-tbdr`, run any one of the following commands:

```
TYPE:    hb_swap IPaddr::165.92.29.77 rp2-tbdr
         or
TYPE:    hb_swap REPsvc1 rp2-tbdr
```
or

```bash
TYPE:     hb_swap crontab::REPcron1 rp2-tbdr
```

If the node argument is omitted, the node to which the resource group is being given is deemed to be the local host. If `rsh` is not enabled, this is the way the script must be run. `rsh` is used to deliver the request message to the cluster daemon that is taking over. So, to perform the example swap without using `rsh`, the following command could be run on rp2-tbdr:

```bash
TYPE:     hb_swap REPsvc1
```

### 18.3.3 Managing the Heartbeat Cluster Service

The heartbeat service is configured to start automatically at Run Level 3 and may be started and stopped and started manually. This service may be managed using `chkconfig(8)`. Use the `/sbin/service` command to stop/start/restart and check the status of the heartbeat service.

```bash
TYPE:     /sbin/service heartbeat start
TYPE:     /sbin/service heartbeat stop
TYPE:     /sbin/service heartbeat restart
TYPE:     /sbin/service heartbeat status
```

### 18.3.4 EDEX/IPVS, LDM Failovers

EDEX runs on dx3 and dx4. There is no “failover” as it is managed as a cluster. Both machines can accomplish all the tasks. If one machine goes down, then the other has to handle the full load, whereas otherwise they share the load. EDEX mostly communicates with qpid by reading messages off of it. However, it also receives connections from CAVE. This is where IPVS comes in. IPVS monitors the EDEX cluster, and “serves” connections to it. This means that CAVE connections are routed through IPVS onto one of the EDEX servers. If one of the servers, dx3 or dx4, is down, then IPVS would route all the connections to the only available node in the cluster. If they are both up, then it will try to keep an equal number of connections to each, to balance the load.

The EDEX and IPVS are managed through System V init scripts, for example, service edex_camel start for EDEX and service pulse start for IPVS. IPVS currently runs on CPs.

LDM is managed through heartbeat. It will start and stop the I dm feed. By default it runs on cp1f.

### 18.4 AWIPS II to AWIPS I Fallback

AWIPS II is designed to co-exist with the AWIPS I software that it is replacing. This includes the capability to switch between AWIPS I operation and AWIPS II operation.
The intent is to allow a site – Weather Forecast Office, River Forecast Center, etc. – to revert to AWIPS I operation if significant problems occur in the operation of the AWIPS II software.

**NOTE:** The fallback should not be attempted without appropriate approval from a qualified ESA or ITO (see below) who initially installed the AWIPS II system and currently manages the AWIPS II software to run on AWIPS I.

**ESA:** The ESA serves as the local office point of contact for system changes that affect hardware, software, and communications infrastructure. The ESA is responsible for coordinating assigned EMRS maintenance requests with appropriate persons at the office, working with the MIC as appropriate. The ESA is responsible for reporting completion of an assigned EMRS maintenance request via the EMRS system.

**ITO:** The ITO serves as the local office point of contact for system changes that affect software (application, operational, database, and operating system) for which the ITO has been assigned or delegated responsibility by the MIC/HIC. The ITO is responsible for coordinating assigned EMRS maintenance requests with appropriate persons at the office, working with the MIC and the ESA as appropriate. The ITO is responsible for reporting completion of an assigned EMRS maintenance request via the EMRS system.

Many of the operations required to switch between AWIPS I and AWIPS II operation are controlled by a single script: `rehost_awips.sh`. `rehost_awips.sh` is part of the AWIPS II installation package.

AWIPS II-to-AWIPS I fallback consists of a series of operations that back up the AWIPS II databases, switch from AWIPS II to AWIPS I operation, and import the previously backed-up AWIPS II databases. Scripts are provided for the first two steps; a manual procedure is provided for the third step.

The process of switching from AWIPS I to AWIPS II consists of a series of operations that back up the AWIPS I databases, switch from AWIPS I to AWIPS II operation, and import the previously backed-up AWIPS I databases. Scripts are provided for the first two steps; a manual procedure is provided for the third step.

### 18.4.1 AWIPS I/AWIPS II Co-existence

The AWIPS II software is designed to co-exist\(^1\) with the existing AWIPS I software. This co-existence is intended to minimize the impact to operations during the initial installation of the AWIPS II software, and allow the AWIPS II software installation and much of the initial setup to be performed while the existing – AWIPS I – software is in operation.

---

\(^1\) “Co-exist” means that both software packages can be installed on the same system. Only one package can be used at a time, however.
This design also provides a mechanism for reverting the system to AWIPS I operation if significant problems are encountered following the AWIPS II installation.

This installation design relies on three factors:

1. The AWIPS II software is installed into a separate file system, generally located at /awips2 on the affected servers and workstations.

2. A switchover script, rehost_awips.sh, is provided as part of the AWIPS II installation package. This script provides semi-automated switching between AWIPS I and AWIPS II operations.

3. When similar functionality exists between AWIPS I and AWIPS II, soft links using the AWIPS I names are used to point the user to the correct executable for the current operational mode (AWIPS I or AWIPS II).

Prior to installation, the AWIPS II installation package should be staged on a network-accessible file system. The location of the staging area is referred to in this module as <AWIPS2_INSTALL_DIR>.

It is important to realize, however, that while both AWIPS I and AWIPS II can be installed, only one of the software packages may be in operation at any one time.

**Note:** The operational modes are referred to as follows:

- *AWIPS I operational mode* means that the system is running AWIPS I software. This may also be referenced as *AWIPS I mode*.
- *AWIPS II operational mode* means that the system is running AWIPS II software and rehosted applications. This may also be referenced as *AWIPS II mode*.

It is also important to recognize that AWIPS II stores much of its data differently from AWIPS I, using PostgreSQL databases and HDF5 files where AWIPS I used NetCDF files.

### 18.4.2 Swapping Procedures

In the event that your site needs to swap operations from AWIPS I to AWIPS II mode, the latest procedures will be sent to your office by the appropriate NWS or NCF personnel.

The procedures boil down to a few general steps:

1. Backing up the postgres database and vtec information.
2. Running the rehost script to swap operations to the other mode.
3. Importing the databases and vtec information.
4. Verifying the new operations.
** CAUTION **

Swapping operational modes from AWIPS II to AWIPS I requires that all server software be shut down. As a result, all client software must also be shut down during the swap.

18.4.3 Limitations on System Swapping

When swapping between AWIPS I mode and AWIPS II mode, certain limitations must be considered.

1. The actual swap requires approximately an hour. Much of this time is devoted to exporting and importing the common databases, and running the swap script. During most of the switchover, no data is being ingested.

2. AWIPS II stores much of its decoded data in a different format than AWIPS I. As a result, there is no practical way to import much of the decoded data from AWIPS I into AWIPS II and vice versa.

3. Even though both AWIPS I and AWIPS II use the text database fxatext, the table structure differs. As a result, there is no practical way to import decoded text from AWIPS I into AWIPS II and vice versa.

Following the swap between AWIPS I and AWIPS II, standard data recovery procedures can be used to backfill missing data. In most cases, there will be a delay in data availability as new data is being ingested.
Chapter 19
System Backup and Recovery Procedures
Chapter 19. System Backup and Recovery Procedures

Table of Contents

19.1 DX NAS Backups ............................................................................................................... 1
19.2 Other Backups .................................................................................................................. 2
  19.2.1 NAS Weekly Backup Strategy ................................................................................ 2
19.3 REP/NS1 Backup (RFCs Only) ....................................................................................... 3
19.4 Recovery Procedures ....................................................................................................... 3
  19.4.1 Volume Table ......................................................................................................... 3
  19.4.2 How To Recover a Single File ................................................................................. 4
19.4.3 How to Recover an Entire Volume ........................................................................... 5
19.4.4 How to Recover an Entire Volume to a Temporary Location ................................ 5
19.4.5 Other Helpful Commands ....................................................................................... 6

List of Tables

Table 19.2-1. /data/fxa Directories for Backup ................................................................. 2
Backups include a mix of cron-generated backups and snapshots that are taken using Linux system utilities. Backups should be transparent to the sites except that sites will need to swap out some tapes on a weekly basis. Because sites may occasionally want to keep a tape, it is recommended that the sites have more than two tapes on hand so that tapes are still available for rotation even when a site wants to keep a tape.

19.1 DX NAS Backups

The weekly Network Attached Storage (NAS) backup is controlled by a root cron on the Archive Server (AX). This weekly backup (nas_backup_weekly) executes Mondays at 0310Z and backs up NAS volumes to an external tape drive connected to the AX. The weekly backup dumps all of the file systems created on the NAS to tape (i.e., the entire NAS). Another cron launches a daily backup (nas_backup_daily) at 0210Z and merely creates a tar archive file in /data/fxa/DAILY_BACKUP. This cron runs in conjunction with the a2dx1apps package. Tapes used for the weekly backups should be rotated on a weekly basis, and it is recommended that sites purchase additional blank tapes so they can keep a tape (i.e., move it out of the rotation) if needed or desired.

Checkpoint Backups

Checkpoints occur weekly at 0000Z. The system preserves four weekly snapshots.

**NOTE:** With the introduction of the Sun StorageTek 5320, nightly checkpoints are also taken.

Snapshot is enabled for the following file systems:

1. /home
2. /awips/adapt
3. /awips/gis
4. /awips/dev
5. /data/local
6. aiidata

aiidata is a directory on the NAS. It is located at nas1:/aiidata. Various systems have mounts to all or part of aiidata. Generally, these mounts are at /awips2/edex/data or a subdirectory. On DX3/4, nas1:/aiidata is mounted at /awips2/edex/data. On PX1/2, nas1:/aiidata is mounted on /awips2/edex/data. On the LX and XT workstations, nas1:/aiidata/share is mounted on /awips2/edex/data/share.

**NOTE:** The /data/local snapshot should be monitored for dynamic “live” data.

Checkpoints should not be done on partitions with a lot of dynamic files activity, such as /data/fxa. The /awips/hydroapps file system could not be included in the snapshot strategy either, due to the mix of static and dynamic data within the same file system.
19.2 Other Backups

A daily online backup will `tar` key directories that are not within the partitions that have checkpoints configured. The results are saved in the `/data/fxa/DAILY_BACKUP` directory.

Table 19.2-1 lists the directories that are backed up by the NAS daily backup, including the `/awips/hydroapps` file system backups. You can view the daily NAS daily backup log at `/data/logs/daily_backup.log` on the server that currently hosts the dx1apps package.

<table>
<thead>
<tr>
<th>/data/fxa Directories for Backup</th>
</tr>
</thead>
<tbody>
<tr>
<td>/data/fxa/archive</td>
</tr>
<tr>
<td>/data/fxa/customFiles</td>
</tr>
<tr>
<td>/data/fxa/rps-lists</td>
</tr>
<tr>
<td>/data/fxa/scripts</td>
</tr>
<tr>
<td>/data/fxa/siteConfig</td>
</tr>
<tr>
<td>/data/fxa/userPrefs</td>
</tr>
<tr>
<td>/data/fxa/workFiles</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>/awips/hydroapps Files/ Directories for OHD-related Backup</th>
</tr>
</thead>
<tbody>
<tr>
<td>/awips/hydroapps/Apps_defaults*</td>
</tr>
<tr>
<td>/awips/hydroapps/1x/geo_data</td>
</tr>
<tr>
<td>/awips/hydroapps/precip_proc/local/data/app</td>
</tr>
<tr>
<td>/awips/hydroapps/whfs/local/data/app</td>
</tr>
<tr>
<td>/awips/hydroapps/whfs/local/data/geo</td>
</tr>
<tr>
<td>/awips/hydroapps/set_hydro_env</td>
</tr>
<tr>
<td>/awips/hydroapps/1x/rfc/idma/images</td>
</tr>
<tr>
<td>/awips/hydroapps/whfs/bin/pa/app-defaults</td>
</tr>
<tr>
<td>/awips/hydroapps/1x/rfc/nwsrfs/ofsf/files</td>
</tr>
</tbody>
</table>

19.2.1 NAS Weekly Backup Strategy

A cron is initiated from the Archive Server (AX) on a weekly basis which performs a backup to tape of all NAS volumes. The LTO-II tape drive is hooked to the Archive Server via an external SCSI cable.

The `/awips/ops/bin/nas_weekly_backup` script performs the work of the backup. This script was modified for the deployment of the StorageTek NAS to run only from the AX and utilizes the `tar` command to write the entire contents of the NAS to tape. Each NAS share (or file volume) will reside on one tape “file.” A tape file is merely a pointer to the start of a new data set. To eliminate the problems with backing up a live file system, a temporary checkpoint is created for each file volume prior to the backup and used for creating the backup. This checkpoint is then removed at the end of the backup.
The script will automatically rewind and eject the tape at the end of each backup. Sites are responsible for rotating tapes when they arrive at the office Monday morning.

19.3 REP/NS1 Backup (RFCs Only)

No script was provided to the RFCs to back up their LTO-II. However, the following checkpoints are taken of the /awips/rep file system:

- **Nightly.** Once a day at 0000Z for 7 days (then tapes should be rotated or the tape will be overwritten)
- **Weekly.** Once a week at 0000Z for 4 weeks (then tapes should be rotated or the tape will be overwritten).

These snapshots are conservative and may not be possible at some RFCs. Because River Ensemble Processor (REP) requirements do not state how the device is to be used, the snapshots may need to be adjusted on a per-site basis depending on file system usage. Sites with very dynamic data sets will need to reduce the frequency or quantity of snapshots.

**NOTE:** This backup strategy will not change with the replacement of the NetApp FAS250, except that the snapshots will be named checkpoints.

19.4 Recovery Procedures

19.4.1 Volume Table

In order to recover a file or an entire volume, you must consult the following table:

<table>
<thead>
<tr>
<th>Volume No.</th>
<th>Volume</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>dataGFE</td>
</tr>
<tr>
<td>1</td>
<td>dataX400</td>
</tr>
<tr>
<td>2</td>
<td>awipsADAPT</td>
</tr>
<tr>
<td>3</td>
<td>DSshared</td>
</tr>
<tr>
<td>4</td>
<td>dataADAPT</td>
</tr>
<tr>
<td>5</td>
<td>awipsGIS</td>
</tr>
<tr>
<td>6</td>
<td>awipsDEV</td>
</tr>
<tr>
<td>7</td>
<td>awipsHOME</td>
</tr>
<tr>
<td>8</td>
<td>dataLOCAL</td>
</tr>
<tr>
<td>9</td>
<td>awipsHYDRO</td>
</tr>
<tr>
<td>10</td>
<td>awipsGFESuite</td>
</tr>
<tr>
<td>11</td>
<td>dataFXA</td>
</tr>
<tr>
<td>12</td>
<td>GFESuite2</td>
</tr>
<tr>
<td>13</td>
<td>Aiidata</td>
</tr>
<tr>
<td>14</td>
<td>Qpid</td>
</tr>
<tr>
<td>15</td>
<td>Localapps</td>
</tr>
</tbody>
</table>
This table is derived from a variable called 'arBackupList' in this file:
ax:/awips/ops/bin/nas_backup_weekly

Here is the variable as included in the file.

If [ `cat /awips/Release_ID` == AWIPSII ], then

    echo -en "\n\n\t[INFO]: Setting _arBackupList to include the AWIPSII partitions.\n"
    _arBackupList=( "/tmp/data/GFE" "/tmp/data/x400"
"/tmp/awips/adapt" "/tmp/DS_shared" "/tmp/data/adapt"
"/tmp/awips/gis"
"/tmp/awips/dev" "/tmp/home" "/tmp/data/local"
"/tmp/awips/hydroapps" "/tmp/awips/GFESuite" "/tmp/data/fxa"
"/tmp/awips2/GFESuite" "/tmp/awips2/edex/data"
"/tmp/awips2/gpid/messageStore" "/tmp/localapps" )

else

    echo -en "\n\n\t[INFO]: Setting _arBackupList to only include the AWIPSII partitions.\n"
    _arBackupList=( "/tmp/data/GFE" "/tmp/data/x400"
"/tmp/awips/adapt" "/tmp/DS_shared" "/tmp/data/adapt"
"/tmp/awips/gis"
"/tmp/awips/dev" "/tmp/home" "/tmp/data/local"
"/tmp/awips/hydroapps" "/tmp/awips/GFESuite" "/tmp/data/fxa" )

19.4.2 How To Recover a Single File

Assume that you need to recover this file:

/data/fxa/engDict

The six steps required to recover the file follow.

1. Log onto AX as the root user
   
   \texttt{ssh root@ax}

2. Identify the filesystem upon which the file is located.

   The volume on which the file must be located. To do this, type the following commands:

   \texttt{cd /data/fxa}
   \texttt{df -h .}

   You will see this output:

   \begin{tabular}{|l|c|c|c|c|}
   \hline
   Filesystem & Size used & Avail & Use\% & Mounted on \\
   \hline
   nas1:/dataFXA & 500G 424G & 77G 85\% & /data/fxa \\
   \hline
   \end{tabular}
This output tells us that the file to be recovered (engDict) is located on the dataFXA volume. Consulting the Volume Table, we see that dataFXA is volume number 11.

3. Eject the tape and place a tape in that you know contains the file you need to recover.

4. Rewind the tape to the beginning (commands issued from the AX as user root):
   \[ \texttt{mt -f nst0 rewind} \]

5. Forward the tape to the beginning of volume 11 (dataFXA):
   \[ \texttt{mt -f nst0 fsf 11} \]

6. Restore the file:
   \[ \texttt{tar -xzvf /dev/nst0 -C /data/fxa engDict} \]

This completes the steps required to recover the file.

19.4.3 How to Recover an Entire Volume

This procedure will restore an entire volume, overwriting all files with data from the backup tape. If you do not wish to overwrite all of your data, skip to the next section (19.4.4, "How to Recover an Entire Volume to a Temporary Location").

1. Identify the volume you would like to restore.
   [In this example, we will restore volume 6 (awipsDEV), which is mounted on /awips/dev.]

2. Insert a tape into the backup tape drive that contains the volume you would like to restore.

3. Rewind the tape to the beginning (commands issued from the AX as user root).
   \[ \texttt{mt -f nst0 rewind} \]

4. Forward to the beginning of volume 6 (awipsDEV).
   \[ \texttt{mt -f nst0 fsf 6} \]

5. Type the following command to restore the files in this volume.
   \[ \texttt{tar -xzvf /dev/nst0a -C /awips/dev} \]

19.4.4 How to Recover an Entire Volume to a Temporary Location

This procedure will copy an entire volume from backup tape to a temporary location on disk. This is useful if you need to choose an array of files to restore without overwriting existing data.
1. Identify the volume to restore. In this example we will restore to a temporary location the dataLOCAL volume (volume number 8).

2. Identify the size of the volume you need to restore.

   \texttt{df -h /data/local}

   You should see output similar to this:

   \begin{tabular}{lllll}
   Filesystem & Size used & Avail & Use\% & Mounted on \\
   nas1:/vol/data/local & 8.2G & 1.7G & 6.5G & 21\% /data/local \\
   \end{tabular}

   There is no way to tell how large the volume is on the tape device, so we have to assume that the size of the data on tape is relatively the same as the size existing on disk (after all, it is a backup). So, because the "Use\%" is less than 50\%, this allows us to assume that we have enough space to make a complete copy of the backup tape's data_local volume inside /data/local.

3. Create a temporary directory to hold our restored files on /data/local

   \texttt{cd /data/local}

   \texttt{mkdir RESTORED}

4. Log into the AX as user root

   \texttt{ssh root@ax}

5. Rewind the tape.

   \texttt{mt -f nst0 rewind}

6. Step the tape forward to our desired volume (volume 8, dataLOCAL).

   \texttt{mt -f nst0 fsf 8}

7. Restore the files to our RESTORED directory

   \texttt{tar -xzvf /dev/nst0 -C /data/local/RESTORED}

   \textbf{Note:} Suppose there is not enough space to put your RESTORED directory in /data/local/ ... You could restore the data_local volume to //data/fxa/RESTORED if there is enough space in the dataFXA volume.

\section*{19.4.5 Other Helpful Commands}

The status command tells you what volume and block you are currently located at. You always need to be at block 0 of the volume you are concerned with (e.g., volume 9 block 0). If you are not, you need to rewind the tape and step it forward to the volume point before restoring.

The status command is:

\texttt{ssh ax "mt -f nst0 status"}

If you are not sure if the file you need is on the volume, you can list them using tar -tzf
/dev/nst0

tar -tzf /dev/nst0 engDict

The preceding command will list the file if it is present or tell you that it is not found.

Use this command to retrieve a list of ALL the files in the backup volume:

tar -tzf /dev/nst0
Chapter 20. System Console

List of Tables

Table 20-1. ATS1 Port and Logical Port Assignments ................................................................. 3
Table 20-2. ATS2 Port and Logical Port Assignments ................................................................. 4
Table 20-3. LTS1 Port and Logical Port Assignments ................................................................. 5
The Cyclades terminal server connects the console ports for the site hardware to the system console and the server private LAN. These connections provide the Monitor and Control console functionality either via the system console or via a private LAN Telnet session. The Cyclades terminal server provides the NCF remote system console functionality via Cyclades modem connections, private LAN Telnet sessions, or the WAN.

The local system console is an HP Thin Client, and can be used to monitor either the AWIPS or LDAD systems by switching to the A or B position of the A/B switch box.

- There is one Cyclades (two at an RFC) for AWIPS and one Cyclades for LDAD. The system console (HP Thin Client) can be directed to either Cyclades, via an A/B switch box.

- To access either Terminal Server, you must first log into the Thin Client and then double-click on the ‘TeemTalk’ icon on the desktop to open the terminal emulating application.

- If an AWIPS or LDAD connection cannot be established, at the Terminal Server’s shell prompt, as root, issue the following:
  
  ```
  ps -ef |grep [####]
  (where ‘####’ is the logical port number of the inaccessible device)
  kill -9 [pid]
  (where ‘pid’ is the process ID of the session accessing the device).
  ```

- Console access to the devices can be acquired from a menu or directly by telnet/ssh, if the logical port number is known.

  To view a list of devices from a menu:

  On AWIPS Terminal Server 1/2 (ATS1/2), issue the following:
  
  ```
  ts_menu
  ```

  On LDAD Terminal Server 1 (LTS1), issue the following:
  
  ```
  menush
  ```

  To access a device directly using the logical port number,
  
  ```
  telnet ats1 7001    or    ssh ats1 7001
  ```

The Cyclades terminal server is used to access all the server and workstation console ports in the beginning and at the end of the process. Additionally, miscellaneous devices connect through the Cyclades including the switches, routers, Terminal Interface Unit (TIU), VIR, and CPSBNs.

The AWIPS Cyclades ATS1 menu follows.
The LDAD Cyclades LTS1 menu follows.

Note that Cyclades does not have a “Login Screen.” Instead, it has just a plain text prompt asking for a username and password as shown below.

```
ats1 login:
Password:
```

From the Cyclades Terminal Server, issue the following key sequence to disconnect from a console connection,

```
Ctrl-[
```

Then type ‘e’ to exit and then ‘q’ to quit the ts_menu. If accessed directly to the logical port number, ‘Ctrl-[]’ will disconnect the session and return you to the terminal server shell prompt.

Routers can be directly accessed only through the terminal server from RFC and stand-alone WFO systems.
ATS1 Port and Logical Port Assignments for ATS1 follow in Table 20-1.

<table>
<thead>
<tr>
<th>Device Name</th>
<th>Port Num</th>
<th>Logical Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>DX1</td>
<td>1</td>
<td>7001</td>
</tr>
<tr>
<td>DX2</td>
<td>2</td>
<td>7002</td>
</tr>
<tr>
<td>DX3</td>
<td>3</td>
<td>7003</td>
</tr>
<tr>
<td>DX4</td>
<td>4</td>
<td>7004</td>
</tr>
<tr>
<td>PX1</td>
<td>5</td>
<td>7005</td>
</tr>
<tr>
<td>PX2</td>
<td>6</td>
<td>7006</td>
</tr>
<tr>
<td>AX</td>
<td>7</td>
<td>7007</td>
</tr>
<tr>
<td>NAS1</td>
<td>8</td>
<td>7008</td>
</tr>
<tr>
<td>OPEN</td>
<td>9</td>
<td>7009</td>
</tr>
<tr>
<td>OPEN</td>
<td>10</td>
<td>7010</td>
</tr>
<tr>
<td>PSW</td>
<td>11</td>
<td>7011</td>
</tr>
<tr>
<td>DMZ</td>
<td>12</td>
<td>7012</td>
</tr>
<tr>
<td>RTR1</td>
<td>13</td>
<td>7013</td>
</tr>
<tr>
<td>RTR2</td>
<td>14</td>
<td>7014</td>
</tr>
<tr>
<td>HSW1</td>
<td>15</td>
<td>7015</td>
</tr>
<tr>
<td>HSW2</td>
<td>16</td>
<td>7016</td>
</tr>
<tr>
<td>FW1</td>
<td>17</td>
<td>7017</td>
</tr>
<tr>
<td>FW2</td>
<td>18</td>
<td>7018</td>
</tr>
<tr>
<td>VIR</td>
<td>19</td>
<td>7019</td>
</tr>
<tr>
<td>TIU</td>
<td>20</td>
<td>7020</td>
</tr>
<tr>
<td>LX1</td>
<td>21</td>
<td>7021</td>
</tr>
<tr>
<td>LX2</td>
<td>22</td>
<td>7022</td>
</tr>
<tr>
<td>LX3</td>
<td>23</td>
<td>7023</td>
</tr>
<tr>
<td>LX4</td>
<td>24</td>
<td>7024</td>
</tr>
<tr>
<td>LX5</td>
<td>25</td>
<td>7025</td>
</tr>
<tr>
<td>LX6</td>
<td>26</td>
<td>7026</td>
</tr>
<tr>
<td>OPEN</td>
<td>27</td>
<td>7027</td>
</tr>
<tr>
<td>OPEN</td>
<td>28</td>
<td>7028</td>
</tr>
<tr>
<td>OPEN</td>
<td>29</td>
<td>7029</td>
</tr>
<tr>
<td>OPEN</td>
<td>30</td>
<td>7030</td>
</tr>
<tr>
<td>XT1</td>
<td>31</td>
<td>7031</td>
</tr>
<tr>
<td>XT2</td>
<td>32</td>
<td>7032</td>
</tr>
<tr>
<td>XT3</td>
<td>33</td>
<td>7033</td>
</tr>
<tr>
<td>XT4</td>
<td>34</td>
<td>7034</td>
</tr>
<tr>
<td>XT5</td>
<td>35</td>
<td>7035</td>
</tr>
<tr>
<td>XT6</td>
<td>36</td>
<td>7036</td>
</tr>
<tr>
<td>CPSBN1</td>
<td>37</td>
<td>7037</td>
</tr>
<tr>
<td>CPSBN2</td>
<td>38</td>
<td>7038</td>
</tr>
<tr>
<td>M&amp;C</td>
<td>39</td>
<td>7039</td>
</tr>
<tr>
<td>DX1-DAS</td>
<td>40</td>
<td>7040</td>
</tr>
<tr>
<td>DX2-DAS</td>
<td>41</td>
<td>7041</td>
</tr>
</tbody>
</table>
ATS2 Port and Logical Port Assignments for ATS1 follow in Table 20-2.

**Table 20-2. ATS2 Port and Logical Port Assignments**

<table>
<thead>
<tr>
<th>Device Name</th>
<th>Port Num</th>
<th>Logical Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPEN</td>
<td>1</td>
<td>7001</td>
</tr>
<tr>
<td>OPEN</td>
<td>2</td>
<td>7002</td>
</tr>
<tr>
<td>OPEN</td>
<td>3</td>
<td>7003</td>
</tr>
<tr>
<td>OPEN</td>
<td>4</td>
<td>7004</td>
</tr>
<tr>
<td>OPEN</td>
<td>5</td>
<td>7005</td>
</tr>
<tr>
<td>OPEN</td>
<td>6</td>
<td>7006</td>
</tr>
<tr>
<td>OPEN</td>
<td>7</td>
<td>7007</td>
</tr>
<tr>
<td>RP1</td>
<td>8</td>
<td>7008</td>
</tr>
<tr>
<td>RP2</td>
<td>9</td>
<td>7009</td>
</tr>
<tr>
<td>GSW1</td>
<td>10</td>
<td>7010</td>
</tr>
<tr>
<td>GSW1</td>
<td>11</td>
<td>7011</td>
</tr>
<tr>
<td>NS1</td>
<td>12</td>
<td>7012</td>
</tr>
<tr>
<td>OPEN</td>
<td>13</td>
<td>7013</td>
</tr>
<tr>
<td>OPEN</td>
<td>14</td>
<td>7014</td>
</tr>
<tr>
<td>OPEN</td>
<td>15</td>
<td>7015</td>
</tr>
<tr>
<td>TIU-2</td>
<td>16</td>
<td>7016</td>
</tr>
<tr>
<td>TIU-3</td>
<td>17</td>
<td>7017</td>
</tr>
<tr>
<td>TIU-4</td>
<td>18</td>
<td>7018</td>
</tr>
<tr>
<td>OPEN</td>
<td>19</td>
<td>7019</td>
</tr>
<tr>
<td>OPEN</td>
<td>20</td>
<td>7020</td>
</tr>
<tr>
<td>LX7</td>
<td>21</td>
<td>7021</td>
</tr>
<tr>
<td>LX8</td>
<td>22</td>
<td>7022</td>
</tr>
<tr>
<td>LX9</td>
<td>23</td>
<td>7023</td>
</tr>
<tr>
<td>LXA</td>
<td>24</td>
<td>7024</td>
</tr>
<tr>
<td>LXB</td>
<td>25</td>
<td>7025</td>
</tr>
<tr>
<td>LXC</td>
<td>26</td>
<td>7026</td>
</tr>
<tr>
<td>LXD</td>
<td>27</td>
<td>7027</td>
</tr>
<tr>
<td>OPEN</td>
<td>28</td>
<td>7028</td>
</tr>
<tr>
<td>OPEN</td>
<td>29</td>
<td>7029</td>
</tr>
<tr>
<td>OPEN</td>
<td>30</td>
<td>7030</td>
</tr>
</tbody>
</table>

* ACR – Router3  ** AFC – PX3/PX4

ATS2 Port and Logical Port Assignments for ATS1 follow in Table 20-2.
<table>
<thead>
<tr>
<th>Device Name</th>
<th>Port Num</th>
<th>Logical Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>XT7</td>
<td>31</td>
<td>7031</td>
</tr>
<tr>
<td>XT8</td>
<td>32</td>
<td>7032</td>
</tr>
<tr>
<td>XT9</td>
<td>33</td>
<td>7033</td>
</tr>
<tr>
<td>XTA</td>
<td>34</td>
<td>7034</td>
</tr>
<tr>
<td>XTB</td>
<td>35</td>
<td>7035</td>
</tr>
<tr>
<td>XCT</td>
<td>36</td>
<td>7036</td>
</tr>
<tr>
<td>XTD</td>
<td>37</td>
<td>7037</td>
</tr>
<tr>
<td>OPEN</td>
<td>38</td>
<td>7038</td>
</tr>
<tr>
<td>OPEN</td>
<td>39</td>
<td>7039</td>
</tr>
<tr>
<td>OPEN</td>
<td>40</td>
<td>7040</td>
</tr>
<tr>
<td>OPEN</td>
<td>41</td>
<td>7041</td>
</tr>
<tr>
<td>OPEN</td>
<td>42</td>
<td>7042</td>
</tr>
<tr>
<td>OPEN</td>
<td>43</td>
<td>7043</td>
</tr>
<tr>
<td>OPEN</td>
<td>44</td>
<td>7044</td>
</tr>
<tr>
<td>OPEN</td>
<td>45</td>
<td>7045</td>
</tr>
<tr>
<td>OPEN</td>
<td>46</td>
<td>7046</td>
</tr>
<tr>
<td>OPEN</td>
<td>47</td>
<td>7047</td>
</tr>
<tr>
<td>OPEN</td>
<td>48</td>
<td>7048</td>
</tr>
</tbody>
</table>

LTS1 Port and Logical Port Assignments for ATS1 follow in Table 20-3.

Table 20-3. LTS1 Port and Logical Port Assignments

<table>
<thead>
<tr>
<th>Device / Interface Type</th>
<th>Port Num</th>
<th>Logical Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>7e1out</td>
<td>1</td>
<td>7001</td>
</tr>
<tr>
<td>8n1out</td>
<td>2</td>
<td>7002</td>
</tr>
<tr>
<td>pppInteractive</td>
<td>3</td>
<td>RRS IP Address</td>
</tr>
<tr>
<td>pppInteractive</td>
<td>4</td>
<td>RRS IP Address</td>
</tr>
<tr>
<td>csportd</td>
<td>5</td>
<td>2500</td>
</tr>
<tr>
<td>pppInteractive</td>
<td>6</td>
<td>RRS IP Address</td>
</tr>
<tr>
<td>csportd</td>
<td>7</td>
<td>2700</td>
</tr>
<tr>
<td>csportd</td>
<td>8</td>
<td>2800</td>
</tr>
<tr>
<td>csportd</td>
<td>9</td>
<td>2900</td>
</tr>
<tr>
<td>csportd</td>
<td>10</td>
<td>3000</td>
</tr>
<tr>
<td>csportd</td>
<td>11</td>
<td>3100</td>
</tr>
<tr>
<td>csportd</td>
<td>12</td>
<td>3200</td>
</tr>
<tr>
<td>csportd</td>
<td>13</td>
<td>3300</td>
</tr>
<tr>
<td>csportd</td>
<td>14</td>
<td>3400</td>
</tr>
<tr>
<td>csportd</td>
<td>15</td>
<td>3500</td>
</tr>
<tr>
<td>csportd</td>
<td>16</td>
<td>3600</td>
</tr>
<tr>
<td>csportd</td>
<td>17</td>
<td>3700</td>
</tr>
<tr>
<td>csportd</td>
<td>18</td>
<td>3800</td>
</tr>
<tr>
<td>csportd</td>
<td>19</td>
<td>3900</td>
</tr>
<tr>
<td>Device / Interface Type</td>
<td>Port Num</td>
<td>Logical Port</td>
</tr>
<tr>
<td>-------------------------</td>
<td>----------</td>
<td>--------------</td>
</tr>
<tr>
<td>csportd</td>
<td>20</td>
<td>4000</td>
</tr>
<tr>
<td>csportd</td>
<td>21</td>
<td>4100</td>
</tr>
<tr>
<td>csportd</td>
<td>22</td>
<td>4200</td>
</tr>
<tr>
<td>csportd</td>
<td>23</td>
<td>4300</td>
</tr>
<tr>
<td>csportd</td>
<td>24</td>
<td>4400</td>
</tr>
<tr>
<td>csportd</td>
<td>25</td>
<td>4500</td>
</tr>
<tr>
<td>csportd</td>
<td>26</td>
<td>4600</td>
</tr>
<tr>
<td>csportd</td>
<td>27</td>
<td>4700</td>
</tr>
<tr>
<td>VIR</td>
<td>28</td>
<td>7028</td>
</tr>
<tr>
<td>LLSW</td>
<td>29</td>
<td>7029</td>
</tr>
<tr>
<td>FAX</td>
<td>30</td>
<td>5000</td>
</tr>
<tr>
<td>LS2</td>
<td>31</td>
<td>7031</td>
</tr>
<tr>
<td>LS3</td>
<td>32</td>
<td>7032</td>
</tr>
</tbody>
</table>
Chapter 21
System Shutdown and Startup Procedures
## Chapter 21. System Shutdown and Startup Procedures

### Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>21.1 System Shutdown Procedures</td>
<td>1</td>
</tr>
<tr>
<td>21.2 AWIPS II System Startup Procedures</td>
<td>8</td>
</tr>
<tr>
<td>21.3 Verification of System Startup While Running AWIPS II Configuration</td>
<td>11</td>
</tr>
<tr>
<td>21.4 AWIPS II Checklist</td>
<td>16</td>
</tr>
<tr>
<td>21.5 LDM Ingest Checklist</td>
<td>18</td>
</tr>
</tbody>
</table>
As with all complex computer systems, a prescribed order of steps must be completed to shut down or restart AWIPS. It is necessary to follow this process in order to preserve the integrity of the data files and the safe termination of any hung or running processes.

A complete reboot of the system is not usually required, but if it is, the system manager must perform the AWIPS shutdown and restart.

This chapter includes shutdown and startup procedures for AWIPS II configuration only.

As noted in Chapter 2, AWIPS System Architecture, some dependencies exist within the AWIPS II systems. These dependencies dictate the best order for shutting down software and hardware, and are briefly described here:

- The HDF5 repository and postGreSQL persistence are served over a block-based Fibre Channel (FC) protocol implemented currently via the Direct Attached Storage (DAS) at WFOs and RFCs connected via FC to DX1 and DX2. At NCEP centers the same is achieved via the NetApp filer device.

- Data ingest processes such as the AWIPS Local Data Manager (Downstream) (LDM(D)), Radar Configuration Manager (RCM), and Local Data Acquisition and Dissemination (LDAD) rehosted applications write raw data into the /data_store mounted on the hosts off the NAS (NAS2).

- Environmental Data Exchange (EDEX) reads and processes raw data files from the /data_store mount.

- EDEX is dependent on both Queue Processor Interface Daemon (QPID) and the postGreSQL database engine running before it starts.

- AWIPS II site data configuration and localization files are stored on an NFS protocol mount physically hosted on the NAS device.

**NOTE:** You must be logged in as root to perform system shutdown and startup. Please notify the Network Control Facility at 301.713.9344 before shutting down any part of the system. This will minimize any miscommunication when the NCF begins getting ITO alarms.

### 21.1 System Shutdown Procedures

If a user is already logged into an LX or XT workstation (i.e., an X-session is running) go to the KMenu (the KMenu is indicated by the RedHat “shadow man” icon on the bottom of the panel) and left-click and select Logout. When the logout menu appears, select TURN OFF COMPUTER. If the workstation is at a GDM login prompt, select Actions from the GDM menu and Shut Down. When prompted to choose Cancel or Shut Down, select Shut Down. If you want to use a command line (manual) shutdown, then log in to a workstation as user root. Open a Terminal window. On each workstation LX, and XT,
TYPE: `shutdown -h 0`
- The workstations will power off automatically once the shutdown is complete; this is indicated by the LED either above or back-lighting the power-supply switch (located on the front of the workstation chassis) turning dark.
- Power off the LX, XT, and monitors when the monitor displays all black.

Complete the following steps to perform an orderly shutdown of an AWIPS II system

1. **[RFC Only] – Shut down rp2apps.**
   ```
   ssh rp2f hostname
   Use return hostname in the following command.
   Enter: `ssh << hostname >>`
   Enter: `hb_halt rp2apps`
   ```

2. **[RFC Only] – Shut down rp1apps.**
   ```
   ssh rp1f hostname
   Use return hostname in the following command.
   Enter: `ssh << hostname >>`
   Enter: `hb_halt rp1apps`
   ```

3. **[RFC Only] – Shut down RP1 and RP2.**
   ```
   Enter: `ssh rp2`
   Enter: `shutdown -h now && exit`
   Enter: `ssh rp1`
   Enter: `shutdown -h now && exit`
   ```

4. Log in and shut down the AX.
   ```
   Enter: `ssh ax`
   Enter: `service postgresql stop`  [RFC Only]
   Enter: `shutdown -h now && exit`
   ```

5. Stop a2px2apps.
   The a2px2apps runs peripheral system-level and LDAD application software. The LDAD software in particular is dependent upon having QPID running and the `/data_store` mount read/write accessible. The EDEX manual endpoint `/awips2/edex/data/manual` is also a prerequisite for the functionality of the
applications managed by the resource group a2px2apps. Other than LDAD external processes, no other system-wide dependencies exist on the a2px2apps running.

Enter: ssh px2f hostname

Get the hostname from the previous step and use in the next step.

Enter: ssh <hostname>
Enter: hb_halt a2px2apps
Enter: hb_stat [Ensure the package has stopped.]
Enter: exit [Back to dx1 shell.]

6. Stop a2px1apps.

The a2px1apps also runs peripheral system-level processes as well as rehosted application software such as MSAS, LAPS, and FSI. The rehosted software in particular is dependent upon having QPID running and the /data_store mount read/write accessible. The EDEX manual endpoint /awips2/edex/data/manual is also a prerequisite for functionality of the applications managed by the resource group a2px1apps. Other than the rehosted processes, no other system-wide dependencies exist on the a2px1apps running.

Enter: ssh px1f hostname

Get the hostname from the previous step and use in the next step.

Enter: ssh <hostname>
Enter: hb_halt a2px1apps
Enter: hb_stat [Ensure the package has stopped.]
Enter: exit [Back to dx1 shell.]

7. Shut down LS2 and LS3 servers.

Because the LDAD Server (LS) software is only dependent on the a2px2apps package, we can safely shut these down, now that the Linux Preprocessor (PX) server software has been shut down.

Enter: ssh ls3
Enter: shutdown -h now && exit

(This will stop all software.)

Connect to the LS2 via ssh and shut down the server.

Enter: ssh ls2
Enter: shutdown -h now && exit

(This will stop all software.)
8. Stop EDEX on dx4 (and dx5, dx6 at NCEP Centers)
   Enter: ssh dx4
   Enter: service edex_camel stop
   Enter: ps -fu awips [Ensure processes are no longer running.]
   Enter: exit [Return to dx1 shell.]

9. Stop RCM.
   Enter: ssh dx1f
   Enter: service edex_rcm stop
   Enter: ps -wef|grep rcm [Ensure process is no longer running.]
   Enter: exit

10. Stop a2dx2apps.
    Enter: ssh dx2f hostname
    Get the hostname from the previous step and use in the next step.
    Enter: ssh <hostname>
    Enter: hb_halt a2dx2apps
    Enter: hb_stat [Ensure the package has stopped.]
    Enter: exit

11. Stop EDEX on dx3
    Enter: ssh dx3
    Enter: service edex_camel stop
    Enter: ps -fu awips [Ensure processes are no longer running.]
    Enter: exit [Return to dx1 shell.]

12. Stop LDM on cpsbn1 and cpsbn2 and a2cp1apps on the host running the package.
    Enter: ssh cp1f hostname
    Use the return value from the above command in place of the word $HOST below.
    Enter: ssh $HOST
    Enter: hb_halt a2cp1apps
    Enter: hb_stat [Ensure the package has stopped.]
    Enter: exit
    Enter: ssh cpsbn1
Enter:  service ldmcp stop
Enter:  ps -fu ldm  [Ensure the processes have stopped.]
Enter:  exit
Enter:  ssh cpsbn2
Enter:  service ldmcp stop
Enter:  ps -fu ldm  [Ensure the processes have stopped.]
Enter:  exit

13. If this site has remote CPs (GUM, HFO, PBP and VRH), shut down qpid and pulse on px1.

Enter:  ssh px1
Enter:  service pulse stop
Enter:  ps -wef|grep pulse  [Ensure the processes have stopped.]
Enter:  service qpidd stop
Enter:  ps -fu awips  [Ensure the processes have stopped.]

14. Stop a2dx1apps.

Nothing left running relies on the PostgreSQL database engine running, so it is safe to stop all software running with the package.

Enter:  ssh dx1f hostname
Get the hostname from the previous step and use it in the next step.

Enter:  ssh <hostname>
Enter:  hb_halt a2dx1apps
Enter:  hb_stat  [Ensure the package has stopped.]
Enter:  exit

15. Shut down CPSBN1, CPSBN2, DX1, DX2, DX3, DX4, PX1, PX2.

Connect to cpsbn2 via the System Console
Enter:  shutdown -h now

Connect to cpsbn1 via the System Console
Enter:  shutdown -h now

Connect to px2 via the System Console
Enter:  shutdown -h now

Connect to px1 via the System Console
Enter: `shutdown -h now`

Connect to dx4 via the System Console
Enter: `shutdown -h now`

Connect to dx3 via the System Console
Enter: `shutdown -h now`

Connect to dx2 via the System Console
Enter: `shutdown -h now`

Connect to dx1 via the System Console
Enter: `shutdown -h now`

At NCEP Centers, also shut down the DX5 but keep the DX6 up until Step 16 is completed.

16. Shut down NAS.

Connect to the nas via the System Console
Enter: `halt -f`

NCEP Centers ONLY:

Connect via a shell originated on DX6, which was left up in the previous step, e.g., use the console server to connect to DX6 and login as root. Issue the following commands from the connection.

From a shell prompt on DX6 issue:

Enter: `umount -t nfs`

Once a command prompt returns:

Enter: `ssh nas1`

Enter: `halt -f 0`

At this point, shut down the clustered controller partner `nas2`:

Enter: `ssh nas2` (after exiting back to a DX6 shell prompt)

Enter: `halt -f 0`

Note that the ‘-f’ flag prevents failover of the clustered resources from nas1/nas2.

Now it is safe to shut down the DX6 from the console connection issue `shutdown -h 0`

17. Shut down DAS through the system console. Note: This must be done by completing the dx2-das head first, followed by the dx1-das.

Enter: `halt -f`
NCEP Centers: The NCEP Centers do not have a separate DAS device; the DAS/NAS functionality at a WFO/RFC is combined into the NetApp filer.

18. Power off the printers.

19. Verify that power has been shut down to the racks and the following devices:
   - All 4-port hubs (Phubs) and 5-port switches
   - All HP 24-port switches
   - All VIR communication switch panels
   - Modem nest
   - Routers
   - Terminal Server
   - ADTRAN
   - Channel Service Unit / Digital Service Unit (CSU/DSU)
   - System console
   - Plaintree LAN switches
   - Power Strips
   - DVB receivers
   - Direct-Attached Storage (DAS).

Note: The following steps are included here as a quick reference for the Linux Workstation and X-Terminal Manual Shutdown and Startup.

1. On the middle screen of the three-headed Linux workstation, click on the RedHat “shadowman” icon on the tool bar.

2. Select: Logout

3. After the windows have closed and you are logged out, on the GDM login GUI:
   Select: Actions
   Select: Shut Down

4. Select Y when prompted about whether you want to halt the workstation.

   NOTE: The Linux workstation will take several minutes to halt. You can follow its shutdown on the Cyclades Terminal Server.

5. The workstation should power itself off unless a problem is encountered. If it does not power off, hold down the power button until the power light goes out. (The network activity light will stay on.) Wait a moment, and then power the CPU on again.

When the GDM login screen displays, the workstation is ready for use.
21.2 *AWIPS II System Startup Procedures*

Complete the following steps to perform an orderly startup of an AWIPS II system.

1. Verify that power has been applied to the racks and the following devices:
   - All 4-port hubs (Phubs) and 5-port switches
   - All HP 24-port switches
   - All VIR communication switch panels
   - Modem nest
   - Routers
   - Terminal server
   - ADTRAN
   - CSU/DSU
   - System console
   - Power Strips
   - DVB receivers
   - Direct-Attached Storage (DAS)

2. Verify that the following conditions are met before proceeding:
   - NAS is powered on and up.
   - All switch settings on each VIR communication switch panel are powered on and set to A.

3. Power on the HP Thin Client (if it is not already powered on) and login as user root.

4. On the HP Thin Client, if no menu is displayed while being logged into ATS1/2, enter `ts_menu` to get the ATS menu prompt.

5. Power (or cycle power) to ETO tape drive and FCBR fiber bridge.

   **NCEP ONLY:** Power up the NAS1/2 (FAS3160A) NetApp Filer

7. Power on the CPs (CPSBNs).
   a. Verify that the CPSBN1 is accessible.
      
      **TYPE:** `ssh cpsbn1-<siteID>`
      - Where `<siteID>` is the site identifier

      **TYPE:** `exit`

   b. Verify that the CPSBN2 is accessible.
      
      **TYPE:** `ssh cpsbn2-<siteID>`
      - Where `<siteID>` is the site identifier
TYPE: exit


9. Ensure the DX1 server boots, and starts the a2dx1apps package.
   Connect to the dx1 via the System Console and log in as root once booted
   
   Enter: hb_stat

   If the package is NOT running, perform the following two steps:
   
   Enter: hb_run a2dx1apps
   Enter: hb_stat [Check for package run]

10. If not already running, start a2cp1apps, and QPID on CPSBN1. Then verify that they
    are accessible and that software is running. This is necessary because most processes
    have a dependency on QPID, but the processes running on the CPSBN do not
    depend on any other process or mount to function properly.

    Note: If site with remote CPs, then power up px1 at this time and start up pulse
    and qpid as indicated below.

    Connect to cpsbn1 via the System Console

    Check to see if a2cp1apps is running.
    
    Enter: hb_stat
    Enter: tail -f /var/log/cluster

    If the package fails to start after a few minutes, force the package start.
    
    Enter: hb_run a2cp1apps
    Enter: hb_stat [Check for package running.]

    Sites with remote CPs:

    Connect to cpsbn1 via the System Console

    Check if the LDM, QPID and pulse processes are running
    
    Enter: ps -fu ldm
    Enter: ps -fu awips
    Enter: ps -wefgrep pulse

    If the processes are NOT running, perform the following two steps:
    
    Enter: service ldmcp start
    Enter: ps -fu ldm [Check for processes.]
If QPID is not running, then perform the following two steps:

Enter:  service qpidd start
Enter:  ps -fu awips  [Check for qpidd.]
Enter:  ps -wef|grep pulse

If IPVS (pulse) is not running perform following two steps:

Enter:  service pulse start
Enter:  ps -wef|grep pulse  [Check for pulse]

Disconnect from cpsbn1.

Log into CPSBN2 and check to see if the LDM is running. If it is not running, use the above command to start the software before exiting the ssh session.

Enter:  ssh cpsbn2
Enter:  ps -fu ldm
Enter:  ps -wef|grep dvbs | grep -v grep
Enter:  exit

11. Power on the remaining servers (DX2, DX3, DX4, PX1, PX2, PX3, PX4, AX, LS3, LS2, RP1, and RP2).

12. Check for EDEX and start on DX3 if it is not already running.

Connect to the dx3 via the System Console and log in as root once booted

Enter:  ps -fu awips  [Make sure no processes are running.]
Enter:  service edex_camel start
Enter:  service edex_camel status  [Make sure all processes show running.]
Enter:  ps -fu awips  [Ensure all processes are running.]

13. Check for EDEX and start on DX4 if it is not already running.

Connect to the dx4 via the System Console and log in as root once booted

Enter:  ps -fu awips  [Make sure no processes are running.]
Enter:  service edex_camel start
Enter:  service edex_camel status  [Make sure all processes show running.]
Enter:  ps -fu awips  [Ensure all processes are running.]
NCEP Only: Power up the DX5 and DX6 using the same steps used for the DX4 in Step 13.

14. Check for the a2px1apps package and start on PX1 if it isn’t already running.

   Connect to the px1 via the System Console and log in as root once booted
   
   Enter: df /data_store
   Enter: hb_stat

   Look for the a2px1apps package running. If it is NOT running, issue the following two commands:
   
   Enter: hb_run a2px1apps
   Enter: hb_stat [Check for package running.]

15. Check for the a2px2apps package and start on PX2 if it isn’t already running.

   Connect to the px2 via the System Console and log in as root once booted
   
   Enter: df /data_store
   Enter: hb_stat

   Look for the a2px2apps package running. If it is NOT running, issue the following two commands:
   
   Enter: hb_run a2px2apps
   Enter: hb_stat [Check for package running.]

21.3 Verification of System Startup While Running AWIPS II Configuration

IMPORTANT ➔ Before continuing, ensure that all servers, workstations, and devices are powered-on and accessible via the network.

1. Verify that all HA packages are up and that a valid Start Time is reported via the hb_stat command.

   An example of a typical output of the hb_stat command, on DX1 and DX2, follows.
An example of a typical output of the `hb_stat` command, on DX3 and DX4, follows.

**dx[3|4]-tbdw is not part of an heartbeat cluster**

An example of a typical output of the `hb_stat` command, on PX1 and PX2, follows.

2. Verify that the pertinent mounts are present.

When running in AWIPS II configuration, the AWIPS I NFS and local mounts will still be active on all nodes at all times. In addition to those AWIPS I mounts, the following AWIPS II-specific mounts should be present.

- On **PX2**:
  - `/dev/mapper/vg00-lvol_awips2` ➔ `/awips2`
  - `nas1:/aiidata` ➔ `/awips2/edex/data`
  - `nas1:/GFESuite2` ➔ `/awips2/GFESuite`
  - `nas2:/data_store` ➔ `/data_store`
Chapter 21  System Shutdown and Startup Procedures

- On PX1:
  - /dev/mapper/vg00-lvol_awips2 → /awips2
  - nas1:/aiidata -> /awips2/edex/data
  - nas1:/GFESuite2 → /awips2/GFESuite
  - nas2:/data_store /data_store

- On DX1:
  - /dev/mapper/vg00-lvol_awips2 → /awips2
  - /dev/mapper/vg_aiidb-awipsiiddb -> /awips2/data
  - nas1:/aiidata/utility -> /awips2/edex/data/utility
  - nas1:/GFESuite2 → /awips2/GFESuite
  - nas2:/data_store /data_store

- On DX2:
  - /dev/mapper/vg00-lvol_awips2 → /awips2
  - /dev/mapper/vg_aiildm-awipsiildm → /data_store
  - nas1:/aiidata/utility -> /awips2/edex/data/utility
  - /dev/mapper/vg_aiihdf5-awipsiihdf5 -> /awips2/edex/data/hdf5
  - nas1:/GFESuite2 → /awips2/GFESuite

- On DX3 and DX4:
  - /dev/mapper/vg00-lvol_awips2 → /awips2
  - nas1:/aiidata → /awips2/edex/data
  - nas2:/data_store /data_store

3. Check for running processes.

- On DX3 and DX4 issue the following command:

  TYPE:  ps -ef | grep edex | cut -d “-” -f1,2 |
  grep -v grep

<table>
<thead>
<tr>
<th>awips</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>12247</td>
<td>1</td>
<td>013:40</td>
<td>00:00:00 ksh</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>/awips2/edex/data/share/hydroapps/precip_proc/bin/start_hpe</td>
</tr>
<tr>
<td>awips</td>
<td>27731</td>
<td>1</td>
<td>013:18</td>
</tr>
<tr>
<td></td>
<td>27792 27731</td>
<td>013:18</td>
<td>00:00:12 java -jar /awips2/edex/bin/yajsw/wrapper.jar</td>
</tr>
<tr>
<td>awips</td>
<td>27814 27792</td>
<td>53</td>
<td>13:18</td>
</tr>
<tr>
<td></td>
<td>27882 1</td>
<td>013:18</td>
<td>00:00:00 /bin/bash /awips2/edex/bin/start.sh -noConsole</td>
</tr>
<tr>
<td>awips</td>
<td>27943 27882</td>
<td>013:18</td>
<td>00:00:12 java -jar /awips2/edex/bin/yajsw/wrapper.jar</td>
</tr>
<tr>
<td>awips</td>
<td>27978 27943</td>
<td>29</td>
<td>13:18</td>
</tr>
<tr>
<td></td>
<td>28047 1</td>
<td>013:18</td>
<td>00:00:00 /bin/bash /awips2/edex/bin/start.sh -noConsole</td>
</tr>
<tr>
<td>awips</td>
<td>28108 28047</td>
<td>013:18</td>
<td>00:00:03 java -jar /awips2/edex/bin/yajsw/wrapper.jar</td>
</tr>
<tr>
<td>awips</td>
<td>28131 28108</td>
<td>14</td>
<td>13:18</td>
</tr>
<tr>
<td>awips</td>
<td>28203 1</td>
<td>013:18</td>
<td>00:00:00 /bin/bash /awips2/edex/bin/start.sh -noConsole</td>
</tr>
<tr>
<td>awips</td>
<td>28264 28203</td>
<td>013:18</td>
<td>00:00:04 java -jar /awips2/edex/bin/yajsw/wrapper.jar</td>
</tr>
<tr>
<td>awips</td>
<td>28290 28264</td>
<td>8</td>
<td>13:18</td>
</tr>
</tbody>
</table>
NOTE: Three start.sh scripts should be running (four if IngestDat is running); they are the parent processes of the three wrappers (the Java wrapper serves as, among other things, a watchdog for the JVM should it crash and need to be restarted). The System V init script edex_camel kicks off one start.sh, which launches the wrapper, which in turn launches the three JVMs: 1) ingestGrib; 2) request; and 3) ingest. In total, NINE (9) EDEX ESB Java-related processes should be running on DX3 and DX4 respectively. In addition, a single start_hpe process should be running; it controls the related HPE (e.g., DHRgather) processes.

- On DX1, issue the following command:

```
TYPE: ps -fu awips | grep -v idle | cut -d "-" -f1,2
```

<table>
<thead>
<tr>
<th>UID</th>
<th>PID</th>
<th>PPID</th>
<th>C</th>
<th>STIME</th>
<th>TTY</th>
<th>TIME</th>
<th>CMD</th>
</tr>
</thead>
<tbody>
<tr>
<td>awips</td>
<td>15139</td>
<td>1</td>
<td>1 Jul19 ?</td>
<td>00:57:02</td>
<td>/awips2/java/bin/java -cp /awips2/rcm/data/config/res::/awips2/rcm/lib/activation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>awips</td>
<td>24009</td>
<td>1</td>
<td>0 Jul18 ?</td>
<td>00:01:34</td>
<td>/awips2/postgresql/bin/postmaster -D /awips2/data</td>
<td></td>
<td></td>
</tr>
<tr>
<td>awips</td>
<td>24018 24009</td>
<td>0 Jul18 ?</td>
<td>00:00:39</td>
<td>postgres: logger process</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>awips</td>
<td>24020 24009</td>
<td>0 Jul18 ?</td>
<td>00:00:28</td>
<td>postgres: checkpointer process</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>awips</td>
<td>24021 24009</td>
<td>0 Jul18 ?</td>
<td>00:00:36</td>
<td>postgres: writer process</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>awips</td>
<td>24022 24009</td>
<td>0 Jul18 ?</td>
<td>00:00:14</td>
<td>postgres: wal writer process</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>awips</td>
<td>24023 24009</td>
<td>0 Jul18 ?</td>
<td>00:09:31</td>
<td>postgres: autovacuum launcher process</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>awips</td>
<td>24024 24009</td>
<td>0 Jul18 ?</td>
<td>00:27:02</td>
<td>postgres: stats collector process</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

NOTE: There should be a single postmaster process running which drives the postGreSQL AWIPS II database engine along with 6 subsequent process threads (logger, writer, wal writer, autovacuum, stats collector, checkpoint).

- On CPSBN1 check for the LDM processes running (ps –ef | grep ldm).
  - LDM should have pqact; edexBridge (connected to the server running Qpid); and ldmd.conf

- On DX2, there should be no AWIPS II-related process running. DX2 serves as a “hot spare” or “hot-standby” to the DX1 and the a2dx1apps resource.

- On PX1, issue the following command:

```
TYPE: ps -fu fxawips
```
Chapter 21 System Shutdown and Startup Procedures

NOTE: EIGHT (8) httpd threads should be running; they control the AWIPS II System Monitor. Two getty serial ports (named /dev/ttyn[a,b]) should be initialized; they are used by the asyncScheduler, which should also be running with the two async processes, asyncPilMsgServer and asyncPollData.pl. The AWIPS II hmMonitorServer and NWWSSchedule should be running. The GUI controller for IPVS (named piranha.gui) should be listed as well as the IPVS process nanny, connected on port 9581 to the IPADDR of the DX3 and DX4.

4. On PX2, issue the following commands:

TYPE:    ps -fU ldad,fxa,ldm | cut -d "-" -f1,2; ps -ef | egrep 'iscsi|nfs' | \  
grep ^V grep

NCEP Only: Note that some of the baseline fxalpha rehosted code is not operational at NCEP Centers; therefore, output might be different. Contact the NCF if you believe that system startup on the PX_SERVERS was not completed properly.

NOTE: The above command is all on one line. The “\” character at the end is an indication to the reader of such, and should not be included in the command issuance.

A table of running processes on PX2 follows.
### Chapter 21 System Shutdown and Startup Procedures

#### 21.4 AWIPS II Checklist

Start the browser on the workstation and check the System Monitor.

**DX1**

- Is postgres running (ps -ef | grep postgres)? Logging (/awips2/data/pg_log)?
- Is RadarServer running (ps -ef | grep Radar)? Logging (/awips2/rcm/data/logs)?
- Any partitions full (df)? Everything mounted (/awips2, /data_store, /awips2/data, /awips2/GFESuite, nas1:/awips2/edex/data/utility)?
- Heartbeat running, a2dx1apps running, dx1f up (ping dx1f)?
- MHS (ps -ef | grep mhs; ypcat hosts | grep mhs)? Sendmail? (ps -ef | grep sendmail; msg_send -atbdr -c0;)

---

**NOTE:** The AWIPS II LDAD and FSI processes should be running and match those in the output above (with the exception of the node domain name which will match your siteID).
Chapter 21 System Shutdown and Startup Procedures

DX2

- Heartbeat running, a2dx2apps running, dx2f up (ping dx2f)?
- Any partitions full (df)? Everything mounted (/awips2, /data_store, /awips2/GFESuite, nas1:/awips2/edex/data/utility, /awips2/edex/data/hdf5)?

DX3/4

- Is edex ingest, ingestGrib, ingestDat and request running (ps -ef | grep edex)?
- Are they logging ingesting data (/awips2/edex/logs, edex-ingest, edex-ingestGrib, ingestDat, request, ingest-satellite, ingest-radar, ingest-shef, ingest-text, ingest-purge, ingest-gen_areal_ffg)?
- Any partitions full (df)? Everything mounted (/awips2, /data_store, /awips2/edex/data, /awips2/GFESuite)?
- Clustering loopback device up (ifconfig lo:0)?
- (Also check PX3/4)

NCEP Only: Also check DX5 and DX6.

CPSBN1

- Is ldm running (ps -ef | grep ldm, ps –wef|grep dvbs)?
- Is data being received (su – ldm -c 'ldmadmin watch', check logs in ~ldm/logs)?
- Is qpid running (ps -ef | grep qpid)?
- IPVS running (ps -ef|grep pulse, ipvsadm)?
- Heartbeat running, a2cp1apps running, cp1f up (ping cp1f, hb_stat)?
- Any partitions full (df)? Everything mounted (/awips2, nas1:/awips2/qpid/messageStore)?

PX1

- Any partitions full (df)? Everything mounted (/awips2, /awips2/edex/data/share, /data_store)?
- Heartbeat running, a2px1apps running, px1f up (ping px1f)?
- Rehosted apps running (ps -ef | grep fxa)?

PX2

- Any partitions full (df)? Everything mounted (/awips2, dx2f:/data_store, /awips2/edex/data/share)?
- Heartbeat running, a2px2apps running, px2f up (ping px2f)?
Rehosted apps running (ps -ef | grep ldad; ps -ef|grep fsi)?

Workstations

DNS, NIS, NTP working? Any partitions full (df)? Everything mounted (/awips2/exdata/data/share, /data_store, /awips2)?

Launch CAVE. Does it come up okay? Data current in the menus and selected products load? (Load Radar (i.e., koax) > Composite Ref; Satellite > IR Window; Obs > Station Plot (load the latest under METAR); Maps > Hires TOPO Image (checkbox); Load GFE, make sure gfeConfig is selected in Config column, click OK.)

21.5 LDM Ingest Checklist

In order to add a line to the pqact.conf or pqact.local file for LDM ingest, certain information is necessary. The following is a general checklist of information that can be used when adding new data from the SBN into the system.

Note the WMO header or pattern for desired product: _______________________

Note the LDM FEEDTYPE for the desired product: _______________________

Note the raw archive location for the desired product: _______________________

Create entry in the pqact.conf or pqact.local file: _______________________

Restart or send HUP to ldmd process on cpsbn.

The following steps may be useful in finding the above information.

1. Find the WMO header or pattern for the desired product.
   If the product is being ingested into an AWIPS I system, look at the /data/fxa/customFiles/acqPatternAddOns.txt file. It is possible that you will discover the WMO pattern needed for ingesting. A listing of WMO header bulletins can also be found starting at the following URL: http://www.nws.noaa.gov/tg/table.html.

2. Find the LDM FEEDTYPE for the desired product.
   In all cases, the FEEDTYPE “ALL” will match all the FEEDTYPES listed. However, tools are available that may help identify the proper FEEDTYPE, if that is of interest. As user ldm the following commands might be useful:

   # ldmadmin watch

This command will show all data coming from the upstream LDM server. This should be every product that the upstream is ingesting. Watching can identify specific feedtypes of the products being ingested. A sample output of the command follows.
# notifyme -l -h $LDM_HOST -v -p '^[WMO ID Pattern]' 

This command will print to the screen every product available from <LDM Upstream IP/Hostname> that matches <WMO ID Pattern>. For example, the following would print all products starting T available from CPSBN2:

```bash
notifyme -l -h cpsbn2 -v -p '^T.*'
```

**Note:** The pattern passed to notifyme, and put into the pqact.conf file, must be a valid regular expression.

A sample output of the command follows.

```bash
Jan 21 15:17:37 notifyme[6887] INFO: Resolving adaml to 165.92.24.36 took 0.000398 seconds
```
Both of these commands will give other information that can be used in writing the data to disk in the raw archive in a more useful format.

3. Note the raw archive location for the product.

   In general, the raw archive used is /data_store/<product type>. For example, radar products are put into a subtree of /data_store/radar based on the information. It is good practice to continue to follow this paradigm. However, as long as the –edex argument is used in the pqact.conf entry, the product should be decoded no matter where the physical storage location ends up as long as the EDEX server has access to that location.

4. Create the pqact.conf entry.

   Create an entry in the pqact.conf file.

5. Restart or send HUP signal to LDM server.

   Restart or send an HUP signal to the LDM server.
Chapter 22
Maintenance Management Procedures
Chapter 22. Maintenance Management Procedures

Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>22.1 General System Administration Activities</td>
<td>1</td>
</tr>
<tr>
<td>22.1.1 Oversight and Administration of Site OT&amp;E Activities</td>
<td>1</td>
</tr>
<tr>
<td>22.1.2 Coordination with NCF and AWIPS Contractor</td>
<td>1</td>
</tr>
<tr>
<td>22.1.3 Performance and Administration of System Software Upgrades</td>
<td>2</td>
</tr>
<tr>
<td>22.1.4 Performance and Administration of System Hardware Upgrades and</td>
<td>3</td>
</tr>
<tr>
<td>Modifications</td>
<td></td>
</tr>
<tr>
<td>22.1.5 Coordination with Regional and National NWS AWIPS Staffs</td>
<td>3</td>
</tr>
<tr>
<td>22.1.6 Site Hardware Configurations</td>
<td>3</td>
</tr>
<tr>
<td>22.1.7 System Software Configurations</td>
<td>3</td>
</tr>
<tr>
<td>22.1.8 Site Applications and Extension Software Configurations</td>
<td>3</td>
</tr>
<tr>
<td>22.1.9 Site AWIPS System Security</td>
<td>4</td>
</tr>
<tr>
<td>22.1.10 Site Documentation Data</td>
<td>4</td>
</tr>
<tr>
<td>22.1.11 Global and Local AWIPS System Software Backups</td>
<td>4</td>
</tr>
<tr>
<td>22.1.12 Daily Monitoring and Administration of Operational System</td>
<td>4</td>
</tr>
<tr>
<td>22.1.13 EMRS Maintenance Data Reporting and Monitoring</td>
<td>4</td>
</tr>
<tr>
<td>22.2 System Maintenance Activities</td>
<td>5</td>
</tr>
<tr>
<td>22.2.1 Daily Actions</td>
<td>5</td>
</tr>
<tr>
<td>22.2.2 Weekly Actions</td>
<td>6</td>
</tr>
<tr>
<td>22.3 Maintenance Troubleshooting Checklist</td>
<td>6</td>
</tr>
<tr>
<td>22.3.1 Routine Maintenance Checklist</td>
<td>6</td>
</tr>
<tr>
<td>22.3.2 Unscheduled Maintenance Checklist</td>
<td>7</td>
</tr>
<tr>
<td>22.4 AWIPS Maintenance and System Administration Reporting</td>
<td>7</td>
</tr>
<tr>
<td>22.4.1 Modification/Software/Maintenance Note and Request for Change (RC)</td>
<td>8</td>
</tr>
<tr>
<td>Implementation</td>
<td></td>
</tr>
<tr>
<td>22.5 Demarcation of AWIPS Hardware Maintenance Responsibility</td>
<td>9</td>
</tr>
</tbody>
</table>

List of Tables

Table 22.4-1. AWIPS Equipment Codes ............................................................... 7
Table 22.4.1-1. Reporting Guidelines for Modifications .................................... 9
Because of the complexity of the AWIPS hardware, software, and other integrated devices, it is necessary to perform an extensive suite of maintenance activities on a regularly scheduled basis. Significant daily system monitoring and maintenance tasks are required to maintain a high level of successful operational capability.

The guidelines and tasks in this chapter are recommendations and form a base for the site-specific system(s) maintenance program. Maintenance shall be performed following NWSI 30-2113 AWIPS maintenance policy. For this chapter only, “ESA” is defined to mean Electronic Systems Analyst (ESA), Information Technology Officer (ITO), or electronic maintenance staff, as assigned.

22.1 General System Administration Activities

This section describes an overall system administrative and equipment maintenance plan and addresses specific activities for AWIPS II operational systems. It is intended as a reference for ESAs in maintaining their AWIPS sites. It is not all inclusive, and it is not directive in nature.

22.1.1 Oversight and Administration of Site OT&E Activities

Major functional releases undergo an Operational Test and Evaluation (OT&E) phase, and the ESA will play a large part in these activities. The test and evaluation phase is critical for generating feedback about the system to AWIPS developers and National Weather Service Headquarters (WSH) managers.

During OT&E, many facets of the site’s AWIPS operations and administration are evaluated by local staff and managers, and by assigned NWS personnel at WSH. For example, products, display capabilities, enhancements, communications, data feed from radar and the Satellite Broadcast Network (SBN), grids, data reception, data storage and retrieval, and many other areas of the overall system operation are scrutinized.

The ESA participates in the logging of OT&E data collection, ensuring that the evaluation data are forwarded to the NWS in a timely manner. Further, the ESA coordinates all software and hardware trouble reports with the Network Control Facility and administers all software and hardware modifications to the system.

22.1.2 Coordination with NCF and AWIPS Contractor

Coordination with the NCF is necessary to ensure monitoring of all site system hardware and software, and the efficient repair of any anomalies or failures.
The NCF’s telephone number, 301.713.9344, is intended for use by any operations or administrative staff member. The number should be prominently posted in all WFO/RFC operations areas.

The ESA performs preliminary examination of system failures and discusses findings with NCF staff. If necessary, the ESA should be able to engage in dialog with software developers (Raytheon, Systems Engineering Center [SEC], Global Systems Division [GSD]) and to discuss hardware issues with designers, engineers, and technicians associated with the Raytheon, Verizon (FTS2001), NOAANet, Globecommm, and the NCF.

When on-site repairs are necessary, the ESA should monitor the work of assigned contractor personnel and assist those technical contractors, when needed.

All maintenance actions, software, and hardware are documented in appropriate reports with the NCF and in the NWS Engineering Management Reporting System (EMRS).

22.1.3 Performance and Administration of System Software Upgrades

During the life cycle of the system, the ESA should be able and available to perform the many software installations and upgrades that will be necessary to ensure that the system is always configured according to all established configuration management guidelines. The ESA will be the site focal point for all system upgrades and will ensure that these upgrades are accomplished in accordance with authorized system upgrade instruction notes. The ESA will also ensure that all system upgrades at his or her site are documented in EMRS.

Software upgrades can take any of the following forms:

- **Application Releases.** The AWIPS Contractor distributes application releases to the sites by CD-ROM or DVD, depending on the size of the release. The ESA loads the software from the CD-ROM or DVD following the install instructions; the install may be monitored remotely by the AWIPS Contractor personnel.

- **Maintenance Releases and Emergency Releases.** Depending on the size of the release, maintenance upgrades and emergency releases may be distributed to the sites on CD-ROM or DVD, or downloaded to the sites over the WAN. The ESA accomplishes the upgrade using scripts and install instructions provided by the software developers. The installation may be monitored remotely by AWIPS Contractor personnel.

- **Patches.** Often, small short-term patches in system software are accomplished by developers, and an isolated program file is downloaded to the appropriate site system by the NCF, the AWIPS Contractor, or NWS developers. In all cases, the ESA will be notified of the changes being made.
22.1.4 Performance and Administration of System Hardware Upgrades and Modifications

Occasionally, AWIPS hardware configuration items must be upgraded or replaced with a newer device. As the site focal point for all such modifications, the ESA will interface directly with Contractor personnel to coordinate the time for the upgrade, and work with the actual hardware upgrade or replacement. The ESA will ensure that these upgrades are accomplished at the site in accordance with authorized instruction notes; the ESA will also ensure that all upgrades are documented in EMRS.

22.1.5 Coordination with Regional and National NWS AWIPS Staffs

The ESA should maintain a regular dialog with regional AWIPS staff members to ensure that all regional directives and guidelines are applied to the operation and maintenance of his or her system.

22.1.6 Site Hardware Configurations

In coordination with regional and national AWIPS staff, the ESA maintains records of all installed equipment. The ESA also ensures that all modifications and repairs are accomplished properly and that no unauthorized system configuration changes or additions are installed without approval from NWS Headquarters.

Many hardware records, materials reports, and documentation are delivered with the system. Some of these will be updated with each software or hardware modification. The ESA maintains all system hardware records.

22.1.7 System Software Configurations

In coordination with regional and national AWIPS staff, the ESA maintains records of all installed software systems. The ESA ensures that all system upgrades, modifications, and local applications are accomplished properly and that no unauthorized system configuration changes or additions are installed without approval from NWS Headquarters.

Each national software build and incremental upgrade will require updates to local site configuration files, map databases, and other records.

22.1.8 Site Applications and Extension Software Configurations

Ultimately, local sites will adapt their older PC-based software to AWIPS platforms and will generate new programs for local use on AWIPS. The ESA will maintain a record of all local applications suites and extensions and ensure that all site software operates properly in the AWIPS environment.
ESAs must be constantly aware of all local applications installed on their local systems and ensure that they do not adversely affect any approved and established system and applications software.

22.1.9 Site AWIPS System Security

In coordination with regional and national AWIPS staff, the ESA must be fully involved in all system security efforts. These responsibilities include the management of national security systems, as well as local site security issues.

The NWS operates a national firewall system in the National Weather Service Modernization and Test Integration System (NMT) in Silver Spring, Maryland. This firewall isolates all AWIPS intranet sites from unauthorized external influences. It is the site ESA’s responsibility to ensure that all access through the national firewall is registered. Refer to Appendix F, NWS/AWIPS Security Policy, for information on registering for firewall access.

Local firewalls will be developed and deployed as necessary to accommodate acquisition of other data (e.g., LDAD firewall).

22.1.10 Site Documentation Data

All site documentation should be maintained by the ESA. This includes operations manuals, system support documentation, installation records, maintenance records, system administration logs, and backup documentation and media.

22.1.11 Global and Local AWIPS System Software Backups

The ESA should ensure that all system backup procedures are properly established and correctly accomplished on a suitable schedule. These backups include system software and configuration data, database transaction records, local applications, and configuration software and data, as applicable.

22.1.12 Daily Monitoring and Administration of Operational System

The ESA must participate in the daily operational monitoring and administration of the local AWIPS. He or she will coordinate closely with the site operational personnel to ensure that all system functions are working correctly and take immediate action whenever anomalies are noted. Refer to checklist in Appendix I, Section I.11

22.1.13 EMRS Maintenance Data Reporting and Monitoring

The ESA will measure how effectively his or her site has satisfied EMRS reporting requirements by reviewing AWIPS maintenance activity, system upgrade, and modification reports via the EMRS web page
The ESA will take action to comply fully with guidelines identified in NWS Instruction 30-2104, Maintenance Data Documentation, available at http://www.nws.noaa.gov/directives/.

22.2 System Maintenance Activities

This section describes the type and frequency of AWIPS hardware maintenance actions recommended for NWS field ESAs and maintenance staff.

22.2.1 Daily Actions

ESAs should complete the following actions on a daily basis.

- **Coordinate with operational personnel.** The ESA should check with the operational staff members each morning to discuss the general operation of the system and any problems or anomalies that users may have noted. Ensure that all unusual situations described are entered in the daily site logs.

- **Ensure that daily logs are completed.** All assigned daily AWIPS log sheets should be reviewed and properly completed. All warranted action items noted should be resolved as quickly as possible.

- **Determine requirements for additional local backups.** As AWIPS users become familiar with the system, they will create local scripts and executable programs to be used for various purposes. These programs and data sets must be backed up at the site.

- **Check the Satellite Ground Station.** Visually inspect the ground station antenna for obstructions and proper heater operation. Clean as necessary.

- **Perform a general system check.** Open a terminal window and log in to each server and workstation. Verify that each device responds properly. Check for and remove core dumps. Check disk statistics and processes on each device, ensuring that all necessary programs are operating and that no runaway processes are present. Kill runaway processes as necessary. Check LAN and WAN connectivity by pinging another local device and then pinging a remote system, such as one of the NCF servers or an adjacent WFO device.

- **Launch the system data monitor, and verify that all data are current.** Investigate any evidence of late or missing data.

- **Review log files.** As necessary, review appropriate system log files, looking for error messages and other indications of communications, data retrieval, data routing, data storage, and data display anomalies and failures. Make appropriate notes in the site logs and take action to correct any noted deficiencies.

- **Verify D2D functionality.** In coordination with the users, operate D2D at each workstation, verifying proper menu availability and product display capability.
• **Visually check each workstation area for neatness and cleanliness.** Clean all monitor screens with appropriate glass cleaner and a soft, lint-free cloth. Check for other contaminants, such as spills in keyboards, and clean as necessary.

• **Verify printer functionality.** Check each system printer for proper operation. Verify that all copies are complete and colors are correct.

### 22.2.2 Weekly Actions

ESAs should complete the following actions on a weekly basis.

• **Back up applications and data sets.** Perform backups of all locally developed applications and data sets, as necessary.

• **Check power cables and connections.** Check the areas around each workstation, ensuring that all cables are properly distributed between the monitors and the workstation base unit and that connections are secure. Check cables, as well as all power cables and connections.

• **Clean work surfaces and base units.** With a soft, damp cloth, clean areas around monitors and workstation surface areas, removing any dust and other debris. Ensure that all dust and other debris are cleaned from all surfaces of workstation base unit, especially around cooling fans.

• **Check printer connections.** Check the areas around each system printer to ensure that all cables are properly distributed between the printer and servers and that connections are secure. Check all power cables and connections.

• **Clean printer surfaces.** With a soft, damp cloth, clean areas around printers and associated work surface areas, removing any dust and other debris. Open the printers and remove dust and other debris.

### 22.3 Maintenance Troubleshooting Checklist

This section includes two quick checklists – one for routine maintenance and one for unscheduled maintenance. The Routine Maintenance checklist includes items that should be checked daily. The Unscheduled Maintenance checklist includes items to be checked as needed.

### 22.3.1 Routine Maintenance Checklist

- Printer queue
- All devices. Partition sizes (using and `df` command for Linux servers)
- Cron system monitoring
22.3.2 Unscheduled Maintenance Checklist

- Hardware monitor
- Printer maintenance
- NCF interaction

22.4 AWIPS Maintenance and System Administration Reporting

AWIPS maintenance and system administration activities are documented in EMRS NWS Instruction 30-2104, Maintenance Data Documentation, issued under NWS Policy Directive 30-21, Systems Maintenance. Appendix F of this System Manager’s Manual describes policy and procedures necessary for documenting AWIPS maintenance and system administration.

As discussed in EMRS NWS Instruction 30-2104, in most cases one WS Form A-26 (http://www.weather.gov/wsom/manual/archives/NA149505.HTML#APPENDIX%20A) is initiated for each maintenance action performed. A single WS Form A-26 may be initiated on a biweekly basis to report all system administration duties. Initiate a WS Form A-26 for all designated AWIPS equipment, systems, or subsystems when the Field Electronics Staff:

- Resolves equipment failures
- Performs routine maintenance
- Relocates equipment
- Conducts special activities or data collection
- Activates deactivates, or modifies equipment
- Accomplishes system administration.

Contact the WSH Maintenance, Logistics, and Acquisition Division, Configuration Branch (W/OPS13), at 301.713.1892, for information or assistance regarding WS Form A-26.

Four equipment codes have been defined for use when documenting AWIPS maintenance activity in EMRS. The AWIPS equipment codes are listed in Table 22.4-1.

<table>
<thead>
<tr>
<th>Equipment Type</th>
<th>Equipment Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWIPS Local Data Acquisition and Dissemination Server</td>
<td>LDAD</td>
</tr>
<tr>
<td>AWIPS Remote Display</td>
<td>ARD</td>
</tr>
<tr>
<td>AWIPS Linux Servers and Workstations</td>
<td>AWLNX</td>
</tr>
<tr>
<td>AWIPS System Administration Reporting Code</td>
<td>AWIPS</td>
</tr>
</tbody>
</table>
NOTE: Use EMRS Equipment Code “FAC” to report incidental facilities maintenance/activities performed at the AWIPS site.

AWIPS equipment codes are activated in EMRS by the Field Staff. Submit one WS Form A-26 to report all activities associated with site preparation, installation, and checkout of the equipment, system, or subsystem. Field Staff may be responsible for activating particular AWIPS equipment as directed in an official Modification Note, Maintenance Note, Software Note, or Request for Change Implementation Memorandum. Follow the equipment-specific instructions included in the equipment change documentation to activate AWIPS equipment in EMRS. Prior to official activation, the NWS Electronics Staff will report all pre-acceptance activities using the reportable equipment code “AWIPS” and the serial number “001.”

AWIPS system administration is defined as all activities related to managing software operating systems and overseeing systems performance. This includes managing user access and privileges, configuring devices, making backups, training users, managing system security, installing approved operating system software changes and resolving fault isolation issues (e.g., software versus hardware failures).

Configuration Management (CM) information related to scheduled routine maintenance, corrective maintenance, and equipment management activities is documented by the AWIPS maintenance contractor. However, CM information related to activities performed by the NWS Field Staff is documented in EMRS using WS Form A-26. When AWIPS equipment fails, the NCF must be notified. The NCF may determine that the unit will be repaired by the maintenance contractor, returned to the National Reconditioning Center (NRC), or repaired by the Field Staff. When the removal and replacement of the failed unit is performed by the Field Staff, specific CM data reporting is required. These requirements are:

- For each part that fails, enter the correct Agency Stock Number and Serial Number of the failed part.
- For each replacement part, enter the correct Vendor Part Number and Serial Number of the replacement part.

NOTE: When replacing an entire subsystem or a part of the subsystem having a uniquely identified serial number, complete two separate WS Forms A-26. The first WS Form A-26 (Deactivation A-26) contains the serial number of the subsystem being removed. The second WS Form A-26 (Activation A-26) contains the serial number of the newly installed subsystem.

22.4.1 Modification/Software/Maintenance Note and Request for Change (RC) Implementation

In addition to the data reporting requirements outlined above, follow specific CM data reporting requirements outlined in each official instruction note. Ensure that the
modification number has been entered in the WS Form A-26 and that the date-of-action is completed. General reporting requirements are defined in Table 22.4.1-1.

<table>
<thead>
<tr>
<th>Modification Type</th>
<th>TM Code</th>
<th>AT Code</th>
<th>Mod. No. Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware Mod</td>
<td>M (Modification)</td>
<td>M (Modify)</td>
<td>The Mod #</td>
</tr>
<tr>
<td>Software Mod</td>
<td>M</td>
<td>M</td>
<td>S and the #</td>
</tr>
<tr>
<td>Software Patch Mod</td>
<td>M</td>
<td>M</td>
<td>SP and the #</td>
</tr>
<tr>
<td>Contractor Interface Note</td>
<td>M</td>
<td>M</td>
<td>CI and the #</td>
</tr>
<tr>
<td>System Administration Note</td>
<td>M</td>
<td>M</td>
<td>SA and the #</td>
</tr>
<tr>
<td>Information Note</td>
<td>E</td>
<td>M</td>
<td>I and the #</td>
</tr>
<tr>
<td>AWIPS System Security Note</td>
<td>M</td>
<td>M</td>
<td>SS and the #</td>
</tr>
<tr>
<td>Maintenance Note</td>
<td>E (Equip. Mgmt.)</td>
<td>M</td>
<td>M and the #</td>
</tr>
<tr>
<td>RC Implementation</td>
<td>E</td>
<td>M</td>
<td>The RC #</td>
</tr>
<tr>
<td>OtherMods</td>
<td>S (Special Acct.)</td>
<td>M</td>
<td>None</td>
</tr>
<tr>
<td>Application Software Installation</td>
<td>M</td>
<td>M</td>
<td>None</td>
</tr>
</tbody>
</table>

22.5 Demarcation of AWIPS Hardware Maintenance Responsibility

The AWIPS Contractor is responsible for providing AWIPS hardware maintenance. However, several Government hardware and communications interfaces to AWIPS require maintenance outside the scope of Contractor-provided maintenance. The ESA has the systems maintenance responsibilities for equipment such as the WSR-88D radar and various non-AWIPS workstations (i.e., other than those delivered with AWIPS). Routine and nonroutine hardware and software maintenance is performed by ESAs. Additionally, the NWS electronics staff is typically responsible for performing all system administration.

AWIPS maintenance providers are:

**Raytheon**
- Hardware maintenance and logistics support
- AWIPS Network Control Facility

**Globecomm Systems, Inc. (also known as GSI)**
- Services for the AWIPS SBN

**Verizon (FTS2001) and NOAANet (Sprint) (Government-furnished equipment and services)**
- Telecommunications services (FTS2001)
**NLSC (Government Maintenance)**

When equipment fails, the NCF will attempt to diagnose the cause of the problem. After determining whether the failure is software or hardware related, the NCF will take the appropriate action. During certain types of failures, the NCF may request assistance from the local electronics maintenance staff. Depending on the nature of the failure and the impact on the site’s operations, local electronics maintenance staff may work with the NCF or its maintenance subcontractor to repair and restore AWIPS functionality. Any Government-provided external interface to AWIPS is the responsibility of the NWS.

In accordance with NWS directives, any maintenance or system administration performed by NWS Field Staff in support of these activities, regardless of the problem source (hardware or software), shall be reported via EMRS. System administrative work shall be reported biweekly.

Examples of these tasks follow.

- Install system software upgrades.
- Ensure that all AWIPS functions operate normally after upgrades.
- Maintain knowledge of locally developed software.
- Ensure that the security policy is adhered to for local computer networks and interfaces.
- Administer locally assigned passwords.
- Perform backup/recovery for file systems.
- Perform manual system shutdown/startup.
- Manage system log files.
- Manage product retention/replenish parameters.
- Coordinate and oversee the maintenance contractor’s work.
- Advise NCF of progress or resolution of maintenance-related problems.

Site hardware maintenance tasks to be performed by the ESA follow.

- Manage AWIPS image tapes (create, store, insert, and remove tapes).
- Perform a hard reset of devices.
- Assist the NCF when troubleshooting possible cable or external interface problems.
- Visually verify device status lights or configurations when requested by the NCF.

Refer to EHB-13, Series II, for additional maintenance guidance.

Site administrators who experience problems or have concerns regarding the NCF or its maintenance providers should contact the Government point of contact for NCF operations:
Kevin Conaty
Voice 301.713.0864 x170
Fax 301.713.1409
Kevin.Conaty@noaa.gov
Chapter 23
NCF Archive
Chapter 23. NCF Archive

Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>23.1</td>
<td>Preparing to Launch the NCF Archive GUI</td>
<td>1</td>
</tr>
<tr>
<td>23.2</td>
<td>Executing and Using the NCF Archive GUI</td>
<td>1</td>
</tr>
<tr>
<td>23.3</td>
<td>Displaying Retrieved Archive Products</td>
<td>7</td>
</tr>
<tr>
<td>23.4</td>
<td>Changing NCF Archive Configuration at a Site</td>
<td>8</td>
</tr>
</tbody>
</table>

List of Exhibits

Exhibit 23.2-1. NCF Archive Window ................................................................. 2
Exhibit 23.2-2. Archive Request Window ............................................................ 2
Exhibit 23.2-3. Archive Help Screen for Archive Request Window .......................... 4
Exhibit 23.2-4. Archive Response Browser Window (All Requests Completed) ............. 5
Exhibit 23.2-5. Archive Response Browser Window (with Both Completed and Outstanding Requests) ................................................................. 5
Exhibit 23.2-6. Request Window ............................................................................ 6
Exhibit 23.2-7. NCF Archive Actions Window ......................................................... 7
Exhibit 23.4-1. Archive Config Window in NCF Archive ......................................... 9
The Network Control Facility Product Archive provides the capability to automatically store Guidance Products, Radar Data, Observations, and Official User Products for the most recent 15 days. Data/products are archived by WMO header and date/time group (DTG). Each AWIPS site can query the NCF Archive contents via an interactive GUI and retrieve these data/products for local site storage. Archive request parameters are currently sent from the site to the NCF via an SMTP message. At the NCF, the requested products are retrieved from the archive and transmitted to the requesting site via an SMTP message. The requesting site stores these products to files and updates the Archive Response Browser window to show receipt. At this time, only text products can be displayed. Text products can be viewed in a Telnet window using as described in Section 23.3.

23.1 Preparing to Launch the NCF Archive GUI

Change to a directory appropriate for temporary storage; create a new archive files subdirectory if one does not already exist. It is strongly recommended this subdirectory be created in the /tmp directory.

```
TYPE:     cd /tmp
TYPE:     mkdir archtest
TYPE:     cd archtest
```

**NOTE:** The NCF Archive GUI should be launched from this temporary directory. Any archived data/products received from the NCF will be stored in this directory. Files stored in non-temporary files will not be purged and may cause your file system to fill up. Files stored in /tmp are purged upon device reboot; however, it is recommended that you manually remove any files that you are not using from the /tmp/archtest directory.

23.2 Executing and Using the NCF Archive GUI

- To Execute the NCF Archive GUI on a workstation

```
TYPE:   /awips/ops/bin/arch_gui &
```

The NCF Archive GUI can be launched on any server or workstation; however, there will be less load on the system when the primary data server is used.

To execute the NCF Archive GUI remotely

```
TYPE: ssh dx1f"/awips/ops/bin/arch_gui"
```

Exhibit 23.2-1 displays the NCF Archive Window.
Chapter 23 NCF Archive

NOTES:

- The Store button on the NCF Archive menu has not been implemented and an information menu box will pop up with this information if that button is selected.

- The scan interval, max products, and max list can be changed using the Config button on the NCF Archive window. However, there is a 20-product maximum and a 200-list maximum. There is no predefined maximum on the scan interval; however, the users probably will not want to increase the default. Refer to section 23.4 for more information. Once the main menu is up, you can proceed to enter your request.

To Request Archived Data/Products

Select: Request from the NCF Archive Actions section of the NCF Archive window

- Displays the Archive Request window. See Exhibit 23.2-2.
Fill in request parameters as follows (every field is optional):

- **WMO Header.** This is the standard WMO header format “TTAAii CCCC <DTG>”; e.g., SAUS97 KMGM 120100. Wildcards may be used (see Note below).

- **NNNXXX.** The standard AFOS-based NNNXXX (e.g., MTRIAD) to request products in this manner.

- **SBN Channel.** The SBN Channel for the products you want to retrieve (GOES, NOAAPORT_OPT, NWSTG, and NWSTG2).

- **Start Date/Time and End Date/Time.** The time period (range) for which you want to retrieve products (i.e., the time the product was stored at the NCF). The NCF Archive GUI supports numerous DTG formats. It assumes times are in GMT. Some examples of operative DTG formats include the following:
  
  o 6/08/10 00:00:00Z
  o 6/08/10 11:31:20GMT
  o June 08, 2010
  o 11:30

- **Max Products.** The maximum number of products you want returned. This can be left blank to use the default (20), or you can specify the number of products to be returned. There are limits on the number of products returned.

**NOTE:** The NCF Archive GUI allows you to use asterisks (*) as wildcards (e.g., SAUS* or SAUS31 KWBC*). The Help feature explains this and provides examples. Exhibit 23.2-3 depicts the Help window for this screen. User-friendly help menus are available throughout this application.

Select: **Send** from the Request Actions section of the Archive Request window

- If your request exceeds the product limit, an information window pops up. This window gives you the option of accepting the default number of products. Select **OK** to continue.

- Displays **Send Confirmation** window with your archive request ID (e.g., TBDW-26369).

Select: **OK** on the Send Confirmation window

- Transmits your request to the NCF.

- Displays the NCF Archive window showing that the message was transmitted to the NCF.
Exhibit 23.2-3. Archive Help Screen for Archive Request Window

You can monitor the status of your request using the **Archive Response Browser**, launched by clicking on **Responses**. Refer to Exhibits 23.2-4 and 23.2-5. Both completed and outstanding requests are displayed. Outstanding requests have asterisks (***') across the **Receive Time** and **Response File** columns. When the requested products have arrived, the **Archive Browser**, **Receive Time**, and **Response File** fields are updated. All requests are displayed in the **Archive Response Browser** until they are deleted.

**NOTE:** Selecting the **Delete** button in the **Archive Response Browser** removes the request from the browser and deletes its query, header, and any retrieved products from the temporary store directory if the **arch_gui** was launched from the data server.

**NOTE:** The directory in which the retrieved products have been placed is reflected in the top part of the **Archive Browser** screen. Be sure to use a **/tmp** directory!
Exhibit 23.2-4. Archive Response Browser Window (All Requests Completed)

Exhibit 23.2-5. Archive Response Browser Window (with Both Completed and Outstanding Requests)

To view the products retrieved from the NCF Archive,
Select: The desired request from the **Archive Browser**
- Highlights the desired request.

Select: **View** from **Browser Options** in the **Archive Browser**
- Displays the Archive data for first 200 matches. Refer to Exhibit 23.2-6.
- Displays the maximum number of products and the number of matches in the request window.
- Adjusts the start and end times to narrow or refine your search.
- Displays each product’s file name in the **Path** column. The retrieved products are numbered sequentially.

![Request Window Image]

**Exhibit 23.2-6. Request Window**

**NOTES:**

- Products listed in the request window that do not have a path name have not been retrieved, but they are available and can be requested. If you want one of these products, highlight it and select the **Request** button at the bottom of the window to initiate a new request to the NCF for that specific data/product. This is a new request, with its own **Request ID**, so you must return to the **Archive Response Browser**. Highlight this new request to open a new window in which that retrieved product will be listed.
• The **Acquire** button allows you to store products to the database. The **NCF Archive Actions** window will display a message indicating that the file has been sent to the **acqserver** on the primary DX. An example is shown in Exhibit 23.2-7 (the appropriate text is highlighted).

• The **Display** button is not grayed out but you will receive a “not yet implemented” dialog box if you select it. It will be enabled in a later release and will allow you to display products on D2D and the Text Browser.

• The **Delete** button may be used to delete products from the request window. Selecting the **Delete** button removes the product both from the **Request Window** and from the temporary store directory if the **arch_gui** was launched from the data server.

![NCF Archive Actions Window](image)

**Exhibit 23.2-7. NCF Archive Actions Window**

### 23.3 Displaying Retrieved Archive Products

The display function of the NCF Archive is not currently implemented because this application was developed as a “proof of concept” and was not originally intended to be deployed to the field. However, retrieved text products can be displayed by using the **cat** or **more** commands in a Terminal window on any workstation.

In the same Terminal window from which you launched the **NCF Archive GUI**,

**TYPE:** `pwd`

- Check to ensure you are in the temporary directory in which the retrieved products were placed. If you are not in this directory, use the **cd** command to change to it.

**TYPE:** `ls -l`

- Displays a list of all the retrieved products. The file name format is `<device-requestID.prodNumber>` (e.g., TBDW-
26369.001, TBDW-26369.002), like the following sample output:

-rw-rw-rw- 1 smithel   fxalpha         226 Jun  8 17:31 TBW3-26369.qry
-rw-rw-rw- 1 root       sys          35568 Jun  8 17:33 TBW3-26369.hdr
-rw-rw-rw- 1 root       sys          2699 Jun  8 17:33 TBW3-26369.001
-rw-rw-rw- 1 root       sys          116 Jun  8 17:33 TBW3-26369.002
-rw-rw-rw- 1 root       sys          235 Jun  8 17:33 TBW3-26369.003
-rw-rw-rw- 1 root       sys          119 Jun  8 17:33 TBW3-26369.004
-rw-rw-rw- 1 root       sys          157 Jun  8 17:33 TBW3-26369.005
-rw-rw-rw- 1 root       sys          2607 Jun  8 17:33 TBW3-26369.006
-rw-rw-rw- 1 root       sys          140 Jun  8 17:33 TBW3-26369.007
-rw-rw-rw- 1 root       sys          116 Jun  8 17:33 TBW3-26369.008
-rw-rw-rw- 1 root       sys          113 Jun  8 17:33 TBW3-26369.009
-rw-rw-rw- 1 root       sys          125 Jun  8 17:33 TBW3-26369.010

Use the cat or more commands to view a specific text product.

**TYPE:**  
cat <RequestID.prodNumber>

or

**TYPE:**  
more <RequestID.prodNumber>

- Where <RequestID.prodNumber> is the text product to be retrieved (i.e., enter cat TBW3-26369.007 from the preceding example).
- Displays the retrieved text product in the Terminal window.

**NOTE:** The asterisk wild card can be used to view more than one product (e.g., more TBDW-26369*), but it will return a lot of data. It is strongly recommended that you use the more command in some capacity to view multiple products.

### 23.4 Changing NCF Archive Configuration at a Site

Click the Config button on the main menu to change configuration options. A maximum of 20 products in Max Products and a maximum of 200 lists in Max List Entries can be selected. There is no predefined maximum on the scan interval; however, the users probably will not want to increase the default.

**To Change NCF Archive Configuration at a Site**

**Select:**  
Config button from the NCF Archive window

- Displays the Archive Config window; refer to Exhibit 23.4-1.
- The configuration menu allows you to change configuration options, as shown in Exhibit 23.4-1.
Exhibit 23.4-1. Archive Config Window in NCF Archive
Chapter 24
Data Archive Server
Chapter 24. Data Archive Server

Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>24.1 WFO Archive Server</td>
<td>1</td>
</tr>
<tr>
<td>24.2 RFC Archive Server</td>
<td>1</td>
</tr>
</tbody>
</table>
This chapter describes the data archivers for the WFOs and the RFCs.

24.1 **WFO Archive Server**

The WFO Archive Server (WAX) is only used for tape backup. It runs the weekly nas backup script because the tape device is connected to it. Please refer to Section 16.3 of the *AWIPS CAVE-D2D User’s Manual* for information on archiving and archive case creation.

24.2 **RFC Archive Server**

The RFC Archive Server (RAX) supports the RFC mission of requiring collection and analysis of hydrologic data over much larger areas than the typical WFO. Because of a lack of storage space, much of the hydrologic data available from field collection sites historically has not been stored on the HP Data Servers. Thus there was no opportunity to process it. The RFC Archive Server addresses both of these issues. RAX provides additional disk storage capacity to allow this previously unstored data to be stored in a new database, and it also provides the processing power and associated software to process large amounts of hydrologic data. RAX is not a redundant server; however, it does have an SCSI RAID Level 5 storage array (Ultra32, ServeRAID-5i SCSI Controller). RAID-5 allows for a single drive to fail at any given time without any data loss. It also has a DVD Drive/Recorder (DVR-A04 Pioneer DVR), and data can be selected and moved to more permanent storage or written to a CD or DVD.

**NOTE:** Only DVD-R and CD-R media should be used to archive data. DVD+R media is no longer supported.

The hardware specifications are as follows:

- Dedicated system, Rack mounted
- Intel Xeon 2.4GHz/400MHz
- 2 - 512MB PC2100 CL2.5 ECC DDR SDRAM RDIMM
- Ultra 320, ServeRAID-5i SCSI Controller (single channel)
- Six 73.4GB 10K rpm Ultra160 SCSI HS
- 10/100/1000 Port Ethernet Server Adapter
- Tape drive - 40/80GB DLTVS HH Int. SCSI Drive (Half-High) and Ultra 160 PCI Adapter(required for Tape device when using ServeRAID5i
Chapter 25. AWIPS II/EDEX Administration Guide

Table of Contents

25.1 Document Conventions and Assumptions ................................................................. 1
25.2 AWIPS II Deployment ............................................................................................... 2
  25.2.1 EDEX Deployment Considerations ................................................................. 2
  25.2.2 Operational Considerations ......................................................................... 4
  25.2.3 Additional AWIPS II Software Packages ..................................................... 5
25.3 Basic Data Flow Concepts ...................................................................................... 6
  25.3.1 Basic Data Flow for ORPG/SPG Products .................................................... 8
25.4 System Management .............................................................................................. 9
  25.4.1 Managing EDEX on the DX3/4 Cluster .......................................................... 9
  Example 25.4.1-1: Checking Status of EDEX Processes ........................................... 10
  Example 25.4.1-2: Starting EDEX Processes ......................................................... 11
  Example 25.4.1-3: Starting EDEX Process (When Already Running) ....................... 11
  Example 25.4.1-4: Using ps to Check for EDEX Processes ....................................... 12
  25.4.2 Managing Software on CPSBN1/21/2 ............................................................ 16
  25.4.3 Managing Software on DX1/2 ..................................................................... 22
  25.4.4 Managing Software on PX1/2 ..................................................................... 26
  25.4.5 Managing Software on DX3/4 .................................................................... 26
25.5 System Configuration ............................................................................................ 31
25.6 EDEX Process Log and Log Format .................................................................... 32
  25.6.1 Process Logs .............................................................................................. 32
  25.6.2 Additional Log Files .................................................................................. 32
  25.6.3 Process Log File Naming Convention ......................................................... 32
  Example 25.6.3-1: Log File Names ........................................................................ 32
  25.6.4 Log Formats .............................................................................................. 32
  25.6.5 EDEX Logging Levels ............................................................................... 33
  Example 25.6.5-1: Monitoring EDEX Process Log for Errors ................................ 33
  25.6.6 Controlling Logging Levels ....................................................................... 34
25.7 EDEX System Monitoring ..................................................................................... 34
  25.7.1 Using Standard Linux Tools ..................................................................... 35
  Example 25.7.1-1: Determining the EDEX Version ................................................. 35
  Example 25.7.1-2: Determining EDEX Status ...................................................... 36
  Example 25.7.1-3: Determining EDEX Process Restarts ........................................ 37
  Example 25.7.1-4: Determining Available Data Types from Ingest ......................... 38
  Example 25.7.1-5: Monitoring Data Ingest ......................................................... 42
Example 25.7.1-6: Monitoring Data Ingest Latency .................................................................45
Example 25.7.1-7: Monitoring Ingest Failures ........................................................................48
Example 25.7.1-8: Verifying EDEX Ready to Execute ............................................................49
Example 25.7.1-9: Monitoring General EDEX Server Status ..................................................50
Example 25.7.1-10: Monitoring Just the EDEX Processes .......................................................52
  25.7.2 Using the JConsole Tool ..........................................................................................52
Example 25.7.2-1: Using JConsole to Monitor an EDEX Process ...........................................53
Example 25.7.2-2: Using JConsole to Examine Data Ingest Status ........................................55
Example 25.7.2-3: Using JConsole to Monitor Request Status ...............................................58
Example 25.7.2-4: Using JConsole to Change Logging Thresholds .......................................59
25.8 Other EDEX-Related Monitoring .................................................................................62
  25.8.1 Monitoring the Database .........................................................................................62
Example 25.8.1-1: Identifying the Database Used by EDEX ....................................................62
Example 25.8.1-2: Verifying that PostgreSQL Server Is Running ..........................................63
Example 25.8.1-3: Verifying Database Availability ...............................................................64
Example 25.8.1-4: Determining Data Type Tables in Metadata Database ...............................65
Example 25.8.1-5: Determine Latest Insertion Time of GRIB Metadata .................................66
Example 25.8.1-6: Determining Database Session Information Using psql ............................66
Example 25.8.1-7: Determining Data Type Tables in Metadata Database (Revisited) .............69
  25.8.2 Monitoring the Data Store ........................................................................................72
Example 25.8.2-1: Determining the Disk Space Used by the Data Store .................................72
Example 25.8.2-2: Modifying Data Retention Time ................................................................73
Example 25.8.2-3: Determining Available Data Types (in the Data Store) ..............................74
  25.8.3 Monitoring QPID ......................................................................................................78
Example 25.8.3-1: Verifying QPID Execution ........................................................................78
Example 25.8.3-2: Monitoring QPID Execution ....................................................................79
Example 25.8.3-3: Monitoring QPID Logging .......................................................................79
  25.8.3.1 QPID Monitoring Tools .......................................................................................81
Example 25.8.3.1-1: Determining the Status of the QPID Cluster ...........................................82
Example 25.8.3.1-2: Monitoring QPID Queue Status ............................................................83
Example 25.8.3.1-3: Monitoring Individual QPID Queues ....................................................84
  25.8.4 Monitoring IP Virtual Server (IPVS) .......................................................................86
Example 25.8.4-1: Verify/Monitor pulse Operation ...............................................................87
Example 25.8.4-2: Verify/Monitor lvsd Operation ..................................................................88
Example 25.8.4-3: Verify/Monitor nanny Operation .............................................................89
Example 25.8.4-4: Monitoring IPVS Using the ipvsadm Tool .................................................91
  25.8.5 Monitoring the LDM .................................................................................................94
Example 25.8.5-1: Verifying LDM Operation on the CP Cluster ............................................94
Example 25.8.5-2: Monitoring LDM Data Flow on the CP Cluster...........................................95
Example 25.8.5-3: Monitoring LDM Log on the CP Cluster.....................................................98
25.8.6 Monitoring Radar Server...........................................................................................99
Example 25.8.6-1: Verifying Radar Server Operation..............................................................99
Example 25.8.6-2: Monitoring RCM Performance Using top...............................................100
Example 25.8.6-3: Monitoring RCM Using JConsole.............................................................101
Example 25.8.6-4: Monitoring the Radar Server Log..............................................................102
25.8.7 Monitoring EDEX Subscription Script Running..........................................................103
Example 25.8.7-1: Identifying Registered Subscriptions – Using psql....................................104
Example 25.8.7-2: Identifying Registered Subscriptions – Using the CLI..............................108
Example 25.8.7-3: Identifying Text DB Triggers – Using the CLI..........................................110
Example 25.8.7-4: Determining if Subscriptions Are Firing from the EDEX Process Logs......112
Example 25.8.7-5: Determining if Subscriptions Are Firing Using JConsole.........................113
25.8.8 Monitoring EDEX Smart Init Processing....................................................................114
Example 25.8.8-1: Monitoring the EDEX Ingest Process Log for Smart Init Processing.........115
Example 25.8.8-2: Monitoring Smart Init Calculations (in the EDEX ingest process log).......116
Example 25.8.8-3: Monitoring Smart Init Using JConsole.....................................................117
25.9 HDF5 Tools ....................................................................................................................120
25.9.1 HDF5 Tools - Procedures............................................................................................120
Example 25.9.1-1: The h5ls Tool ............................................................................................120
Example 25.9.1-2: The h5dump Tool.....................................................................................120
Example 25.9.1-3: The h5stat Tool.........................................................................................121
Example 25.9.1-4: The h52gif Tool.........................................................................................121
25.9.2 The µEngine Web Test Driver....................................................................................121
Example 25.9.2-1: Retrieving a Satellite Image via the µEngine Web Test Driver...............121
25.10 CAVE/Workstation Setup Considerations....................................................................124
25.10.1 CAVE Startup Modes...............................................................................................124
25.10.2 Cave Installation.......................................................................................................124
25.10.3 CAVE Startup ..........................................................................................................125
Example 25.10.3-1: Validating the LX VIZ Installation..........................................................125
Example 25.10.3-2: Validating the XT VIZ Installation..........................................................126
Example 25.10.3-3: Cave Connection Problems.....................................................................126
25.11 Verifying That AWIPS II Processes Are Running.........................................................128
25.12 EDEX Startup and Shutdown Procedures.....................................................................134
25.12.1 AWIPS II/EDEX Software Dependencies............................................................134
25.12.2 Starting AWIPS II/EDEX Software.......................................................................135
25.12.3 Verifying Software Execution....................................................................................137
25.12.4 Stopping EDEX Software..........................................................................................138
25.12.5 Verifying Software Shutdown..........................................................140
25.12.6 Alternative Shutdown Scenarios ......................................................141
25.13 Troubleshooting Basic AWIPS II Data Flow...........................................146
Example 25.13-1: Troubleshooting Satellite Data Flow.................................147
Example 25.13-2: Troubleshooting Radar Data Flow.................................148

List of Exhibits
Exhibit 25.2.1-1. AWIPS II EDEX Installation.............................................2
Exhibit 25.2.1-2. EDEX Install Structure.....................................................3
Exhibit 25.3-1. Basic SBN Data Flow............................................................7
Exhibit 25.3.1-1. Local Radar Data Flow in AWIPS II..............................8
Exhibit 25.7.1-1. Basic top Display...............................................................51
Exhibit 25.7.1-2. Top Display with Selected Processes...............................52
Exhibit 25.7.2-1. Initial JConsole Display....................................................54
Exhibit 25.7.2-2. JConsole MBean Tab, Initial Display...............................55
Exhibit 25.7.2-3. JConsole MBean Tab with org.apache.camel Expanded....56
Exhibit 25.7.2-4. Partial Listing of Camel Routes..........................................57
Exhibit 25.7.2-5. Radar Decoder Route with Attributes Displayed.............57
Exhibit 25.7.2-6. EDEX Request in JConsole with Routes Expanded.........58
Exhibit 25.7.2-7. Micro Engine Http Thrift Bean Attributes.........................59
Exhibit 25.7.2-8. EDEX Logging Controls via JConsole.............................60
Exhibit 25.7.2-9. Set Level Operation of the Logging MBean....................60
Exhibit 25.7.2-10. Changing a Logging Level.............................................61
Exhibit 25.8.3-1. Top Display Monitoring QPID..........................................79
Exhibit 25.8.3.1-1. Typical Output of qpid-stat for Queue Monitoring..........83
Exhibit 25.8.3.1-2. Typical Output of qpid-stat Monitoring Ingest Queues....84
Exhibit 25.8.3.1-3. Typical Output of qpid-queue-stats Monitoring Ingest Queues 85
Exhibit 25.8.4-1. Client Interaction with CAVE via IPVS.........................86
Exhibit 25.8.4-2. Basic Hierarchy of Tools in IPVS......................................87
Exhibit 25.8.4-3. Top Output Monitoring pulse..........................................88
Exhibit 25.8.4-4. Top Output Monitoring lvsd...........................................89
Exhibit 25.8.4-5. Top Output Monitoring nanny........................................90
Exhibit 25.8.4-6. Monitoring IPVS Connections Using watch....................92
Exhibit 25.8.4-7. Monitoring IPVS Connection Statistics Using watch........93
Exhibit 25.8.4-8. Monitoring IPVS Rate Statistics Using watch................94
Exhibit 25.8.6-1. Using top to Monitor the Radar Server..........................100
Exhibit 25.8.6-2. Using JConsole to Monitor Radar Server......................101
This AWIPS II/EDEX Administration Guide provides information on the implementation of AWIPS II within the existing AWIPS system. It is intended as a resource for AWIPS System Managers who need a quick source of information for use in diagnosing and administering the AWIPS II software.

The structure of this administration guide differs from that of other SMM chapters. Sections 25.1, 25.2 and 25.3 describe conventions used throughout the chapter, outline AWIPS II deployment considerations, and address the basic data flow concepts respectively. The remaining sections focus on specific procedures for monitoring EDEX (the Environmental Data Exchange).

The procedural sections include a series of “examples,” which illustrate techniques that can be used to monitor or manage the AWIPS II/EDEX processes. The examples are broken down into “recipes,” which are actually step-by-step procedures (or “recipes”) for performing a specific task.

As you review the specific EDEX monitoring procedures presented here, remember that the following steps provide a general guide to checking EDEX operation.

- Check for running EDEX services.
- Check system log file for exceptions.
- Check for data ingest.
- Check for products in database.
- Check CAVE operation.

Finally, keep in mind that, unlike many of the diagrams presented elsewhere in the SMM, the diagrams in this chapter should be considered conceptual. For example, a diagram may have a single box representing a cluster.

### 25.1 Document Conventions and Assumptions

In order to provide a consistent presentation of information that reflects the known state of the AWIPS II system, several conventions have been adapted for use in this administration guide. In addition, assumptions were made about the AWIPS II installation environment. A list of these assumptions and conventions follows.

1. It is assumed that AWIPS II software on a server is installed into /awips2.
2. It is assumed that EDEX software is being installed on DX3 and DX4. On fielded systems, the server names may be dx3-xxx and dx4-xxx, where xxx is the site identifier.
3. In Examples involving user interaction with a server, it is assumed that the user will open a terminal and log into the server. A generic prompt ($) is used. User inputs are shown in an italic Courier New font; computer output is shown in a plain font, also Courier New.
4. All Examples utilize the Linux ssh utility to perform remote logins to the server; other remote login tools are available, for example, putty. If another tool is used, the Examples may need to be adjusted.

5. When you log on to a server, the server returns a banner for display by the client. To save space in the recipes, this banner is omitted from listings.

6. To simplify command line interactions, it is assumed that all tools are located on the user’s PATH. If that is not the case, command lines should be adjusted.

25.2 **AWIPS II Deployment**

EDEX is deployed to DX3 and DX4. Additional EDEX-related components are installed on other servers.

25.2.1 **EDEX Deployment Considerations**

On each DX3/4 box, the deployment consists of a set of directories under /awips2, as shown in Exhibit 25.2.1-1.

```
/awips2
 |   ... GFESuite
 |   ... adapt
 |   ... edex
 |   ... fxa
 |   ... java
 |   ... notification
 |   ... psql
 |   ... python
 |   ... qpid
 |   ... tools
(On DX 3/4)
```

**Exhibit 25.2.1-1. AWIPS II EDEX Installation**

**NOTE:** There is a single installation of EDEX; all EDEX instances run from this single installation. This increases operational flexibility while simplifying installation.

The installation structure under the edex directory is shown in Exhibit 25.2.1-2.
Exhibit 25.2.1-2. EDEX Install Structure

From an administration/monitoring standpoint, the important directories are `bin`, `conf`, `data`, and `logs`. For a list of specific files of interest in this installation tree, see Table 25.2.1-1.

Table 25.2.1-1. EDEX Installed Directories

<table>
<thead>
<tr>
<th>Directory</th>
<th>File</th>
<th>Description/Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>bin</td>
<td>start.sh</td>
<td>Basic startup script for EDEX. Normally, this file will not be changed once the system has been installed.</td>
</tr>
<tr>
<td></td>
<td>setup.env</td>
<td>Basic localization information for EDEX.</td>
</tr>
<tr>
<td></td>
<td><code>&lt;process&gt;</code>.pid</td>
<td>Contains the Product Identifier (PID) of the wrapper instance controlling an EDEX process. (<code>&lt;process&gt;</code> represents the name of the EDEX process; valid process names are <code>ingestDat</code>, <code>ingestGrib</code>, <code>ingest</code>, and <code>request</code>. These files exist only when the EDEX processes are executing.</td>
</tr>
<tr>
<td></td>
<td>versions.sh</td>
<td>Script to query the RPM database and return Name, Version, Release ID, Build Machine, and Install Date for AWIPS II RPMs.</td>
</tr>
<tr>
<td>conf</td>
<td>banner.txt</td>
<td>Contains the banner logged at each EDEX startup. The banner can be used to identify the installed version of EDEX.</td>
</tr>
<tr>
<td></td>
<td>modes.xml</td>
<td>Defines the EDEX operational modes: <code>ingestDat</code>, <code>ingestGrib</code>, <code>ingest</code>, and <code>request</code>. This file should not be modified; changing this file will adversely affect EDEX operation.</td>
</tr>
<tr>
<td></td>
<td>log4j-ingest.xml, log4j.xml, log4j-datatypeagent.xml, log4j-registry.xml</td>
<td>Defines EDEX logging. For details, see section 25.6.</td>
</tr>
<tr>
<td>data</td>
<td>--</td>
<td>Directory tree that contains the files utilized and/or generated by the EDEX processes. In the EDEX cluster, this directory is mounted to a shared directory (on the NAS).</td>
</tr>
</tbody>
</table>
25.2.2 Operational Considerations

EDEX has been re-engineered to allow it to run multiple configurations out of a single installation. This is done via command line switches. EDEX currently supports four command line switches – `ingest`, `ingestGrib`, `ingestDat`, and `request`. In operational settings, the syntax for starting EDEX is

```
/awips2/edex/bin/start.sh [mode]
```

The operational modes currently supported are shown in Table 25.2.2-1.

<table>
<thead>
<tr>
<th>EDEX Mode</th>
<th>Token</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ingest</td>
<td>ingest</td>
<td>Starts an EDEX (radar, satellite, text, trigger, gen_areal_ffg, shef-performance, smartInit, archive, GFEPerformance, archive, activeTableChange, performance, gen_areal_qpe, hibernate) process that supports Ingest operations.</td>
</tr>
<tr>
<td>DAT Ingest</td>
<td>ingestDAT</td>
<td>Starts an EDEX process that supports the ingest process for the Decision Aids Toolset (DAT).</td>
</tr>
<tr>
<td>GRIB Ingest</td>
<td>ingestGrib</td>
<td>Starts an EDEX process that supports ingest of GRIB products.</td>
</tr>
<tr>
<td>Request</td>
<td>request</td>
<td>Starts an EDEX process that supports Request operations.</td>
</tr>
</tbody>
</table>

If no mode is specified, a single EDEX process that supports all Ingest and Request operations is started. If an invalid mode is specified, EDEX will run long enough to generate a log for the specified (invalid) mode, but will then shut down.

Which operational modes will be used when starting EDEX depends upon the available memory on the EDEX server. The Ingest, GRIB Ingest and Request processes will always be started; the DAT Ingest process is started only on servers having at least 4 GByte of memory.
25.2.3 Additional AWIPS II Software Packages

In addition to EDEX installed on DX3 and DX4, AWIPS II installs additional software packages on several AWIPS servers. (See Exhibit 25.2-1.) These additional software packages provide functionality to support EDEX and are listed in Table 25.2.3-1.

Table 25.2.3-1. Additional AWIPS II Software Packages

<table>
<thead>
<tr>
<th>Package</th>
<th>Server</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>PostgreSQL*</td>
<td>DX1/2</td>
<td>Hosts the AWIPS II database, including rehosted application databases.</td>
</tr>
<tr>
<td>PyPIES</td>
<td>DX2</td>
<td>AWIPS II data store management software – performs read/write operations in the HDF5 data store.</td>
</tr>
<tr>
<td>QPID</td>
<td>CP1/2</td>
<td>Message broker – handles IPC between EDEX instances, LDM and EDEX, RCM and EDEX, EDEX and CAVE, etc.</td>
</tr>
<tr>
<td>RCM</td>
<td>DX1/2</td>
<td>AWIPS II Radar Server – provides the bridge between ORPG/SPG and EDEX ingest.</td>
</tr>
<tr>
<td>IPVS</td>
<td>CP1/2</td>
<td>IP Virtual Server – provides IP aliasing between DX3 and DX4.</td>
</tr>
<tr>
<td>LDM</td>
<td>CP1/2</td>
<td>Provides a bridge for data between the SBN and EDEX and RCM and EDEX.</td>
</tr>
</tbody>
</table>

*Postgres Database (v9.2.3 and v9.2.4): EDEX uses Postgres as the repository for storing metadata extracted during the data ingest process. The database also contains the text database, the maps database, and the hydro database.

In a typical AWIPS II installation, the Postgres database, by default, is installed on the dx1 server under the /awips2/postgresql directory. The following is an overview of some important directories and files in the postgres installation.

- `/awips2/postgresql/bin` ○ `start_developer_postgres.sh` and `start_postgres.sh`.
  Either of these scripts can be used to start the postgres server on a development workstation.

- `/awips2/psql`. This is the command line interface for interacting with the postgres database. A typical usage of this command to connect to the metadata database is as follows: `psql -d metadata -U awips -h dx1`. You would then enter the password for user awips, which is awips. Once connected, you are now able to interact with the database using SQL or the set of meta-commands provided by psql. Detailed documentation about using psql and psql’s meta-commands can be found here: [http://www.postgresql.org/docs/8.3/static/app-psql.html](http://www.postgresql.org/docs/8.3/static/app-psql.html)

Further documentation about the client applications in this directory can be found here: [http://www.postgresql.org/docs/8.3/static/reference-client.html](http://www.postgresql.org/docs/8.3/static/reference-client.html)

- `/awips2/postgresql/doc`. This directory contains a complete set of html documents detailing the usage of postgres.

- `/awips2/postgresql/include`. This directory contains code used internally by postgres and should not be manually modified.

- `/awips2/postgresql/lib`. This directory contains libraries used by postgres and should not be manually modified.
• `/awips2/postgresql/man`. UNIX man pages for client applications included in the PostgreSQL installation.

The `/awips2/data` directory on DX1 is used by PostgreSQL to store table information (schemas, table spaces, etc.) and user configurable files. Some important directories and files contained in this directory are:

• `/awips2/data/postgresql.conf`. This file controls a number of items defining how PostgreSQL behaves behind the scenes including memory usage, logging, and querying. Modifying items in this file can have significant performance implications. Therefore, modifications should be carefully considered. A series of documentation explaining the various configuration items contained in this file is here: [http://www.postgresql.org/docs/8.3/static/runtime-config-file-locations.html](http://www.postgresql.org/docs/8.3/static/runtime-config-file-locations.html).

• `/awips2/data/pg_hba.conf`. This file controls client connection permissions and authentication. Detailed documentation about this file and other client authentication concerns can be found here: [http://www.postgresql.org/docs/8.3/static/client-authentication.html](http://www.postgresql.org/docs/8.3/static/client-authentication.html).

• `/awips2/data/pg_log`. This directory contains the PostgreSQL logs. Logging behavior can be enabled/disabled and modified in the aforementioned `postgresql.conf` file.

• `/awips2/data/ident.conf`. This file controls PostgreSQL ident-based authentication. It maps # ident user names (typically UNIX user names) to their corresponding PostgreSQL user names.


**PGAdmin3**: PGAdmin3 is a graphical interface to view PostgreSQL databases. Refer to the PGAdmin3 documentation for usage details: [http://www.pgadmin.org/docs/](http://www.pgadmin.org/docs/).

### 25.3 Basic Data Flow Concepts

EDEX is fairly agnostic as to the actual source of its data; however, there are two specific routes by which data may be delivered to EDEX:

1. The data source moves a file into the manual processing directory (manual ingest); or
2. The data source sends a message to the message broker (QPID) defining the data that is ready to process. The message contains the path to a file containing the data to ingest.

Option 1, manual ingest, is beyond the scope of this administration guide.

Exhibit 25.3-1 shows the basic flow of data from the Satellite Broadcast Network (SBN) to the EDEX ingest processes.

Data received over the SBN includes non-local radar products, satellite imagery, GRIB data, ASCII text-based products such as METAR, and Terminal Aerodrome Forecast
(TAF) products and weather bulletins. All SBN data ingest shares the same general pattern. This general pattern is shown in Exhibit 25-3.1.

Exhibit 25.3-1. Basic SBN Data Flow

As shown in Exhibit 25.3-1:

1. Data is received from the SBN by the LDM. (Included is a sequence number to identify the data.)

2. The LDM performs two functions:
   a. LDM writes the product as a file to Raw Data Storage.
   b. A “data available” message is sent via the EDEX bridge to the QPID message broker.

3. The EDEX Ingest process obtains the “data available” message from QPID and determines the data decoder to use to ingest the data.
   a. The data decoder reads the raw data from Raw Data Storage and decodes it.

4. Once decoded, the data is persisted for later retrieval.
   a. The data is sent to the DBMS.
      i. Information about the data is persisted to the AWIPS II metadata database.
      ii. Some types of data, such as text products, are written to tables in the AWIPS II database.
b. Certain types of data (satellite imagery, radar imagery, GRIB data, and certain point data) are written to Hierarchical Data Format – 5 (HDF5) files in HDF5 data storage. This is performed via the PyPIES process on DX2.

5. In some cases, EDEX will automatically run scripts on arrival of the data.

To verify this data flow, log into the various servers in the system and examine log files to trace the data from the source to its final destination. You can also examine the database and file systems to verify that the data has been ingested.

If you discover that the data flow has been interrupted, make sure that the applications involved in the data flow are running. If one of the applications is not running, diagnose and correct the problem; then restart the application.

### 25.3.1 Basic Data Flow for ORPG/SPG Products

Local radar data is obtained from the ORPG/SPG and written to Raw Data Storage by the AWIPS II Radar Server. The Radar Server then passes a “data available” message to QPID. The basic data flow is shown in Exhibit 25.3.1-1.

As shown in Exhibit 25.3.1-1:

1. The RCM obtains a local radar product from the ORPG/SPG.
2. The Radar Server performs two operations for the data it receives.
   a. The Radar Server writes the product as a file to Raw Data Storage.
   b. The Radar Server posts a “data available” message to the QPID message broker.
3. The EDEX Ingest process obtains the “data available” message from QPID and determines the data decoder to use to ingest the data.
   a. The data decoder reads the raw data from Raw Data Storage and decodes it.
4. Once decoded, the data is persisted for later retrieval.
   a. The data is sent to the DBMS.
i. Information about the data is persisted to the AWIPS II metadata database.

ii. Some types of data, such as text products, are written to tables in the AWIPS II database.

b. Certain types of data, including radar imagery, are written to HDF5 files in Processed Data Storage. This is performed via the PyPIES process on DX2.

Note: In some cases, EDEX will automatically run scripts on arrival of the data.

To verify the local radar data flow, log into the various servers in the system and examine log files to trace the data from the source to its final destination. You can also examine the database and file systems to verify that the data has been ingested.

If you discover that the data flow has been interrupted, make sure the applications involved in the data flow are running. If one of the applications is not running, diagnose and correct the problem; then restart the application.

25.4 System Management

AWIPS II system management involves the control of both EDEX on the DX3/4 and the additional software packages described Section 25.2.3. This section covers basic system management for these software systems.

25.4.1 Managing EDEX on the DX3/4 Cluster

Once EDEX has been installed, the recommended means of control is via the edex_camel script, which is located in /etc/init.d. This script conforms to the standard interface for system control scripts:

```
/etc/init.d/edex_camel option [service [service] ...]
```

The standard options – start, stop, and restart – are supported, as are additional, mode-specific options. The options that are currently available are listed in Table 25.4.1-1.

<table>
<thead>
<tr>
<th>Option</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>start</td>
<td>Starts all EDEX processes</td>
</tr>
<tr>
<td>stop</td>
<td>Stops all EDEX processes</td>
</tr>
<tr>
<td>restart</td>
<td>Restarts all EDEX processes</td>
</tr>
<tr>
<td>reload</td>
<td>Requests a reload of all EDEX processes</td>
</tr>
<tr>
<td>status</td>
<td>Obtains status information for the EDEX processes</td>
</tr>
</tbody>
</table>

If no option is specified, a usage message is displayed. If one or more services are specified, only those services are used. For a list of available EDEX processes, see Section 25.2.2, Operational Considerations.

For example, to check the status of the request instance, the command line in use is

```
/etc/init.d/edex_camel status request
```
NOTE: *edex_camel* controls the EDEX processes on a single server. Because of the nature of the script, it should be used only by the root user. (There is one exception to the rule, i.e., any user may use the *status* option to determine if each EDEX process is running.)

The EDEX process logs routine status messages as well as error messages. Note that EDEX keeps separate logs on DX3 and DX4; as a result, both servers logs must be examined when monitoring the system and confirming system startup and shutdown. The locations of the EDEX logs on DX3 and DX4 are listed in Table 25.4.1-2.

<table>
<thead>
<tr>
<th>Server</th>
<th>EDEX Process</th>
<th>Log Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>DX3</td>
<td>EDEX Ingest (radar, satellite, text, trigger, gen_areal_ffg, shef-performance, smartInit, archive, GFEPerformance, gen_areal_qpe, activeTableChange, performance)</td>
<td>/awips2/edex/logs</td>
</tr>
<tr>
<td>DX3</td>
<td>EDEX DAT Ingest</td>
<td>/awips2/edex/logs</td>
</tr>
<tr>
<td>DX3</td>
<td>EDEX GRIB Ingest</td>
<td>/awips2/edex/logs</td>
</tr>
<tr>
<td>DX3</td>
<td>EDEX Request</td>
<td>/awips2/edex/logs</td>
</tr>
<tr>
<td>DX4</td>
<td>EDEX Ingest</td>
<td>/awips2/edex/logs</td>
</tr>
<tr>
<td>DX4</td>
<td>EDEX DAT Ingest</td>
<td>/awips2/edex/logs</td>
</tr>
<tr>
<td>DX4</td>
<td>EDEX GRIB Ingest</td>
<td>/awips2/edex/logs</td>
</tr>
<tr>
<td>DX4</td>
<td>EDEX Request</td>
<td>/awips2/edex/logs</td>
</tr>
</tbody>
</table>

Details on the EDEX process logs, including log file naming conventions, log entry formatting, and logging configuration, are covered in Section 25.6, EDEX Process Log and Log Format.

**Example 25.4.1-1: Checking Status of EDEX Processes**

This recipe presents two scenarios: 1) checking the status of all EDEX processes; and 2) checking the status of a specific EDEX process. The steps to follow for each scenario are:

**Scenario 1: To Check the Status of the EDEX Processes on DX3**

```
$ ssh root@dx3
root@dx3’s password:
$ cd /etc/init.d
$ ./edex_camel status

EDEX Camel (ingest) is running (wrapper PID 24173)
EDEX Camel (ingest) is running (java PID 24175)
EDEX Camel (ingestGrib) is running (wrapper PID 24272)
EDEX Camel (ingestGrib) is running (java PID 24274)
EDEX Camel (ingestDat) is running (wrapper PID 17309)
EDEX Camel (ingestDat) is running (java PID 17314)
EDEX Camel (request) is running (wrapper PID 24497)
EDEX Camel (request) is running (java PID 14130)
```
Scenario 2: To Check the Status of the EDEX Request Process on DX3

$ ssh root@dx3
root@dx3’s password:
$ cd /etc/init.d
$ ./edex_camel status request

EDEX Camel (request) is running (wrapper PID 24497)
EDEX Camel (request) is running (java PID 14130)

Example 25.4.1-2: Starting EDEX Processes

To start EDEX processes on DX4, use this recipe:

$ ssh root@dx4
root@dx4’s password:
$ cd /etc/init.d
$ ./edex_camel start

Starting EDEX Camel (ingest): WARNING: EDEX ingest instance already running, not starting another instance
OK
Starting EDEX Camel (ingestGrib): WARNING: EDEX ingestGrib instance already running, not starting another instance
OK
Starting EDEX Camel (ingestDat): WARNING: EDEX ingestDat instance already running, not starting another instance
OK
Starting EDEX Camel (request): WARNING: EDEX request instance already running, not starting another instance
OK

edex_camel does some checking prior to taking a start/stop action. Specifically, it checks for existing EDEX processes in each case. Then, for a start action:

1. It displays a warning if the process is already running, and
2. It displays a warning if the process is not already running.

In both cases, the action is terminated after the warning is issued.

Example 25.4.1-3: Starting EDEX Process (When Already Running)

To attempt to start the EDEX ingest process on DX3 when it is already running, these are the steps to follow:

$ ssh root@dx3
root@dx3’s password:
$ cd /etc/init.d
Starting EDEX Camel (ingest): WARNING: EDEX ingest instance already running, not starting another instance

OK

Because of the **SEDA Load Balancing** provided by the QPID message broker, an EDEX process running on DX3 or DX4 may be stopped with minimal impact to overall system operation. The remaining EDEX process, on the other DX box, will automatically pick up the full load. Restarting the EDEX process will result in the load balance being restored.

One additional concept on system management is this: EDEX is a Java process that runs as a Java Virtual Machine (JVM). To improve the reliability/robustness of EDEX, AWIPS II uses **Java Service Wrapper** (wrapper) from Tanuki Software to manage the JVM running each EDEX process. Wrapper is an independent process that starts and monitors the EDEX JVM; should the JVM crash, wrapper will restart the EDEX process automatically. Both the crash and the restart are logged to the EDEX process log.

**Example 25.4.1-4: Using ps to Check for EDEX Processes**

The Linux `ps` tool is commonly used to obtain basic information on processes running on a Linux server. Because `ps` will provide a bunch of information, the Linux `grep` tool is normally used to filter its output. Use `ps` to obtain information on the EDEX ingest process on DX3. The steps to follow:

```
$ ssh root@tbdr
root@tbdr’s password:
$ ps aux | grep ingest
```

`awips` output:

```
awips     5765  0.0  0.0 116848  1488 ?        S    Mar21   0:00
 /bin/bash /awips2/edex/bin/start.sh -noConsole -h ingest
awips     5848 23.0 18.1 6594632 1462316 ?     SNsl Mar21 1329:57
 /awips2/java/bin/java -Dedex.run.mode=ingest -
 Daw.site.identifier=RHA -Dedex.home=/awips2/edex -
 XX:MaxPermSize=128m -Dorg.apache.camel.jmx.disabled=true -
 Duser.timezone=GMT -XX:+UseConcMarkSweepGC -XX:+CMSIncrementalMode -
 Dqpid.dest_syntax=BURL -Ddb.addr=dx1f -Ddb.port=5432 -
 Ddc.db.name=dc_ob7rha -Dfxa.db.name=fxatext -Dhm.db.name=hmdb -
 Dih.db.name=hd_ob92rha -Ddb.metadata.pool.min=10 -
 Ddb.metadata.pool.max=50 -Ddata.archive.root=/data_store -
 Dedex.arch=64-bit -Dlog4j.configuration=log4j-ingest.xml -
 Dweb.port=8080 -DinitializeHibernatables=true -Dthrift.stream.maxsize=200 -
 Djstx.proxy=org.apache.catalina.connector.JSSEClientProtocol -
 Djstx.config=/awips2/edex/conf/resources/jstx.conf -
 Djava.library.path=/awips2/edex/lib/dependencies/org.jep.linux64/:/a
 wips2/java/jre/lib/amd64/server:/awips2/java/jre/lib/amd64:/awips2/j
 ava/jre/.../lib/amd64:/awips2/edex/lib/illusion:/awips2/edex/lib/nat
 ive/linux32/awips1:/awips2/java/lib/awips2/python/lib:/awips2/ps
 gilib:/awips2/edex/lib/dependencies/org.jep.linux64:/awips2/":"/a
 wips2/edex/bin/yajsw/"/wrapperApp.jar:/awips2/edex/bin/yajsw/wrapp
 er.jar:/awips2/edex/conf:/awips2/edex/conf/cache:/awips2/edex/conf/s
 pring:/awips2/edex/conf/resources:/awips2/edex/lib/dependencies/org.
```


-Xmx1024m -Dwrapper.port=15004 -Dwrapper.key=2092553382596361639 -Dwrapper.tmp.path=/tmp -Djna_tmpdir=/tmp -Dwrapper.config=/awips2/edex/conf/wrapper.conf

org.rzo.yajsw.app.WrapperJVMMain

awips 6064 0.0 0.0 116848 1488 ? S Mar21 0:00
/bin/bash /awips2/edex/bin/start.sh -noConsole -h ingestDat

awips 6161 0.9 7.8 6552988 635300 ? SNSl Mar21 53:46

-Xmx1024m -Dwrapper.port=15004 -Dwrapper.key=2092553382596361639 -Dwrapper.tmp.path=/tmp -Djna_tmpdir=/tmp -Dwrapper.config=/awips2/edex/conf/wrapper.conf

org.rzo.yajsw.app.WrapperJVMMain

awips 6064 0.0 0.0 116848 1488 ? S Mar21 0:00
/bin/bash /awips2/edex/bin/start.sh -noConsole -h ingestDat

awips 6161 0.9 7.8 6552988 635300 ? SNSl Mar21 53:46
/awips2/java/bin/java -Dedex.run.mode=ingestDat -Daw.site.identifier=RHA -Dedex.home=/awips2/edex -XX:MaxPermSize=128m -Dorg.apache.camel.jmx.disabled=true -Duser.timezone=GMT -XX:+UseConcMarkSweepGC -XX:+CMSIncrementalMode -
At this point, it is easy to see that further tuning of the `ps | grep` approach for monitoring an EDEX process is required. This requires more information about the EDEX system and is covered in more detail in section 25.7, EDEX System Monitoring.

### 25.4.2 Managing Software on CPSBN1/21/2

CPSBN1 and CPSBN2 function as a heartbeat cluster; failover is managed automatically or manually. Note that only a single high-availability package is currently used on this cluster. The package name is a2cp1apps. In failover mode, package execution shifts to the backup server. LDM runs only on the SBN CP server. The a2cp1apps package manages the cp1f floating server name. Note that floating names are available only when the high-availability packages are running; they cannot be used to start or stop the high-availability packages.

The AWIPS II services used on CPSBN1/2 are listed in Table 25.4.2-1.

#### Table 25.4.2-1. CPSBN1/2 Services

<table>
<thead>
<tr>
<th>System</th>
<th>AWIPS II Services</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPSBN1 and CPSBN2</td>
<td>LDM, Queue Processor Interface Daemon (Apache AMQP Message Broker) (QPID), Java, Python, rehost</td>
</tr>
</tbody>
</table>

The AWIPS II services on CPSBN1/2 are managed using the standard Linux `service` tool, which requires root access to run and is normally located /sbin. They can also be managed by directly executing the System V scripts located in /etc/init.d. In addition, LDM may be controlled directly using the `ldadmin` tool. The root user must switch user to ldm prior to running ldmadmin, however.

The AWIPS II services may be started with the Linux `service` command; the basic command format is `service <name> start`. To start the AWIPS II services on either CPSBN1 or CPSBN2, log into the target server using the root account and password.

1. To start ldmcp:
   
   Enter: `service ldmcp start`

2. To start qpidd:
   
   Enter: `service qpidd start`

3. To start pulse:
   
   Enter: `service pulse start`

The AWIPS II services may be stopped with the Linux `service` command; the basic command format is `service <name> stop`. To stop the AWIPS II services on either CPSBN1 or CPSBN2, log into the target server using the root account and password.

1. To stop ldmcp:
   
   Enter: `service ldmcp stop`

2. To stop qpidd:
   
   Enter: `service qpidd stop`
3. To stop pulse:

   **Enter:** `service pulse stop`

The AWIPS II services may be status checked with the Linux `service` command; the basic command format is `service <name> status`. It is also useful to use the Linux `ps` command to operation.

The CP1/2 cluster is a heartbeat cluster, although the architecture is somewhat different from that of DX1/2. On the CP1/2 cluster, the LDM processes will normally run on both CPSBN1 and CPSBN2 and are not heartbeat managed. The heartbeat manages QPID and pulse (IPVS), which run on one of the servers. This server is accessed via the cp1f floater. You will log into the servers using the `root` login and password.

The software to check on the CP1/2 cluster consists of:

- The heartbeat service (CPSBN1 and CPSBN2)
- The LDM processes (CPSBN1 and CPSPN2)
- The QPID message broker (cp1f)
- The pulse (IPVS) service (cp1f)

Dataflow to check on the CP1/2 cluster:

- LDM data flow (on CPSBN1 and CPSBN2)

For verification, perform these steps:

a. Open a terminal window on your workstation.

b. Log into CPSBN1 using the `root` login.

   **Enter:** `ssh root@cpsbn1`

c. Verify the heartbeat service is running.

   **Enter:** `service heartbeat status`

   ```
   heartbeat OK [pid 28624 et al] is running on cpsbn1-tbdw [cpsbn1-tbdw]...
   ```

d. Verify LDM processes are running.

   **Enter:** `ps -ef | grep ldm | grep -v grep | cut -b -80`

   ```
   ldm 30195  1 0 Jul10 ?  00:00:00 ldm -I 0.0.0.0 -P 388
   -M 256 -m
   ```

   **Enter:** `ps -wef | egrep noaaportIngest`
Verify LDM data flow.

Enter: su - ldm
Enter: ldmadmin watch

**NOTE:** Look for a continuous stream of “INFO:” messages; press <Ctrl-D> to terminate the watch.
Enter:

**exit**

f. Log off CPSBN1.

Enter:

**exit**

g. Log into CPSBN2 using the root login.

Enter:

**ssh root@cpsbn2**

h. Verify the heartbeat service is running.

Enter:

**service heartbeat status**

heartbeat OK [pid 5029 et al] is running on cpsbn2-tbdw [cpsbn2-tbdw]...

i. Verify the LDM processes are running.

Enter:

**ps -ef | grep ldm | grep -v grep | cut -b -80**

```
ldm  10274  1  0 Mar17 ?  00:00:02 ldm -I 165.92.24.51 -P 388 -M 2
ldm  10276 10274  0 Mar17 ?  01:28:26 pqact -e
ldm  10277 10274  0 Mar17 ?  00:12:28 edexBridge -vxl /usr/local/ldm/l
```

Enter:

**ps -wef | egrep noaaportIngest**

```
root     11738 11734  0  2013 ?  03:43:59 noaaportIngestester -b
50000  -m 224.0.1.1 -n 3 -u mhs -r 1 -s NMC
root     11739 11734  0  2013 ?  00:14:00 noaaportIngestester -b
50000  -m 224.0.1.2 -n 4 -u mhs -r 1 -s GOES
root     11740 11734  0  2013 ?  03:56:30 noaaportIngestester -b
50000  -m 224.0.1.3 -n 5 -u mhs -r 1 -s NMC2
root     11741 11734  0  2013 ?  00:23:14 noaaportIngestester -b
50000  -m 224.0.1.4 -n 6 -u mhs -r 1 -s NOAAPORT_OPT
root     11743 11734  0  2013 ?  00:31:30 noaaportIngestester -b
50000  -m 224.0.1.5 -n 7 -u mhs -r 1 -s NMC3
ldm      17115 16965  0 15:19 pts/l  00:00:00 grep noaaportIngestest
```
j. Verify LDM data flow.

Enter:  su - ldm

Enter:  ldmadmin watch

Note: Look for a continuous stream of “INFO:” messages; press <Ctrl-D> to terminate the watch.
Enter:  exit

k. Log off CPSBN2.

Enter:  exit

l. Log into cp1f using the root account.

Enter:  ssh root@cp1f

m. Verify QPID is running.

Enter:  service qpidd status

qpidd (pid 29150) is running...

Enter:  ps –ef | grep qpidd | grep –v grep | cut –b -80

awips 29150 1 16 Oct20 ? 09:04:15
/awips2/qpid/sbin/qpidd –daemon

n. Verify pulse (IPVS) is running.

Enter:  service pulse status

pulse (pid 3906) is running..

Enter:  ps –ef | grep pulse | grep –v grep | cut –b -80

root 3906 1 0 May24 ? 00:00:00 pulse

o. Log off cp1f.

Enter:  exit

p. Error results: As a general rule, if one of the processes is not running, the ps command will not print anything. If the service commands return a “not running” message, heartbeat is not running and automated failover will fail. If the floater (cp1f) login fails, this means that the high-availability package controlling the software is not running.
25.4.3 Managing Software on DX1/2

DX1 and DX2 function as a heartbeat cluster; failover is managed automatically or manually. Note that both DX1 and DX2 normally run separate high-availability packages; a2dx1apps on DX1 and a2dx2apps on DX2. In failover mode, both packages shift to the same server. Access to DX1 and DX2 is via server name alias: dx1f points to the server running the a2dx1apps package; dx2f points to the server running the a2dx2apps package. The high-availability packages are also used to manage several shared mounts to the Direct Attached Storage (DAS) and Network Attached Storage (NAS (NAS1 and NAS2)). Note that the floating names are available only when the high-availability packages are running; they cannot be used to start or stop the high-availability packages. The AWIPS II services used on DX1/2 are listed in Table 25.4.3-1. The NAS and DAS file system mounts are listed in Table 25.4.3-2.

<table>
<thead>
<tr>
<th>Service</th>
<th>Applications Controlled</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>edex_postgres</td>
<td>PostgreSQL DBMS</td>
<td>managed by a2dx1apps</td>
</tr>
<tr>
<td>edex_rcm</td>
<td>Radar Server</td>
<td>managed by a2dx1apps</td>
</tr>
<tr>
<td>httpd-pypies</td>
<td>PyPIES</td>
<td>managed by a2dx2apps</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Mount</th>
<th>Mount Point</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>das1:/aiihdf5</td>
<td>/awips2/edex/data/hdf5</td>
<td>managed by a2dx1apps</td>
</tr>
<tr>
<td>das1:/aiidb</td>
<td>/awips2/data</td>
<td>managed by a2dx1apps</td>
</tr>
<tr>
<td>nas2:/data_store</td>
<td>/data_store</td>
<td>managed by a2dx2apps</td>
</tr>
<tr>
<td>nas1:/aidata/utility</td>
<td>/awips2/edex/data/utility</td>
<td>managed by a2dx1apps</td>
</tr>
<tr>
<td>nas1:/GFESuite2</td>
<td>/awips2/GFESuite</td>
<td>managed by a2dx1apps</td>
</tr>
</tbody>
</table>

In addition to the AWIPS II services listed in Table 25.4.3-1, several re-hosted AWIPS II services continue to operate on DX1/2; these services are managed by the appropriate high-availability package.

In addition to using the heartbeat utilities for controlling the AWIPS II services, they may be managed individually. The edex_postgres, edex_rpm, and httpd-pypies are controlled using the Linux `service` command.

The DX1/2 cluster is a heartbeat cluster with both nodes normally running AWIPS II processes. The two cluster nodes are accessible via the virtual server names dx1f and dx2f. You will be able to log into the appropriate node using the floating name dx1f or dx2f, and the root account and password.

The software to check on the DX1/2 cluster consists of:

- The heartbeat service (DX1 and DX2)
- The PostgreSQL database server (on dx1f)
- The Radar Server (on dx1f)
• The PyPIES data server (on dx2f)

Dataflow to check on the DX1/2 cluster:
• Radar Server data flow (on dx1f)

For verification, perform these steps:

a. Open a terminal window on your workstation.

b. Log into dx1f using the root login.

   Enter: ssh root@dx1f

c. Verify that the heartbeat service is running.

   Enter: service heartbeat status

   heartbeat OK [pid 1515 et al] is running on dx1-tbdw [dx1-tbdw]...

d. Verify that the PostgreSQL server is running.

   Enter: ps -ef | grep postmaster | grep -v grep | cut -b -80

   awips 8003 1 0 Oct20 ? 00:00:25 /awips2/postgresql/bin/postmaste

e. Verify that the Radar Server (RCM) is running.

   Enter: ps -ef | egrep "[/]rcm/" | tr -s " " | cut -b -80

   awips 8387 1 2 Oct20 ? 01:27:45 /awips2/java/bin/java -cp /awips2/rcm/data/confi

f. Verify Radar Server data flow.

   Enter: tail -f /awips2/rcm/data/logs/radarserver.log

   Note: The latest radarserver log will be named as radarserver.log.<YYYY-MM-DD>.

   Note: Look for messages containing “Stored message in”; press <Ctrl-C> to terminate tail.
g. Log off dx1f.
   Enter: exit

h. Log into dx2f using the root login.
   Enter: ssh root@dx2f

i. Verify the heartbeat service is running.
   Enter: service heartbeat status

   heartbeat OK [pid 1515 et al] is running on dx2-tbdw [dx2-tbdw]...

j. Verify that the PyPIES data server is running.
   Enter: ps -ef | grep httpd_pypies | grep -v grep | cut -b -80
### Log off `dx2f`

**Enter:** `exit`

### Error Results

As a general rule, if any of the `ps` commands has an empty return, the specific process being checked is not running. If the service commands return a “not running” message, heartbeat is not running and automated failover will fail. If either login fails, this means the high-availability package controlling the software is not running.
25.4.4 Managing Software on PX1/2

No AWIPS II specific processes have been added to PX1/2. Rather, PX1/2 are used to run various re-hosted applications. As in AWIPS I, the PX1/2 cluster is a heartbeat cluster. The main operational change is the names of the high-availability packages. For AWIPS II they are a2px1apps and a2px2apps.

The AWIPS II high-availability packages on PX1 are controlled using the heartbeat utilities: hb_run, hb_halt, and hb_stat. In normal operation, only the a2px1apps high-availability package is running on PX1.

1. To start a2px1apps:
   
   Enter: `hb_run a2px1apps`

2. To stop a2px1apps:

   Enter: `hb_halt a2px1apps`

3. To verify a2px1apps:

   Enter: `hb_stat`

The AWIPS II high-availability packages on PX2 are controlled using the heartbeat utilities: hb_run, hb_halt, and hb_stat. In normal operation, only the a2px2apps high-availability package is running on PX2.

1. To start a2px2apps:

   Enter: `hb_run a2px2apps`

2. To stop a2px2apps:

   Enter: `hb_halt a2px2apps`

3. To verify a2px2apps:

   Enter: `hb_stat`

Logging by the re-hosted apps on PX1/2 is generally unchanged. Specifically, log locations are unchanged for the re-hosted applications.

25.4.5 Managing Software on DX3/4

DX3 and DX4 are independent servers, both of which run the EDEX server software. **There is no failover between DX3 and DX4.** Both servers are load balanced and share data processing responsibilities; if one server is stopped, the other will pick up the entire data processing load. Processing load balancing between DX3 and DX4 is provided by QPID as data messages are removed from QPID queues and processed by the EDEX instance having available processing capacity. Client request balancing for DX3 and DX4 is managed by Internet Protocol Virtual Server (IPVS) (the pulse service). Both QPID and IPVS run on CPSBN1 and CPSBN2.
Unlike the other servers, the AWIPS II software must be verified on both DX3 and DX4. Access to both servers is by direct login. The steps in this procedure must be performed on both DX3 and DX4 using the root login and password.

The software to check on DX3 and DX4 consists of:
- EDEX server software.

Dataflow to check on the DX3/4:
- EDEX ingest processes (on DX3 and DX4).

For verification, perform these steps:

a. Open a terminal window on your workstation.

b. Log into DX3 using the root login.

   Enter: `ssh root@dx3`

c. Determine the available memory on the server.

   Enter: `free -g`

   \[
   \begin{array}{ccccccc}
   \text{total} & \text{used} & \text{free} & \text{shared} & \text{buffers} & \text{cached} \\
   \text{Mem:} & 7 & 6 & 0 & 0 & 0 & 1 \\
   -/+ buffers/cache: & 5 & 2 \\
   \text{Swap:} & 2 & 0 & 2 \\
   \end{array}
   \]

d. Verify the EDEX server software is running.

   \textbf{Note:} There are four possible instances of EDEX: ingest, request, ingestGrib, and ingestDat. If the available memory indicated by the `free -g` command is less than 4 GB, ingestDat will not be running.

   Enter: `service edex_camel status`

   EDEX Camel (ingest) is running (wrapper PID 27792)
   EDEX Camel (ingest) is running (java PID 27794)
   EDEX Camel (ingestGrib) is running (wrapper PID 27896)
   EDEX Camel (ingestGrib) is running (java PID 27898)
   EDEX Camel (ingestDat) is running (wrapper PID 28001)
   EDEX Camel (ingestDat) is running (java PID 28003)
   EDEX Camel (request) is running (wrapper PID 28119)
   EDEX Camel (request) is running (java PID 28121)

   Enter: `ps -ef | grep edex.run.mode | grep -v grep | cut -b -80`

   awips 837 834 12 Apr19 ? 08:35:40 /awips2/java/bin/java -Dedex.run
   awips 17456 17451 7 Apr19 ? 05:13:35 /awips2/java/bin/java -Dedex.run
   awips 25828 25826 7 Apr20 ? 1-16:54:51 /awips2/java/bin/java -Dedex.run
   awips 31216 30171 5 Apr20 ? 02:37:31 /awips2/java/bin/java -Dedex.run

e. Verify EDEX ingest process data flow.
Enter: `cd /awips2/edex/logs`

Enter: `tail -f edex-ingest-`date +%Y%m%d`.log`

**Note:** Look for a continuous display of messages containing “processed in:”; press `<Ctrl-C>` to terminate tail.

```
INFO  2013-02-13 00:00:23,532 [genericThreadPool-53] Ingest: EDEX: Ingest - obs:: /data_store/metar/20130213/00/SAXX60_KWBC_130000_289198456.2013021300 processed in: 0.2420 (sec) Latency: 0.2520 (sec)
INFO  2013-02-13 00:00:23,950 [gfeNotifyThreadPool-1] BasicMessageConsumer: Closing consumer:1[29252995]
INFO  2013-02-13 00:00:23,952 [gfeNotifyThreadPool-1] AMQSession: Dispatcher is not null
INFO  2013-02-13 00:00:23,952 [gfeNotifyThreadPool-1] AMQSession: Rejecting messages from _queue for Consumer tag(1) (PDispatchQ) requeue:true
INFO  2013-02-13 00:00:23,952 [gfeNotifyThreadPool-1] AMQSession: No messages in _queue to reject
INFO  2013-02-13 00:00:23,952 [gfeNotifyThreadPool-1] AMQSession: Closing session: org.apache.qpid.client.AMQSession_0_10@e2db96
INFO  2013-02-13 00:00:23,952 [Dispatcher-Channel-1] Dispatcher: Dispatcher-Channel-1 thread terminating for channel
1:org.apache.qpid.client.AMQSession_0_10@e2db96
INFO  2013-02-13 00:00:23,952 [gfeNotifyThreadPool-1] AMQConnection: Connection:amqp://guest:********@gfeNotify/edex?brokerlist='tcp://cp1f:5672?connectdelay='5000'&connecttimeout='5000'&retries='9999'&sync_ack='true'&sync_publish='all'&maxprefetch='0'
INFO  2013-02-13 00:00:23,953 [gfeNotifyThreadPool-1] AMQProtocolSession: Close was successful.
INFO  2013-02-13 00:00:23,953 [gfeNotifyThreadPool-1] AMQProtocolSession: Closing ProtocolSession for Session:0-10
INFO  2013-02-13 00:00:23,953 [gfeNotifyThreadPool-1] AMQProtocolSession: Session closed successfully.
INFO  2013-02-13 00:00:23,953 [gfeNotifyThreadPool-1] AMQProtocolSession: Closing ProtocolSession for Session:0-10
INFO  2013-02-13 00:00:23,953 [gfeNotifyThreadPool-1] AMQProtocolSession: Session closed successfully.
INFO  2013-02-13 00:00:23,953 [genericThreadPool-53] MetarToShefTransformer: Metar to SHEF now use config file: metar.cfg with options:-a -b -p1 -p6 -p24 -w -strip -q1
INFO  2013-02-13 00:00:23,252 [genericThreadPool-70] Ingest: EDEX: Ingest - sfcobs:: /data_store/maritime/20130213/00/SMVD15_KWNB_130000_289198501.2013021300 processed in: 0.5240 (sec) Latency: 0.9540 (sec)
```

Enter: `tail -f edex-ingestGrib-`date +%Y%m%d`.log`

**Note:** Look for a continuous display of messages containing “processed in:”; press `<Ctrl-C>` to terminate tail.
f. Log off DX3.
   **Enter:** `exit`

g. Log into DX4 using the `root` login.
   **Enter:** `ssh root@dx4`

h. Determine the available memory on the server.
   **Enter:** `free -g`

```
total    used    free   shared    buffers   cached
Mem:       7       7       0       0       0       2
-/+ buffers/cache:       4       3
Swap:      2       0       1
```

i. Verify the EDEX server software is running.

   **Note:** There are four possible instances of EDEX: ingest, request, ingestGrib, and ingestDat. If the available memory indicated by the `free -g` command is less than 4 GB, ingestDat will not be running.

   **Enter:** `service edex_camel status`
EDEX Camel (ingest) is running (java PID 23995)
EDEX Camel (ingestGrib) is running (wrapper PID 24110)
EDEX Camel (ingestGrib) is running (java PID 24112)
EDEX Camel (ingestDat) is running (wrapper PID 24213)
EDEX Camel (ingestDat) is running (java PID 24215)
EDEX Camel (request) is running (wrapper PID 24333)
EDEX Camel (request) is running (java PID 24336)

Enter: `ps -ef | grep edex.run.mode | grep -v grep | cut -b -80`

```
awips 23995 23992 99 Oct21 ? 1-08:00:31
    /awips2/java/bin/java -Dedex.run
awips 24112 24110 9 Oct21 ? 03:07:58 /awips2/java/bin/java
    -Dedex.run
awips 24215 24213 2 Oct21 ? 00:41:29 /awips2/java/bin/java
    -Dedex.run
awips 24336 24333 1 Oct21 ? 00:32:08 /awips2/java/bin/java
    -Dedex.run
```

j. Verify EDEX ingest process data flow.

Enter: `cd /awips2/edex/logs`

Enter: `tail -f edex-ingest-`date +%Y%m%d`.log`

Note: Look for a continuous display of messages containing “processed in:”; press <Ctrl-C> to terminate tail.

```
INFO 2014-01-16 00:00:24,227 [genericThreadPool-32] TafDao: Discarding duplicate: /taf/2014-01-15_23:41:00.0[2014-01-16_00:00:00.0--2014-01-17_00:00:00.0]/PGWT/null/null/1523412
INFO 2014-01-16 00:00:24,227 [genericThreadPool-32] Ingest: EDEX: Ingest - taf::
    /data_store/forecast/20140116/00/FTPQ31_KWBC_160000_51039307.2014011600 processed in: 0.0070 (sec) Latency: 0.0770 (sec)
INFO 2014-01-16 00:00:24,230 [genericThreadPool-61] Ingest: EDEX: Ingest - sfcobs::
    /data_store/maritime/20140115/22/SNVD01_KWBC_152200_51039304.2014011600 processed in: 0.0100 (sec) Latency: 0.0800 (sec)
INFO 2014-01-16 00:00:24,232 [genericThreadPool-75] Ingest: EDEX: Ingest - obs::
    /data_store/metar/20140116/00/SPXX61_KWBC_160000_51039311.2014011600 processed in: 0.0080 (sec) Latency: 0.0820 (sec)
INFO 2014-01-16 00:00:24,252 [genericThreadPool-74] ACARSDecoder: null - Decoded 37 obs
INFO 2014-01-16 00:00:24,256 [genericThreadPool-75] Ingest: EDEX: Ingest - obs::
    /data_store/metar/20140115/23/SAUK42_KWBC_152350_51039316.2014011600 processed in: 0.0200 (sec) Latency: 0.1060 (sec)
INFO 2014-01-16 00:00:24,263 [genericThreadPool-32] Ingest: EDEX: Ingest - taf::
    /data_store/forecast/20140116/00/FTUS80_KWBC_160000_51039314.2014011600 processed in: 0.0320 (sec) Latency: 0.1130 (sec)
```
Enter: `tail -f edex-ingestGrib-`date +%Y%m%d`.log`

**Note:** Look for a continuous display of messages containing “processed in:”; press `<Ctrl-C>` to terminate tail.

```
INFO 2013-05-20 00:00:57,510 [gribThreadPool-3] Ingest: EDEX: Ingest - grib2:: /data_store/grib2/20130519/23/RUC2/GRID130/2300Z_F010_HGHT-LHDK62_KWBG_192300_23017093.grib2.2013052000 processed in: 0.2460 (sec) Latency: 0.2580 (sec)
INFO 2013-05-20 00:00:57,546 [gribThreadPool-4] Ingest: EDEX: Ingest - grib2:: /data_store/grib2/20130519/23/RUC2/GRID130/2300Z_F010_TMPK-LTDK82_KWBG_192300_23017087.grib2.2013052000 processed in: 0.2870 (sec) Latency: 0.2970 (sec)
INFO 2013-05-20 00:00:57,570 [gribThreadPool-2] Ingest: EDEX: Ingest - grib2:: /data_store/grib2/20130519/23/RUC2/GRID130/2300Z_F010_HGHT-LHDK35_KWBG_192300_23017096.grib2.2013052000 processed in: 0.3120 (sec) Latency: 0.3170 (sec)
INFO 2013-05-20 00:00:57,580 [gribThreadPool-3] Ingest: EDEX: Ingest - grib2:: /data_store/grib2/20130519/23/RUC2/GRID130/2300Z_F010_TMPK-LTDK50_KWBG_192300_23017099.grib2.2013052000 processed in: 0.0610 (sec) Latency: 0.3260 (sec)
INFO 2013-05-20 00:00:57,824 [gribThreadPool-2] Ingest: EDEX: Ingest - grib2:: /data_store/grib2/20130519/23/RUC2/GRID130/2300Z_F010_HGHT-LHDK90_KWBG_192300_23017105.grib2.2013052000 processed in: 0.2070 (sec) Latency: 0.5690 (sec)
INFO 2013-05-20 00:00:57,826 [gribThreadPool-4] Ingest: EDEX: Ingest - grib2:: /data_store/grib2/20130519/23/RUC2/GRID130/2300Z_F010_TMPK-LTDK85_KWBG_192300_23017104.grib2.2013052000 processed in: 0.2340 (sec) Latency: 0.5710 (sec)
INFO 2013-05-20 00:00:57,836 [gribThreadPool-1] Ingest: EDEX: Ingest - grib2:: /data_store/grib2/20130519/23/RUC2/GRID130/2300Z_F010_HGHT-LHDK82_KWBG_192300_23017097.grib2.2013052000 processed in: 0.2770 (sec) Latency: 0.5830 (sec)
```

k. Log off DX4.

**Enter:** `exit`

l. Error results: If the EDEX software is not running, the `ps` command will display nothing. If fewer than three lines of output are produced, one or more EDEX processes are not running.

**NOTE:** The `free --g` command in step c. can be refined to limit the output. Entering `free --g | grep Mem | tr -s [space:] | cut --d ' ' -f 2` will display a single number identifying the memory size in gigabytes.

### 25.5 System Configuration

Most system configuration is handled at system installation.
25.6 **EDEX Process Log and Log Format**

The EDEX processes generate a number of log files as they execute. This section offers an overview of these logs and information on the primary process log. More Recipes on monitoring the process logs are included in Section 25.7, [EDEX System Monitoring](#).

25.6.1 **Process Logs**

EDEX process logs are located in the /awips2/edex/logs directory. This directory will contain several logs; one log is created for each EDEX process, each day. EDEX is normally configured to retain up to 30 log files for each EDEX process.

25.6.2 **Additional Log Files**

In addition to the EDEX process logs, the JVM occasionally generates an error log in the JVM’s run directory. For the EDEX processes, the run directory is /awips2/edex/bin.

This log is created when 1) the JVM crashes, and 2) it is able to generate a log file before the crash. In the event of a JVM crash, this log could hold valuable information about the cause of the crash. Reports on EDEX crashes should include the information in this log.

The JVM error log files are named `hs_err_pidxxxx.log`, where `xxxx` is the PID of the JVM that generated the log.

25.6.3 **Process Log File Naming Convention**

The EDEX process logs are located in the /awips2/edex/logs directory. The logs are named `edex-{process}-{date}.log`.

- `{process}` identifies the EDEX process that creates the log. The EDEX processes, as discussed in section 25.2.2, Operational Considerations, are ingest, ingestGrib, and request.
- `{date}` is the date the log was created; the date format is YYYYMMDD.

**Example 25.6.3-1: Log File Names**

The log created by the EDEX Ingest process on October 22, 2011, is named `edex-ingest-20111022.log`.

25.6.4 **Log Formats**

The EDEX processes use a console appender to capture all logging to a single log file; this includes both formatted log messages and unformatted output such as that produced by writing to Linux’s STDOUT and STDERR. As a result, most entries in an EDEX process log follow a standard format, which, as specified in the log4j configuration, is

```%5p %d [%t] %c{1}: %m%n```

The components of this format definition are:
1. **%-5p** – The “p” specifies that the logging level is to be output. The “%-5” specifies the formatting; it is to be left justified in a five-character field.

2. **%d** – This component specifies that the date/time of the message is to be printed. The format used is YYYYY-MM-DD hh:mm:ss,xxx where xxx is milliseconds.

3. **[%t]** – This component specifies that the thread generating the message is listed. For EDEX, this is generally the camel route’s thread pool; it can usually be ignored for diagnostic purposes.

4. **%c{1}** – This component specifies that the logger handling the message is listed. The “{1}” specifies that only the final component of the logger name is listed. Usually, this is the name of the class that logs the message. The colon is printed immediately following the logger name.

5. **%m%n** – The “%m” specifies that the message from the application is logged; the “%n” specifies that it should be followed by a new-line character.

To summarize, the standard log format is

```
LEVEL DATE-TIME [THREAD] LOGGER: MESSAGE
```

If the log message contains multiple lines, as is the case when a Java stack trace is included, only the first line of the message follows this format. Of the standard elements in a log message, the logging level and the logger can be used with the Linux `tail` and `grep` tools to provide real-time filtering of log messages. In addition, there are some standard message patterns that can be useful in monitoring for specific information/problems in an EDEX instance.

### 25.6.5 EDEX Logging Levels

EDEX logs messages at one of these five severity levels: DEBUG; INFO; WARNING; ERROR; and FATAL. Each message logged by the application is given one of these levels. In addition, the logging facility is configured with a logging level. Only messages that meet or exceed the configured logging level are written to the log file. For example, if the logging facility is configured to a logging level of `WARNING`, only messages with a level of `WARNING` and above (i.e., `WARNING`, `ERROR`, and `FATAL`) are logged.

Out of the box, most EDEX logging is at the INFO level; in this case INFO, WARNING, ERROR, and FATAL messages are logged.

**Example 25.6.5-1: Monitoring EDEX Process Log for Errors**

To monitor for errors in the EDEX Ingest process on DX4, use `tail` to display the EDEX log. To filter for errors, pipe the results of `tail` to `grep`, using the appropriate filter on `grep`. (Note: For this example, assume the current date is October 22, 2011). The steps to follow:

```
$ ssh awips@dx4
awips@dx4’s password:
```
NOTE: Messages shown would appear on a single line in the terminal window.

25.6.6 Controlling Logging Levels

This section will be updated as information becomes available.

25.7 EDEX System Monitoring

This section discusses a few of the options available for monitoring the EDEX processes. The EDEX processes are deployed on DX3 and DX4 (see Exhibit 25.2-1).

System monitoring is somewhat complicated by the general design and deployment of EDEX, as described earlier. Factors to be considered include the following:

1. Each EDEX process is designed around the Service Oriented Architecture (SOA) concept and is, therefore, a collection of services that perform specific tasks or functions in response to events. In the ingest stream discussed in section 25.3, Basic Data Flow Concepts, the event that triggers an action in a Data Decoder service is the arrival of a message in that decoder’s message queue. Other events include time-based (Quartz triggered) events and client requests (http/thrift events). A good way to visualize SOA events is to picture your favorite GUI application. As you manipulate the GUI, events are created that cause the application to perform processing for you. EDEX utilizes the SOA concept to provide somewhat similar, event-driven processing.
2. Because each EDEX process logs primarily to a single log file,¹ some filtering is needed to extract information specific to a service from the combined log file of the EDEX process.

3. EDEX services replace many of the separate processes running on the AWIPS I servers. As noted previously, the result is that messages from all services in each EDEX process are combined in a single log file.

### 25.7.1 Using Standard Linux Tools

Table 25.7.1-1 lists several standard Linux tools that can be used to monitor the EDEX processes. Other useful monitoring tools include the Java Jconsole tool, which is packaged with EDEX. The PG Admin tool and HDF View, which are not packaged with EDEX, may be used to check the database and the data store.

#### Table 25.7.1-1. Useful Tools for Monitoring EDEX

<table>
<thead>
<tr>
<th>Tool</th>
<th>Location</th>
<th>Monitoring Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>ps</td>
<td>/bin</td>
<td>Can be used to get information about specific processes</td>
</tr>
<tr>
<td>cat</td>
<td>/bin</td>
<td>Used to display a text file in a terminal</td>
</tr>
<tr>
<td>tail</td>
<td>/usr/bin</td>
<td>Used to provide a dynamic picture of process logs</td>
</tr>
<tr>
<td>grep</td>
<td>/bin</td>
<td>Used to filter content of process logs; used to filter output of other tools</td>
</tr>
<tr>
<td>top</td>
<td>/usr/bin</td>
<td>Provides a dynamic view of the memory and cpu usage of the EDEX processes</td>
</tr>
<tr>
<td>tee</td>
<td>/usr/bin</td>
<td>Sends command line output to a file</td>
</tr>
<tr>
<td>less</td>
<td>/usr/bin</td>
<td>Used to give a page-oriented view of files; includes search capabilities</td>
</tr>
<tr>
<td>edex_camel</td>
<td>/etc/init.d</td>
<td>Determines the PIDs of the EDEX processes</td>
</tr>
<tr>
<td>JConsole</td>
<td>/awips2/java/bin</td>
<td>Monitors the internal state of an EDEX process</td>
</tr>
</tbody>
</table>

The rest of this section is a series of Examples that illustrate various monitoring activities for EDEX processes. All of these examples make the following simple assumptions:

1. EDEX is installed on DX3. For other servers, the initial login should be modified as appropriate.

2. EDEX is installed under /awips2. Supporting software such as Java and Python is also installed under /awips2.

3. The administrator is logging into the server using ssh from a Linux workstation.

#### Example 25.7.1-1: Determining the EDEX Version

The installed EDEX version is listed in a file named `banner.txt`, which is located in `/awips2/edex/conf`. The steps to follow:

```
$ ssh awips@dx3
awips@dx3’s password:
$ cd /awips2/edex/conf
$ cat banner.txt
```

¹ The basic logging should stay the same, although some of the messages being logged may change.
**Example 25.7.1-2: Determining EDEX Status**

The status of the EDEX processes may be determined using the `status` option of the `edex_camel` script. This option is also helpful for determining the PID of each EDEX process. Unlike other options involving `edex_camel`, the `status` option is usable by only root user account on the server. The steps to follow:

```
$ ssh root@dx3
root@dx3’s password:
$ cd /etc/init.d
$ ./edex_camel status
EDEX Camel (ingest) is running (wrapper PID 24173)
EDEX Camel (ingest) is running (java PID 24175)
EDEX Camel (ingestGrib) is running (wrapper PID 24272)
EDEX Camel (ingestGrib) is running (java PID 24274)
EDEX Camel (ingestDat) is running (wrapper PID 17309)
EDEX Camel (ingestDat) is running (java PID 17314)
EDEX Camel (request) is running (wrapper PID 24497)
EDEX Camel (request) is running (java PID 14130)
```

This output indicates that three EDEX processes are running. Note that the reported PIDs will depend on the actual PIDs on the server being monitored.

**TIP:** To determine the status of a specific EDEX process, specify the process name on the command line. For example: `./edex_camel status request`.

**TIP:** If one of the EDEX processes is not running, the status reported will be “EDEX Camel (process) is not running” where process is the name of the process that is not running. (See section 25.2.2, Operational Considerations for a list of EDEX processes.)
Example 25.7.1-3: Determining EDEX Process Restarts

Each time wrapper starts a JVM running an EDEX process, it prints *Launching a JVM...* in the process log. This provides an easy means of determining the number of times an EDEX process has been started. The steps to follow:

```
$ ssh awips@dx3
awips@dx3’s password:
$ cd /awips2/edex/logs
$ ls edex*-20111023.log
edex-ingest-20111023.log
edex-ingestDat-20111023.log
edex-ingest-gen_areal_ffg-20111023.log
edex-ingest-gen_areal_gpe-20111023.log
edex-ingestGrib-20111023.log
edex-ingest-purge-20111023.log
edex-ingest-radar-20111023.log
edex-ingest-satellite-20111023.log
edex-ingest-shef-20111023.log
edex-ingest-shef-performance-20111023.log
edex-ingest-smartInit-20111023.log
edex-ingest-text-20111023.log
edex-ingest-unrecognized-files-20111023.log
edex-request-20111023.log
edex-request-productSrvRequest-20111023.log
edex-ingest-archive-20111023.log
edex-ingest-activeTableChange-20111023.log
edex-ingestDat-performance-20111023.log
edex-ingestGrib-performance-20111023.log
edex-ingest-performance-20111023.log
edex-request-performance-20111023.log
edex-ingest-gen_areal_qpe-20111023.log
start-edex-ingest-20111023.log
start-edex-ingestDat-20111023.log
start-edex-ingestGrib-20111023.log
start-edex-request-20111023.log
edex-request-hibernate-20111023.log
edex-ingest-hibernate-20111023.log
edex-ingestGrib-hibernate-20111023.log
```

```
$ grep "Launching" edex-ingest-20111023.log
edex-ingest-20111023.log:Launching a JVM...
edex-ingest-20111023.log:Launching a JVM...
```

This output shows that the EDEX *ingest* process was started twice on Oct 23, 2011. In this case the EDEX installation was also performed on Oct 23, so we know *wrapper* restarted the *ingest* process once – this indicates a process crash that should be investigated.

**TIP:** To determine the starts logged in each process log, use the `-c` (count) option on `grep`. 
NOTE: The names of the available log files depend on when EDEX was first started and the length of time the EDEX processes have actually been running on the server.

Example 25.7.1-4: Determining Available Data Types from Ingest

This Example shows how to monitor ingest of a specific data type on one of the EDEX servers. This Example obtains a list of data types being ingested. The list can then be used to selectively monitor ingest of a specific data type. The steps to follow:

```bash
$ ssh awips@dx3
awips@dx3’s password:
$ cd /awips2/edex/logs

Note: For today’s log file, for example use ls edex-*-20111023.log
```

```
edex-ingest-20111023.log
edex-ingestDat-20111023.log
edex-ingest-gen_areal_ffg-20111023.log
edex-ingest-gen_areal_qpe-20111023.log
edex-ingestGrib-20111023.log
edex-ingest-purge-20111023.log
edex-ingest-radar-20111023.log
edex-ingest-satellite-20111023.log
edex-ingest-shef-20111023.log
edex-ingest-shef-performance-20111023.log
edex-ingestGrib-20111023.log
edex-ingestGrib.performance-20111023.log
```

```
$ grep -Po "Ingest: .+::" edex-ingest-20111023.log > ingest-types.txt
```

```
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - sfcobs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - sfcobs::
```
Ingest: EDEX: Ingest - sfcobs::
Ingest: EDEX: Ingest - sfcobs::
Ingest: EDEX: Ingest - warning::
Ingest: EDEX: Ingest - ldadhydro::
Ingest: EDEX: Ingest - ldadmesonet::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - acars::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - acars::
Ingest: EDEX: Ingest - redbook::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - acars::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - sfcobs::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - binlightning::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - redbook::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - sfcobs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - bufrascat::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - taf::
Ingest: EDEX: Ingest - taf::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - acars::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - sfcobs::
Ingest: EDEX: Ingest - sfcobs::
Ingest: EDEX: Ingest - taf::
Ingest: EDEX: Ingest - taf::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - warning::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - sfcobs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - sfcobs::
Ingest: EDEX: Ingest - sfcobs::
Ingest: EDEX: Ingest - sfcobs::
Ingest: EDEX: Ingest - sfcobs::
Ingest: EDEX: Ingest - sfcobs::
Ingest: EDEX: Ingest - sfcobs::
Ingest: EDEX: Ingest - warning::
Ingest: EDEX: Ingest - ldadhydro::
Ingest: EDEX: Ingest - ldadmesonet::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - acars::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - obs::
Ingest: EDEX: Ingest - bufrhdw::
Ingest: EDEX: Ingest - sfcobs::

This output shows a sorted list of data types being processed on this server.

**TIP:** The -P option on grep allows you to use a Perl 5 regular expression as your search pattern. The -o option limits the output to only the portion of the line that matches the pattern.

**TIP:** This recipe uses the `sort` and `uniq` tools to help identify/filter information. See the man pages for details.

**Example 25.7.1-5: Monitoring Data Ingest**

Each time a data file is successfully ingested, the EDEX Ingest process logs a message. The general format of the message is `INFO <date> <time> [thread] Ingest: type:: <message>.` (Note that there are two spaces between INFO and the start of the date stamp.) The strategy is to tail the active log file and use grep to limit the output to show successful data ingest messages. This recipe presents two scenarios for monitoring data ingest: 1) monitoring all data ingest; and 2) monitoring ingest of a specific type of data.

**Scenario 1: Monitoring General Data Ingest**

For general data ingest, we can ignore the `type` token and simply look for INFO messages from ingest. (A Perl 5 regex that matches is `^INFO .+ Ingest: .`) The steps to follow:

```
$ ssh awips@dx3
awips@dx3’s password:
```

Example 25.7.1-5: Monitoring Data Ingest

Each time a data file is successfully ingested, the EDEX Ingest process logs a message. The general format of the message is `INFO <date> <time> [thread] Ingest: type:: <message>`. (Note that there are two spaces between INFO and the start of the date stamp.) The strategy is to tail the active log file and use grep to limit the output to show successful data ingest messages. This recipe presents two scenarios for monitoring data ingest: 1) monitoring all data ingest; and 2) monitoring ingest of a specific type of data.

**Scenario 1: Monitoring General Data Ingest**

For general data ingest, we can ignore the `type` token and simply look for INFO messages from ingest. (A Perl 5 regex that matches is `^INFO .+ Ingest: .`) The steps to follow:

```
$ ssh awips@dx3
awips@dx3’s password:
```
$ cd /awips2/edex/logs

$ ls edex-*-20111023.log

edex-ingest-20111023.log
edex-ingestDat-20111023.log
edex-ingest-gen_area0_gpe-20111023.log
edex-ingestGrib-20111023.log
edex-ingest-purge-20111023.log
edex-ingest-radar-20111023.log
edex-ingest-satellite-20111023.log
edex-ingest-shef-20111023.log
edex-ingest-shef-performance-20111023.log
edex-ingest-smartInit-20111023.log
edex-ingest-text-20111023.log
edex-ingest-unrecognized-files-20111023.log
edex-request-20111023.log
edex-request-productSrvRequest-20111023.log
edex-ingest-archive-20111023.log
edex-ingest-activeTableChange-20111023.log
edex-ingestDat-performance-20111023.log
edex-ingestGrib-performance-20111023.log
edex-ingest-performance-20111023.log
edex-request-performance-20111023.log
edex-ingest-gen_area0_gpe-20111023.log
start-edex-ingest-20111023.log
start-edex-ingestDat-20111023.log
start-edex-ingestGrib-20111023.log
start-edex-request-20111023.log
edex-request-hibernate-20111023.log
edex-ingest-hibernate-20111023.log
edex-ingestGrib-hibernate-20111023.log

$ tail -f edex-ingest-<YYYYMMDD.log> | grep -P "^INFO .+ Ingest:"

INFO  2014-01-15 01:38:36,860 [genericThreadPool-74] Ingest: EDEX: Ingest - acars:: /data_store/acars/acars_raw_decrypted/20140115/01/IUAD01_EGRR_150139_48804923.bufr.2014011501 processed in: 0.0070 (sec) Latency: 0.0770 (sec)
INFO  2014-01-15 01:38:36,893 [genericThreadPool-75] Ingest: EDEX: Ingest - binlightning:: /data_store/binlightning/20140115/01/SFUS41_KWBC_150137_48804936.nldn.2014011501 processed in: 0.0320 (sec) Latency: 0.1100 (sec)
INFO  2014-01-15 01:38:36,905 [genericThreadPool-100] RedbookDecoder: Storing new version of /redbook/2014-01-15_00:00:00.0/PYMA50/PE50000NAPLTUPA/null/null/0/72
Time spent in persist: 87
To stop monitoring, press <ctrl-c>.

Scenario 2: Monitoring Specific Data Ingest

To monitor a specific data ingest, it is necessary to include the type token in the grep.

(A Perl 5 regex that matches is ^INFO .+ Ingest: type::. Replace type with the name of the data type to monitor.) For this example, we will monitor radar ingest. The steps to follow:

$ ssh awips@dx3
awips@dx3’s password:
$ cd /awips2/edex/logs
$ ls edex-*-20130213.log
$ tail -f edex-ingest-20130213.log | grep -P "^INFO .+ Ingest: pirep::"
INFO 2013-02-13 00:05:14,693 [genericThreadPool-55] Ingest: EDEX: Ingest - pirep:: /data_store/pirep/20130213/00/UBOR90_KWBC_130005_289206471.201302130 0 processed in: 0.0060 (sec) Latency: 0.0530 (sec)


INFO 2013-02-13 00:05:14,699 [genericThreadPool-55] Ingest: EDEX: Ingest - pirep:: /data_store/pirep/20130213/00/UBAR90_KWBC_130005_289206480.201302130 0 processed in: 0.0040 (sec) Latency: 0.0560 (sec)

INFO 2013-02-13 00:05:14,703 [genericThreadPool-55] PirepDao: Discarding duplicate: /pirep/2013-02-12_23:56:00.0/4700/null/null/43.53911986588092/-104.92787018891087/40000

INFO 2013-02-13 00:05:14,703 [genericThreadPool-55] Ingest: EDEX: Ingest - pirep:: /data_store/pirep/20130213/00/UBBW90_KWBC_130005_289206487.201302130 0 processed in: 0.0030 (sec) Latency: 0.0580 (sec)

Time spent in persist: 105

(as EDEX Ingest writes to the log file, lines identifying successful text ingest are displayed.)

To stop monitoring, press <ctrl-c>.

**NOTE:** In both scenarios, the Linux *cut* utility can be used to limit the displayed output. For example, to eliminate the date stamp from the output, pipe the tail/grep command from either scenario into *cut –d ' ' –f 1,4,6-15*. (This also eliminates the [thread] from the output but includes the data type and file name.)

**Example 25.7.1-6: Monitoring Data Ingest Latency**

Latency is the elapsed time between when the AWIPS II system first contacts a data file and when the data has been ingested, decoded, and is ready for retrieval by either CAVE or another client application. This information is logged into the EDEX process logs for every file that is ingested. The general format of the message is **INFO <date> <time> [thread] Ingest: <type>:: <file> processed in: <time> Latency: <time>**. (Note that there are two spaces between INFO and the start of the date stamp.) This pattern can be used to monitor the EDEX Ingest process logs for ingest latency, using the pattern to filter out undesirable messages.

This recipe presents two scenarios: 1) general latency monitoring; and 2) monitoring latency for a specific data type. Both scenarios use the Linux *tail* and *grep* tools to provide real-time display of the log files while filtering to eliminate extra output. They also use the Linux *cut* tool to eliminate some extra information from the lines being displayed.
Scenario 1: Monitoring General Data Latency

For general data latency, we can ignore the type token and simply capture the INFO messages from Ingest. To simplify the output, we will use cut to eliminate the date and time stamps, the thread identifier, the data type, and the file name from the output.

The Perl 5 regex:  ^INFO .+ Ingest:

The cut command: cut -d ' ' -f 1,4,6,9-

The steps to follow:

$ ssh awips@dx3
awips@dx3's password:
$ cd /awips2/edex/logs
$ ls edex-*-20111023.log

edex-ingest-20111023.log
edex-ingestDat-20111023.log
edex-ingest-gen_areal_ffg-20111023.log
edex-ingest-gen_areal_qpe-20111023.log
edex-ingestGrib-20111023.log
edex-ingest-purge-20111023.log
edex-ingest-radar-20111023.log
edex-ingest-satellite-20111023.log
edex-ingest-shef-20111023.log
edex-ingest-shef-performance-20111023.log
edex-ingest-smartInit-20111023.log
edex-ingest-text-20111023.log
edex-ingest-unrecognized-files-20111023.log
edex-request-20111023.log
edex-request-productSrvRequest-20111023.log
edex-ingest-archive-20111023.log
edex-ingest-activeTableChange-20111023.log
edex-ingestDat-performance-20111023.log
edex-ingestGrib-performance-20111023.log
edex-ingest-performance-20111023.log
edex-request-performance-20111023.log
edex-ingest-gen_areal_qpe-20111023.log
start-edex-ingest-20111023.log
start-edex-ingestDat-20111023.log
start-edex-ingestGrib-20111023.log
start-edex-request-20111023.log
edex-request-hibernate-20111023.log
edex-ingest-hibernate-20111023.log
edex-ingestGrib-hibernate-20111023.log

$ tail -f edex-ingest-20111023.log | grep -P "^INFO .+ Ingest:" | cut -d ' ' -f 1,4,6,9-

(as EDEX Ingest writes to the log file, lines identifying successful ingest with processing and latency times are displayed.)

To stop monitoring, press <ctrl-c>. 
Scenario 2: Monitoring Data Latency for a Specific Data Type

To monitor a specific data type, the type is important. Other than that, the process is nearly identical to scenario 1. For this example, we will monitor radar data.

The Perl 5 regex: ^INFO .+ Ingest: radar::

The cut command: cut -d '' -f 1,4-6,9-15

The steps to follow:

$ ssh awips@dx3
awips@dx3’s password:
$ cd /awips2/edex/logs
$ ls edex-*.

Note: For today’s log file, for example, use ls edex-*-20111023.log

edex-ingest-20111023.log
edex-ingestDat-20111023.log
edex-ingest-gen_areal_ffg-20111023.log
edex-ingest-gen_areal_gpe-20111023.log
edex-ingestGrib-20111023.log
edex-ingest-purge-20111023.log
edex-ingest-radar-20111023.log
edex-ingest-satellite-20111023.log
edex-ingest-shef-20111023.log
edex-ingest-shef-performance-20111023.log
edex-ingest-smartInit-20111023.log
edex-ingest-text-20111023.log
edex-ingest-unrecognized-files-20111023.log
edex-request-20111023.log
edex-request-productSrvRequest-20111023.log
edex-ingest-archive-20111023.log
edex-ingest-activeTableChange-20111023.log
edex-ingestDat-performance-20111023.log
edex-ingestGrib-performance-20111023.log
edex-ingest-performance-20111023.log
edex-request-performance-20111023.log
edex-ingest-gen_areal_gpe-20111023.log
start-edex-ingest-20111023.log
start-edex-ingestDat-20111023.log
start-edex-ingestGrib-20111023.log
start-edex-request-20111023.log
edex-request-hibernate-20111023.log
edex-ingest-hibernate-20111023.log
edex-ingestGrib-hibernate-20111023.log

$ tail -f edex-ingest-radar-20111023.log | grep -P "^INFO .+ Ingest: radar::" | cut -d ' ' -f 1,4,6,7,9-
To stop monitoring, press <ctrl-c>.

Typical messages will be similar to this:

```
INFO 2011-10-23 00:00:02,132 [radarThreadPool-1] Ingest: EDEX: Ingest - radar::
data_store/radar/BWI/TV0/SDUS51_BWI_TV0_222358_461873052.rad
processed in: 0.0090 (sec) Latency: 0.9870 (sec)
```

```
INFO 2011-10-23 00:00:02,305 [radarThreadPool-1] Ingest: EDEX: Ingest - radar::
data_store/radar/KEAX/N3Q/SDUS23_2351_KEAX_N3Q_461873056.rad
processed in: 0.0670 (sec) Latency: 1.1600 (sec)
```

**NOTE:** You can expect the latency time to vary somewhat during the day depending on the rate at which data is arriving at the particular time of day. Extended periods of high latency are an indication of general system problems.

**TIP:** To monitor other data types, change `radar` to the name of the data type. See Example 25.7.1-4 to determine available data types.

**Example 25.7.1-7: Monitoring Ingest Failures**

Unfortunately, as of this writing, there is no standard message logged when an ingest process fails. The closest available is a message with the general format of `ERROR <date> <time> [thread] Logger:< message>`. (A Perl 5 regex that matches this pattern is `ERROR.* Logger:`.) This pattern has two problems. First, it does not name the file being ingested when the error occurred, and second, it is not exclusive to file ingest. Because all ingest is handled by the EDEX ingest or GRIB ingest process, the second problem is less of an issue. The steps to follow:

```
$ ssh awips@dx3
awips@dx3’s password:
$ cd /awips2/edex/logs
$ ls edex-*-20111023.log
```

**Note:** For today’s log file, for example use `ls edex-*-20111023.log`

```
edex-ingest-20111023.log
edex-ingestDat-20111023.log
edex-ingest-gen_areal_ffg-20111023.log
edex-ingest-gen_areal_qpe-20111023.log
edex-ingestGrib-20111023.log
edex-ingest-purge-20111023.log
edex-ingest-radar-20111023.log
edex-ingest-satellite-20111023.log
edex-ingest-shef-20111023.log
edex-ingest-shef-performance-20111023.log
edex-ingest-smartInit-20111023.log
edex-ingest-text-20111023.log
edex-ingest-unrecognized-files-20111023.log
edex-request-20111023.log
```
edex-request-productSrvRequest-20111023.log
edex-ingest-archive-20111023.log
edex-ingest-activeTableChange-20111023.log
edex-ingestDat-performance-20111023.log
edex-ingestGrib-performance-20111023.log
edex-ingest-performance-20111023.log
edex-request-performance-20111023.log
edex-ingest-gen_areal_gpe-20111023.log
start-edex-ingest-20111023.log
start-edex-ingestDat-20111023.log
start-edex-ingestGrib-20111023.log
start-edex-request-20111023.log
edex-request-hibernate-20111023.log
edex-ingest-hibernate-20111023.log
edex-ingestGrib-hibernate-20111023.log

$ tail -f edex-ingest-20111023.log | grep -P "^ERROR .+ Logger:"

(as EDEX Ingest writes errors to the log file they are displayed.)

To stop monitoring, press <ctrl-c>.

Example 25.7.1-8: Verifying EDEX Ready to Execute

When an EDEX process is started, it performs a series of initial operations before it is operational, that is, before it is ready to process data or handle client requests. EDEX logs three distinct messages during the startup process:

- *Launching a JVM... is logged as wrapper starts the EDEX process:*
- *AWIPS II EDEX ESB Platform* is logged as the EDEX process starts.
- *EDEX ESB is now operational* is logged once the process is operational.

`grep` can be used to search the log at startup to determine if the EDEX process is operational. The steps to follow:

```
$ ssh awips@dx3
awips@dx3’s password:
$ cd /awips2/edex/logs
$ ls edex*-20111023.log
```

edex-ingest-20111023.log
edex-ingestDat-20111023.log
edex-ingest-gen_areal_ffg-20111023.log
edex-ingest-gen_areal_qpe-20111023.log
edex-ingestGrib-20111023.log
edex-ingest-purge-20111023.log
edex-ingest-radar-20111023.log
edex-ingest-satellite-20111023.log
edex-ingest-shef-20111023.log
edex-ingest-shef-performance-20111023.log
edex-ingest-smartInit-20111023.log
edex-ingest-text-20111023.log
edex-ingest-unrecognized-files-20111023.log
edex-request-20111023.log
edex-request-productSrvRequest-20111023.log
edex-ingest-archive-20111023.log
edex-ingest-activeTableChange-20111023.log
edex-ingestDat-performance-20111023.log
edex-ingestGrib-performance-20111023.log
edex-ingest-performance-20111023.log
edex-request-performance-20111023.log
edex-ingest-gen_areal_qpe-20111023.log
start-edex-ingest-20111023.log
start-edex-ingestDat-20111023.log
start-edex-ingestGrib-20111023.log
start-edex-request-20111023.log
edex-request-hibernate-20111023.log
edex-ingest-hibernate-20111023.log
edex-ingestGrib-hibernate-20111023.log
$ grep -Pw "Launching|Platform|operational" edex-ingest-20111023.log

Launching a JVM...
* AWIPS II EDEX ESB Platform
* EDEX ESB is now operational

Launching a JVM...
* AWIPS II EDEX ESB Platform
* EDEX ESB is now operational

This output shows that the EDEX ingest process started and became operational twice on October 23, 2011. Assuming EDEX ingest has not been stopped, it also shows that the EDEX ingest process is ready to process data.

**TIP:** The `-P` option on `grep` allows you to use a *Perl 5* regular expression as the search pattern. The `-w` option instructs `grep` to search for whole word matches only.

### Example 25.7.1-9: Monitoring General EDEX Server Status

The Linux `top` utility can be used to monitor the general status of a server. `top` is designed to provide dynamic real-time view of the system. It provides a system summary and a list of executing tasks on the server.

The basic steps for running `top`:

```bash
$ ssh awips@dx3
awips@dx3’s password:
$ top
```

To exit `top`, either press `q` or `<ctrl-c>`. Typical output is shown in Exhibit 25.7.1-1.

Interesting values to watch are in the PID, VIRT, RES, SHR, %CPU, %MEM and TIME columns. From the top main page:

- **PID:** Process Id.
• **VIRT**: Virtual Image (kb). The total amount of virtual memory used by the task. Includes code, data, and shared libraries plus pages that have been swapped out.

• **RES**: Resident size (kb). The non-swapped physical memory a task has used.

• **SHR**: Shared Mem size (kb). The amount of shared memory used by a task. It simply reflects memory that could be potentially shared with other processes.

• **%CPU**: CPU usage. The task’s share of the elapsed CPU time since the last screen update, expressed as a percentage of total CPU time.

• **%MEM**: Memory usage (RES). A task’s currently used share of available physical memory.

• **TIME**: CPU Time. Total CPU time the task has used since it started. When ‘Cumulative mode’ is “On,” each process is listed with the cpu time that it and its dead children have used.

---

```text
Exhibit 25.7.1-1. Basic top Display
```

```text
Exhibit 25.7.1-1. Basic top Display
```

**top - 03:25:10 up 73 days, 10:24,  1 user,  load average: 2.84, 2.78, 2.77

Tasks: 213 total,  1 running, 212 sleeping,  0 stopped,  0 zombie
Cpu(s):  6.8%us,  2.5%sy,  0.0%ni,  86.9%id,  0.5%wa,  0.4%hi,  2.9%si, 0.0%st
Mem:   830672k total, 7517240k used,   789432k free,    18520k buffers
Swap:  2199544k total,     6920k used, 2192624k free,  1608884k cached

PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+ COMMAND
24175 awips     22   0 1666m 1.3g  17m S 72.8 16.6   5315:37 java
17314 awips     20   0 1683m 918m  14m S 14.3 11.3 380:07.88 java
22469 root      15   0  2632 1264  924 R  0.3  0.0   0:00.11 top
  1 root       15  0 2160 540 512 S  0.0  0.0   0:12.62 init
   2 root      RT -5   0   0   0 S  0.0  0.0   1:28.10 migration/0
   3 root      34 19   0   0   0 S  0.0  0.0   0:18.69 ksoftird/0
   4 root      RT -5   0   0   0 S  0.0  0.0   0:00.00 watchdog/0
   5 root      RT -5   0   0   0 S  0.0  0.0   0:05.76 migration/1
   6 root      34 19   0   0   0 S  0.0  0.0   0:00.87 ksoftird/1
   7 root      RT -5   0   0   0 S  0.0  0.0   0:00.00 watchdog/1
   8 root      RT -5   0   0   0 S  0.0  0.0   0:07.58 migration/2
   9 root      34 19   0   0   0 S  0.0  0.0   0:00.35 ksoftird/2
  10 root      RT -5   0   0   0 S  0.0  0.0   0:00.00 watchdog/2
  11 root      RT -5   0   0   0 S  0.0  0.0   0:06.77 migration/3
  12 root      39 19   0   0   0 S  0.0  0.0   0:02.91 ksoftird/3
  13 root      RT -5   0   0   0 S  0.0  0.0   0:00.00 watchdog/3
  14 root      RT -5   0   0   0 S  0.0  0.0   0:08.81 migration/4
  15 root      34 19   0   0   0 S  0.0  0.0   0:00.34 ksoftird/4
  16 root      RT -5   0   0   0 S  0.0  0.0   0:00.00 watchdog/4
  17 root      RT -5   0   0   0 S  0.0  0.0   0:05.04 migration/5
  18 root      34 19   0   0   0 S  0.0  0.0   0:01.11 ksoftird/5
  19 root      RT -5   0   0   0 S  0.0  0.0   0:00.00 watchdog/5
  20 root      RT -5   0   0   0 S  0.0  0.0   0:08.65 migration/6
  21 root      34 19   0   0   0 S  0.0  0.0   0:00.36 ksoftird/6
  22 root      RT -5   0   0   0 S  0.0  0.0   0:00.00 watchdog/6
  23 root      RT -5   0   0   0 S  0.0  0.0   0:04.72 migration/7
  24 root      34 19   0   0   0 S  0.0  0.0   0:01.04 ksoftird/7
```
Example 25.7.1-10: Monitoring Just the EDEX Processes

`Top` supports several command line options; one of the most useful is the ability to select the processes to monitor. This is done using the `-p` flag. The basic syntax is

```
top -p PID1[, PID2[, PID3 ...]]
```

This can be used to have `top` monitor just the EDEX processes. The steps to follow:

```
$ ssh root@dx3
root@dx3’s password:
$ /etc/init.d/edex_camel status | grep java
EDEX Camel (ingest) is running (java PID 24175)
EDEX Camel (ingestGrib) is running (java PID 24274)
EDEX Camel (ingestDat) is running (java PID 17314)
EDEX Camel (request) is running (java PID 14130)
$ top -p 24175, 24274, 17314, 14130
```

To exit `top`, press either `q` or `<ctrl-c>`. Typical output is shown in Exhibit 25.7.1-2.

<table>
<thead>
<tr>
<th>PID</th>
<th>USER</th>
<th>PR</th>
<th>NI</th>
<th>VIRT</th>
<th>RES</th>
<th>SHR</th>
<th>S</th>
<th>%CPU</th>
<th>%MEM</th>
<th>TIME+</th>
<th>COMMAND</th>
</tr>
</thead>
<tbody>
<tr>
<td>24175</td>
<td>awips</td>
<td>22</td>
<td>0</td>
<td>1677m</td>
<td>1.3g</td>
<td>17m</td>
<td>S</td>
<td>117.6</td>
<td>16.6</td>
<td>5323:16</td>
<td>java</td>
</tr>
<tr>
<td>24274</td>
<td>awips</td>
<td>22</td>
<td>0</td>
<td>1529m</td>
<td>1.3g</td>
<td>17m</td>
<td>S</td>
<td>6.0</td>
<td>16.7</td>
<td>458:16.28</td>
<td>java</td>
</tr>
<tr>
<td>14130</td>
<td>awips</td>
<td>17</td>
<td>0</td>
<td>2718m</td>
<td>1.8g</td>
<td>17m</td>
<td>S</td>
<td>4.0</td>
<td>22.9</td>
<td>73:18.21</td>
<td>java</td>
</tr>
<tr>
<td>17314</td>
<td>awips</td>
<td>20</td>
<td>0</td>
<td>1683m</td>
<td>918m</td>
<td>14m</td>
<td>S</td>
<td>0.0</td>
<td>11.3</td>
<td>381:46.94</td>
<td>java</td>
</tr>
</tbody>
</table>

Exhibit 25.7.1-2. Top Display with Selected Processes

25.7.2 Using the JConsole Tool

The JConsole tool has been part of the standard Java release since Java 1.5. It provides a GUI interface into an executing Java program and is a basic tool for monitoring the EDEX Processes. JConsole utilizes the Java Management Extension (JMX) to provide a low-impact view into the internals of a Java application.

EDEX has been configured to support JConsole connectivity options that allow both local and remote monitor monitoring.

In this context, “local” means logged onto the server as the user, normally `awips`, running EDEX. This also means that we must first identify the PID of the AWIPS Process we want to monitor. Finally, because JConsole is a GUI based application, you will need to export display from the EDEX server being monitored to the workstation. If you log into the server using `ssh`, the `-X` flag may be used to export display.
Remote monitoring allows a user to monitor the EDEX without logging into the server running EDEX. In order to do so you need the hostname of the server and the port the EDEX instance exposes for JConsole monitoring. The ports currently exposed by EDEX are listed in Table 25.7.2-1.

### Table 25.7.2-1. EDEX JConsole Ports

<table>
<thead>
<tr>
<th>Process</th>
<th>Token</th>
<th>JConsole Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>EDEX Request</td>
<td>request</td>
<td>1616</td>
</tr>
<tr>
<td>EDEX Ingest</td>
<td>ingest</td>
<td>1617</td>
</tr>
<tr>
<td>EDEX GRIB Ingest</td>
<td>ingestGrib</td>
<td>1618</td>
</tr>
<tr>
<td>EDEX DAT Ingest</td>
<td>ingestDAT</td>
<td>1619</td>
</tr>
</tbody>
</table>

**NOTE:** Remote monitoring requires that Java 1.5 or later be installed on the computer running JConsole.

The next few recipes discuss using JConsole to interact with the running EDEX processes.

**NOTE:** When possible, remote monitoring should be used for monitoring; this results in a lower processing load on the EDEX server.

### Example 25.7.2-1: Using JConsole to Monitor an EDEX Process

This recipe presents the basics for using JConsole. There are two scenarios: local monitoring of the EDEX process; and remote monitoring of the EDEX process. In the next few recipes, it will be assumed that the steps provided in this recipe are used to start JConsole.

**Scenario 1: Using JConsole in Local Monitoring Mode**

For local monitoring, you need to be logged into the EDEX server using the same account that is running EDEX. Normally, this is the `awips` account.

The steps to follow to initiate a JConsole connection to an operating EDEX ingest process:

```
$ ssh -X awips@dx3
awips@dx3’s password:
$ /etc/init.d/edex_camel status ingest | grep java
EDEX Camel (ingest) is running (java PID 13603)
$ /awips2/java/bin/jconsole 13603 &
```

For the other EDEX processes, replace `ingest` with the token for the desired EDEX process. (See section 25.2.2, [Operational Considerations](#) for a list of EDEX processes.)
Scenario 2: Using JConsole in Remote Monitoring Mode

For remote monitoring you will need two items – the host name of the EDEX server and the connection port. The command line format for remote connection is:

```
jconsole host:port.
```

In this recipe, we will connect to the EDEX Ingest process on DX3. The port used by the EDEX request process is 1616, see section 25.7.2, Using the JConsole Tool (Table 25.7.2-1) for a list of ports for the other EDEX processes.

**NOTE:** This scenario assumes that Java 1.5 or better has been installed on the workstation and that the JConsole executable is in the path.

The steps to follow to initiate a remote JConsole connection to an operating EDEX Request process:

(Open a terminal window on your workstation.)

```
$ jconsole dx3:1616 &
```

In both scenarios, JConsole starts with the **Overview** tab selected. The display will be similar to Exhibit 25.7.2-1.

![Exhibit 25.7.2-1. Initial JConsole Display](image-url)

JConsole has six tabs that provide different views into the running application. Of these, the most useful are:
- **Overview**: Provides a general view of the application
- **Memory**: Provides detailed information on memory usage by the application.
- **MBeans**: Provides access into detailed information and (some) control of the processes.

**TIP**: This recipe uses one of the command line options available for starting JConsole. Use /awips2/java/bin/jconsole -h to see the other options.

**Example 25.7.2-2: Using JConsole to Examine Data Ingest Status**

The MBeans tab on JConsole provides a means to examine the inner workings of the various components that make up EDEX. The various components of EDEX, as shown in Exhibit 25.7.2-2, are known as MBeans; this means they expose attributes that can be displayed in JConsole. They may also expose operations that can be executed from JConsole. This recipe looks at some of the component data that is available via that page. For illustrative purposes, we will examine the data available in the Radar decoder service. Other data types may be examined using the same basic steps.

Exhibit 25.7.2-2. JConsole MBean Tab, Initial Display

Follow the basic steps in Example 25.7.2-1/Scenario 2 to start JConsole monitoring the EDEX ingest process on DX3. Once JConsole is running, click the **MBeans** tab. This will display the **MBeans** browser, which is a fairly standard tree browser. On the left is an expandable tree view of the process’ MBeans, while the right side is used to display information about the selected node of the MBeans tree.

The initial tree display presents the top-level view of the components visible in JConsole. Of these, **log4j** and **org.apache.camel** are the nodes containing EDEX components, as
shown in Exhibit 25.7.2-3. Clicking the toggle in front of a node name in the tree expands that node; selecting a node name displays available information on that node.

The nodes under org.apache.camel provide information about both the EDEX components and the Camel framework on which they are running. (The node is named org.apache.camel because EDEX runs within the Camel framework.) Of the nodes under org.apache.camel the one providing the most status information is the routes node, as shown in Exhibit 25.7.2-4, which displays a partial listing of Camel routes. Roughly speaking, a Camel route equates to an EDEX service. (Technically, this list contains Camel contexts.)

The names of the route nodes are <host>/node. In particular, the name of the Camel route node for Radar ingest is <host>/node. Note that the <host> name is a fully qualified domain name.

**NOTE:** The images in this recipe were captured from a development server (awips-dev1.oma.ray.com) running the latest development build of EDEX. On a production system, the computer name will change appropriately.

Clicking the toggle to expand the radar-camel node, we see that radar-camel has two beans listed – radarFileConsumerRoute and radarIngestAlert. See Exhibit 25.7.2-5.

Ingest components will normally have a route with a name like dataIngestRoute. Note the naming pattern; the data type name is followed by the IngestRoute. This bean provides the information on the status of the ingest component.
Clicking the toggle on the radarIngestRoute, we see that the radarIngestRoute can display Attributes, Operations, and Notifications. Selecting Attributes displays the attributes in the data display panel.
Of the attributes listed, the most interesting are the three “Exchanges” attributes, and the “Processing Time” attributes. These are generally self-explanatory, although the units are not. The processing time attributes are given in milliseconds.

**TIP:** You can generally double click on any numeric attribute to get a real-time graphical view of that attribute.

**Example 25.7.2-3: Using JConsole to Monitor Request Status**

The *MBeans* tab on JConsole, as shown in Exhibit 25.7.2-6, provides a means to examine the inner workings of the various components that make up EDEX. The various components of EDEX are known as *MBeans*; this means they expose attributes that can be displayed in JConsole. They may also expose operations that can be executed from JConsole. This recipe looks at some of the component data that is available via that page. For illustrative purposes, we will examine μEngine (Micro Engine) requests.

[Exhibit image]

**Exhibit 25.7.2-6. EDEX Request in JConsole with Routes Expanded**

Follow the basic steps in Example 25.7.2-1/Scenario 2, to start JConsole monitoring the EDEX Request process on DX3. Once JConsole is running, click the *MBeans* tab. This will display the *MBeans* browser, which is a fairly standard tree browser. On the left is an expandable tree view of the process’ *MBeans*, while the right side is used to display information about the selected node of the *MBeans* tree.

Click the toggle to expand the *org.apache.camel* node of the tree. As with monitoring the ingest process in Example 25.7.2-2, the most useful information is in the tree under the *routes* node. Click the toggle to expand this node.
Micro Engine requests are handled by the `uengine-camel` route (remember, routes are listed as server/route). When we expand this node, we see that the `uengine-camel` has two Beans listed – `uEngineHttpJaxb` and `uEngineHttpThrift`. These beans represent the two ways a client can submit requests to the Micro Engine. In AWIPS II, CAVE makes all requests via the `uEngineHttpThrift` bean. Thus, the attributes of this bean provide information on the number and time spent on CAVE requests.

Click on the Micro Engine (`uengine-camel`) node and then expand the `uEngineHttpThrift` bean. Finally select the `attributes` node. The result is shown in Exhibit 25.7.2-7.

![Exhibit 25.7.2-7. Micro Engine Http Thrift Bean Attributes](image)

**Example 25.7.2-4: Using JConsole to Change Logging Thresholds**

As discussed previously (see section 25.6.5, EDEX Logging Levels), the messages being logged to the EDEX process log are controlled in part by changing the threshold level of the appropriate logger. The EDEX server provides a mechanism where this threshold may be changed using JConsole. It is important to realize that this change is temporary – the threshold value will revert to its configured value when the system is restarted.

To illustrate the process, we will change the logging level for the Subscription Service’s Script Runner to DEBUG. This threshold change will provide additional log messages in the EDEX process log. The Subscription Service’s Script Runner is a Service in the EDEX Ingest process.

To get started, follow the steps in Example 25.7.2-1/Scenario 2 to connect to the EDEX ingest process on DX3. Once connected, select the `MBeans` tab and expand first the `log4j` node and then the `settings` bean, as shown in Exhibit 25.7.2-8.
Exhibit 25.7.2-8. EDEX Logging Controls via JConsole

The settings MBean has no Attributes; controls are listed under the Operations node. Selecting the Operations node displays the list of available operations. (See Exhibit 25.7.2-9.)

Exhibit 25.7.2-9. Set Level Operation of the Logging MBean
Unfortunately, JConsole does not provide a very user friendly display of the available operations. The display lists the method name for the operation correctly, but the user inputs are identified simply as p1, p2, p3, etc. The JConsole GUI does identify the expected type of each input and does provide a text box for entering the value.

For setting EDEX Logging Thresholds, the Operation is called `setLevel`. `setLevel` takes two string arguments – the name of the logger (labeled as p1) and the new logging threshold (labeled as p2). The logging thresholds are simply the logging levels listed in Section 25.6.5, EDEX Logging Levels: DEBUG; INFO; WARNING; ERROR; and FATAL.

To change the logging threshold for the Subscription Service’s script runner, we will set the following loggers to DEBUG:

- `com.raytheon.edex.services.ScriptRunner`
- `com.raytheon.edex.subscription`
- `com.raytheon.edex.uengine.runners`

**NOTE:** This changes the logging threshold for any class having one of these strings as the prefix of its package. (The first one is a class, but the others could result in unexpected additional logging. Sorry, but that’s the way LOG4J works.)

Refer to Exhibit 25.7.2-9. Enter the first logger name, `com.raytheon.edex.services.ScriptRunner`, in the p1 text box, then enter DEBUG in the p2 text, and then click `setLevel`. The update status message is displayed in a separate dialog, as shown in Exhibit 25.7.2-10. Click OK to close the dialog.
Repeat the process for the other two loggers (com.raytheon.edex.subscription and com.raytheon.edex.uengine.runners).

To verify the change, monitor the EDEX log. You should start seeing DEBUG level logging. In particular, you should start see DEBUG level messages similar to ScriptRunner: script runner fired: type= timer, trigger= [1273594140002].

**TIP**: The logging threshold will only be in effect until EDEX is restarted.

### 25.8 Other EDEX-Related Monitoring

While the EDEX processes run on DX3 and DX4, they rely on processes running on other servers for operation. Key among these are the database, running on DX1/2, the QPID Message Broker, running on CP1/2, the Radar Server, running on DX1/2, and the LDM data manager, running on CP1/2. These processes are discussed in this section. Also discussed is the HDF5 data store, which is accessible via /awips2/edex/data/hdf5 from either DX3 or DX4.

**NOTE**: There are tools other than those discussed in this section that may be used for monitoring. Some of these tools are standard Linux utilities. This section is limited to tools that are either 1) part of the standard RedHat Linux distribution, or 2) delivered as part of the AWIPS II distribution.

#### 25.8.1 Monitoring the Database

This document limits the discussion of database monitoring to EDEX-related monitoring. In the recipes presented, only Linux tools and database tools delivered with the EDEX installation are discussed.

**Example 25.8.1-1: Identifying the Database Used by EDEX**

When EDEX is installed, the installer sets the Database Host and Port into the EDEX environment file, /awips/edex/bin/setup.env. This file is sourced at EDEX startup to set the runtime environment for EDEX. Examining this file allows you to determine the server name and port of the database EDEX is using to store data.

The lines containing the information start `export DB_ADDR` and `export DB_PORT`. We can use `grep` to extract this information from the startup script. The steps to follow:

```
$ ssh awips@dx3
awips@dx3’s password:
$ cd /awips2/edex/bin
$ grep -P "^export DB" setup.env
export DB_ADDR=dx1f
export DB_PORT=5432
```

**TIP**: The information obtained here is used each time we want to connect to the database.
Example 25.8.1-2: Verifying that PostgreSQL Server Is Running

The Linux `ps` tool may be used to verify that the PostgreSQL server is running. It can also be used to determine the source and status of connections to the database. This check is performed from the database server.

To simplify the output somewhat, we will choose the `-o` option of the `ps` tool. This option allows you to specify the outputs; we will limit the outputs to the process id (pid) and the command line used to start the process (args). The basic steps to follow:

```
$ ssh awips@dx1
awips@dx1’s password:
$ ps ax -o pid,args | grep -v grep | grep post
<<filtered output from ps>>
$
```

The output at this point depends on a number of factors; however, if nothing is displayed it means the PostgreSQL server is not running. If PostgreSQL is running you will see output similar to this:

```
14898 /awips/pgsql/bin/postmaster -D /awips/data -p 5432
14905 postgres: logger process
14907 postgres: writer process
14908 postgres: wal writer process
14909 postgres: autovacuum launcher process
14910 postgres: stats collector process
14913 postgres: awips postgres 127.0.0.1(50547) idle
14914 postgres: awips fxatext 127.0.0.1(50548) idle
```

This output indicates that the PostgreSQL server consists of the “postmaster” and a number of “postgres” children. The children may be divided into two types: 1) background processes; and 2) client service processes. The background processes have a command line that ends with `process`. Client service processes, which handle client interactions with the PostgreSQL server, include login information from the client, including the IP address of the client. We can use this information for status checking.

To determine if the PostgreSQL server is running, we can change the command to filter for `postmaster` rather than the more general `post`. The basic steps are:

```
$ ssh awips@dx1
awips@dx1’s password:
$ ps ax -o pid,args | grep -v grep | grep postmaster
14898 /awips/pgsql/bin/postmaster -D /awips/data -p 5432
$
```
As before, if the PostgreSQL server is not running, no output will be displayed.

To get a list of client connections, we can change the original command somewhat to filter for IP addresses. The basic steps are:

```bash
$ ssh awips@dx1
awips@dx1's password:
$ ps ax -o pid,args | grep-vgrep | grep -P "post.+
\d+.\d+.\d+.\d+"
14913 postgres: awips postgres 127.0.0.1(50547) idle
14914 postgres: awips fxatext 127.0.0.1(50548) idle
$
```

In this case, if there are no client connections, no output is displayed.

Each line of output includes, in order, the following entries: the Process ID (PID) and name of each process, the login name of the client, the IP address of the client, the port used by the client, and the status of the client’s connection. In this example, the client is logged in using the `awips` account and has connections to the `postgres` and `fxatext` databases. The login is from the database server itself – indicated by the “local” IP address of 127.0.0.1, and both connections are idle.

Example 25.8.1-3: Verifying Database Availability

Database availability can be verified by attempting to log into the database using the `psql` command line database client, which is normally installed with the AWIPS II software. To complete the login, you may need to specify a password – that depends on the authentication levels set for the database. At this writing, the new AWIPS II databases require passwords; the legacy (AWIPS I) databases utilized by EDEX do not.

To simplify connecting to the database using `psql`, several command line flags may be used. The basic syntax to use is

```
psql -h host -p port -d database -U user
```

where:

- **host** is the host name identified in Example 25.8.1-1.
- **port** is the port identified in Example 25.8.1-1.
- **database** is the database you wish to access.
- **user** is the database user for the login.

For a complete listing of command line options, run `psql --h` from the command line. Running `psql --h` will also identify the defaults for any command line options.

To test the availability of the EDEX database, we will attempt to connect to the `metadata` database as user `awips`. Because are interested in confirming the availability of the database to EDEX, we will attempt the connection from DX3. The steps to follow:

```
$ ssh awips@dx3
```
awips@dx3’s password:
$ psql -h dx1 -p 5432 -d metadata -U awips

psql (9.2.4)
Type "help" for help.

metadata=#

**NOTE:** Please provide the password if prompted.

This listing shows a complete session connecting to the metadata database as user awips and then logging out. It is important to note that the request for a password shows the connection is accessible. If the database is not available, *psql* will return a message similar to this:

```
Psql: could not connect to server: Connection refused
Is the server running on host "dx1" and accepting TCP/IP connections on port 5432?
```

**NOTE:** To save space, the welcome banner was omitted from this listing. It is also omitted from other recipes involving *psql*.

**Example 25.8.1-4: Determining Data Type Tables in Metadata Database**

AWIPS II uses the metadata database to store information about ingested data. To get a list of the main table names that store the metadata, we can query a table named `plugin_info` in that database. The steps to follow:

```
$ ssh awips@dx3
awips@dx3’s password:
$ psql -h dx1 -p 5432 -d metadata -U awips
Password for user awips:
metadata=# select name, tablename from plugin_info
metadata-# where database='metadata'
metadata-# and tablename != '<unspecified>';

<table>
<thead>
<tr>
<th>name</th>
<th>tablename</th>
</tr>
</thead>
<tbody>
<tr>
<td>acarssounding</td>
<td>acarssounding</td>
</tr>
<tr>
<td>level</td>
<td>level</td>
</tr>
<tr>
<td>grib</td>
<td>grib</td>
</tr>
</tbody>
</table>
(more rows are printed - there are 51 rows total)
```

```
metadata=# \q
```

As an example, this result shows that the main table of GRIB is *grib*.
TIP: You can spread the sql query over several lines in PSQL; simply press return to at the end of the line. Note that the prompt changes slightly. To execute the query, terminate the line with a semicolon and then press return.

Example 25.8.1-5: Determine Latest Insertion Time of GRIB Metadata

One basic method for checking on data flow is to query the latest insertion time of the metadata for a specific data type. Most of the data type tables contain a field named inserttime that contains the data/time stamp of the insertion. This is not the same as the valid time of the data,\(^2\) of course, but it can indicate if a decoder is actively performing metadata data inserts.

Note the use of a couple of PostgreSQL functions – to_char() and now(). now() provides the current data/time on the database server; to_char() is used to modify the print format of the time fields. The steps to follow:

```
$ ssh awips@dx3
awips@dx3’s password:
$ psql -h dx1 -p 5432 -d metadata -U awips
Password for user awips:
metadata=# select to_char((select now()),'MM/DD HH24:MI')
metadata=# as "Time Now", to_char((select inserttime
metadata(# from awips.grib order by inserttime desc
metadata(# limit 1),'MM/DD HH24:MI') as "Latest";
           Time Now | Latest
-------------+-------------
      05/11 21:25 | 05/11 21:24
metadata=# \q
```

This result shows the current time as well as the most recent insert time for GRIB data. Note that the last insert was at 21:24 while the current time is 21:25.

TIP: To check the latest insert time for a different data type, simply change grib to the desired data table name.

TIP: Given the length of the query, you may want to save it into a file for later use. Assuming the query has been saved to /path/to/file/query.sql, you execute it by entering \i /path/to/file/query.sql at the psql command prompt.”

Example 25.8.1-6: Determining Database Session Information Using psql

The postgres database contains useful information about the database that may be queried to help determine the state/health of the database. Two useful tables are the pg_database table and the pg_stat_activity table. The pg_database table provides information about

\(^2\) In most tables, the valid time of the data is contained in a field named refitime.
the databases available in the PostgreSQL server; the `pg_stat_activity` provides a snapshot of the current activity in the server. In addition, the `information_schema.columns` table can be used to get a listing of the columns in a specific table. This Example presents two scenarios: 1) determining the available databases; and 2) determining the activity for a specific client.

**Scenario 1: Determine the Databases Available in the PostgreSQL Server on DX1**

To determine the available databases, we connect to the postgres database in the PostgreSQL server on DX1 using `psql`. Note that we make the connection from DX3 because we are interested in the structure of the database used by the EDEX server. (See Example 25.8.1-1 to determine the database used by EDEX on DX3.) We then query the `pg_database` table to determine the available table names. The steps to follow:

```
$ ssh awips@dx3
awips@dx3’s password:
$ psql -h dx1 -p 5432 -d postgres -U awips
Password for user awips:
postgres=# select table_name,column_name from
postgres=# information_schema.columns
postgres=# where table_name = 'pg_database';
table_name        | column_name
-------------------+---------------
pg_database        | datname
pg_database        | datdba
pg_database        | encoding
pg_database        | datistemplate
pg_database        | datallowconn
pg_database        | datconnlimit
pg_database        | datlastsysoid
pg_database        | datfrozenxid
pg_database        | datatablespace
pg_database        | datconfig
pg_database        | datacl
(11 rows)
```

```
postgres=# select datname as "database"
postgres=# from pg_database
postgres=# where datistemplate='f';
    database
------------
    postgres
    metadata
    fxatext
    maps
    hd_ob83oax
```
As this output shows, there are six AWIPS II databases available on DX1. As noted before, postgres is a standard database that contains general information and is not specific to AWIPS II.

**Scenario 2: Determining Status of Database Clients**

To determine client status with the PostgreSQL server, we connect to the postgres database in the PostgreSQL server on DX1 using psql. Note that we make the connection from DX3 because we are interested in client connections of the database used by the EDEX server. (See Example 25.8.1-1 to determine the database used by EDEX on DX3.)

Once connected, we will first query the `pg_stat_activity` table to determine the status of each database connection. The columns to query are somewhat arbitrarily chosen; several “trial and error” queries were attempted before deciding on the query shown here. The as modifier is used on some columns in the query to provide more readable output. In addition, the where clause is used to filter out the activity created by running this query.

Also note the use of a couple of PostgreSQL functions; `to_char()` and `now()`. `now()` provides the current data/time on the database server; `to_char()` is used modify the print format of the time fields. The steps to follow:

```bash
$ ssh awips@dx3
awips@dx3’s password:
$ psql -h dx1 -p 5432 -d postgres -U awips
postgres=# select table_name,column_name from
(7 rows)
```

```
postgres=# \q
```

```
dc_ob7oax
hmbdb
(7 rows)
```

```
postgres=# \q
```

```
As this output shows, there are six AWIPS II databases available on DX1. As noted before, postgres is a standard database that contains general information and is not specific to AWIPS II.

**Scenario 2: Determining Status of Database Clients**

To determine client status with the PostgreSQL server, we connect to the postgres database in the PostgreSQL server on DX1 using psql. Note that we make the connection from DX3 because we are interested in client connections of the database used by the EDEX server. (See Example 25.8.1-1 to determine the database used by EDEX on DX3.)

Once connected, we will first query the `pg_stat_activity` table to determine the status of each database connection. The columns to query are somewhat arbitrarily chosen; several “trial and error” queries were attempted before deciding on the query shown here. The as modifier is used on some columns in the query to provide more readable output. In addition, the where clause is used to filter out the activity created by running this query.

Also note the use of a couple of PostgreSQL functions; `to_char()` and `now()`. `now()` provides the current data/time on the database server; `to_char()` is used modify the print format of the time fields. The steps to follow:

```bash
$ ssh awips@dx3
awips@dx3’s password:
$ psql -h dx1 -p 5432 -d postgres -U awips
postgres=# select table_name,column_name from
(7 rows)
```

```
postgres=# \q
```

```
dc_ob7oax
hmbdb
(7 rows)
```

```
postgres=# \q
```

```
As this output shows, there are six AWIPS II databases available on DX1. As noted before, postgres is a standard database that contains general information and is not specific to AWIPS II.

**Scenario 2: Determining Status of Database Clients**

To determine client status with the PostgreSQL server, we connect to the postgres database in the PostgreSQL server on DX1 using psql. Note that we make the connection from DX3 because we are interested in client connections of the database used by the EDEX server. (See Example 25.8.1-1 to determine the database used by EDEX on DX3.)

Once connected, we will first query the `pg_stat_activity` table to determine the status of each database connection. The columns to query are somewhat arbitrarily chosen; several “trial and error” queries were attempted before deciding on the query shown here. The as modifier is used on some columns in the query to provide more readable output. In addition, the where clause is used to filter out the activity created by running this query.

Also note the use of a couple of PostgreSQL functions; `to_char()` and `now()`. `now()` provides the current data/time on the database server; `to_char()` is used modify the print format of the time fields. The steps to follow:

```bash
$ ssh awips@dx3
awips@dx3’s password:
$ psql -h dx1 -p 5432 -d postgres -U awips
postgres=# select table_name,column_name from
(7 rows)
```

```
postgres=# \q
```

```
dc_ob7oax
hmbdb
(7 rows)
```

```
postgres=# \q
```
This output gives a snapshot of the client activity for each database that is currently engaged. As we can see, there are a total of five client connections, all of which are currently idle. Displaying the start time and then querying for the current time provides an idea of how long each connection has been in use.

**NOTE:** Some of the queries used by AWIPS II are rather long, so the output may not be this brief.

**NOTE:** The sample output was produced by querying an isolated database; in an active AWIPS II installation, the number of connections will be considerably higher.

**Example 25.8.1-7: Determining Data Type Tables in Metadata Database (Revisited)**

When combined, Example 25.8.1-4 and Example 25.8.1-5 present a method for determining the latest insertion time for the AWIPS II GRIB metadata table. This Example presents additional information on how this was done. The intent is to present a pattern of queries that can be adapted to mine other data from the database.

Generally, we will follow these steps:

1. Connect to the database server using `psql`.
2. Query to determine available schema in the database.
3. Query to determine the tables in the schema.
4. Query to determine the fields in the desired table.
5. Query to determine contents of a specific table.

When possible, where clauses should be used to limit queries. For example, many of the PostgreSQL elements have a name that is prefaced with “pg_” These can be filtered out using a where clause similar to where <<field name>> not like 'pg%'.

Following this general outline, the steps to follow are:

```sql
$ ssh awips@dx3
awips@dx3’s password:
$ psql -h dx1 -p 5432 -d metadata -U awips
Password for user awips:
metadata=# select table_name,column_name
metadata=# from information_schema.columns
metadata=# where table_name='schemata';

+-----------------+-------------------------+
| table_name      | column_name             |
| schemata        | catalog_name            |
| schemata        | schema_name             |
| schemata        | schema_owner            |
| schemata        | default_character_set_catalog |
| schemata        | default_character_set_schema |
| schemata        | default_character_set_name |
| schemata        | sql_path                |
+-----------------+-------------------------+
(7 rows)

metadata=# select catalog_name,schema_name
metadata=# from information_schema.schemata
metadata=# where schema_name not like 'pg%' and schema_name not like 'information_%';

+-----------------+-------------------------+
| catalog_name    | schema_name             |
| metadata        | public                  |
| metadata        | awips                   |
| metadata        | subscription            |
| metadata        | vtec                    |
+-----------------+-------------------------+
(4 rows)

metadata=# select table_name,column_name
metadata=# from information_schema.columns
metadata=# where table_name = 'pg_tables';

+-----------------+-------------------------+
| table_name      | column_name             |
| pg_tables       | schemaname              |
| pg_tables       | tablename               |
| pg_tables       | tableowner              |
| pg_tables       | tablespace              |
+-----------------+-------------------------+
```

---

`AWP.MAN.SMM.A2` 25-70

-- Hard copies uncontrolled. Verify effective date prior to use. --
pg_tables | hasindexes
pg_tables | hasrules
pg_tables | hastriggers
(7 rows)

metadata=# select schemaname, tablename
metadata=# from pg_tables
metadata=# where schemaname = 'awips'
metadata=# order by tablename asc;

<table>
<thead>
<tr>
<th>schemaname</th>
<th>tablename</th>
</tr>
</thead>
<tbody>
<tr>
<td>awips</td>
<td>acars</td>
</tr>
<tr>
<td>awips</td>
<td>acarsbuilder</td>
</tr>
<tr>
<td>awips</td>
<td>acarssounding</td>
</tr>
<tr>
<td>awips</td>
<td>acarssoundinglayer</td>
</tr>
<tr>
<td>awips</td>
<td>airep</td>
</tr>
<tr>
<td>awips</td>
<td>airmet</td>
</tr>
<tr>
<td>awips</td>
<td>airmet_location</td>
</tr>
<tr>
<td>awips</td>
<td>goessounding</td>
</tr>
<tr>
<td>awips</td>
<td>grib</td>
</tr>
<tr>
<td>awips</td>
<td>grib_lambertconformal_coverages</td>
</tr>
<tr>
<td>awips</td>
<td>vil</td>
</tr>
<tr>
<td>awips</td>
<td>warning</td>
</tr>
<tr>
<td>awips</td>
<td>warning_ugczone</td>
</tr>
</tbody>
</table>

<<several rows deleted>>

| awips | goessounding |
| awips | grib         |
| awips | grib_lambertconformal_coverages |

<<a bunch of rows deleted>>

| awips | vil       |
| awips | warning   |
| awips | warning_ugczone |

(94 rows)

metadata=# select table_name, column_name
metadata=# from information_schema.columns
metadata=# where table_name = 'grib'
metadata=# and column_name like '%time';

table_name | column_name
-----------|--------------
grib       | forecasttime
| reftime
| inserttime
(3 rows)

metadata=# select to_char((select now()),'MM/DD HH24:MI')
metadata=# as "Time Now", to_char((select inserttime
metadata=# from awips.grib order by inserttime desc
metadata=# limit 1),'MM/DD HH24:MI') as "Latest";

<table>
<thead>
<tr>
<th>Time Now</th>
<th>Latest</th>
</tr>
</thead>
<tbody>
<tr>
<td>05/11 21:25</td>
<td>05/11 21:24</td>
</tr>
</tbody>
</table>
metadata=# \q
$

The final query shows that the latest insert time is approximately 1 minute prior to the current time. From this we know the insertions are recent; it does not identify whether the data being inserted is up to date, however.

Here are a couple more notes on these queries. First, the PostgreSQL query engine is geared toward queries into the `public` schema of a database. To query into other schemata, you need to preface the table name with the schemata name, for example, `information_schema.schemata` or `awips.grib`. Second, note that many, but not all, of the AWIPS II metadata tables include an `inserttime` field, so it may be necessary to use a `where` clause similar to `where column_name like '%time'` to identify time fields to query.

### 25.8.2 Monitoring the Data Store

EDEX uses Hierarchical Data Format (HDF) 5 files to store data that has been ingested. EDEX does not interact directly with the HDF5 repository; rather it interacts with PyPIES, running on the DX2, to access the Data Store. As seen from DX2, the HDF5 Data Store is located in `/awips2/edex/data/hdf5`. This directory contains both the HDF5 data store and other directories, notably the topo directory, which must be shared between DX1 and DX2 to allow the EDEX servers to operate correctly.

For gridded data, stored in the `grib` directory, and forecast grids, stored in the `gfe` directory, the structure is somewhat different. Gridded data is stored (and purged) by model and model run, so the grib directory contains a subdirectory for each model; HDF5 files for each model run are located in each subdirectory. For forecast grids, located under the `gfe` directory, grids are stored by site and specific type and model run.

Although the directory structure varies, there are some general techniques that apply to all data types. These techniques are covered in the Examples in this section.

#### Example 25.8.2-1: Determining the Disk Space Used by the Data Store

Disk space with the Data Store is managed by the EDEX Purge Service. This allows the data stored in the HDF5 repository to remain in synch with the metadata in the database. In order to determine optimal data retention rules, it is desirable to determine the disk usage in the Data Store.

The Linux `du` tool can be used to get a rough idea of the size of the Data Store. This information can then be used to determine if retention rules need to be modified to prevent over utilization of disk space. The basic procedure is

```
$ ssh awips@dx3
awips@dx3’s password:
$ cd /awips2/edex/data/hdf5
$ du -sh --exclude=topo
    42G    .
$ 
```
This result shows that the data store currently contains approximately 42 GBytes of data. Note that the --exclude flag is used to have the topo file not counted.

**Example 25.8.2-2: Modifying Data Retention Time**

The Purge Service utilizes information packaged with each data decoder plug-in to determine the data retention rules for the plug-in. The data retention rules are then applied to delete records from the database and HDF5 files from the Data Store. Most data plug-ins use a simple, retention time-based purging. The rule here is simple: “If the data is more than X hours old, delete it.” The main plug-in that does not use this strategy is the grib plug-in; it uses a purge strategy based on the desired number of model runs to retain.

Data retention time is tracked in the awips.plugin_info table, which is in the metadata database in AWIPS II. The key fields are name, which identifies the data type, retentiontime, which identifies the number of hours to retain the data. When the Purge Service runs, it queries the database for the retention time; because of this, the retention time for a specific data type can be changed without requiring a restart of the EDEX server.

This Example presents two scenarios: 1) determining retention time; and 2) changing retention time. The data type used to illustrate this process is satellite.

**Scenario 1: Determining Retention Time for Satellite Data**

The retention time for satellite data may be obtained with a simple database query. This can be performed from any computer that has psql installed. For consistency with other psql examples, we will perform the test from the EDEX server. The basic steps to follow:

```
$ ssh awips@dx3
awips@dx3’s password:
$ psql -h dx1 -p 5432 -d metadata -U awips
Password for user awips:
metadata=# select name, retentiontime
metadata=# from awips.plugin_info
metadata=# where name = 'satellite';
  name     | retentiontime
-----------+---------------
satellite |            24
(metadata=# \
$)
```

From this result, we see that the retention time for satellite data is 24 hours.
**Scenario 2: Changing Retention Time for Satellite Data**

The retention time for *satellite* data is changed by changing its retention time entry in the *plugin_info* table. This is accomplished using an SQL update command. The update command changes the value in the database. A word of caution: Be very careful running this command; small errors made when using *update* can cause unexpected problems and generally cannot be corrected/rolled back.

For this example, we will first change the *satellite* data retention time to 18 hours and then repeat the query from scenario 1 to confirm the change. The basic procedure is:

```
$ ssh awips@dx3
awips@dx3's password:
$ psql -h dx1 -p 5432 -d metadata -U awips
Password for user awips:
metadata=# update awips.plugin_info
metadata=# set retentiontime = 18
metadata=# where name = 'satellite';
UPDATE 1
metadata=# select name, retentiontime
metadata=# from awips.plugin_info
metadata=# where name = 'satellite';
name    | retentiontime
--------|---------------
satellite |            18
(1 row)
metadata=# \q
$
```

The next time the Purge Server runs, all *satellite* data more than 18 hours old will be purged.

**TIP:** If you are running the EDEX server from a canned data set, you can avoid having to re-ingest the data by modifying the data retention rule. Simply set the retention time to a large number of hours, say 99999 – approximately 11 and a half years – and it will not be purged.

---

**Example 25.8.2-3: Determining Available Data Types (in the Data Store)**

There are several means of determining data availability on the EDEX cluster. Database queries similar to those presented in Example 25.8.1-4 and Example 25.8.1-5 can be used to determine the data type EDEX can process as well as the latest insert time for the various data types. In this Example, we approach the question from the Linux file system. As is frequently the case with file system issues, basic Linux tools such as *ls*, *find*, and *tree* can provide useful information. This Example presents two scenarios; 1) getting a basic listing of available data types; and 2) determining which data types have current HDF5 repositories.
**Scenario 1: Determining available data types**

If `ls` is used to get a listing of the contents of `/awips2/edex/data/hdf5`, we will get a rough indication of the available data types. What we really need, however, is a listing of the subdirectories of `/awips2/edex/data/hdf5` that are the root of a directory tree containing HDF5 files.

A quick check for available data can be performed using the Linux `find` tool. Because available data is in HDF5 files, we use `find` to search for HDF5 files. The basic command is `find . -name "*.h5"`. Unfortunately, the number of HDF5 files on an active EDEX cluster can reach the thousands. As a result, we need to do some filtering of the output from `find` to generate a more compact list. The techniques will be similar to those used in Example 25.8.1-4. The basic steps to follow:

```
$ ssh awips@dx2
awips@dx2's password:
$ cd /awips2/edex/data/hdf5
$ find . -name "*.h5" | grep -Pv "^./gfe" | cut -d '/' -f 2 > ingest-types.txt
$ sort ingest-types.txt > sorted-ingest-types.txt
$ uniq sorted-ingest-types.txt ingest-types.txt
$ cat ingest-types.txt
binlightning
bufrascat
bufrhwd
bufrmosAVN
bufrmosEta
bufrmosGFS
<<several more lines>>
scan
sfcobs
tcg
tcs
topo
vil
```

The final output shows a sorted list of data types that utilize HDF5 for data storage. Note that it does not identify specifics such as GRIB models, satellite regions, or radar types. Those can be determined more easily by searching the database.

**Scenario 2: Determining the latest available data for a data type**

The `-l` flag on the Linux `ls` tool can be used to determine the latest modification time for the contents of a directory. However, what we really want is to find the newest file in a directory structure when only the root of the structure is known.

A quick and dirty check can be performed using some of the options of the Linux `tree` tool. The options to use are `-D` (print modification time), `-f` (print full path information – also suppresses tree style output), `-i` (don’t print indentation lines), and `-t`
(sort by modification time). Finally, we will use the `grep` tool to limit the output to HDF5 files and the `cut` tool to reduce the output somewhat. The steps to follow (using `satellite` as the data type):

```
$ ssh awips@dx2
awips@dx2's password:
$ cd /awips2/edex/data/hdf5
$ tree -Dflt satellite |grep -P "\.h5$" |cut -d '/' -f 1
|-- [Feb 15 5:25] satellite
|-- [Feb 15 4:25] satellite
|-- [Feb 15 2:25] satellite
|-- [Feb 15 0:55] satellite
|-- [Feb 14 23:25] satellite
|-- [Feb 14 22:25] satellite
|   |-- [Feb 14 20:25] satellite
|   |-- [Feb 14 18:55] satellite
|   |-- [Feb 14 17:43] satellite
|   |-- [Feb 14 15:56] satellite
|   |-- [Feb 14 14:25] satellite
|   |-- [Feb 14 13:25] satellite
|   |-- [Feb 14 11:25] satellite
|   |-- [Feb 14 10:25] satellite
|   |-- [Feb 14 8:25] satellite
|   |-- [Feb 14 7:25] satellite
|   `-- [Feb 14 5:25] satellite
|   |-- [Feb 17 14:25] satellite
|   |-- [Feb 17 13:25] satellite
|   |-- [Feb 17 11:57] satellite
|   |-- [Feb 17 10:25] satellite
|   |-- [Feb 17 9:55] satellite
|   |-- [Feb 17 8:25] satellite
|   |-- [Feb 17 7:25] satellite
|   |-- [Feb 17 5:57] satellite
|   |-- [Feb 17 4:57] satellite
|   |-- [Feb 17 2:25] satellite
|   |-- [Feb 17 1:55] satellite
|   |-- [Feb 17 0:55] satellite
|   |-- [Feb 16 22:25] satellite
|   |-- [Feb 16 20:09] satellite
|   |-- [Feb 16 19:25] satellite
|   |-- [Feb 16 17:25] satellite
|   |-- [Feb 16 15:55] satellite
|   |-- [Feb 16 14:25] satellite
|   |-- [Feb 16 13:25] satellite
|   |-- [Feb 16 11:25] satellite
```
In this case, we see that the latest modification to the satellite data stored in HDF5 was at 13.43 on October 25.

Once again, due to the potential number of HDF5 files in the data store on an EDEX cluster, it may be desirable to pipe the output of this command into a file and use a text viewer to examine the file. In this case, we use the Linux `head` tool to view the first few lines of the file. The steps to follow (using `grib` as the data type):

```bash
$ ssh awips@dx2
awips@dx2’s password:
$ cd /awips2/edex/data/hdf5
$ tree -Dflt grid |grep -P "\.h5$" > available-gribs.txt
$ head -n 5 available-gribs.txt
|-- [Jan 16 15:39] grid/1ad-hoc-test/FHAG/1ad-hoc-test-2014-01-16-14-FH-000.h5
  |-- [Jan 16 15:20] grid/1ad-hoc-test/FHAG/1ad-hoc-test-2014-01-16-13-FH-000.h5
  |-- [Jan 16 13:39] grid/1ad-hoc-test/FHAG/1ad-hoc-test-2014-01-16-12-FH-000.h5
  `-- [Jan 16 13:21] grid/1ad-hoc-test/FHAG/1ad-hoc-test-2014-01-16-11-FH-000.h5
|-- [Jan 16 11:12] grid/1test-gfs/FRZ/1test-gfs-2014-01-16-06-FH-192.h5
  |-- [Jan 16 11:12] grid/1test-gfs/FRZ/1test-gfs-2014-01-16-06-FH-171.h5
  |-- [Jan 16 11:12] grid/1test-gfs/FRZ/1test-gfs-2014-01-16-06-FH-174.h5
  |-- [Jan 16 11:12] grid/1test-gfs/FRZ/1test-gfs-2014-01-16-06-FH-177.h5
  |-- [Jan 16 11:12] grid/1test-gfs/FRZ/1test-gfs-2014-01-16-06-FH-180.h5
  `-- [Jan 16 11:12] grid/1test-gfs/FRZ/1test-gfs-2014-01-16-06-FH-183.h5
  |-- [Jan 16 11:12] grid/1test-gfs/FRZ/1test-gfs-2014-01-16-06-FH-186.h5
  `-- [Jan 16 11:12] grid/1test-gfs/FRZ/1test-gfs-2014-01-16-06-FH-189.h5
|-- [Jan 16 11:12] grid/1test-gfs/FRZ/1test-gfs-2014-01-16-06-FH-147.h5
  |-- [Jan 16 11:12] grid/1test-gfs/FRZ/1test-gfs-2014-01-16-06-FH-153.h5
  `-- [Jan 16 11:12] grid/1test-gfs/FRZ/1test-gfs-2014-01-16-06-FH-156.h5
  |-- [Jan 16 11:12] grid/1test-gfs/FRZ/1test-gfs-2014-01-16-06-FH-159.h5
```
This output is a bit deceptive, however, because the time sorting is performed on files within a single directory. A more accurate result is obtained by either sorting the output file or filtering for a specific grib model.

### 25.8.3 Monitoring QPID

QPID is an Advanced Message Queuing Protocol (AMQP) broker that provides for messaging between the various components in the EDEX cluster. It allows for a fairly high level of load balancing within the cluster and helps to minimize the loss of data ingest in the event an EDEX instance experiences an unplanned restart. QPID also allows the LDM to post messages to EDEX; see section 25.3, Basic Data Flow Concepts for details. QPID runs on the CP 1/2 cluster.

#### Example 25.8.3-1: Verifying QPID Execution

QPID execution can be verified by logging onto the CP1 and using either `ps` or `pidof` to verify that the process is running. Note that the actual QPID process is running as qpidd.

First, the steps for using `ps`:

```
$ ssh awips@cpsbn1
awips@cpsbn1’s password:
$ ps ax -o pid,args | grep qpidd | grep –v grep
```

This shows that QPID is running; if QPID is not running, there will be no output.

The linux `pidof` tool outputs the process ID (PID) of a process; if the process is not running, a blank line is output. Here are the steps to follow:

```
$ ssh awips@cpsbn1
awips@cpsbn1’s password:
$ pidof qpidd
```

```
20632
```

```
$
```

**NOTE:** The Linux `pidof` tool is normally installed in `/sbin`, which may not be in a user’s path. If a `command not found` error is displayed, try using `/sbin/pidof`. 
**Example 25.8.3-2: Monitoring QPID Execution**

As with the EDEX processes (see Example 25.7.1-10), the general status of QPID may be monitored using the Linux `top` utility. To do so, we first use `pidof` to determine the PID of the QPID process, and then use the `-p` option on `top` to limit monitoring to the QPID instance. The steps to follow:

```
$ ssh awips@cpsbn1
awips@cp1’s password:
$ pidof qpidd
24565
$ top -p 24565
```

To exit `top`, press either `q` or `<ctrl-c>`. Typical output is shown in Exhibit 25.8.3-1.

```
top - 21:02:06 up 91 days, 22:39, 1 user, load average: 0.22, 0.24, 0.13
Tasks:   1 total,   0 running,   1 sleeping,  0 stopped,   0 zombie
Cpu(s):  0.1%us,  0.1%sy,  0.0%ni, 99.8%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:   6222480k total,  5657220k used,  565260k free,  385132k buffers
Swap:  2047992k total,  88k used,  2047904k free,  3354092k cached
PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
24565 awips     15   0  599m 279m 6048 S  2.3  4.6 192:46.88 qpidd
```

**Exhibit 25.8.3-1. Top Display Monitoring QPID**

**Example 25.8.3-3: Monitoring QPID Logging**

When it is running, QIPD writes `/var/log/messages`; this file is owned by `root` and is readable only by `root`, but it can be monitored to provide information on QPID’s operation. QPID messages are formatted as follows:

```
<date> <time> <server> qpidd[PID] <message>
```

This format allows you to scan `/var/log/messages` for QPID messages; it also allows you to monitor QPID in real time.

**Scenario 1: Scanning for QPID Messages**

This scenario is similar to the recipes for scanning the EDEX process logs, e.g., Example 25.6.5-1 or Example 25.7.1-3. As in those previous recipes, the Linux `grep` utility is our weapon of choice. The steps to follow:

```
$ ssh awips@cpsbn1
awips@cp1’s password:
$ cd /var/log
$ grep qpidd messages
```

```
17 14:35:03 cpsbn1-tbw3 qpidd[28596]: 2013-02-17 14:35:03 warning
task overran 2 times by 2ms (taking 200492ns) on average.
```
The displayed messages will vary depending on conditions encountered by QPID. Normally, QPID logs very few messages. If no messages are displayed and QPID is known to be running (see Example 25.8.3-1), you can assume that QPID is running normally.

Scenario 2: Real-Time Scanning of QPID Messages

This scenario is similar to the recipes for real-time scanning of the EDEX process logs, e.g., Example 25.7.1-5 or Example 25.7.1-6. As in those previous recipes, the trick is to use the Linux `tail` utility to obtain a real-time display of the log and to pipe `tail`’s output to `grep` to filter the display for QPID messages. The basic steps:

```
$ ssh awips@cpsbn1
awips@cp1’s password:
$ cd /var/log
$ tail -f messages | grep qpidd
```

Feb 17 14:37:52 cpsbn1-tbw3 qpidd[28596]: 2013-02-17 14:37:52
warning ManagementAgent::periodicProcessing task overran 1 times by 2ms (taking 3480204ns) on average.

Feb 17 14:38:02 cpsbn1-tbw3 qpidd[28596]: 2013-02-17 14:38:02
warning ManagementAgent::periodicProcessing task overran 2 times by 2ms (taking 4613ns) on average.
Section 25.8.3.1 QPID Monitoring Tools

QPID supports both stand-alone and clustered operation. Clustered operation facilitates failover operations on the CPSBN cluster, allowing failover of QPID operation without loss of messages. QPID management and monitoring is accomplished by a set of Python language tools that are installed on the CPSBN cluster. As these tools provide both monitoring and management/control functions, care must be taken to avoid operational impact. All examples in this section have been designed to provide monitoring functionality without affecting operations.

QPID monitoring tools are located in /awips2/python/bin on the CPSBN1/2 cluster. They may be installed on other systems as well. This section focuses on five of these tools. They are listed and described in Table 25.8.3.1-1.

NOTE: The QPID tools described here can be used with both the clustered and the unclustered QPID. In addition, all the tools have a –h option, which will cause the tool to print a usage message and terminate.
### Table 25.8.3.1-1. Basic QPID Tools

<table>
<thead>
<tr>
<th>Tool</th>
<th>Usage (Arguments)</th>
<th>Description</th>
</tr>
</thead>
</table>
| qpid-cluster    | [options] [host[:port]] | General administration tool. Use to query information and administer the QPID cluster. Monitoring options:  
- C: view client connections for all cluster members  
- c: View client connections for a specific cluster member |
| qpid-stat       | [options] [host[:port]] | Displays statistics on QPID operation Monitoring options:  
- b: show brokers  
- c: show connections  
- e: show exchanges  
- q: show queues  
- S: sort by a specified column name; msg, msgIn, msgOut and bytes are useful columns  
- L: limit the number of lines to display |
| qpid-printevents | [host[:port]] | Provides a real-time view of QPID events. Monitoring options:  
- h: display usage message  
<crtl-C> to terminate |
| qpid-queue-stats | [options] [-a host[:port]] | Provides a real-time view of QPID queue statistics. Monitoring options:  
- f: list of queues to monitor, comma separated, regex are accepted  
<crtl-C> to terminate |
| qpid-config     | [-a host[:port]] [command] | General QPID administration tool. Use to query and administer QPID. Monitoring commands:  
- queues: prints a list of queues  
- exchanges: prints list of exchanges  
With no command, summary statistics are printed. |

### Example 25.8.3.1-1: Determining the Status of the QPID Cluster

QPID runs on the CPSBN1/2 cluster. In normal operation, QPID runs in clustered mode. The *qpid-cluster* tool may be used both to determine the status of the QPID cluster and to identify the members of the cluster. This tool is also used to manage the QPID cluster; care must be exercised to avoid affecting operations.

The basic steps to follow:

```
$ ssh root@cpsbn1
root@cpsbn1’s password:
$ cd /awips2/python/bin
root@cpsbn1-tbw3 bin]# qpid-stat -b

Brokers
  broker  conn  exch  queue
  ============================
  Broker  338  6  205
```
The output above is typical but will vary by site. If QPID is not operating in clustered mode, the following message is printed:

Failed: Exception - Clustering is not installed on the broker

**Example 25.8.3.1-2: Monitoring QPID Queue Status**

The *qpid-stat* tool provides statistics on various QPID activities — queues, brokers, exchanges, and connections. In this example, we monitor the top 10 queues by number of messages in the queue. The output from *qpid-stat* is rather long (generally about 170 characters per line), so *cut* is used to shorten the output length to 145 characters. To provide near real-time display, *watch* is used to update the display every 5 seconds. The basic command for this is **watch –n 5 ./qpid-stat –q –S msg –L 10 | cut –b -135”**.

The full procedure:

```
$ ssh awips@cpsbn1
awips@cpsbn1’s password:
$ cd /awips2/python/bin
$ watch –n 5 "qpid-stat –q –S msg –L 10 | cut –b -135"
<<results of watch are displayed>>
<ctrl-C>
```

Typical output is shown in Exhibit 25.8.3.1-1.

```
Every 5.0s: qpid-stat -q -S msg -L 10 | cut -b -135     Tue Oct 25 15:51:12 2011
Queues
  queue             dur  auto
Del  excl  msg   msgIn  msgOut  bytes  bytesIn  bytesO
===============================================================
Ingest.dhr                           Y
  3.90k  19.0k  15.0k    269k  1.31m    1.04m
scheduledQCScanWork                  1.12k  1.13k     14     0      0        0
watchwarn                                 Y
  10  3.41m  3.41m     560   198m     198m
Ingest.RadarRadarServer               Y
  0      0      0       0    214      214
_reedx.alerts.utility@amq.topic_0f77cb37-292b-4682-9eb4-ab1f9a543dd0  Y
  0   52   52        0  30.6k   30.6k
_reedx.alerts.utility@amq.topic_76c92477-6363-4d2b-b671-71d59afa3aad  Y
  0     95     95        0      0        0
Ingest.wcp                                0
  4      4        0  214     214
Ingest.ffg                                0
  1.65k  1.65k     0    96.3k   96.3k
satNotification                      Y
```

**Exhibit 25.8.3.1-1. Typical Output of qpid-stat for Queue Monitoring**
Example 25.8.3.1-3: Monitoring Individual QPID Queues

The same general approach can be used to monitor specific queues by piping the output of `qpid-stat` into `grep` with an appropriate pattern. For example, the EDEX ingest processes pull messages from QPID queues whose names start with `Ingest`. We can use this information to monitor ingest queues. As before, use `cut` to shorten the output and `watch` to provide near real-time monitoring. The basic command is `watch --n 5 './qpid-stat -q -S msg | grep -P "Queues|queue|===|Ingest" | cut -b -135'` (The `grep` search string selects lines of output containing `Queues`, `queue`, `===`, or `Ingest`. The first three match the header lines output by `qpid-stat`; the last one catches the ingest queue stats.)

The full procedure:

```bash
$ ssh awips@cpsbn1
awips@cpsbn1’s password:
$ cd /awips2/python/bin
$ watch --n 5 './qpid-stat -q -S msg | grep -P "Queues|queue|===|Ingest" | cut -b -135'
<<results of watch are displayed>>
<ctrl-C>
```

Typical output is shown in Exhibit 25.8.3.1-2.

<table>
<thead>
<tr>
<th>Queue Name</th>
<th>Sec</th>
<th>Depth</th>
<th>Enq Rate</th>
<th>Deq Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ingest.Grib</td>
<td>30.00</td>
<td>0</td>
<td>0.40</td>
<td>0.40</td>
</tr>
<tr>
<td>Ingest.Radar</td>
<td>10.00</td>
<td>0</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>Ingest.RadarRadarServer</td>
<td>10.00</td>
<td>0</td>
<td>2.50</td>
<td>2.50</td>
</tr>
<tr>
<td>Ingest.Satellite</td>
<td>10.00</td>
<td>0</td>
<td>0.20</td>
<td>0.20</td>
</tr>
<tr>
<td>Ingest.Shef</td>
<td>30.00</td>
<td>0</td>
<td>0.03</td>
<td>0.03</td>
</tr>
<tr>
<td>Ingest.Text</td>
<td>10.00</td>
<td>0</td>
<td>4.80</td>
<td>4.80</td>
</tr>
<tr>
<td>Ingest.dhr</td>
<td>20.00</td>
<td>0</td>
<td>0.40</td>
<td>0.40</td>
</tr>
<tr>
<td>Ingest.dpa</td>
<td>20.00</td>
<td>0</td>
<td>0.15</td>
<td>0.15</td>
</tr>
<tr>
<td>Ingest.lsr</td>
<td>80.01</td>
<td>0</td>
<td>0.01</td>
<td>0.01</td>
</tr>
<tr>
<td>Ingest.sfcobs</td>
<td>30.00</td>
<td>0</td>
<td>0.07</td>
<td>0.07</td>
</tr>
<tr>
<td>Ingest.Grib</td>
<td>10.00</td>
<td>0</td>
<td>5.60</td>
<td>5.60</td>
</tr>
<tr>
<td>Ingest.Radar</td>
<td>10.00</td>
<td>0</td>
<td>0.50</td>
<td>0.50</td>
</tr>
<tr>
<td>Ingest.RadarRadarServer</td>
<td>10.00</td>
<td>0</td>
<td>1.80</td>
<td>1.80</td>
</tr>
<tr>
<td>Ingest.Satellite</td>
<td>10.00</td>
<td>0</td>
<td>0.10</td>
<td>0.10</td>
</tr>
<tr>
<td>Ingest.Text</td>
<td>10.00</td>
<td>0</td>
<td>4.20</td>
<td>4.20</td>
</tr>
<tr>
<td>Ingest.acars</td>
<td>60.01</td>
<td>0</td>
<td>0.05</td>
<td>0.05</td>
</tr>
<tr>
<td>Ingest.dhr</td>
<td>10.00</td>
<td>0</td>
<td>0.10</td>
<td>0.10</td>
</tr>
<tr>
<td>Ingest.dpa</td>
<td>10.00</td>
<td>0</td>
<td>0.10</td>
<td>0.10</td>
</tr>
<tr>
<td>Ingest.obs</td>
<td>20.00</td>
<td>0</td>
<td>0.40</td>
<td>0.40</td>
</tr>
<tr>
<td>Ingest.redbook</td>
<td>180.02</td>
<td>0</td>
<td>0.01</td>
<td>0.01</td>
</tr>
<tr>
<td>Ingest.taf</td>
<td>40.01</td>
<td>0</td>
<td>0.02</td>
<td>0.02</td>
</tr>
<tr>
<td>Ingest.Grib</td>
<td>10.00</td>
<td>4</td>
<td>14.00</td>
<td>13.60</td>
</tr>
<tr>
<td>Ingest.Radar</td>
<td>10.00</td>
<td>0</td>
<td>1.60</td>
<td>1.60</td>
</tr>
<tr>
<td>Ingest.RadarRadarServer</td>
<td>10.00</td>
<td>0</td>
<td>0.60</td>
<td>0.60</td>
</tr>
<tr>
<td>Ingest.Satellite</td>
<td>10.00</td>
<td>0</td>
<td>0.30</td>
<td>0.30</td>
</tr>
<tr>
<td>Ingest.Shef</td>
<td>20.00</td>
<td>0</td>
<td>0.05</td>
<td>0.05</td>
</tr>
<tr>
<td>Ingest.Text</td>
<td>10.00</td>
<td>0</td>
<td>8.50</td>
<td>8.50</td>
</tr>
<tr>
<td>Ingest.Warning</td>
<td>120.01</td>
<td>0</td>
<td>0.01</td>
<td>0.01</td>
</tr>
<tr>
<td>Ingest.dhr</td>
<td>10.00</td>
<td>0</td>
<td>0.60</td>
<td>0.60</td>
</tr>
</tbody>
</table>

Exhibit 25.8.3.1-2. Typical Output of qpid-stat Monitoring Ingest Queues
The *qpid-queue-stats* tool can also be used to provide real-time monitoring of individual queue status. *qpid-queue-stats* provides some filtering; unlike *watch*, the new output is appended to the screen each time the statistics are updated. No sorting is provided; the output lines are shorter, however. The filtering mechanism is for specific queues; regex pattern matching is supported. To use *qpid-queue-stats* to display Ingest queue statistics, the basic command is `./qpid-queue-stats -f Ingest.*`.

The full procedure:

```
$ ssh awips@cpsbn1
awips@cpsbn1’s password:
$ cd /awips2/python/bin
$ ./qpid-queue-stats -f Ingest.*
<<results of qpid-queue-stats are displayed>>
<ctrl-C>
```

Typical output is shown in Exhibit 25.8.3.1-3.

<table>
<thead>
<tr>
<th>Queue Name</th>
<th>Sec</th>
<th>Depth</th>
<th>Enq Rate</th>
<th>Deq Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ingest.Grib</td>
<td>30.00</td>
<td>0</td>
<td>0.40</td>
<td>0.40</td>
</tr>
<tr>
<td>Ingest.Radar</td>
<td>10.00</td>
<td>0</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>Ingest.RadarRadarServer</td>
<td>10.00</td>
<td>0</td>
<td>2.50</td>
<td>2.50</td>
</tr>
<tr>
<td>Ingest.Satellite</td>
<td>10.00</td>
<td>0</td>
<td>0.20</td>
<td>0.20</td>
</tr>
<tr>
<td>Ingest.Shef</td>
<td>30.00</td>
<td>0</td>
<td>0.03</td>
<td>0.03</td>
</tr>
<tr>
<td>Ingest.Text</td>
<td>10.00</td>
<td>0</td>
<td>4.80</td>
<td>4.80</td>
</tr>
<tr>
<td>Ingest.dhr</td>
<td>20.00</td>
<td>0</td>
<td>0.40</td>
<td>0.40</td>
</tr>
<tr>
<td>Ingest.dpa</td>
<td>20.00</td>
<td>0</td>
<td>0.15</td>
<td>0.15</td>
</tr>
<tr>
<td>Ingest.lsr</td>
<td>80.01</td>
<td>0</td>
<td>0.01</td>
<td>0.01</td>
</tr>
<tr>
<td>Ingest.sfcobs</td>
<td>30.00</td>
<td>0</td>
<td>0.07</td>
<td>0.07</td>
</tr>
<tr>
<td>Ingest.Grib</td>
<td>10.00</td>
<td>0</td>
<td>5.60</td>
<td>5.60</td>
</tr>
<tr>
<td>Ingest.Radar</td>
<td>10.00</td>
<td>0</td>
<td>0.50</td>
<td>0.50</td>
</tr>
<tr>
<td>Ingest.RadarRadarServer</td>
<td>10.00</td>
<td>0</td>
<td>1.80</td>
<td>1.80</td>
</tr>
<tr>
<td>Ingest.Satellite</td>
<td>10.00</td>
<td>0</td>
<td>0.10</td>
<td>0.10</td>
</tr>
<tr>
<td>Ingest.Text</td>
<td>10.00</td>
<td>0</td>
<td>4.20</td>
<td>4.20</td>
</tr>
<tr>
<td>Ingest.acars</td>
<td>60.01</td>
<td>0</td>
<td>0.05</td>
<td>0.05</td>
</tr>
<tr>
<td>Ingest.dhr</td>
<td>10.00</td>
<td>0</td>
<td>0.10</td>
<td>0.10</td>
</tr>
<tr>
<td>Ingest.dpa</td>
<td>10.00</td>
<td>0</td>
<td>0.10</td>
<td>0.10</td>
</tr>
<tr>
<td>Ingest.obs</td>
<td>20.00</td>
<td>0</td>
<td>0.40</td>
<td>0.40</td>
</tr>
<tr>
<td>Ingest.redbook</td>
<td>180.02</td>
<td>0</td>
<td>0.01</td>
<td>0.01</td>
</tr>
<tr>
<td>Ingest.taf</td>
<td>40.01</td>
<td>0</td>
<td>0.02</td>
<td>0.02</td>
</tr>
<tr>
<td>Ingest.Grib</td>
<td>10.00</td>
<td>4</td>
<td>14.00</td>
<td>13.60</td>
</tr>
<tr>
<td>Ingest.Radar</td>
<td>10.00</td>
<td>0</td>
<td>1.60</td>
<td>1.60</td>
</tr>
<tr>
<td>Ingest.RadarRadarServer</td>
<td>10.00</td>
<td>0</td>
<td>0.60</td>
<td>0.60</td>
</tr>
<tr>
<td>Ingest.Satellite</td>
<td>10.00</td>
<td>0</td>
<td>0.30</td>
<td>0.30</td>
</tr>
<tr>
<td>Ingest.Shef</td>
<td>20.00</td>
<td>0</td>
<td>0.05</td>
<td>0.05</td>
</tr>
<tr>
<td>Ingest.Text</td>
<td>10.00</td>
<td>0</td>
<td>8.50</td>
<td>8.50</td>
</tr>
<tr>
<td>Ingest.Warning</td>
<td>120.01</td>
<td>0</td>
<td>0.01</td>
<td>0.01</td>
</tr>
</tbody>
</table>

Exhibit 25.8.3.1-3. Typical Output of qpid-queue-stats Monitoring Ingest Queues

One comment here: These two tools may produce different results. In particular, *qpid-stat* tends to display more Ingest queues than *qpid-queue-stats*. 
25.8.4 Monitoring IP Virtual Server (IPVS)

The IP Virtual Server (IPVS) is used to load balance the connections from clients (AlertViz, CAVE, etc.) to an EDEX cluster, as shown in Exhibit 25.8.4-1. IPVS is hosted on the CP 1/2 cluster. It is configured so that when an incoming connection request is made to the IPVS IP address, it looks to see which server has the fewest number of connections. It then routes the connection to the server with fewer active connections. If both servers have the same number of active connections, it is configured with DX4 weighted higher than DX3. This means that it will route to DX4 by default if both servers have the same number of active connections.

Exhibit 25.8.4-1. Client Interaction with CAVE via IPVS

For clients such as CAVE that interact with the EDEX Request process, IPVS allows a single connection string to be used regardless of the number of EDEX servers that are available. CAVE connects to a virtual address that is managed by IPVS on the CP1/2 cluster, shown here as CP1. (The virtual address used by CAVE is TS1.) Once the request is received, IPVS performs a handoff of one of the EDEX request servers, which fulfills the request. This results in a balancing of requests between DX3 and DX4.

IPVS is a standard part of the Red Hat Linux kernel that is installed on CP1/2.

On the clustered server, IPVS is managed by a number of Linux services and tools.

- **pulse** is a Linux heartbeat daemon for monitoring the health of cluster nodes. On a server that is part of a cluster, service pulse status can be used to determine if pulse is running; top may be used to monitor pulse.
- **lvsd** is the Linux daemon for controlling Red Hat clustering services. It is started by **pulse**. **lvsd** operation may be verified using **ps** and monitored using **top**.

- **nanny** is a Linux tool that monitors the status of services in a cluster. It is used by **pulse** and should not be used directly. **nanny** operation may be verified using **ps** and monitored using **top**.

- **ipvsadm** is the Linux Virtual Server administration management tool. It is used to manage the virtual server table and include some diagnostic/monitoring tools as well. **ipvsadm** must be run as root. When using **ipvsadm** for monitoring, use extreme care in the commands you execute.

The basic interaction of these tools is shown in Exhibit 25.8.4-2.

![Exhibit 25.8.4-2. Basic Hierarchy of Tools in IPVS](image)

**Example 25.8.4-1: Verify/Monitor pulse Operation**

**pulse** is the top-level tool involved in IPVS; it is a Linux daemon that is responsible for managing the cluster. Because IPVS is part of the CP1/2 cluster, **pulse** is ultimately responsible for its execution. If **pulse** is not running on all members of a cluster, cluster failover and other activities will fail. This recipe contains two scenarios: 1) verifying **pulse** execution; and 2) monitoring **pulse** execution using **top**.

**Scenario 1: Verifying Pulse Execution**

**pulse** is a Linux service whose operation may be controlled using the Linux **service** utility. Execution may be verified using **service pulse status**.

The basic steps to follow:

```
$ ssh root@cpsbn1
root@cpsbn1’s password:
$ service pulse status
pulse (pid 27173) is running...
$`

You should see output similar to the sample output above. (The actual PID will be different, of course.)

**Scenario 2: Monitoring Pulse Execution Using top**

The Linux `top` utility provides for a near real-time picture of computer operation; the `-p` flag is used to limit the processing being watched. The syntax is `top -p <PID list>`.

The basic steps follow, and the `top` output monitoring `pulse` is shown in Exhibit 25.8.4-3.

```
$ ssh root@cpsbnx1
root@cpsbn1’s password:
$ service pulse status
pulse (pid 27173) is running...
$ top -p 27173
(Typical top output is shown below, press q to quit top.)
```

![Top Output Monitoring pulse](image)

**Exhibit 25.8.4-3. Top Output Monitoring pulse**

**Example 25.8.4-2: Verify/Monitor lvsd Operation**

`lvsd` is the Linux daemon that is responsible for controlling the clustering. As such, it is part of the chain of tools responsible for the operation of IPVS. As with `pulse`, `lvsd` should be running on all members of a cluster to ensure proper operation of IPVS on the cluster. This Example contains two scenarios: 1) verifying `lvsd` execution; and 2) monitoring `lvsd` execution using `top`.

**Scenario 1: Verifying lvsd Operation**

Since `lvsd` is controlled by `pulse`, we use the Linux `ps` tool to verify that it is executing.

The basic steps to follow:

```
$ ssh root@cpsbn1
root@cpsbn1’s password:
$ ps ax -o pid,args | grep lvsd | grep -v grep
22787 /usr/sbin/lvsd --nofork -c /etc/sysconf/ha/lvs.cf
```

$
The first entry in the line is the `lvsd` process ID (PID); it will vary depending on the actual PID. In normal operation, you should expect to see a single instance of `lvsd`.

**Scenario 2: Monitoring lvsd Operation Using top**

As with other processes, `lvsd` may be monitored using `top`.

The basic steps follow, and the `top` output monitoring `lvsd` is shown in Exhibit 25.8.4-4.

```plaintext
$ ssh root@cpsbn1
root@cpsbn1's password:
$ ps ax -o pid,args | grep lvsd | grep -v grep
22787 /usr/sbin/lvsd --nofork -c /etc/sysconf/ha/lvs.cf
$ top -p 22787
(Typical top output is shown below, press q to quit top.)
```

![Exhibit 25.8.4-4. Top Output Monitoring lvsd](image)

**Example 25.8.4-3: Verify/Monitor nanny Operation**

`nanny` is a Linux tool that is used to monitor the status of services in a cluster. As such, it is used to monitor IPVS to ensure continued operation. Note that `nanny` is started by `lvsd`; as with `lvsd`, `nanny` should be running on all members of a cluster to ensure proper operation of IPVS on the cluster. This recipe contains two scenarios: 1) verifying `nanny` execution; and 2) monitoring `nanny` operation using `top`.

**Scenario 1: Verifying nanny Operation**

As with `lvsd`, `nanny` runs as a child process of `pulse`; as a result `nanny` operation is checked using the Linux `ps` tool.

The basic steps to follow:

```plaintext
$ ssh root@cpsbn1
root@cpsbn1’s password:
$ ps ax -o pid,args | grep nanny | grep -v grep
27213 /usr/sbin/nanny -c -h 155.157.61.161 -p 9581 -s GET /
HTTP/1.0\r\n\r
-x HTTP -a 15 -I /sbin/ipvsadm -t 6 -w l -V
155.157.61.165 -M g -U none --lvs
```
You should see a single line of output for each virtual host. The general format of each line of output is

```
PID /usr/sbin/nanny -c -h <address> -p <port> -s <ping> -x <response> -a <time> -I <ipvsadm> -t <time> -w weight -V <address> -M method -U none -lvs
```

The key values are the two addresses in each line. The first address is the IP for a DX server (DX3 or DX4); the second address is the virtual IP address for the IPVS server.

**TIP:** The output from `ps` for `nanny` is extremely long; `cut` can be used to shorten the output of the `grep` output, making the result easier to read. For example, piping the final `grep` in the example above into `cut -d ' ' -f -7` results in

```
27213 /usr/sbin/nanny -c -h 155.157.61.161 -p 9581
27214 /usr/sbin/nanny -c -h 155.157.61.162 -p 9581
```

**Scenario 2: Monitoring nanny Operation Using top**

The Linux `top` utility is used to monitor `nanny` operation; in this case there are multiple PIDs to monitor. Note the format of the `top` command.

The basic steps to follow and the `top` output monitoring `nanny` is shown in Exhibit 25.8.4-5.

```
$ ssh root@cpsbn1
root@cpsbn1's password:
$ ps ax -o pid,args | grep nanny | grep -v grep | cut -d ' ' -f -7
27213 /usr/sbin/nanny -c -h 155.157.61.161 -p 9581
27214 /usr/sbin/nanny -c -h 155.157.61.162 -p 9581
$ top -p 27213, 27214
(Typical top output is shown below, press q to quit top.)
```

![Top Output Monitoring nanny](image)
Example 25.8.4-4: Monitoring IPVS Using the ipvsadm Tool

`ipvsadm` is the Linux Virtual Server tool. It is used by `lvsd` to assist in managing the virtual servers in a cluster. `ipvsadm` is an extremely powerful tool and must be used with care; it is available only by the root user.

That said, `ipvsadm` does include some options that provide useful diagnostic information.

- `ipvsadm -L [-t <service>]` lists the Virtual Server Table, optionally for the specified service.
- `ipvsadm -L -c` lists current connection output including client system, virtual host and destination.
- `ipvsadm -L -stats` displays the statistics information of services and their servers.
- `ipvsadm -L -rate` displays the rate information of services and their servers (use –n to get numeric output of servers/services).

**CAUTION:** Use extreme care when using `ipvsadm`; several of the options available in `ipvsadm` perform modifications on the IPVS Virtual Server Table. Modifying the Virtual Server Table may have an adverse impact on system operation.

The IP aliasing information used by IPVS is contained in the Virtual Server Table. The Virtual Server Table identifies the aliased server and the real servers available via that alias. It also identifies the weighting scheme used for distributing the requests between the servers and connection information. `ipvsadm` is the tool used to display and manipulate the Virtual Server Table.

To display Virtual Server Table for IPVS:

```
$ ssh root@cpsbn1
root@cpsbn1’s password:
$ ipvsadm -L
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
-> RemoteAddress:Port Forward Weight ActiveConn
InActConn
TCP edexcluster-tbdw:9581 lc
  -> dx4-tbdw:9581 Route 2 2 29
  -> dx3-tbdw:9581 Route 1 2 29 $
```

In this display, captured on a typical AWIPS II installation, we see that there are two servers, DX3 and DX4. The connection port for all three servers is 9581. The weighting scheme for the two servers is 2:1 in favor of DX4. This means that, when both servers are available to handle requests, DX4 will get a new request.

**TIP:** To see a more dynamic view of the Virtual Server Table, use the Linux `watch` command to run `ipvsadm –L`. 
As shown here, `ipvsadm` defaults to showing server information using DNS names; to display IP addresses, add an “-n” flag to the `ipvsadm` command.

```bash
$ ssh root@cpsbn1
root@cpsbn1’s password:
$ ipvsadm -L -n
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
-> RemoteAddress:Port Forward Weight ActiveConn InActConn
TCP  155.157.61.165:9581 lc
   -> 155.157.61.162:9581 Route 2 0 0
   -> 155.157.61.161:9581 Route 1 0 0
```

**NOTE:** The same display, showing the IP addresses of the servers, is available via service `ipvsadm` status; either form may be used. In both cases, you must be logged on as the root user to run the command.

The `ipvsadm` tool can be used to display a snapshot of the connection information between clients and servers. The command to use is `ipvsadm -L -c`; when used with the `watch` command, this provides a near real-time tracking of server routing, as shown in Exhibit 25.8.4-6.

```
IPVS connection entries
pro expire state source virtual
destination
TCP 10:26 ESTABLISHED xt5-tbw3:57414 edexcluster-tbw3:9581 dx4-tbw3:9581
TCP 10:26 ESTABLISHED xt3-tbw3:42347 edexcluster-tbw3:9581 dx4-tbw3:9581
```

**Exhibit 25.8.4-6. Monitoring IPVS Connections Using watch**

To display the IPVS connection information in near real-time:

```bash
$ ssh root@cpsbn1
root@cpsbn1’s password:
$ watch ipvsadm -L -c
```

*(Typical output is displayed below; use <Ctrl-C> to terminate)*

In this display, which was captured on an AWIPS II installation, we see that three clients (lx2, lx4, and px1) are connected to DX3 (dx3-tbdw) and DX4 (dx4-tbdw) via the edexcluster-tbdw:9581.

The `ipvsadm` tool can be used to display statistics about the virtual server, including connections and packet and byte statistics, as shown in Exhibit 25.8.4-7. The command to use is `ipvsadm -L --stats` (note the double dash). As is the case with other `ipvsadm` options, `watch` can be used to provide a near real-time view of the system performance.
Exhibit 25.8.4-7. Monitoring IPVS Connection Statistics Using `watch`

To display IPVS connection statistics in near real-time:

```
$ ssh root@cpsbn1
root@cpsbn1’s password:
$ watch ipvsadm -L --stats
```

(Typical output is displayed below; use <Ctrl-C> to terminate)

In this display, which was captured on a typical AWIPS II installation, we see that the two servers, DX3 (dx3-oma) and DX4 (dx4-oma), are carrying a roughly equal load of connections, packets, and bytes. The actual percentages are:

<table>
<thead>
<tr>
<th>Server</th>
<th>Connections</th>
<th>Packets</th>
<th>Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>DX3</td>
<td>47%</td>
<td>48%</td>
<td>47%</td>
</tr>
<tr>
<td>DX4</td>
<td>53%</td>
<td>52%</td>
<td>53%</td>
</tr>
</tbody>
</table>

**NOTE:** IPVS is not handling outbound traffic so the zeros for OutPkts and OutBytes are normal.

To display IPVS rate statistics in near-real time:

```
$ ssh root@cpsbn1
root@cpsbn1’s password:
$ watch ipvsadm -L --rate
```

(Typical output is displayed below; use <Ctrl-C> to terminate)

The results are displayed in Exhibit 25.8.4-8.
In this display, captured on a typical AWIPS II installation, we see that the two servers, DX3 (dx3-oma) and DX4 (dx4-oma), are handling client requests. DX4 is averaging 60 packets per second (inbound) and 3,479 bytes per second (inbound).

**NOTE:** IPVS is not handling outbound, so the zeros for OutPPS and OutBPS are normal. Connections per second (CPS) also tends to be 0 because CAVE requests are generally infrequent.

### 25.8.5 Monitoring the LDM

The Local Data Management (LDM) is the system used by AWIPS II to transfer data from the SBN to the EDEX ingest processes. (See section 25.3, Basic Data Flow Concepts.) The LDM is covered extensively in other documents; this section presents basic recipes for monitoring the LDM.

There is only one LDM server on SBN CP. It ingests everything from the SBN. The LDM behavior is controlled by the pqact.conf file. The patterns are matched against pqact.conf. LDM writes the product to data_store and posts a QPID message.

The LDM system includes `ldmadmin`, a general administration tool used to start and stop the LDM; it also provides some of the monitoring capabilities used in these examples.

**Example 25.8.5-1: Verifying LDM Operation on the CP Cluster**

Although `ldmadmin` provides a tool for determining if the LDM is running, it can be easier to use the Linux `ps` utility to verify operation on the CP cluster.

**Scenario 1: Verify LDM Operation Using the Linux ps Utility**

```
$ ssh ldm@cpsbn1f
ldm@cpsbn1f’s password:
$ ps -u ldm -o user,pid,args
```
USER       PID       COMMAND
ldm        30195    ldm -l 0.0.0.0 -P 388 -M 256 -m 3600 -o 3600 -q /usr/local/ldm
ldm        30197    pqact -e
ldm        30198    edexBridge -s cp1f
ldm        30195    ldm -l 0.0.0.0 -P 388 -M 256 -m 3600 -o 3600 -q /usr/local/ldm
ldm        30197    pqact -e
ldm        30198    edexBridge -s cp1f

The exact output will vary. Note, however, that the critical lines are those with arguments rpc.ldmd, pqact, and edexBridge. These indicate that the various parts of the LDM are operating.

**Scenario 2: Verify LDM Using the ldadmin Program**

The ldadmin program’s `isrunning` option checks to see if the LDM is running. Unfortunately, this option simply returns 0 or 1 and exits; 0 is returned if the LDM is running, and a 1 is returned if it is not running. After executing `ldadmin isrunning`, you need to check its return value. This sequence should work under most shells:

```
$ ssh ldm@cpsbn1f
ldm@cpsbn1f’s password:
$ ldadmin isrunning
$ echo $? 
0
```

This output indicates that the LDM is running. If it were not running, the `echo $?` command would print a 1.

If you are using the `bash` shell, you can combine things slightly and produce a more readable result. The modified command sequence:

```
$ ssh ldm@cpsbn1f
ldm@cpsbn1f’s password:
$ ldadmin isrunning > /dev/null 2>&1 && echo "Running"
Running
```

Once again, this output indicates that the LDM is running; if it were not, no output would be displayed.

**Example 25.8.5-2: Monitoring LDM Data Flow on the CP Cluster**

The ldadmin program provides for real-time monitoring of the LDM product queue. This can be used to determine quickly if data is passing through the LDM; it can also be filtered by data feed to concentrate on a specific data type. Three scenarios are presented: 1) basic product queue monitoring; 2) identifying data feeds; and 3) monitoring a specific data feed.
Scenario 1: Basic Product Queue Monitoring

The ldmadmin program’s watch option provides a real-time view of the LDM’s message queue. Monitoring this queue provides a quick verification of the status of the LDM. The basic steps to follow:

$ ssh ldm@cpsbn1f
ldm@cpsbn1f’s password:
$ ldmadmin watch
(type ^D when finished)

At this point, you should see output similar to

```
Apr 22 21:51:11 pguilt INFO: 5160 20120422215109.997 NEXRAD3
149811050 SDUS53 KLOT 222141 /pDVLLOT !nids/
Apr 22 21:51:11 pguilt INFO: 1011 20120422215109.997 NEXRAD3
149811051 SDUS53 KILX 222142 /pDPAILX
Apr 22 21:51:11 pguilt INFO: 165 20120422215109.997 NEXRAD3
149811052 SDUS35 KPSR 222148 /pNMDPHX !nids/
Apr 22 21:51:11 pguilt INFO: 6764 20120422215109.997 NEXRAD3
149811053 SDUS23 KILX 222142 /pN3QILX !nids/
Apr 22 21:51:11 pguilt INFO: 18621 20120422215109.997 NEXRAD3
149811054 SDUS25 KRIW 222148 /pN3URIW !nids/
Apr 22 21:51:11 pguilt INFO: 97 20120422215110.999 IDS|DDPLUS
149811055 SPUS31 KKCI 222147
Apr 22 21:51:11 pguilt INFO: 117 20120422215110.999 IDS|DDPLUS
149811056 SAMH31 PKMJ 222200
Apr 22 21:51:11 pguilt INFO: 106 20120422215110.999 IDS|DDPLUS
149811057 SALV31 EVRA 222150
Apr 22 21:51:11 pguilt INFO: 149 20120422215110.999 HDS
149811058 NXUS62 KMFL 222150 /pGSMMIA
Apr 22 21:51:11 pguilt INFO: 134 20120422215110.999 IDS|DDPLUS
149811059 SAFA31 EKCH
```

If data is flowing, this output will scroll past fairly rapidly and the message date time stamp should be current. If no messages are scrolling, it is likely that there is an issue “upstream” from the LDM.

Scenario 2: Identifying LDM Data Feeds

The ldmadmin program’s watch facility has an option to filter output for a specific feed type. The basic command is ldmadmin watch -f <feed>. This scenario presents a method for identifying the available feeds.

The feeds used by LDM are defined in a file named pqact.conf, which is located in <ldm-home>/etc. While this file can be examined to determine the available feed
types, the process can be automated using the known structure of the file and a few basic Linux utilities. The lines defining LDM feeds have the following structure:

```
FEEDTYPE <tab> pattern <tab> action [<tab> options] [<tab> args]
```

Note that lines defining feed types start with at least one upper-case letter. This allows you to filter for the feed definitions and extract the feed names. The steps to follow:

```
$ ssh ldm@cpsbn1
ldm@cpsbn1’s password:
$ cd /etc
$ grep -P "^[A-Z]+" pqact.conf | cut -f 1 > feeds.txt
$ sort feeds.txt > sorted-feeds.txt
$ uniq sorted-feeds.txt ldm-feeds.txt
$ cat ldm-feeds.txt
```

ANY
GPSSRC
GRID
NGRID|HDS
HDS
HRS
IDS|DDPLUS
NEXRAD2
NIMAGE
NNEXRAD
$

The final output is a sorted list of feed names. The list may differ somewhat depending on the exact data flow configuration for the server. Some of the feed type names are rather cryptic; common feed types are identified in Table 25.8.5-1.

### Table 25.8.5-1. Selected LDM Feed Types

<table>
<thead>
<tr>
<th>Feed Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANY</td>
<td>wildcard – matches any feed type</td>
</tr>
<tr>
<td>GRID</td>
<td>NOAAport high-resolution model output</td>
</tr>
<tr>
<td>NEXRAD2</td>
<td>NEXRAD Level-II radar data</td>
</tr>
<tr>
<td>NIMAGE</td>
<td>NOAAport satellite imagery</td>
</tr>
<tr>
<td>NNEXRAD</td>
<td>NEXRAD Level-III products</td>
</tr>
</tbody>
</table>

### Scenario 3: Monitoring the Radar Data Feed

The feed type names determined in Scenario 3 may be used to filter the output of the `ldmadmin` program’s `watch` facility. For level III radar, the feed type name is `NNEXRAD`. The basic procedure:

```
$ ssh ldm@cpsbn1f
ldm@cpsbn1f’s password:
$ ldmadmin watch -f NNEXRAD
```
At this point you should see output similar to

Oct 25 18:13:44 pqutil INFO: 1300 20111025181343.559 NEXRAD3 467095897 SDUS33 KBIS 251809 /pN1PBIS
Oct 25 18:13:44 pqutil INFO: 807 20111025181343.559 NEXRAD3 467095898 SDUS54 KTSA 251808 /pDPASRX
Oct 25 18:13:44 pqutil INFO: 6918 20111025181343.559 NEXRAD3 467095899 SDUS26 KOTX 251809 /pN1QOTX !nids/

As in scenario 1, the key is to look for recent message date/time stamps and a generally constant flow of messages. In this case, a lack of messages indicates that the specific feed is not getting any messages; if frequent products are expected, this indicates a problem “upstream” from the LDM.

**Example 25.8.5-3: Monitoring LDM Log on the CP Cluster**

The LDM writes a process message, i.e., a routing status message and errors, to a log. The `ldmadmin` program provides two facilities for checking the log – `log` and `tail`. **Log** allows the user to page through the current LDM log – similar to the Linux `more` utility. **Tail** accesses the current log in a manner similar to the Linux `tail -f` utility. The basic procedure:

```
$ ssh ldm@cpsbn1f
ldm@cpsbn1f’s password:
$ ldmadmin tail
```

At this point you should see output similar to

Aug 13 08:28:54 cpsbn1-tbw3 rpc.ldmd[12415] NOTE: Terminating process group
Aug 13 14:11:02 cpsbn1-tbw3 rpc.ldmd[11189] NOTE: Starting Up (version: 6.7.1; built: May 3 2011 22:54:00)
Aug 13 14:11:02 cpsbn1-tbw3 rpc.ldmd[11189] NOTE: Using local address 165.92.24.50:388
Aug 13 14:11:20 cpsbn1-tbw3 dx2-tbw3[11207] NOTE: Data-product with signature 57b6643d8b6432d5cd95f8144f6ef5e6 wasn't found in product-queue
Aug 13 14:11:20 cpsbn1-tbw3 dx2-tbw3(feed)[11207] NOTE: Starting Up(6.7.1/6): 20120813131119.276 TS_ENDT {{ANY, .".*"}}, SIG=57b6643d8b6432d5cd95f8144f6ef5e6, Alternate
25.8.6 Monitoring Radar Server

The AWIPS II system consolidated several legacy AWIPS I radar communication and management processes into a new application called “RadarServer.” The Radar Server (RCM) is an independent application that runs on the DX1/2 cluster. It provides a bridge between the Open Radar Products Generator (ORPG) and the EDEX Ingest processes.

Like EDEX, RCM is written in Java. As a result, most monitoring of RCM is similar to the techniques used to monitor the EDEX processes: basic Linux tools and JConsole.

**NOTE:** Radar data comes from two sources: 1) the SBN for nationally distributed data; and 2) the Radar Server for locally collected radar data, including routine data and one-time as well as multiple requests from network radars.

**Example 25.8.6-1: Verifying Radar Server Operation**

Unlike EDEX, the startup script for RCM does not include a status check option. Rather, the Linux `ps` utility is used to verify RCM execution. The basic procedure:

```bash
$ ssh awips@dx1
awips@dx1’s password:
$ ps ax -o args | grep rcm.mqsrvr | grep -v grep
/awips2/bin/java -cp
/awips2/rcm/data/config/res::/awips2/lib/activation-1.1.jar
(several lines of output have been deleted)
```

From this output, we can see that RCM is currently executing. If the Radar Server is not running, there will be no output.
NOTE: Recall that all Java programs show the Java VM as the first argument. The first \texttt{grep} is used to identify the Radar Server. The second \texttt{grep} filters out the first \texttt{grep} from the output.

TIP: Output can be simplified somewhat by piping the second \texttt{grep} into \texttt{wc -l}; in that case, the output should be 0 or 1: 0 if Radar Server is running; 1 otherwise.

If it is desirable to also identify the PID of the Java instance running RCM, add it to the output list in the \texttt{ps} call. With that modification, the procedure becomes:

\begin{verbatim}
$ ssh awips@dx1
awips@dx1's password:
$ ps ax -o pid,args | grep rcm.mqsrvr | grep -v grep
17883 /awips2/bin/java -cp
/awips2/rcm/data/config/res::/awips2/lib/activation-1.1.jar
(several lines of output have been deleted)
$
\end{verbatim}

**Example 25.8.6-2: Monitoring RCM Performance Using top**

As with the EDEX processes, the Linux \texttt{top} utility may be used to monitor RCM. (See Example 25.8.6-3 for additional details.) The basic procedure:

\begin{verbatim}
$ ssh awips@dx1
awips@dx1’s password:
$ ps ax -o pid,args | grep rcm.mqsrvr | grep -v grep
17883 /awips2/bin/java -cp
/awips2/rcm/data/config/res::/awips2/lib/activation-1.1.jar
(several lines of output have been deleted)
$ top -p 17883
\end{verbatim}

To exit \texttt{top}, press either \texttt{q} or \texttt{<ctrl-c>}. Typical output is shown in Exhibit 25.8.6-1.

![Exhibit 25.8.6-1. Using \texttt{top} to Monitor the Radar Server](image)
**What’s it mean?** Even with a reasonably full flow of ORPG products, CPU usage for RCM is normally low, typically no more than 1–2 percent; usage above that value should be investigated further.

**Example 25.8.6-3: Monitoring RCM Using JConsole**

Like EDEX, the Radar Server is a Java program; thus, JConsole may be used to monitor the Radar Server. The techniques are essentially the same as those used for monitoring EDEX. (See Example 25.7.2-1, Example 25.7.2-2, and Example 25.7.2-3 for more details on using JConsole.) This recipe covers the steps required to connect JConsole to the Radar Server. The basic steps follow, and Exhibit 25.8.6-2 displays the results.

```
$ ssh -X awips@dx1
awips@dx1’s password:
$ ps ax -o pid,args | grep rcm.mqsrvr | grep -v grep
 17883 /awips2/bin/java -cp
 /awips2/rcm/data/config/res:/awips2/lib/activation-1.1.jar
(some lines of output have been deleted)
$ /awips2/java/bin/jconsole 2823 &
[1]28181
$
```

![Exhibit 25.8.6-2. Using JConsole to Monitor Radar Server](image)

Exhibit 25.8.6-2. Using JConsole to Monitor Radar Server
NOTE: Radar Server does not support certain connectivity options of JConsole. As a result, only local connections to Radar Server are supported. In this context, “local” means logged onto the server as the user, normally awips, running Radar Server. This also means that we must first identify the PID of the Radar Server in order to monitor it. Once that is done, we can pass the PID to JConsole on the command line.

Example 25.8.6-4: Monitoring the Radar Server Log

Like the EDEX Processes, the Radar Server logs important events to a process log. The process logs are located on the DX 1/2 cluster in /RadarServer/data/logs. The Radar Server maintains a rolling set of logs; a new log is created daily. The current log is named radarserver.log; previous logs have the date of log appended. For example, the log for May 31, 2010 will have the name radarserver.log.2010-05-31.

The basic format of the entries in the Radar Server log is the same as those in the EDEX server logs; see section 25.6.4, Log Formats for more information on the EDEX log message formatting.

As with EDEX Process logs, the basic tools for monitoring Radar Server logs are `tail` and `grep`. This recipe presents two scenarios: 1) general log monitoring; and 2) filtered log monitoring.

**Scenario 1: Monitoring Radar Server Execution**

This scenario uses the Linux `tail` utility to provide a real-time view of events the Radar Server logs to its process log. Specific output is not displayed.

```
$ ssh -X awips@dx1
awips@dx1’s password:
$ cd /awips2/rcm/data/logs
$ tail -f radarserver.log
INFO 22:01:03,089 [TCM Read koax #2] RadarServer: koax: message code=143 size=1870 elev=2.4 sequence=375 vs=2012-04-22 21:58:52 #3
INFO 22:01:03,761 [TCM Read kakq #2] RadarServer: kakq: message code=94 size=8761 elev=5.1 sequence=4019 vs=2012-04-22 22:06:34 #47
INFO 22:01:03,802 [TCM Read kakq #2] RadarServer: Stored message in /data_store/radar/kakq/2/elev5_3/res1/level256/kakq.94.20120422_2206
```
INFO 22:01:03,845 [TCM Read kakq #2] RadarServer: kakq: message
code=143 size=120 elev=5.1 sequence=4057 vs=2012-04-22 22:06:34 #47

To stop monitoring, press <ctrl-c>.

**What to look for:** Generally, you will see several messages display per minute, most of them *INFO* messages. The message frequency depends on the number of ORPG connections and the operational mode of the radars, including the RPS list employed.

**Scenario 2: Monitoring for Radar Product Delivery**

When the Radar Server writes a file to disk and sends a message to EDEX (via QPID), it logs a message giving the location of the file. The format of this message is:

```
INFO <time> [thread] RadarServer: Stored message in <path>
```

We use this information to filter the output of *tail* to display only the “stored message” lines from the Radar Server log. The basic steps:

```
$ ssh -X awips@dx1
awips@dx1’s password:
$ cd /awips2/rcm/data/logs
$ tail -f radarserver.log | grep -P "^INFO.+Stored message"
```

(the display scrolls to display the latest messages as the Radar Server writes the messages to the log file)

To stop monitoring, press <ctrl-c>.

**What to look for:** Generally, you should expect to see several messages per minute being logged. If the logging stops for an extended period of time, there is a problem with the ingest path.

### 25.8.7 Monitoring EDEX Subscription Script Running

The EDEX Subscription Script Runner provides a mechanism for executing arbitrary scripts based on system-level events. Three event types are supported: Quartz timer-based events; general data arrival-based events; and text product arrival-based events. (For text products, the Script Runner functions similarly to the existing text database watch/warn triggers.)

The Script Runner supports three types of script execution: text data triggered (ldad) scripts; generic (system) scripts; and µEngine (python) scripts. Text data triggered scripts are programs that reside on the server and, when triggered, are passed the AFOS PIL of the ingested text data. Generic scripts are also programs that reside on the server. The difference is the arguments passed to the program; any appropriate arguments can be passed to a generic script. A µEngine script is a Python script that is stored in the database. It can access the full capabilities of the EDEX µEngine.

The Subscription Script Runner has three components: EDEX Subscription Service; EDEX Script Runner; and the Command Line Interface (CLI) tool package. The CLI is covered in more detail elsewhere; the recipes in this section address system diagnostics.
Example 25.8.7-1: Identifying Registered Subscriptions – Using psql

EDEX Script Runner Subscriptions are stored in the `subscriptions` table in the AWIPS II database; the `subscriptions` table is in the `subscription` schema of the `metadata` database. We can use the `psql` tool to check for subscriptions. (For more information on the AWIPS II database and the `psql` tool, see Examples 25.21-1 through 25.21-7.) This recipe presents four scenarios: 1) identifying available fields in the `subscription` table; 2) identifying available subscription trigger types, identifying timer-triggered subscriptions; and 4) identifying system runner subscriptions.

Scenario 1: Identifying Available Fields in the Subscription Table

This scenario provides information needed to access the subscription table in the AWIPS II database. In addition to the field names, we obtain the required connection/login information. The steps to follow:

```
$ ssh awips@dx3
awips@dx3’s password:
$ cd /awips2/edex/bin
$ grep -P "^export DB" start.sh
export DB_ADDR=dx1
export DB_PORT=5432
$ psql -h dx1 -p 5432 -d metadata -U awips
Password for user awips:

metadata=# select table_name,column_name from
metadata=# information_schema.columns
metadata=# where table_name = 'subscriptions';
      table_name     | column_name
-----------------+-----------------
    subscriptions | id
    subscriptions | active
    subscriptions | type
    subscriptions | runner
    subscriptions | trigger
    subscriptions | script
    subscriptions | filepath
    subscriptions | arguments
(8 rows)

metadata=# \q
$
```

See Table 25.8.7-1 for information on the fields of the `subscriptions` table.
Table 25.8.7-1. Subscriptions Table Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>(bigint) Unique ID of the subscription</td>
</tr>
<tr>
<td>Active</td>
<td>(boolean) True if the subscription is active – i.e., will fire</td>
</tr>
<tr>
<td>Type</td>
<td>(char 10) Type of trigger event (timer, data, ldad)</td>
</tr>
<tr>
<td></td>
<td>timer – event is triggered at a specific time</td>
</tr>
<tr>
<td></td>
<td>data – event is triggered by data ingest</td>
</tr>
<tr>
<td></td>
<td>ldad – event is triggered by text product ingest</td>
</tr>
<tr>
<td>Runner</td>
<td>(char 10) Type of script runner needed for the subscription</td>
</tr>
<tr>
<td></td>
<td>python – use Python uEngine</td>
</tr>
<tr>
<td></td>
<td>jscript – (deprecated) use JavaScript uEngine</td>
</tr>
<tr>
<td></td>
<td>system – run the script as a separate process</td>
</tr>
<tr>
<td></td>
<td>ldad – run the script using the text script runner</td>
</tr>
<tr>
<td>Trigger</td>
<td>(char 512) pattern to match to trigger the event</td>
</tr>
<tr>
<td>Script</td>
<td>(char 4048) script to run. Used only for python and jscript runners. Is encoded to preserve formatting for python.</td>
</tr>
<tr>
<td>Filepath</td>
<td>(char 512) full path to script to execute. Used only for ldad and system runners.</td>
</tr>
<tr>
<td>Arguments</td>
<td>(char 512) arguments to pass to the script. Used only for ldad and system runners. Optional. Special token, %TRIGGER%, specifying passing the trigger event to the script – used mainly for ldad runner.</td>
</tr>
</tbody>
</table>

**Scenario 2: Identifying Available Subscription Trigger Types**

The `type` field of the `subscriptions` table contains the subscript type. While valid subscription types are timer, ldad, and data, we can query the database to determine which types are currently registered. The steps to follow:

```
$ ssh awips@dx3
awips@dx3’s password:
$ psql -h dx1 -p 5432 -d metadata -U awips
Password for user awips:
metadata=# select distinct type from
metadata=# subscription.subscriptions
metadata=# order by type;
type
-------
timer
(1 row)
```

Metadata # \q
$

**Scenario 3: Identifying Timer-Triggered Subscriptions**

Once we know the available subscription types, we can easily query for subscriptions of a specific type. In this scenario, we query for `timer` type subscriptions. The steps to follow:

```
$ ssh awips@dx3
awips@dx3’s password:
$ psql -h dx1 -p 5432 -d metadata -U awips
Password for user awips:
metadata=# select distinct type from
metadata=# subscription.subscriptions
metadata=# order by type;
type
-------
timer
(1 row)
```
$ ssh awips@dx3
Awips@dx3’s password:
$ psql -h dx1 -p 5432 -d metadata -U awips
Password for user awips:
metadata=# select id,active,type,runner,trigger
metadata-# from subscription.subscriptions
metadata=# where type = 'timer';
<table>
<thead>
<tr>
<th>id</th>
<th>active</th>
<th>type</th>
<th>runner</th>
<th>trigger</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>t</td>
<td>timer</td>
<td>python</td>
<td>0 * * * * ?</td>
</tr>
<tr>
<td>6</td>
<td>t</td>
<td>timer</td>
<td>system</td>
<td>0 0/5 * * * ?</td>
</tr>
</tbody>
</table>
(2 rows)
metadata=# \q
$

The output shows the single timer subscription that is currently registered. Because the value of the active field is true, this script should execute each minute. Note that we did not print out the fields that identify the script being executed. For python runners, the script is stored in the database as a binary encoded string and is not readable; for the other runners it may be useful to identify the file path and arguments. This can be accomplished easily using a slightly more complicated SQL query. The steps to follow:

$ ssh awips@dx3
awips@dx3’s password:
$ psql -h dx1 -p 5432 -d metadata -U awips
Password for user awips:
metadata=# select type,runner,trigger,
metadata-# (case when char_length(script) > 1 then
metadata(# 'SCRIPT' else '' end) as script,
metadata-# filepath,arguments from
metadata-# subscription.subscriptions
metadata=# where type = 'timer';
<table>
<thead>
<tr>
<th>type</th>
<th>runner</th>
<th>trigger</th>
<th>script</th>
<th>filepath</th>
</tr>
</thead>
<tbody>
<tr>
<td>timer</td>
<td>python</td>
<td>0 * * * * ?</td>
<td>SCRIPT</td>
<td></td>
</tr>
<tr>
<td>timer</td>
<td>system</td>
<td>0 0/5 * * * ?</td>
<td>#TRIGGER%</td>
<td>/awips2/fxa/bin/myScript.sh</td>
</tr>
</tbody>
</table>
(2 rows)
metadata=# \q
$

$
To shorten the display, the id and active fields are not included in the query. The result, as displayed in an xTerm, is shown in Exhibit 25.8.7-1.

![Image of xTerm output](image)

### Exhibit 25.8.7-1. Sample Result of Timer Type Query

**NOTE:** To see the actual script, use `decode(script, 'hex') as script` in place of the `case` construct in the above query.

#### Scenario 4: Identifying System Runner Subscriptions

This scenario is nearly the same as Scenario 3; the main change is in the `where` clause. To identify a runner, the field to check is the `runner` field.

```sql
$ ssh awips@dx3
awips@dx3’s password:
$ psql -h dx1 -p 5432 -d metadata -U awips
Password for user awips:
metadata=# select distinct runner from metadata.subscription.subscriptions
metadata=# order by runner;
runner
--------
ldad
python
system
(3 rows)
```

```sql
metadata # select type, runner, trigger,
metadata # (case when char_length(script) > 1 then 'SCRIPT' else '' end) as script,
metadata # filepath,arguments from subscription.subscriptions where type = 'Timer';
```

```sql
timer | python | 0 * * * ? | SCRIPT |
timer | system | 0 9/5 * * * ? | /awips2/fxa/bin/myScript.sh | %TRIGGER%
(2 rows)
```

```sql
metadata-# select type, runner, trigger,
metadata-# (case when char_length(script) > 1 then 'SCRIPT' else '' end) as script,
metadata-# filepath,arguments from subscription.subscriptions where type = 'Timer';
```

```sql
metadata-# select type, runner, trigger,
metadata-# (case when char_length(script) > 1 then 'SCRIPT' else '' end) as script,
metadata-# filepath,arguments from subscription.subscriptions where type = 'Timer';
```
type | runner | trigger | script | filepath
| arguments

-------+--------+---------------+--------+--------------+-----------
timer | system | 0 0/5 * * * ? |        | /awips2/fxa/bin/myScript.sh | %TRIGGER%
(1 row)
metadata # \q
$

From this output, we see that: 1) there is one system runner script; 2) it fires every 5 minutes; and 3) it runs myScript.sh, passing the value of the trigger to the script. This output is shown again in Exhibit 25.8.7-2.

Exhibit 25.8.7-2. Sample Result of System Runner Query

Example 25.8.7-2: Identifying Registered Subscriptions – Using the CLI

The Command Line Interface (CLI) tools are a set of tools that provide basic capabilities for interaction with the EDEX server. The CLI tools are normally installed on DX 3/4 in /awips2/bin, and they are available for installation on workstations that also support CAVE. Of interest here is the subscription tool, which is a custom EDEX client for managing subscriptions. This recipe introduces the subscription tool and provides scenarios for determining available subscriptions. As an EDEX client, it does not interact directly with the database; rather, it interacts with the EDEX server and returns specifically formatted responses. If additional information is desired, the `psql` tool can be used to interface directly with the AWIPS II database. (See Example 25.8.7-1 for details.)

The subscription CLI tool uses a flag/value CLI for requesting information. The flags used for subscription query operations are shown in Table 25.8.7-2.
Table 25.8.7-2. Basic Flags for Subscription CLI Tool

<table>
<thead>
<tr>
<th>Flag</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-o</td>
<td>operation</td>
<td>Operation to perform. Valid values are read, add, delete, and update.</td>
</tr>
<tr>
<td>-p</td>
<td>trigger</td>
<td>The specific trigger for the subscription.</td>
</tr>
<tr>
<td>-t</td>
<td>type</td>
<td>The type of subscription trigger. Valid values are timer, data, and ldad.</td>
</tr>
<tr>
<td>-r</td>
<td>runner</td>
<td>The type of script runner for the subscription. Valid values are python, jscript, system, and ldad, although jscript is deprecated and should not be used.</td>
</tr>
</tbody>
</table>

For querying subscription information, the operation to use is read.

**NOTE:** The subscription tool uses a standard output format for output; the values of the id, active, type, runner, and trigger fields are displayed. The script is also displayed: for python runners, the hex encoded script is displayed; for system and ldad runners, the script’s path and arguments are displayed.

**Scenario 1: Identifying Timer-Triggered Subscriptions**

To determine the subscriptions of a specific trigger type, timer in this case, we use the -t with the appropriate subscription type name. The basic steps to follow:

```
$ ssh awips@dx3
awips@dx3’s password:
$ cd /awips2/bin
$ ./subscription -o read -t timer
```

```
ID     ACTIVE TYPE  RUNNER TRIGGER   SCRIPT
------ ------ ----- ------ --------- ------
3       True   timer python 0 * * * * ?
696d706f72742048656c66f576f726c640a72756e6e6572203d2048656c66f576f726c642e48656c66f576f726c6428290a72756e6e657222e7365744d6573736167652822222220254d455341474525202222222290a72657475726e2072756e6e657222e657865637574652829
6       True   timer system 0 0/5 * * * ? /awips2/fxa/bin/myScript.sh

%TRIGGER%
```

It is possible to filter this output to eliminate much of the HEX output. The trick is to use the Linux `cut` tool and specify the number of characters to retain. For example, to filter output and retain just the first 80 characters of each line, “pipe” the command output into `cut -b -80`. With this refinement, the command sequence becomes

```
$ ssh awips@dx3
awips@dx3’s password:
$ cd /awips2/bin
$ ./subscription -o read -t timer | cut -b -80
```

```
ID     ACTIVE TYPE  RUNNER TRIGGER   SCRIPT
------ ------ ----- ------ --------- ------
3       True   timer python 0 * * * * ?
696d706f72742048656c66f576f726c640a72756e6e6572203d2048656c66f576f726c642e48656c66f576f726c6428290a72756e6e657222e7365744d6573736167652822222220254d455341474525202222222290a72657475726e2072756e6e657222e657865637574652829
```
In either case, there are two timer-triggered scripts. The first one uses a stored Python script and triggers every minute. The second one runs myScript.sh every 5 minutes.

**Scenario 2: Identifying System Runner Subscriptions**

To identify subscriptions using a specific runner, in this case the system runner, use the -r flag with the appropriate runner name. In this case, we also use the Linux cut utility to limit the length of the display lines. The basic steps to follow:

```
$ ssh awips@dx3
awips@dx3’s password:
$ cd /awips2/bin
$ ./subscription -o read -r system | cut -b -80
```

```
ID     ACTIVE TYPE  RUNNER TRIGGER   SCRIPT
------ ------ ----- ------ --------- -----
6 True   timer system 0 0/5 * * * ? /awips2/fxa/bin/myScript.sh
%TRIGGER%
```

**NOTE:** In both Scenario 3 and Scenario 4, only a single selection criterion is provided. In this case, selection criteria are specified using the -t, -p, or -r flags. In general, multiple flags are permitted but they “anded” together. For example, substitution -o read -t timer -r system performs a query for subscriptions that are both timer triggered and use external scripts (i.e., system runner type). The only way to perform or queries is to run separate queries.

**Example 25.8.7-3: Identifying Text DB Triggers – Using the CLI**

Text DB triggers are executed using the EDEX Script Runner Service. Registered Text DB triggers are stored in the subscription table of the AWIPS II database. The AWIPS II CLI tool set provides a modified version of the textdb tool. The CLI textdb tool is a command line tool that supports the same functionality as the AWIPS I textdb tool. The CLI textdb tool allows a user to perform routine queries of the Text DB. It is also used to manage Text DB tools. This example presents two scenarios: 1) using the CLI subscription tool to identify Text DB triggers; and 2) using the CLI textdb tool to identify Text DB triggers.

**Scenario 1: Using the CLI Subscription Tool to Identify Text DB Triggers**

To use the subscription tool to identify registered Text DB triggers, you need to know either the runner type or the trigger type used for Text DB triggers. For Text DB triggers, both the runner and the trigger type are “ldad”. We can use this information to perform the query.

The basic steps to follow:

```
$ ssh awips@dx3
```
awips@dx3’s password:
$ cd /awips2/fxa/bin
$ ./subscription -o read -r ldad

ID     ACTIVE TYPE RUNNER TRIGGER     SCRIPT
------ ------ ----- ------ --------- -----
1179968075 True    ldad   ldad STLZFPNW1
1179968076 True    ldad   ldad SFOLSRSTO
1179968077 True    ldad   ldad SFOCEMSTO
1179968078 True    ldad   ldad WBCZFPLWX
1179968079 True    ldad   ldad STLNOWEAX

$ ./subscription -o read -t ldad

PRODUCTID SCRIPT
--------- -----
STLZFPNW1 /awips/fxa/bin/asyncPilTrigger.sh
SFOLSRSTO /awips/fxa/bin/asyncPilTrigger.sh
SFOCEMSTO /awips/fxa/bin/asyncPilTrigger.sh
WBCZFPLWX /awips/fxa/bin/asyncPilTrigger.sh
STLNOWEAX /awips/fxa/bin/asyncPilTrigger.sh
$

It is interesting that the results obtained are different. Although both queries show that there is a Text DB trigger registered for a product ID of “STLZFPNW1”, the rest of the information is radically different. The reason for this – the subscription tool generally uses the textdb tool to perform queries that it recognizes as Text DB related. A trigger type of ldad indicates the query is Text DB related.

**Scenario 2: Using the CLI textdb Tool to Identify Text DB Triggers**

In order to identify Text DB triggers using the CLI textdb tool, you need to know the Product ID used to register the trigger. The basic steps to follow:

$ ssh awips@dx3

awips@dx3’s password:
$ cd /awips2/bin
$ ./textdb -ldad -r OMAADMOMA

PRODUCTID SCRIPT
---------- -----
OMAADMOMA /awips2/fxa/bin/myScript.sh
$ ./textdb -ldad -r OMAMRTOMA

PRODUCTID SCRIPT
---------- -----
$

This result shows that there is a single trigger registered for Product ID “OMAADMOMA” and no triggers registered to Product ID “OMAMRTOMA.”
Example 25.8.7-4: Determining if Subscriptions Are Firing from the EDEX Process Logs

The subscription script runner is part of the EDEX Request and Ingest process (see Table 25.8.7-3). Where the script runner is located depends on the type of script runner.

Table 25.8.7-3. Script Runners by EDEX Process

<table>
<thead>
<tr>
<th>Runner Type</th>
<th>EDEX Process</th>
</tr>
</thead>
<tbody>
<tr>
<td>timer</td>
<td>Ingest</td>
</tr>
<tr>
<td>data</td>
<td>Ingest</td>
</tr>
<tr>
<td>ldad</td>
<td>Ingest and Request</td>
</tr>
</tbody>
</table>

Note that the ldad runner in the EDEX Request process only fires in response to a text product being inserted into the Text DB via the CLI textdb tool. Because of this, the main log to monitor is the EDEX Ingest process log.

Each time the subscription fires, it logs to the appropriate EDEX process log some basic information. Generally, when the subscription fires successfully, the log messages match this pattern:

```
INFO <date> <time> [thread] Scr
iptRunner: Executed script: runner= <runner>, script= <script> <trigger>
```

The script runner may output additional log messages. When script execution fails, the log messages match this pattern:

```
ERROR <date> <time> [thread] ScriptRunner: Encountered errors executing script: runner= <runner>, script= <script> <trigger>
```

This line will normally be followed by a Java Exception stack trace that may be used to help identify the specific problem. In addition, the script runner may output additional log messages. Note that this information may be combined into a single Perl 5 style regular expression:

```
^ERROR|INFO .+ ScriptRunner:
```

You can use this information to monitor the EDEX process log and verify subscription execution.

The basic steps to follow:

```
$ ssh awips@dx3
awips@dx3’s password:
$ cd /awips2/edex/logs
$ ls edex-*-20111023.log
edex-ingest-20111023.log
edex-ingestDat-20111023.log
edex-ingest-gen_areal_ffg-20111023.log
edex-ingest-gen_areal_qpe-20111023.log
edex-ingestGrib-20111023.log
edex-ingest-purge-20111023.log
edex-ingest-radar-20111023.log
edex-ingest-satellite-20111023.log
```
edex-ingest-shef-20111023.log
edex-ingest-shef-performance-20111023.log
edex-ingest-smartInit-20111023.log
edex-ingest-text-20111023.log
edex-ingest-trigger-20111023.log
edex-ingest-unrecognized-files-20111023.log
edex-request-20111023.log
edex-request-productSrvRequest-20111023.log
edex-request-thriftSrv-20111023.log
edex-ingest-archive-20111023.log
edex-request-GFEPerformance-20111023.log
edex-ingest-GFEPerformance-20111023.log

$ tail -f edex-ingest-20111023.log | grep -P "^ERROR|INFO .+
ScriptRunner: "
(as EDEX Ingest writes to the log file, lines identifying
successful ingest are displayed.)

To stop monitoring, press <ctrl-c>.

**NOTE:** The subscription script runners only log information for registered subscriptions
and Text DB triggers. A lack of log entries simply indicates that no events, i.e.,
product arrivals, matching registered subscriptions have been received. See
Example 25.8.7 for information on using JConsole to monitor script running.

**Example 25.8.7-5: Determining if Subscriptions Are Firing Using JConsole**

As discussed in Example 25.7.2-2 and Example 25.7.2-3, JConsole may be used to
monitor internal operations and/or states in an EDEX process. This includes monitoring
subscription script running. This is done by finding the appropriate Bean on JConsole’s
*MBeans* tab. The basic steps to follow are similar to those in either Example 25.7.2-2 or
25.7.2-3. Briefly, these are:

1. Execute JConsole.
2. Establish a connection to the appropriate EDEX process. Use either *<host-
   name>:1616* or *<host-name>:1617*.
3. Once the connection is complete, click the *MBeans* tab.
4. In the tree browser, expand the *org.apache.camel* node.
5. Under the *org.apache.camel* node, expand the *routes* node.
6. Scroll the tree browser to find the node labeled *<host-name>/script-runner-
camel*.
7. Expand this node.
8. At this point, you will see the available script runner beans. In the EDEX Request
   process, only a single bean is available – “ldadWatchWarnDirect”. In the Ingest
   process, there are three beans – dataArrival, ldadWatchWarn, and
runnerScheduled.
9. Expand the Bean node for the runner type you want to monitor.
10. Under the expanded Bean node, select attributes. This will produce a display similar to that shown in Exhibit 25.8.7-3.

Exhibit 25.8.7-3. Data Arrival Script Runner Bean Attributes

Of the attributes listed, the most interesting are the three “Exchanges” attributes, and the “Processing Time” attributes. These are generally self-explanatory, although the units are not. The processing time attributes are given in milliseconds.

Tip: You can generally double click on any numeric attribute to get a real-time graphical view of that attribute.

NOTE: The images in this recipe were captured from a development server (awips-dev1.oma.ray.com) running the latest development build of EDEX. On a production system, the computer name will change appropriately.

25.8.8 Monitoring EDEX Smart Init Processing

AWIPS II uses the EDEX Ingest process to perform Smart Init processing. As such, basic information on Smart Init execution is logged to the EDEX Ingest process log.

AWIPS II Smart Init processing utilizes an aggregation strategy within EDEX; as GRIB products are ingested within EDEX, product information is forwarded to the aggregation endpoint. EDEX uses a quartz trigger that fires every minute to trigger Smart Init processing.

In the EDEX Ingest process log, Smart Init entries follow a specific general pattern:

```xml
<level> <date> <time> [smartInitThreadPool-#] <source>: <message>
```
This pattern may be used to filter the EDEX Ingest process log for messages generated by Smart Init.

There are two ways to monitor basic Smart Init processing: monitoring the EDEX Ingest process log; and monitoring the EDEX Ingest process using JConsole. The recipes in this section provide information on these two monitoring methods.

**Example 25.8.8-1: Monitoring the EDEX Ingest Process Log for Smart Init Processing**

(This recipe is similar to Example 25.7.1-5, Monitoring Data Ingest.) We use the Linux `tail` tool to display the process log in real time; to isolate Smart Init processing, we use the Linux `grep` tool to filter and limit the output to Smart Init.

A Perl 5 regex that matches Smart Init log entries is:

```
^.+\[smartInit
```

The basic steps to follow:

```
$ ssh awips@dx3
awips@dx3's password:
$ cd /awips2/edex/logs
$ ls edex-20111023*.log
```

```
edex-ingest-20111023.log
edex-ingestDat-20111023.log
edex-ingest-gen_areal_ffg-20111023.log
edex-ingest-gen_areal_qpe-20111023.log
edex-ingestGrib-20111023.log
edex-ingest-purge-20111023.log
edex-ingest-radar-20111023.log
edex-ingest-satellite-20111023.log
edex-ingest-shef-20111023.log
```

```
edex-ingest-shef-performance-20111023.log
edex-ingest-smartInit-20111023.log
edex-ingest-text-20111023.log
edex-ingest-trigger-20111023.log
edex-ingest-unrecognized-files-20111023.log
edex-request-20111023.log
edex-request-productSrvRequest-20111023.log
edex-request-thriftSrv-20111023.log
edex-request-archive-20111023.log
```

```
edex-ingest-GFEPerformance-20111023.log
edex-ingest-activeTableChange-20111023.log
dex-ingestDat-activeTableChange-20111023.log
dex-ingestDat-performance-20111023.log
dex-ingestGrib-activeTableChange-20111023.log
dex-ingestGrib-performance-20111023.log
dex-ingest-performance-20111023.log
dex-request-activeTableChange-20111023.log
dex-request-performance-20111023.log
dex-ingest-gen_areal_qpe-20111023.log
dex-request-GFEPerformance-20111023.log
```

```
$ tail -f edex-ingest-20111023.log | grep -P "^.+\[smartInit"
```
(As EDEX ingest writes to the log file, lines identifying Smart Init operations are displayed.)

To stop monitoring, press <ctrl-C>.

In normal operation, you should expect to see a fairly steady stream of log messages.

**Example 25.8.8-2: Monitoring Smart Init Calculations (in the EDEX ingest process log)**

As discussed in Example 25.7.1-5, Monitoring Data Ingest, the output from the Linux `tail` utility may be filtered to isolate specific types of information. As a general rule, you need to examine the log for patterns that indicate specific types of information and adjust the filter pattern accordingly. In this recipe, we need to identify the basic pattern used for logging Smart Init calculations.

A Perl 5 regex that matches Smart Init calculation lines is `^\+[\w.]+Calc`:

The basic steps to follow:

```
$ ssh awips@dx3
awips@dx3's password:
$ cd /awips2/edex/logs
$ ls edex-\*20111023.log
```

**Note:** For this log file, for example, use `ls edex-\*20111023.log`
To stop monitoring, press <ctrl-C>.

In normal operation, you should expect to see a fairly steady stream of log messages.

**Example 25.8.8-3: Monitoring Smart Init Using JConsole**

To monitor Smart Init processing using JConsole, you generally need to identify two items – the name of the Camel Route and the name of the MBean (or MBeans) to monitor. (See Example 25.7.2-1, Example 25.7.2-2, and Example 25.7.2-3 for details on connecting to EDEX and finding the correct Camel Route and MBean.)

For Smart Init, the name of the Camel Route is `<host name>/gfe-camel-spring`. This route hosts several Beans that handle Smart Init capabilities; the two that we will examine are `smartInitTrigger` and `smartInitWork`. As the names imply, `smartInitTrigger` is the trigger that fires each minute to initiate processing, while `smartInitWork` is the service that runs the Smart Init scripts.

Follow the procedure in Example 25.7.2-1 to connect to the EDEX Ingest process on DX3. Once the connection is made, select the `MBeans` tab and find the `gfe-camel-spring` route in the tree explorer. Click to expand the `gfe-camel-spring` route. JConsole will look similar to Exhibit 25.8.8-1.

To monitor Smart Init trigger firing, expand the `smartInitTrigger` bean and click on `Attributes`. This will display the attributes, as shown in Exhibit 25.8.8-2.

As mentioned in Example 25.7.2-2 and Example 25.7.2-3, the interesting attributes are the “Exchanges” and “Processing Time” attributes. Remember that times are given in milliseconds.

To monitor the Smart Init processing, expand on the `smartInitWork` Bean and click on Attributes. This will display the attributes, as shown in Exhibit 25.8.8-3.

As before, the interesting attributes are the “Exchanges” and “Processing Time” attributes. Remember that times are given in milliseconds.

As mentioned previously, you can usually double click on a numeric attribute in the `attribute values` display panel to get an automatically updating graph of the attribute’s value. An example of this is shown in Exhibit 25.8.8-4.
Exhibit 25.8.8-1. gfe-camel-spring Route in JConsole

Exhibit 25.8.8-2. smartInitTrigger Bean Attributes in JConsole
Exhibit 25.8.8-3. smartInitWork Bean Attributes in Jconsole

Exhibit 25.8.8-4. JConsole Bean Attributes with Graphs

For both the smartInitTrigger bean and the smartInitWork bean, the number of Exchanges Completed should match the number of Exchanges Total. Ideally, the number of Exchanges Failed should be zero.
25.9 **HDF5 Tools**

The HDF5 tool set includes tools that allow examination, export, and modification of HDF5 files. These tools are installed into /awips2/tools on the database server DX2. The tools themselves are installed in /awips2/tools/bin; the supporting libraries are in /awips2/tools/lib.

### 25.9.1 HDF5 Tools - Procedures

The general procedure for running the HDF5 tools may be summarized as follows:

1. Log onto the database server (e.g., DX2).
2. Run the tool(s).

Four of the available HDF5 tools that are installed with AWIPS II in the examples that follow. These tools identified in Table 25.9.1-1.

<table>
<thead>
<tr>
<th>Tool</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>h5ls</td>
<td>Lists specified features of an HDF5 file. This tool is used to determine the contents of the HDF5 file</td>
</tr>
<tr>
<td>h5dump</td>
<td>Dumps the contents of an HDF5 file in ASCII format. This tool is used to examine the actual contents of the HDF5 file.</td>
</tr>
<tr>
<td>h5stat</td>
<td>Reports statistics regarding an HDF5 file and the objects stored in the file.</td>
</tr>
<tr>
<td>h52gif</td>
<td>Converts images in an HDF5 file to a GIF image.</td>
</tr>
</tbody>
</table>

For all remaining examples in this section, assume that some satellite images have been archived in an HDF5 file named satellite-1144554058.h5, which is located in /tmp.

**Example 25.9.1-1: The h5ls Tool**

The h5ls tool produces a list of specific features in an HDF5 file. The list is written to standard out and can be redirected into a text file for viewing. The basic command to use is:

```
h5ls -r file[/OBJECT]
```

Where: file is the path to the HDF5 file and OBJECT is the path within the HDF5 file. The “-r” flag indicates a recursive listing starting with the specified path within the file.

**Example 25.9.-2: The h5dump Tool**

The h5dump tool prints the contents of the HDF5 file to standard out in an ASCII format. The default format for the dump is the HDF Group’s HDF5 Data Description Language; it tends to be verbose. When using this tool, it is best to redirect the output into a text file and use vi or less to examine the contents. The basic h5dump commands to use are:

```
h5dump file
```

Where: file is the path to the HDF5 file, and
**Example 25.9.1-3: The h5stat Tool**

The *h5stat* tool reports statistics on an HDF5 file and the objects (i.e., data) that are stored in the file. The statistics reported are somewhat extensive and may be filtered using various command line options. The basic *h5stat* command is:

\[
\text{h5stat file}
\]

Where *file* is the path to the HDF5 file.

**Example 25.9.1-4: The h52gif Tool**

The *h52gif* tool extracts an image from an HDF5 file and converts it to a gif-formatted image. To render as an image, the object must be stored in the HDF5 file as a rectangle. To convert a rectangular object to a gif image, the basic command is:

\[
\text{h52gif h-file g-file -i image}
\]

Where: *h-file* is the path to the HDF5 file; *g-file* is the path to the output file; and *image* is the path to the image data in the HDF5 file.

### 25.9.2 The µEngine Web Test Driver

Built into the AWIPS II EDEX application is a web-based test driver that provides identification information on satellite, radar, GRIB, and ASCII products. The test driver may be used to help verify product data flow and to request displayable products.

The URL for the test driver is http:<EDEX server>:8080/uEngineWeb/index.html where <EDEX server> is the DNS name of the EDEX server, normally dx3 or dx4.

**NOTE:** IPVS, which provides IP aliasing/load balancing for CAVE-EDEX interactions, does not currently handle web-based interactions with the EDEX server.

**Example 25.9.2-1: Retrieving a Satellite Image via the µEngine Web Test Driver**

For this recipe, we use the µEngine Web test driver to access and display satellite data. The assumption used is simple: If the data can be displayed, it can be ingested. The µEngine Web test driver is used to allow more control over the selection of specific data than is easily available in CAVE.
a. On a workstation with network access to the EDEX server, open a web browser and enter the µEngine Web’s URL into the address bar. A display of the basic µEngine Web’s test page follows.

![AWIPS Test Driver Interface](image1)

AWIPS Test Driver Interface Connected to localhost

b. On the test driver, select the location, parameter, and time from the dropdown lists.

![AWIPS Test Driver Interface](image2)

You should also select a color map that is more appropriate to satellite data.

c. Scroll down the page and click Request Product. After a short pause, the image will be displayed. (You may need to scroll down to see the actual image.)
d. Double click on the image to display a larger version; depending on the browser settings, it may display in a separate tab or in a separate browser.

If there are any problems, a popup error dialog will be displayed.

**NOTE:** Prior to using the µEngine Web test page to display the data, the procedure outlined in Example 25.9.1-5 may be used to identify the data to display.

**HINT:** The µEngine Web test page only allows you to select data that has been ingested.
25.10 CAVE/Workstation Setup Considerations

Most of this appendix has been oriented toward server operation and troubleshooting. This section discusses potential problems with the main AWIPS II client application, CAVE. In AWIPS II, CAVE (Common AWIPS Visualization Environment) provides much of the functionality of D-2D, GFE, etc. in a single, integrated environment.

25.10.1 CAVE Startup Modes

In addition to the standard startup, CAVE has the capability to start in multiple modes: Text WS, AvnFPS Menu, AvnFPS Setup. A startup mode is activated by passing –\textit{component switch} via the command line when starting CAVE. (See Table 25.10.1-1.)

<table>
<thead>
<tr>
<th>Mode</th>
<th>Switch</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text WS</td>
<td>textws</td>
<td>Implements Text Workstation functionality from AWIPS.</td>
</tr>
<tr>
<td>AvnFPS Menu</td>
<td>avnmenu</td>
<td>Implements the AvnFPS Menu functionality from AWIPS.</td>
</tr>
<tr>
<td>AvnFPS Setup</td>
<td>avnsetup</td>
<td>Implements the AvnFPS Setup functionality from AWIPS.</td>
</tr>
</tbody>
</table>

25.10.2 Cave Installation

CAVE is installed on both the LX and the XT workstations, and has two main components: CAVE and Alert VIZ. CAVE can be started in two modes (see section 25.10.1, CAVE Startup Modes, Table 25.10.1-1). In normal operation, it is started in \textit{standard} mode on the LX workstation, and in \textit{textws} mode on the XT workstation.

CAVE is normally installed in \texttt{/usr/local/viz}. That installation includes CAVE, Alert VIZ, GFEClient, and support software such as Java and Python. Table 25.10.2-1 lists the expected directories under \texttt{/usr/local/viz}.

<table>
<thead>
<tr>
<th>Directory</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>alertviz</td>
<td>Contains the Alert VIZ application and support files.</td>
</tr>
<tr>
<td>basemaps</td>
<td>Contains shape files used by CAVE.</td>
</tr>
<tr>
<td>bin</td>
<td>Contains executables and scripts used in connection with AWIPS II VIZ software.</td>
</tr>
<tr>
<td>cave</td>
<td>Contains the CAVE application and support files.</td>
</tr>
<tr>
<td>caveData</td>
<td>Used by the installer.</td>
</tr>
<tr>
<td>fx a</td>
<td>Used by the installer.</td>
</tr>
<tr>
<td>gfeclient</td>
<td>Contains the GFE Client application.</td>
</tr>
<tr>
<td>include</td>
<td>Contains header files for the notification API.</td>
</tr>
<tr>
<td>java</td>
<td>Contains Java runtime and documentation files.</td>
</tr>
<tr>
<td>lib</td>
<td>Contains library files used in connection with AWIPS II VIZ software.</td>
</tr>
<tr>
<td>notification</td>
<td>Contains the notification API.</td>
</tr>
<tr>
<td>psql</td>
<td>Used by the installer.</td>
</tr>
<tr>
<td>python</td>
<td>Used by the installer.</td>
</tr>
<tr>
<td>setup</td>
<td>Used by the installer.</td>
</tr>
<tr>
<td>Uninstaller</td>
<td>Contains the uninstaller.</td>
</tr>
</tbody>
</table>
25.10.3 CAVE Startup

Starting CAVE depends on how the system is installed. For the recipes in this section, we assume a launcher has been created to run the CAVE startup script, cave.sh., with the appropriate startup mode flag. (See section 25.10.1, CAVE Startup Modes.) This is also assumed for the Alert VIZ application. Note that Alert VIZ should be started prior to starting CAVE.

If a launcher is not available for CAVE, the following steps can be used to start CAVE.

In AWIPS II, there are three methods for starting CAVE:
1. Open a console window and Enter: /awips2/cave/cave.sh.
2. Open a console window and Enter: cd /awips2/cave/;./cave.sh.

TIP: Both CAVE, in its various modes, and Alert VIZ write information to STDOUT and STDERR. This information may be useful for diagnosing runtime problems. Normally, this information is lost as the terminal window scrolls; it is also lost when starting either application using a launcher. It is possible to capture this information to a file for later review using Linux output redirection. To do so, add 2>&1 | tee ~/logs/<app name>.txt to the command line. You can also add a similar redirect to the launcher.

In each of the recipes that follow, it is assumed you are logged on to the appropriate workstation and have opened a terminal window.

Example 25.10.3-1: Validating the LX VIZ Installation

To validate the CAVE installation, you check to see if the software has been installed on the workstation. In addition, you will need to verify that the NFS mount is available and that the EDEX servers and companion XT workstation are accessible from the workstation. For this recipe, assume you are logged on to the LX1 workstation and have an open terminal window.

The basic steps to follow:

```bash
$ cd /usr/local/viz
$ ls -l
```

drwxr-xr-x  8 awips fxalpha 4096 Oct 30 12:58 alertviz
drwxr-xr-x 13 awips fxalpha 4096 Nov  5 16:21 cave
drwxr-xr-x  3 root  root    4096 Oct 30 13:16 edex
drwxr-xr-x  4 awips fxalpha 4096 Oct 30 12:58 fxa
drwxr-xr-x  5 awips fxalpha 4096 Oct 30 12:58 GFESuite
drwxr-xr-x  9 awips fxalpha 4096 Oct 30 12:58 java
drwxr-xr-x  7 awips fxalpha 4096 Oct 30 12:57 notification
Chapter 25 AWIPS/EDEX Administration Guide

What’s it mean? A correctly set up workstation box will have the directory structure and data (HDF5) mount, and will be able to ping the EDEX servers and its XT workstation successfully.

TIP: This check can be performed remotely by using ssh to log into the LX workstation.

Example 25.10.3-2: Validating the XT VIZ Installation

Validating the XT VIZ install is similar to validating the LX installation. The only difference is that the XT workstation needs to be able to access the LX. For this reason, you can follow the steps in Example 25.10.3-1, replacing xt1 with lx1 in the final ping.

Example 25.10.3-3: Cave Connection Problems

The first time a user runs CAVE or Alert VIZ (on an LX workstation) or Text WS (on an XT workstation), connection information is requested. This information points CAVE, Alert VIZ, or Text WS to the EDEX localization server. It may also point CAVE or the Text WS to Alert VIZ. This information, as well as certain user preferences, is stored in a
directory structure under *caveData* in the user’s home directory. Should the *caveData* directory tree be deleted, a VIZ application will request connection information the next time it is started. It does this via the dialogs shown in Exhibits 25.10.3-1 and 25.10.3-2.

Recall from Section 25.10.1, **CAVE Startup Modes** that AlertVIZ must be started or available before starting CAVE. Exhibit 25.10.3-3 shows the startup if CAVE is unable to find Alert VIZ at startup. In this exhibit, CAVE was able to find its localization server.
NOTE: You will also see this dialog if you attempt to start Alert VIZ on the LX workstation and the localization server is invalid.

If you see either version of this dialog when starting CAVE, Text WS or Alert VIZ, replace *localhost* with the name or IP address of the EDEX request server, enter the desired localization site, and, if the Alert Server box is displayed, replace *localhost* with the name or IP address of the Alert Server. Finally, click the *validate* button. If the entries are valid, the red highlights will be removed. Note that the localization site is not validated. Also note that the text (i.e., “Error: Unable to connect to localization server”) is not changed when the entries are valid; only the highlight is removed.

### 25.11 Verifying That AWIPS II Processes Are Running

This section presents alternative methods for validating operation of the EDEX processes. These may be used in place of the methods outlined earlier.

**On DX3/4:**

```
ps -ef | grep -v grep | grep edex.run.mode | cut -b -80
```

```
awips    27008 27004 99 17:12 ?   01:50:01
/awips2/java/bin/java -Dedex.run
awips    27211 27209 19 17:12 ?   00:22:01
/awips2/java/bin/java -Dedex.run
awips    27329 27323 3 17:12 ?    00:03:41
/awips2/java/bin/java -Dedex.run
awips    27434 27432 4 17:12 ?    00:05:09
/awips2/java/bin/java -Dedex.run
```
NOTE: There should be four lines of output, one for each of the EDEX processes: ingest, ingestDat, ingestGrib, and request. The results will be similar on both DX3 and DX4.

On dx1f:

```bash
ps -fu awips | grep -v idle | cut -d "-" -f1,2
```

<table>
<thead>
<tr>
<th>UID</th>
<th>PID</th>
<th>PPID</th>
<th>C</th>
<th>STIME</th>
<th>TTY</th>
<th>TIME</th>
<th>CMD</th>
</tr>
</thead>
<tbody>
<tr>
<td>awips</td>
<td>631</td>
<td>9444</td>
<td>1</td>
<td>18:14</td>
<td>?</td>
<td>00:00:32</td>
<td>/awips2/httpd_pypies/usr/sbin/httpd -f /awips2/httpd_pypies/etc/httpd/conf/httpd.conf</td>
</tr>
<tr>
<td>awips</td>
<td>636</td>
<td>9444</td>
<td>1</td>
<td>18:14</td>
<td>?</td>
<td>00:00:36</td>
<td>/awips2/httpd_pypies/usr/sbin/httpd -f /awips2/httpd_pypies/etc/httpd/conf/httpd.conf</td>
</tr>
<tr>
<td>awips</td>
<td>645</td>
<td>9444</td>
<td>1</td>
<td>18:14</td>
<td>?</td>
<td>00:00:36</td>
<td>/awips2/httpd_pypies/usr/sbin/httpd -f /awips2/httpd_pypies/etc/httpd/conf/httpd.conf</td>
</tr>
<tr>
<td>awips</td>
<td>6094</td>
<td>9444</td>
<td>1</td>
<td>18:20</td>
<td>?</td>
<td>00:00:28</td>
<td>/awips2/httpd_pypies/usr/sbin/httpd -f /awips2/httpd_pypies/etc/httpd/conf/httpd.conf</td>
</tr>
<tr>
<td>awips</td>
<td>6579</td>
<td>9444</td>
<td>1</td>
<td>18:22</td>
<td>?</td>
<td>00:00:33</td>
<td>/awips2/httpd_pypies/usr/sbin/httpd -f /awips2/httpd_pypies/etc/httpd/conf/httpd.conf</td>
</tr>
<tr>
<td>awips</td>
<td>6758</td>
<td>9444</td>
<td>0</td>
<td>18:22</td>
<td>?</td>
<td>00:00:21</td>
<td>/awips2/httpd_pypies/usr/sbin/httpd -f /awips2/httpd_pypies/etc/httpd/conf/httpd.conf</td>
</tr>
<tr>
<td>awips</td>
<td>8935</td>
<td>9444</td>
<td>0</td>
<td>18:28</td>
<td>?</td>
<td>00:00:18</td>
<td>/awips2/httpd_pypies/usr/sbin/httpd -f /awips2/httpd_pypies/etc/httpd/conf/httpd.conf</td>
</tr>
<tr>
<td>awips</td>
<td>8936</td>
<td>9444</td>
<td>1</td>
<td>18:28</td>
<td>?</td>
<td>00:00:23</td>
<td>/awips2/httpd_pypies/usr/sbin/httpd -f /awips2/httpd_pypies/etc/httpd/conf/httpd.conf</td>
</tr>
<tr>
<td>awips</td>
<td>8937</td>
<td>9444</td>
<td>1</td>
<td>18:28</td>
<td>?</td>
<td>00:00:30</td>
<td>/awips2/httpd_pypies/usr/sbin/httpd -f /awips2/httpd_pypies/etc/httpd/conf/httpd.conf</td>
</tr>
<tr>
<td>awips</td>
<td>8947</td>
<td>9444</td>
<td>1</td>
<td>18:28</td>
<td>?</td>
<td>00:00:34</td>
<td>/awips2/httpd_pypies/usr/sbin/httpd -f /awips2/httpd_pypies/etc/httpd/conf/httpd.conf</td>
</tr>
<tr>
<td>awips</td>
<td>9002</td>
<td>1</td>
<td>0</td>
<td>Oct21</td>
<td>?</td>
<td>00:00:26</td>
<td>/awips2/postgresql/bin/postmaster -D /awips2/data</td>
</tr>
<tr>
<td>awips</td>
<td>9016</td>
<td>9002</td>
<td>0</td>
<td>Oct21</td>
<td>?</td>
<td>00:00:02</td>
<td>postgres: logger process</td>
</tr>
<tr>
<td>awips</td>
<td>9022</td>
<td>9002</td>
<td>0</td>
<td>Oct21</td>
<td>?</td>
<td>00:00:16</td>
<td>postgres: writer process</td>
</tr>
<tr>
<td>awips</td>
<td>9023</td>
<td>9002</td>
<td>0</td>
<td>Oct21</td>
<td>?</td>
<td>00:00:00</td>
<td>postgres: wal writer process</td>
</tr>
<tr>
<td>awips</td>
<td>9024</td>
<td>9002</td>
<td>0</td>
<td>Oct21</td>
<td>?</td>
<td>00:00:03</td>
<td>postgres: autovacuum launcher process</td>
</tr>
<tr>
<td>awips</td>
<td>9025</td>
<td>9002</td>
<td>0</td>
<td>Oct21</td>
<td>?</td>
<td>00:01:36</td>
<td>postgres: stats collector process</td>
</tr>
</tbody>
</table>
Chapter 25 AWIPS/EDEX Administration Guide

awips     9149  9002  0 17:31 ?   00:00:25 postgres: awips
metadata 165.92.29.133(60513) PARSE
awips     9442  9438  0 Oct21 ?   00:00:00 bash -c
    /awips2/python/bin/python
awips     9443  9442  0 Oct21 ?   00:09:33
      /awips2/python/bin/python -u /awips2/python/lib/python2.7/site
awips     10055  1 1 Oct21 ?   01:24:22
    /awips2/java/bin/java -cp
     /awips2/rcm/data/config/res::/awips2/rcm/lib/activation
awips     11844  9444  1 18:30 ?   00:00:26
      /awips2/httpd_pypies/usr/sbin/httpd -f
     /awips2/httpd_pypies/etc/httpd/conf/httpd.conf
awips     11853  9444  1 18:30 ?   00:00:29
    /awips2/httpd_pypies/usr/sbin/httpd -f
     /awips2/httpd_pypies/etc/httpd/conf/httpd.conf
awips     11854  9444  1 18:30 ?   00:00:26
    /awips2/httpd_pypies/usr/sbin/httpd -f
     /awips2/httpd_pypies/etc/httpd/conf/httpd.conf
awips     11875  9444  1 18:30 ?   00:00:24
    /awips2/httpd_pypies/usr/sbin/httpd -f
     /awips2/httpd_pypies/etc/httpd/conf/httpd.conf
awips     12137  9002  0 18:30 ?   00:00:00 postgres: autovacuum
    worker process   metadata
awips     17121  9444  1 18:41 ?   00:00:21
    /awips2/httpd_pypies/usr/sbin/httpd -f
     /awips2/httpd_pypies/etc/httpd/conf/httpd.conf
awips     17146  9002  0 18:42 ?   00:00:00 postgres: autovacuum
    worker process   metadata
awips     20664  9444  1 18:46 ?   00:00:12
    /awips2/httpd_pypies/usr/sbin/httpd -f
     /awips2/httpd_pypies/etc/httpd/conf/httpd.conf
awips     20666  9444  1 18:46 ?   00:00:17
    /awips2/httpd_pypies/usr/sbin/httpd -f
     /awips2/httpd_pypies/etc/httpd/conf/httpd.conf
awips     20667  9444  1 18:46 ?   00:00:11
    /awips2/httpd_pypies/usr/sbin/httpd -f
     /awips2/httpd_pypies/etc/httpd/conf/httpd.conf
awips     21347  9444  0 18:48 ?   00:00:09
    /awips2/httpd_pypies/usr/sbin/httpd -f
     /awips2/httpd_pypies/etc/httpd/conf/httpd.conf
awips     21355  9444  0 18:48 ?   00:00:08
    /awips2/httpd_pypies/usr/sbin/httpd -f
     /awips2/httpd_pypies/etc/httpd/conf/httpd.conf
awips     22928  9444  0 18:51 ?   00:00:07
    /awips2/httpd_pypies/usr/sbin/httpd -f
     /awips2/httpd_pypies/etc/httpd/conf/httpd.conf
awips     26147  9444  1 18:59 ?   00:00:06
    /awips2/httpd_pypies/usr/sbin/httpd -f
     /awips2/httpd_pypies/etc/httpd/conf/httpd.conf
awips     31650  9444  1 19:01 ?   00:00:02
    /awips2/httpd_pypies/usr/sbin/httpd -f
     /awips2/httpd_pypies/etc/httpd/conf/httpd.conf

AWP.MAN.SMM.A2

– Hard copies uncontrolled. Verify effective date prior to use. –
NOTE: There should be a single postmaster process running which drives the postgreSQL AWIPS II database engine along with five (5) subsequent process threads (logger, writer, wal writer, autovacuum, stats collector). The ISC routing table service (edex_irt) as well as the AWIPS II RadarServer (edex_rcm) should be running as a stand-alone process.

On cp1f:

```
ps -fU fx,ldm | cut -d "-" -f1,2; ps -ef | egrep 'iscsi|nfs' | grep -v grep
```

```
UID    PID    PPID  C   STIME TTY   TIME CMD
ldm    30195  1   0  Jul10 ?  00:00:00 ldmd -I 0.0.0.0
ldm    30197 30195  1  Jul10 ?  03:40:16 pqact -e
ldm    30198 30195  0  Jul10 ?  00:52:55 edexBridge -s cp1f
root   30199 30195  4  Jul10 ?  13:40:01 noaaportIngester -m 224.0.1.1
root   30200 30195  0  Jul10 ?  00:55:51 noaaportIngester -m 224.0.1.2
root   30201 30195  1  Jul10 ?  05:27:35 noaaportIngester -m 224.0.1.3
root   30202 30195  0  Jul10 ?  00:41:11 noaaportIngester -m 224.0.1.4
root   30204 30195  0  Jul10 ?  01:07:01 noaaportIngester -m 224.0.1.5
root   4207   67   0  Jun06 ?  00:59:39 [nfsiod]
root   27655 5652  0 16:12 ?  00:00:00 /bin/sh
/var/opt/OV/bin/instrumentation/stale_nfs.sh stale_nfs_mounts
```

NOTE: The LDM should have five (5) noaaportIngester processes, edexBridge (connected to the server running Qpid), ldmd and pqact running.

On px1f:

```
apache 3738  14491  0 Oct23 ?  00:00:04 /usr/sbin/httpd
apache 3739  14491  0 Oct23 ?  00:00:05 /usr/sbin/httpd
apache 3740  14491  0 Oct23 ?  00:00:05 /usr/sbin/httpd
apache 3741  14491  0 Oct23 ?  00:00:05 /usr/sbin/httpd
apache 3742  14491  0 Oct23 ?  00:00:05 /usr/sbin/httpd
apache 3743  14491  0 Oct23 ?  00:00:05 /usr/sbin/httpd
apache 3744  14491  0 Oct23 ?  00:00:05 /usr/sbin/httpd
apache 3745  14491  0 Oct23 ?  00:00:03 /usr/sbin/httpd
fxa     6766 13164  0 19:10 ?  00:00:00 crond
fxa     6772  6766  0 19:10 ?  00:00:00 csh -c
```

```
awips adapt/hwr/bin/hwrn
```

AWP.MAN.SMM.A2 25-131

-- Hard copies uncontrolled. Verify effective date prior to use. --
NOTE: There should be eight (8) httpd threads running which control the AWIPSII System Monitor. There should be two getty serial ports initialized named /dev/tty[n][a,b] which are used by the asyncScheduler, which should also be running with the two async processes, asyncPilMsgServer and asyncPollData.pl. The AWIPS II hmMonitorServer and NWWSSchedule should be running. The GUI controller for IPVS named piranha.gui should be listed as well as the IPVS process nanny connected on port 9581 to the IPADDR of the DX3 and DX4.
On px2f:

```
ps -fU ldad,fxa | cut -d "-" -f1,2; ps -ef | egrep 'iscsi|nfs' | grep
-v grep
```

<table>
<thead>
<tr>
<th>UID</th>
<th>PID</th>
<th>PPID</th>
<th>C</th>
<th>STIME</th>
<th>TTY</th>
<th>TIME</th>
<th>CMD</th>
</tr>
</thead>
<tbody>
<tr>
<td>fxa</td>
<td>13193</td>
<td>1</td>
<td>0</td>
<td>19:13</td>
<td>?</td>
<td>00:00:00</td>
<td>/awips/adapt/hwr/bin/Linux/nwrxmit -g /awips/ on NWRBrowser</td>
</tr>
<tr>
<td>fxa</td>
<td>13841</td>
<td>1</td>
<td>0</td>
<td>Oct13</td>
<td>?</td>
<td>00:00:01</td>
<td>/awips/fxa/bin/asyncPilMsgServer</td>
</tr>
<tr>
<td>fxa</td>
<td>13844</td>
<td>1</td>
<td>0</td>
<td>Oct13</td>
<td>?</td>
<td>00:00:07</td>
<td>/awips/fxa/bin/asyncScheduler</td>
</tr>
<tr>
<td>fxa</td>
<td>14277</td>
<td>13844</td>
<td>0</td>
<td>Oct13</td>
<td>?</td>
<td>00:00:04</td>
<td>test1 23 20 1 test1</td>
</tr>
<tr>
<td>fxa</td>
<td>14278</td>
<td>13844</td>
<td>0</td>
<td>Oct13</td>
<td>?</td>
<td>00:01:21</td>
<td>test2 25 23 2 test2</td>
</tr>
<tr>
<td>fxa</td>
<td>14279</td>
<td>13844</td>
<td>0</td>
<td>Oct13</td>
<td>?</td>
<td>00:01:33</td>
<td>ttyn1h 27 25 8</td>
</tr>
<tr>
<td>ttyn1h</td>
<td>/dev/ttyn1h</td>
<td>9600</td>
<td>8</td>
<td>NONE</td>
<td>1 NONE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>fxa</td>
<td>14280</td>
<td>1</td>
<td>0</td>
<td>Oct13</td>
<td>?</td>
<td>00:00:00</td>
<td>/usr/bin/perl /awips/fxa/bin/asyncPollData.pl</td>
</tr>
<tr>
<td>fxa</td>
<td>14288</td>
<td>1</td>
<td>0</td>
<td>Oct13</td>
<td>?</td>
<td>00:00:00</td>
<td>/awips/fxa/bin/hmMonitorServer</td>
</tr>
<tr>
<td>fxa</td>
<td>14306</td>
<td>1</td>
<td>0</td>
<td>Oct13</td>
<td>?</td>
<td>00:01:46</td>
<td>/awips/fxa/bin/NWWSSchedule</td>
</tr>
<tr>
<td>fxa</td>
<td>30288</td>
<td>1</td>
<td>0</td>
<td>Sep19</td>
<td>?</td>
<td>00:48:45</td>
<td>/awips/fxa/bin/purgeProcess -commit</td>
</tr>
<tr>
<td>root</td>
<td>4515</td>
<td>35</td>
<td>0</td>
<td>Sep19</td>
<td>?</td>
<td>00:00:00</td>
<td>[iscsi_eh]</td>
</tr>
<tr>
<td>root</td>
<td>4762</td>
<td>1</td>
<td>0</td>
<td>Sep19</td>
<td>?</td>
<td>00:00:00</td>
<td>brcm_iscsiuito</td>
</tr>
<tr>
<td>root</td>
<td>4768</td>
<td>1</td>
<td>0</td>
<td>Sep19</td>
<td>?</td>
<td>00:00:00</td>
<td>iscsid</td>
</tr>
<tr>
<td>root</td>
<td>4769</td>
<td>1</td>
<td>0</td>
<td>Sep19</td>
<td>?</td>
<td>00:00:00</td>
<td>iscsid</td>
</tr>
<tr>
<td>root</td>
<td>5298</td>
<td>35</td>
<td>0</td>
<td>Sep19</td>
<td>?</td>
<td>00:00:00</td>
<td>[iscsi_q_3]</td>
</tr>
<tr>
<td>root</td>
<td>5698</td>
<td>35</td>
<td>0</td>
<td>Sep19</td>
<td>?</td>
<td>00:01:29</td>
<td>[nfsiod]</td>
</tr>
<tr>
<td>root</td>
<td>5970</td>
<td>35</td>
<td>0</td>
<td>Sep19</td>
<td>?</td>
<td>00:00:00</td>
<td>[nfsd4]</td>
</tr>
<tr>
<td>root</td>
<td>5971</td>
<td>1</td>
<td>0</td>
<td>Sep19</td>
<td>?</td>
<td>00:00:00</td>
<td>[nfsd]</td>
</tr>
<tr>
<td>root</td>
<td>5972</td>
<td>1</td>
<td>0</td>
<td>Sep19</td>
<td>?</td>
<td>00:00:00</td>
<td>[nfsd]</td>
</tr>
<tr>
<td>root</td>
<td>5973</td>
<td>1</td>
<td>0</td>
<td>Sep19</td>
<td>?</td>
<td>00:00:00</td>
<td>[nfsd]</td>
</tr>
<tr>
<td>root</td>
<td>5974</td>
<td>1</td>
<td>0</td>
<td>Sep19</td>
<td>?</td>
<td>00:00:00</td>
<td>[nfsd]</td>
</tr>
<tr>
<td>root</td>
<td>5975</td>
<td>1</td>
<td>0</td>
<td>Sep19</td>
<td>?</td>
<td>00:00:00</td>
<td>[nfsd]</td>
</tr>
<tr>
<td>root</td>
<td>5976</td>
<td>1</td>
<td>0</td>
<td>Sep19</td>
<td>?</td>
<td>00:00:00</td>
<td>[nfsd]</td>
</tr>
<tr>
<td>root</td>
<td>5977</td>
<td>1</td>
<td>0</td>
<td>Sep19</td>
<td>?</td>
<td>00:00:00</td>
<td>[nfsd]</td>
</tr>
<tr>
<td>root</td>
<td>5978</td>
<td>1</td>
<td>0</td>
<td>Sep19</td>
<td>?</td>
<td>00:00:00</td>
<td>[nfsd]</td>
</tr>
</tbody>
</table>

**NOTE:** There should be eight (8) nfsd threads running and two (2) iscsid daemons running which control the /data_store filesystem export. The AWIPSII LDAD and FSI processes should also be running and should match those in the output above (with the exception of the node domain name, which will match your site ID).
25.12 EDEX Startup and Shutdown Procedures

Although EDEX is installed only on the DX 3/4 cluster, it is supported by applications running on several servers. The actual EDEX server software is installed on both DX3 and DX4. Please Note: For large memory servers\(^3\) an additional EDEX process, EDEX/DAT, is present. Additional software necessary for proper functionality of the EDEX is installed and runs on the DX1/2 cluster, the CP1/2 cluster and the PX1/2 cluster. DX1/2 and PX1/2 are set up into heartbeat clusters; although the software is installed on both servers, it is running on only one server at a time in each cluster. IP aliasing is used to provide a single login to the active member of each cluster. In the case of CP1/2, no automatic failover currently exists. As a result, AWIPS II software normally runs on CP1; in the case of failover, references to CP1 should be changed to CP2.

25.12.1 AWIPS II/EDEX Software Dependencies

EDEX is designed to be a distributed processing system. This provides additional system stability by reducing the impact of individual server shutdowns. For example, since the EDEX server software runs on both DX3 and DX4, it is possible to bump\(^4\) DX3 without completely interrupting data processing.

There are some dependencies among the various software systems, however. For example, the Apache Advanced Message Protocol Message Broker (QPID) service on the CP1/2 cluster must be running before any EDEX processes on DX3 or DX4 can be started. Exhibit 25.12.1-1 shows the dependencies among the EDEX software. Because EDEX is the same on both DX3 and DX4, it is shown as a single bubble in the dependency graph.

In Exhibit 25.12.1-1, operational dependencies are shown as lines between the bubbles representing processes. Two types of dependencies are identified: Hard\(^5\) dependencies are shown as solid lines; soft\(^6\) dependencies are shown as dotted lines. Dependencies run top to bottom; the upper of a pair of dependent processes must be running before the lower one can be started. Conversely, the lower of a pair of dependent processes should be stopped before the upper process may be stopped.

From Exhibit 25.12.1-1 we see that there is a dependency between EDEX (on DX3/4) and QPID. As a result, 1) QPID must be started before starting EDEX, and 2) EDEX must be stopped before stopping QPID. The dependency between EDEX (on DX3/4) and RCM, however, is soft; EDX will start even if RCM has not been started, but will log communication errors related to RCM.

---

\(^3\) In this case, large memory means at least four gigabytes (GB) of onboard memory.

\(^4\) Shut down and restart.

\(^5\) In a hard dependency, a dependent process will not start if the other process is not running. Thus, EDEX (on DX3 or DX4) will not start unless QPID is running.

\(^6\) In a soft dependency, the dependent process does not require the other process to start, but the output of the dependent process may be lost. In this case, LDM(D) and RCM are data producers for which EDEX (on DX3 and DX4) is a data consumer; if EDEX is not running, the data provided by LDM(D) and RCM does not get processed and may be lost.
Also in Exhibit 25.12.1-1, note that no dependency exists between RCM and Local Data Manager (LDM); as a result, LDM and RCM may be started in any order.

25.12.2 Starting AWIPS II/EDEX Software

Because of the previously discussed software dependencies within the EDEX system, the EDEX software should be started in a specific order. Please refer to Chapter 21 for complete system startup.

Refer to Exhibit 25.12.1-1 to determine software dependencies. What follows is the approved sequence for a full start of the EDEX software system.

The approved steps to follow to perform a complete startup of the AWIPS II/ EDEX system, assuming no processes are running on any server, are:

1. Log into DX1.
   
   **Enter:** ssh root@dx1

2. Start the DX1 high-availability package.
   
   **Enter:** hb_run a2dx1apps

3. Start QPID
   
   Start LDM, pulse, and QPID on CPSBN1. Then verify that they are accessible and that software is running. This is necessary because most processes have a
dependency on QPID, but the processes running on the CPSBN do not depend on any other process or mount to function properly.

**Note:** If site with remote CPs, then power up px1 at this time and start up pulse and qpid as indicated below.

Connect to cpsbn1 via the System Console

Check to see if a2cp1apps is running.

**Enter:** `hb_stat`

**Enter:** `tail -f /var/log/cluster`

If the package fails to start after a few minutes, force the package start.

**Enter:** `hb_run a2cp1apps`

Check to see if LDM software is running

**Enter:** `ps -fu ldm`

If the processes are NOT running, perform the following two steps:

**Enter:** `service ldmcp start`

**Enter:** `ps -fu ldm`  
[check for processes]

**Enter:** `ps -fu awips`

If QPID is not running, then perform the following two steps:

**Enter:** `service qpidd start`

**Enter:** `ps -fu awips`  
[check for qpidd]

**Enter:** `ps –wef|grep pulse`

If IPVS (pulse) is not running perform following two steps:

**Enter:** `service pulse start`

4. Start EDEX.

**Enter:** `ssh dx3 “service edex_camel start”`

**Enter:** `ssh dx4 “service edex_camel start”`

5. Start the PX1 high-availability package.

**Enter:** `ssh px1 “hb_run a2px1apps”`

6. Start the PX2 high-availability package.

**Enter:** `ssh px2 “hb_run a2px2apps”`
7. Start the DX2 high-availability package.

   Enter: `ssh dx2 “hb_run a2dx2apps”`

   **NOTE:** IPVS, which runs on the CP1/2 cluster, is part of the Red Hat Linux Kernel. As a result, it is automatically started when the CP servers are started.

### 25.12.3 Verifying Software Execution

Verifying EDEX software execution is accomplished by logging into each of the servers and running one or more commands. The output from these commands is then evaluated to determine if the expected software is running.

Because software dependencies are not critical in checking for software execution, the commands are grouped by server; this allows a minimum number of logins in the verification process.

The System V service scripts can be used to validate execution of some of the EDEX processes; in addition, the methods presented here use variations of a simple Linux `ps` command. Although here, they are being performed by the root user, normally any user can execute the `ps`-based commands; only the root user can run the System V service.

The basic steps to verify software execution are:

1. **On DX1/2:**
   
   ```
   ssh root@dx1f
   
   hb_stat
   ps -ef | grep postmaster | grep -v grep | cut -b -80
   service edex_postgres status
   ps -ef | grep Radar | grep -v grep | cut -b -80
   service edex_rcm status
   ps -ef | grep logging | grep -v grep | cut -b -80
   service httpd-pypies status
   
   ssh root@dx2f
   ps -ef | grep httpd_pypies | grep -v grep | cut -b -80
   ```

2. **On CP1/2:**
   
   ```
   ssh root@cpsbn1
   
   ps -ef | grep ldmd | grep -v grep | cut -b -80
   service ldmcp status
   ps -ef | grep gpidd | grep -v grep | cut -b -80
   service awips2_qpidd_cluster_status
   ps -ef | grep pulse | grep -v grep | cut -b 80
   ```

3. **On PX1/2:**
   
   ```
   ssh root@px2f
   
   hb_stat
   ps -fu fxa -u ldad
   ```
4. On DX3 and DX4:

```
ssh root@dx3
ps -ef | grep esb.Main | grep -v grep | cut -b -80
service edex_camel status
```

```
ssh root@dx4
ps -ef | grep esb.Main | grep -v grep | cut -b -80
service edex_camel status
```

**NCEP Only:** For DX5 and DX6, use the same step shown above for DX3 and DX4.

**NOTE:** Each command is entered on a single line. Also note that these commands can be performed via remote login, `ssh`, from DX1.

When the EDEX server software is running, each of these commands will return one or more lines of output. The number of lines of output may vary depending on the number of instances of processes that are running. The expected number of lines of output for each command is shown in Table 25.12.3-1.

<table>
<thead>
<tr>
<th>Server</th>
<th>Command</th>
<th>Lines</th>
</tr>
</thead>
<tbody>
<tr>
<td>DX1/2</td>
<td>`ps -ef</td>
<td>grep postmaster</td>
</tr>
<tr>
<td></td>
<td>`ps -ef</td>
<td>grep Radar</td>
</tr>
<tr>
<td></td>
<td>`ps -ef</td>
<td>grep httpd_pypies</td>
</tr>
<tr>
<td></td>
<td>`ps -ef</td>
<td>grep logging</td>
</tr>
<tr>
<td>CP1/2</td>
<td>`ps -ef</td>
<td>grep ldmd</td>
</tr>
<tr>
<td></td>
<td>`ps -ef</td>
<td>grep qpidd</td>
</tr>
<tr>
<td>PX1/2</td>
<td><code>ps -fu fxa -u ldad</code></td>
<td>&gt; 1</td>
</tr>
<tr>
<td>DX3</td>
<td>`ps -ef</td>
<td>grep esb.Main</td>
</tr>
<tr>
<td>DX4</td>
<td>`ps -ef</td>
<td>grep esb.Main</td>
</tr>
</tbody>
</table>

**NOTE:** To determine the correct number of lines (three or four) to expect on DX3 and DX4, you need to check the installed memory on each server. If the amount of memory is greater than 4 GB, you should see four lines of output; otherwise you should see three lines of output.

In addition to the commands shown above, it may be desirable to check the various process log files to determine if the software has started.

### 25.12.4 Stopping EDEX Software

Because of the previously discussed software dependencies within the EDEX system, the EDEX software should be stopped in a specific order.

Refer to Exhibit 25.12.1-1 to determine software dependencies. The standard procedure is to stop LDM prior to stopping EDEX on DX3 and DX4.
NOTE: The PX1/2 and DX1/2 clusters both utilize high-availability packages to manage software. Because each high-availability packages manages the virtual server name assigned to the package, you cannot use the virtual server name to log in when stopping the software. The correct procedure is:

Log into the server via the virtual server name.

Use `hostname` to determine the actual server name.

For the a2px1apps package, the command sequence is:

```
ssh px1f "hostname"
ssh xxx "hb_halt a2px1apps"
```

where xxx is the server name determined by the hostname command. Both these commands are normally executed from DX1 after logging in as the root user.

The approved steps to perform a complete shutdown of the EDEX system are:

1. Log into DX1.
   
   ```
   Enter: ssh root@dx1
   ```

2. Stop LDM on cpsbn1/1 cluster.
   
   ```
   Enter: ssh cpsbn1 "service ldmcp stop"
   Enter: ssh cpsbn2 "service ldmcp stop"
   ```

3. Stop the DX2 high-availability package.
   
   ```
   Enter: ssh dx2f "hostname"
   Enter: ssh xxx "hb_halt a2dx2apps"
   ```

4. Stop the PX2 high-availability package.
   
   ```
   Enter: ssh px2f "hostname"
   Enter: ssh xxx "hb_halt a2px2apps"
   ```

5. Stop the PX1 high-availability package.
   
   ```
   Enter: ssh px1f "hostname"
   Enter: ssh xxx "hb_halt a2px1apps"
   ```

6. Stop EDEX.
   
   ```
   Enter: ssh dx4 "service edex_camel stop"
   Enter: ssh dx3 "service edex_camel stop"
   ```

   **Note:** For dx5, dx6 at NCEP Centers, use the command shown above, replacing dx3 and dx4 with dx5 and dx6.

7. Stop the DX1 high-availability package.

   ```
   Enter: ssh dx1f "hostname"
   Enter: ssh xxx "hb_halt a2dx1apps"
   ```
8. Stop CPSBN1 high-availability package.
   Enter: `ssh cp1f “hostname”`
   Enter: `ssh xxx “hb_halt a2cplapps”`

9. Stop QPID.
   `ssh cpsbn1 “service edex_qpid stop”`

25.12.5 Verifying Software Shutdown

Verifying EDEX software shutdown is accomplished by logging into each of the servers
and running one or more commands. The output from these commands is then evaluated
to determine if the expected software has been stopped.

Because software dependencies are not critical in checking for software shutdown, the
commands are grouped by server; this allows a minimum number of logins in the
verification process.

The System V service scripts can be used to validate shutdown of some of the EDEX
processes; in addition, the methods presented here use variations of a simple Linux `ps`
command. Although these are shown as being performed by the root user, the
`ps`-based
commands can normally be executed by any user; the System V service scripts can only be run by the root user.

The basic steps to verify successful software halt are:

1. On DX1/2:
   ```
   ssh root@dx1
   hb_stat
   ps –ef | grep postmaster | grep –v grep | cut –b -80
   service edex_postgres status
   ps –ef | grep Radar | grep –v grep | cut –b -80
   service edex_rcm status
   ps –ef | grep logging | grep –v grep | cut –b -80
   service httpd-pypies status
   ssh root@dx2
   ps –ef | grep httpd_pypies | grep –v grep | cut –b -80
   ```

2. On CP1/2:
   ```
   ssh root@cpsbn1
   ps –ef | grep ldm | grep –v grep | cut –b -80
   service ldmcp status
   ps –ef | grep qpidd | grep –v grep | cut –b -80
   service qpidd status
   ps –ef | grep pulse | grep –v grep | cut –b -80
   service pulse status
   ssh root@cpsbn2
   ps –ef | grep ldm | grep –v grep | cut –b -80
   service ldmcp status
   ```
3. On PX1/2:

```
ssh root@px1
ps -fu fxu -u ldad
hb_stat
ssh root@px2
ps -fu fxu -u ldad
hb_stat
```

4. On DX3 and DX4:

```
ssh root@dx3
ps -ef | grep esb.Main | grep -v grep | cut -b -80
service edex_camel status
ssh root@dx4
ps -ef | grep esb.Main | grep -v grep | cut -b -80
service edex_camel status
```

**NOTE:** Each command is entered on a single line. Also note that these commands can be performed via remote login, ssh, from DX1.

When the software has been successfully stopped, each of the listed `ps` commands will display no output and each `service` command will provide a “service not running” message; the exact wording will vary depending on the service.

In addition to the commands shown above, it may be desirable to check the various process log files to determine that the software stopped execution.

### 25.12.6 Alternative Shutdown Scenarios

In addition to the complete startup and shutdown processes, it may be desirable to perform a partial shutdown and restart of the EDEX systems. Although a number of partial shutdown/restart scenarios are possible, only three are considered here: restarting the EDEX services on DX3; restarting PostgreSQL on DX1/2; restarting QPID on CP1/2.

The general rule: *When stopping an EDEX system, all dependent systems must be stopped first.* With this rule in mind, the three scenarios are:

**Scenario 1: Restarting the EDEX Services on DX3**

We can stop and restart EDEX on DX3 without stopping and restarting the other EDEX system components. (Note that for CAVE to operate successfully, at least one EDEX request process must be running.)

The steps for this scenario are:

1. On DX3, stop EDEX services:

```
ssh root@dx3
service edex_camel stop
```
2. On DX3, verify software shutdown:
   
   ```bash
   ssh root@dx3
   ps -ef | grep esb.Main | grep -v grep | cut -b -80
   service edex_camel status
   ```

3. On DX3, start EDEX services:
   
   ```bash
   ssh root@dx3
   service edex_camel start
   ```

4. On DX3, verify software execution:
   
   ```bash
   ssh root@dx3
   ps -ef | grep esb.Main | grep -v grep | cut -b -80
   service edex_camel status
   ```

Scenario 1 may be used when it is necessary to modify EDEX configuration files. Because the EDEX services on DX3 and DX4 are independent, each can be stopped and restarted without affecting the rest of the system. Other EDEX system components are not affected.

**NOTE:** Both EDEX shutdown and restart can take some time. As a result, if EDEX has not shut down (Step 2), wait a short time and retry the test. The same is true for the startup.

**Scenario 2: Restarting PostgreSQL on DX1/2**

In Exhibit 25.12.1-1, we see that there are dependencies between PostgreSQL and the EDEX services on DX3 and DX4. As a result, the EDEX services on DX3 and DX4 should be stopped before stopping PostgreSQL. The steps for this scenario are:

1. On DX3, stop EDEX services:
   
   ```bash
   ssh root@dx3
   service edex_camel stop
   ```

2. On DX3, verify software shutdown:
   
   ```bash
   ssh root@dx3
   ps -ef | grep esb.Main | grep -v grep | cut -b -80
   service edex_camel status
   ```

3. On DX4, stop EDEX services:
   
   ```bash
   ssh root@dx4
   service edex_camel stop
   ```

4. On DX4, verify software shutdown:
   
   ```bash
   ssh root@dx4
   ps -ef | grep esb.Main | grep -v grep | cut -b -80
   service edex_camel status
   ```
5. On DX1/2, stop PostgreSQL:
   `ssh root@dx1f
service edex_postgres stop`

6. On DX1/2, verify software shutdown:
   `ssh root@dx1f
   ps -ef | grep postmaster | grep -v grep | cut -b -80
   service edex_postgres status`

7. On DX1/2, start Postgresql:
   `ssh root@dx1f
   service edex_postgres start`

8. On DX1/2, verify software startup:
   `ssh root@dx1f
   ps -ef | grep postmaster | grep -v grep | cut -b -80
   service edex_postgres status`

9. On DX3, start EDEX services:
   `ssh root@dx3
   service edex_camel start`

10. On DX3, verify software startup:
    `ssh root@dx3
    ps -ef | grep esb.Main | grep -v grep | cut -b -80
    service edex_camel status`

11. On DX4, start EDEX services:
    `ssh root@dx4
    service edex_camel start`

12. On DX4, verify software startup:
    `ssh root@dx4
    ps -ef | grep esb.Main | grep -v grep | cut -b -80
    service ede_camel status`

**NOTE:** Refer to the previous discussion on validating system start and stop for information on interpreting the “verify” steps.

Scenario 2 may be followed when it is necessary to stop the database briefly; for example, to modify its configuration files, which require a software restart. Keep in mind that, although AWIPS II applications access the database via EDEX, local applications and other non-AWIPS II applications may access the database directly. Any applications that access the database should be shut down before performing a database restart.

**NOTE:** Scenario 2 assumes a short downtime (maximum of 10 to 15 minutes); for longer downtimes, you should also stop and restart LDM(D) and RCM.
Scenario 3: Restarting QPID on CP1/2

As seen in Exhibit 25.12.1-1, there are hard dependencies between QPID and EDEX (on both DX3 and DX4) and RCM. In fact, the only EDEX components that are independent of QPID are PostgreSQL and PyPIES. The steps for this scenario are:

1. On CP1/2, stop LDM:
   
   ```
   ssh ldm@cp1f
   ldmadmin stop
   ```
2. On CP1/2, verify software shutdown:
   
   ```
   ssh root@cp1f
   ps –ef | grep ldm | grep –v grep | cut –b -80
   ```
3. On PX2, stop rehosted software:
   
   ```
   ssh root@px2f “hostname”
   ssh root@xxx “hb_halt a2px2apps
   ```
4. On PX2, verify software shutdown:
   
   ```
   ssh root@xxx
   ps –fu fxa –u ldad
   ```
5. On DX1/2, stop RCM:
   
   ```
   ssh root@dx1f
   service edex_rcm stop
   ```
6. On DX1/2, verify software shutdown:
   
   ```
   ssh root@dx1f
   ps –ef | grep Radar | grep –v grep | cut –b -80
   service edex_rcm status
   ```
7. On DX3, stop EDEX services:
   
   ```
   ssh root@dx3
   service edex_camel stop
   ```
8. On DX3, verify software shutdown:
   
   ```
   ssh root@dx3
   ps –ef | grep esb.Main | grep –v grep | cut –b -80
   service edex_rcm status
   ```
9. On DX4, stop EDEX services:
   
   ```
   ssh root@dx4
   service edex_camel stop
   ```
10. On DX4, verify software shutdown:
    
    ```
    ssh root@dx4
    ps –ef | grep esb.Main | grep –v grep | cut –b -80
    service edex_rcm status
    ```
11. On CP1/2, stop QPID:
   ```
   ssh root@cp1f
   service qpidd stop
   ```

12. On CP1/2, verify software shutdown:
   ```
   ssh root@cp1f
   ps -ef | grep qpidd | grep -v grep | cut -b -80
   service qpidd status
   ```

13. On CP1/2, start QPID:
   ```
   ssh root@cp1f
   service qpidd start
   ```

14. On CP1/2, verify software startup:
   ```
   ssh root@cp1f
   ps -ef | grep qpidd | grep -v grep | cut -b -80
   service qpidd status
   ```

15. On DX3, start EDEX services:
   ```
   ssh root@dx3
   service edex_camel start
   ```

16. On DX3, verify software startup:
   ```
   ssh root@dx3
   ps -ef | grep esb.Main | grep -v grep | cut -b -80
   service edex_camel status
   ```

17. On DX4, start EDEX services:
   ```
   ssh root@dx4
   service edex_camel start
   ```

18. On DX4, verify software startup:
   ```
   ssh root@dx4
   ps -ef | grep esb.Main | grep -v grep | cut -b -80
   service edex_camel status
   ```

19. On DX1/2, start RCM:
   ```
   ssh root@dx1f
   service edex_rcm start
   ```

20. On DX1/2, verify software startup:
   ```
   ssh root@dx1f
   ps -ef | grep Radar | grep -v grep | cut -b -80
   service edex_rcm status
   ```

21. On PX2, start rehosted processes:
   ```
   ssh root@px2 "hb_run a2px2apps"
   ```
22. On PX2, verify software start:
   
   ```bash
   ssh root@px2 "hb_stat a2px2apps"
   ssh root@px2 "ps -fu fxa -u ldad"
   ```

23. On CP1/2, start LDM(D):
   
   ```bash
   ssh ldm@cp1f
   ldmadmin start
   ```

24. On CP1/2, verify software startup:
   
   ```bash
   ssh root@cp1f
   ps -ef | grep ldm | grep -v grep | cut -b -80
   ```

### 25.13 Troubleshooting Basic AWIPS II Data Flow

This section “connects the dots,” the dots being the recipes presented in this Administration Guide. In doing so, the goal is to provide a series of steps to follow to diagnose data ingest problems. The data flow consists of a series of steps like links in a chain. The approach used here is to start at the point where the problem is detected and then search upstream until the broken link is found. Once that link is repaired, check the data flow and continue the search as needed. This can be represented as a flow chart. See Exhibit 25.13-1.

![Exhibit 25.13-1. Basic Data Flow Troubleshooting](image)
The next two examples illustrate this basic approach to troubleshooting the data flow.

**Example 25.13-1: Troubleshooting Satellite Data Flow**

The basic flow of satellite data into AWIPS II was discussed under Section 25.3, *Basic Data Flow Concepts*. Exhibit 25.13-1 traces data from the SBN to the EDEX ingest component. Missing from Exhibit 25.13-1, however, is the Data Store, which contains the decoded data after it has been processed by the AWIPS Ingest processes. Exhibit 25.13-2 presents a more complete picture of the data flow from the SBN through EDEX to the Data Store. (The exhibit does not show the client application; it only shows the dataflow.)

![Exhibit 25.13-2. Expanded EDEX Data Flow](image)

Suppose that CAVE is not displaying the satellite data. From Exhibit 25.13-2, we see that the place to start is at the metadata database and the data store and then work upstream. Follow the applicable examples in this chapter.

1. Verify that satellite data is available in the data store.
2. Verify that satellite metadata is available in the database.
3. Verify that satellite data is being ingested.
4. Verify that QPID satellite queue is current.
5. Verify that satellite data is being provided by the LDM.
Example 25.13-2: Troubleshooting Radar Data Flow

Radar data follows the same basic data flow as satellite data with one exception: radar data is available via the ORPG as well as the SBN. ORPG data is handled in AWIPS II by the Radar Server (RCM).

Suppose that CAVE is not displaying radar data. From Exhibit 25.13-2, we see that the place to start is at the metadata database and the data store and then work upstream. In addition, the Radar Server must be checked, of course. Follow the applicable examples in this chapter.

1. Verify that radar data is available in the data store.
2. Verify that radar metadata is available in the database.
3. Verify that radar data is being ingested.
4. Verify that QPID radar queue is current.
5. Verify that radar data is being provided by LDM.
6. Verify that radar data is being provided by RCM.
Chapter 26
Thin Client
Chapter 26. Thin Client

Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>26.1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>26.2 Configuring the Apache Proxy Server</td>
<td>1</td>
</tr>
<tr>
<td>26.2.1 Hardware/Software Requirements</td>
<td>1</td>
</tr>
<tr>
<td>26.2.2 Configuration File</td>
<td>2</td>
</tr>
<tr>
<td>26.2.3 Client Information</td>
<td>3</td>
</tr>
<tr>
<td>26.2.4 Security Considerations</td>
<td>3</td>
</tr>
<tr>
<td>26.2.5 Log Files</td>
<td>3</td>
</tr>
<tr>
<td>26.2.6 Troubleshooting</td>
<td>3</td>
</tr>
<tr>
<td>26.3 Client Installation and Configuration</td>
<td>4</td>
</tr>
<tr>
<td>26.3.1 Installing and Starting the Linux Client</td>
<td>4</td>
</tr>
<tr>
<td>26.3.1.1 Hardware Prerequisites</td>
<td>4</td>
</tr>
<tr>
<td>26.3.1.2 Software Prerequisites</td>
<td>4</td>
</tr>
<tr>
<td>26.3.1.3 Directory Structure</td>
<td>5</td>
</tr>
<tr>
<td>26.3.1.4 User awips</td>
<td>5</td>
</tr>
<tr>
<td>26.3.1.5 Repository Access and Setup</td>
<td>6</td>
</tr>
<tr>
<td>26.3.1.6 Installing Thin Client</td>
<td>7</td>
</tr>
<tr>
<td>26.3.1.7 Networking Configuration</td>
<td>8</td>
</tr>
<tr>
<td>26.3.1.8 Starting Thin Client</td>
<td>9</td>
</tr>
<tr>
<td>26.3.2 Installing the Win32 Client</td>
<td>9</td>
</tr>
<tr>
<td>26.3.3 CAVE Features Not Supported by Thin Client</td>
<td>18</td>
</tr>
<tr>
<td>26.3.4 Removing Configuration and Cached Files</td>
<td>18</td>
</tr>
<tr>
<td>26.3.5 Other Known Limitations</td>
<td>18</td>
</tr>
</tbody>
</table>

List of Exhibits

<table>
<thead>
<tr>
<th>Exhibit</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exhibit 26.3.2-1. Runtime Environment Setup Screen</td>
<td>10</td>
</tr>
<tr>
<td>Exhibit 26.3.2-2. End-User License Agreement</td>
<td>11</td>
</tr>
<tr>
<td>Exhibit 26.3.2-3. Install Screen for AWIPS II Runtime Environment Setup</td>
<td>11</td>
</tr>
<tr>
<td>Exhibit 26.3.2-4. Runtime Environment Setup (Ready to install)</td>
<td>12</td>
</tr>
<tr>
<td>Exhibit 26.3.2-5. Runtime Environment Setup (Installation)</td>
<td>12</td>
</tr>
<tr>
<td>Exhibit 26.3.2-6. Runtime Environment Setup (Installation Progress)</td>
<td>13</td>
</tr>
<tr>
<td>Exhibit 26.3.2-7. Runtime Environment Setup (Completion)</td>
<td>13</td>
</tr>
<tr>
<td>Exhibit 26.3.2-8. AWIPS II CAVE (Extraction)</td>
<td>14</td>
</tr>
</tbody>
</table>
Exhibit 26.3.2-9. AWIPS II CAVE Setup ................................................................. 14
Exhibit 26.3.2-10. AWIPS II CAVE Setup (End-User License Agreement) .......... 15
Exhibit 26.3.2-11. AWIPS II CAVE Setup (Custom Setup) ................................. 15
Exhibit 26.3.2-12. Windows Firewall Alert Dialog ............................................... 16
Exhibit 26.3.2-13. AWIPS II CAVE Setup (Ready to install) ................................. 16
Exhibit 26.3.2-14. AWIPS II CAVE Setup (Installation Screen) ......................... 17
Exhibit 26.3.2-15. AWIPS II CAVE Setup (Completion) ...................................... 17

List of Tables

Table 26.3.2-1. Win32 CAVE Hardware/Software Requirements ............................. 9
26.1 Introduction

The Thin Client component implements support for NWS enterprise requirements for remote access to baseline AWIPS II capabilities. The Thin Client will be able to take advantage of new and enhanced AWIPS II capabilities in a seamless manner as they are deployed to the baseline.

The Thin Client is targeted to the following users:

1. Center Weather Service Units (CWSU)
2. Weather Service Offices (WSO)
3. Incident Support Specialists (ISS) including Incident Meteorologists (IMET)
4. NCEP Centers, to provide partial support for their Continuity of Operations Planning (COOP) requirements and remote access requirements
5. River Forecast Centers (RFC), to provide partial support for backup requirements.

The Thin Client will also be used to brief state and local emergency managers during incident operations.

The Thin Client component adds the following features to CAVE/AlertViz:

- Support for access to EDEX via a single well-known port (e.g., http port 80). This allows CAVE to run on firewall restricted networks
- Support for compression of data sent to/from EDEX
- Local caching of localization, map, and weather data
- Timed refresh of data.

Both the Linux version and the Win32 version support all the features of CAVE and AlertViz.

This chapter describes the configuration needed to support the Thin Client component of CAVE.

26.2 Configuring the Apache Proxy Server

The Thin Client function of CAVE uses an Apache httpd server to perform data compression and map responses from EDEX Services and Pypies Ports to a single port. Users outside the NOAA firewall will be required to log in with their NOAA username and password.

26.2.1 Hardware/Software Requirements

Clients must be able to access the compression proxy server on port 80 for users inside the NOAA firewall and port 443 (https) for users outside the NOAA firewall.
The compression proxy server must be able to access EDEX on the “Services” and “Pypies” ports. The Apache httpd server providing the compression proxy service can also provide other web services. The server can run on the same system as EDEX.

Because the purpose of the compression proxy server is to compress the data stream between EDEX and the clients, it is important that the compression proxy server have a high-speed, high-capacity, low-latency connection to the EDEX servers.

Memory and CPU utilization will depend on the number of simultaneous users, but a machine with at least 2GB of RAM is recommended. There is no local storage of data; the only disk space required is space for the httpd installation and log files.

At WFOs that support CWSUs, the compression proxy server runs on pxlf. At sites that support external users (e.g., IMETs, DSS meteorologists) the compression proxy server runs on the LDAD server.

### 26.2.2 Configuration File

This section describes how to create a Configuration File. Two steps are involved:

1. Create a file called /etc/httpd/conf.d/thincclient.conf with the following contents where “edex_host” and “pypies_host” are replaced with the names or ip addresses of those machines.

   DeflateFilterNote Input instream
   DeflateFilterNote Output outstream
   DeflateFilterNote Ratio ratio

   LogFormat '%t "%r" %{outstream}n/%{instream}n
   (%{ratio}n)%' ' deflate

   CustomLog logs/compression_log deflate

   <Location /services>
     SetOutputFilter DEFLATE
   </Location>
   <Location /pypies>
     SetOutputFilter DEFLATE
     SetInputFilter DEFLATE
   </Location>

   ProxyPass /services http://edex_host:9581/services
   ProxyPassReverse /services http://edex_host:9581/services
   ProxyPass /pypies http://pypies_host:9582/
   ProxyPassReverse /pypies http://pypies_host:9582/

2. Start or reload the httpd service.

   # service httpd start
   or
   # service httpd reload
26.2.3 Client Information

When Thin Client is started for the first time, users will need to enter the addresses of the proxied services either in the Localization Dialog at startup or in the CAVE Preferences dialog after startup. The addresses will have the following format for users inside the NOAA firewall.

- Localization Server: http://hostname.noaa.gov/services
- Services: http://hostname.noaa.gov/services
- Pypies: http://hostname.noaa.gov/pypies

For users outside the NOAA firewall:

- Localization Server: https://hostname.noaa.gov/services
- Services: https://hostname.noaa.gov/services
- Pypies: https://hostname.noaa.gov/pypies

The string “hostname.noaa.gov” must be replaced with the name or IP address of the host running the Apache httpd proxy compression server.

26.2.4 Security Considerations

TBD. This will be site dependent.

26.2.5 Log Files

The httpd server by default creates access_log and error_log. compression_log is created when using the Apache httpd as a compression proxy server.

These files are created in /var/log/httpd and managed by Apache and logrotate.

- access_log: Logs all client requests.
- error_log: Logs errors from the proxy server.
- compression_log: Logs the amount of compression and service time.

The location, contents, and formatting of these logs are defined by the Apache configuration files.

26.2.6 Troubleshooting

The following three principal issues are unique to the Thin Client:

1. Can the Apache proxy successfully fetch data from the EDEX and Pypies servers? Errors of this kind will be logged in the “error_log”.
2. Can the Thin Client access the Apache proxy? Successful connections will be logged in the “access_log”. Unsuccessful connections are typically due to firewall or client connectivity issues and there will be nothing in the local logs.
3. Is compression working? Compression is logged in /var/log/httpd/compression_log. Only the first connection to the server is uncompressed; all others will show a compression percentage.

26.3 Client Installation and Configuration

26.3.1 Installing and Starting the Linux Client

This section provides instructions and information to support installation of the Thin Client on a Linux platform for various scenarios. If you are updating a previous installation (for example, on an AWIPS Remote Display (ARD) workstation at a CWSU) you can skip directly to Subsection 26.3.1.6, “Installing Thin Client.”

26.3.1.1 Hardware Prerequisites

Hardware requirements for running the Thin Client are almost the same as the requirements for running CAVE. (The Thin Client is, after all, CAVE running with some additional data compression and caching functionality.) The only additional requirement for a Thin Client workstation or laptop is to allow each user to store at least 2GB of cache files under his or her home directory.

The hardware requirements for Thin Client follow.

- At least 4GB memory. More is better. Performance improves with more memory up to about 12GB. Do not run more than one instance of Thin Client if your laptop has less than 12 GB of memory. [Note: These memory requirements apply to 64-bit builds of the Thin Client. You may be able to get by with 2GB of memory if you are running a 32-bit build.]

- Graphics adapter with at least 500 MB of dedicated on-board memory. More is better, up to about 1.5GB. Graphics adapter must support OpenGL 2.2 or later. Note: Nvidia adapters seem to work best. That is not an explicit requirement, but other graphics adapters seem not to work very well with AWIPS II.

- At least 2GB per user of cache storage space in the users’ home directories.

26.3.1.2 Software Prerequisites

Thin Client requires the same operating system (OS) and infrastructure software as CAVE. Thin Client has also been successfully installed on CentOS versions 5.5 through 5.9. Thin Client (that is, CAVE) builds are available for both 64- and 32-bit RHEL 5.x. Use the 64-bit OS if at all possible. That is the system used on the LXs, and that is the system for which the startup scripts and parameters are optimized.

At present CAVE cannot be installed on RHEL (or CentOS) 6.x. CAVE depends on certain packages (that is, RPMs) that are not included in default RHEL installations. At present those dependencies are not specified in the CAVE RPMs, so the yum installer will not automatically pull them in. Use the following command to verify that the packages are installed on the target platform:
rpm --query compat-readline43 openmotif compat-libf2c \libXp libgfortran

The packages are available in standard RHEL and CentOS repositories, so you can install them with commands like:

```
yum install compat-readline43
```

### 26.3.1.3 Directory Structure

The directory hierarchy in which Thin Client gets installed should be mounted at two places: /usr/local/viz and /awips2. The Eclipse installer does not understand symbolic links, so they cannot be used. Here is one possible approach.

Create a directory in which you will install Thin Client. You should do this on a partition with a lot of space available.

```
TYPE: export INST=/data/tc
TYPE: mkdir $INST
TYPE: mkdir /awips2
TYPE: mkdir /usr/local/viz
TYPE: mount --bind $INST /awips2
TYPE: mount --bind $INST /usr/local/viz
```

Add the necessary lines to /etc/fstab to have the directory mounted at boot time. Here are examples.

```
/data/tc  /awips2   ext3    bind
/data/tc  /usr/local/viz ext3    bind
```

**IMPORTANT NOTE:** These are only examples. In your case, they must be adapted to your platform. An incorrect fstab file can render the system unbootable.

### 26.3.1.4 User awips

Make sure you have User **awips** belonging to group fxalpha before you proceed.

Enter the command:

```
TYPE: id awips
```

You should see output like:

```
uid=501(awips) gid=501(fxalpha) groups=501(fxalpha)
```

The user and group numbers do not matter. If you do not have the group and user, create them with the following commands.

```
TYPE: groupadd fxalpha
```
26.3.1.5  Repository Access and Setup

The user will use the yum utility to install CAVE. Yum pulls the necessary RPM packages from a repository – a special directory hierarchy containing the packages and metadata. At a high level, the user needs to accomplish the following two tasks:

1. Create the repository if it does not already exist.
2. Create or edit the configuration file ("repo" files); this file tells yum where to find the repository and how to access it.

These two tasks can be accomplished in a number of ways. A few examples follow.

Example 1: Standard CWSU ARD Setup: Installation via http

The repository access and setup work for ARDs was done with the first installation of Thin Client on ARD. It need not be repeated. It is described here because it may be a useful starting point for other configurations.

The ARD workstations installed at CWSUs have network connections to their supporting WFOs, so they can use the repository in /data/fxa/INSTALL/awips2 at the WFO. This is the same repository that is used to install AWIPS II software at the WFO. Using this repository to install or update CAVE on the ARD is convenient, and it also ensures that the ARD gets a version of CAVE that is compatible with the EDEX at the WFO.

Create a file on the ARD called “/etc/yum.repos.d/awips2.repo”. The file should contain the following:

```bash
[awips2repo]
name=AWIPS II Repository
baseurl=http://px1f/tcrepo/dataFxa/INSTALL/awips2/$basearch
enabled=1
protect=0
gpgcheck=0

[awips2noarch]
name=AWIPS II Repository
baseurl=http://px1f/tcrepo/dataFxa/INSTALL/awips2/noarch
enabled=1
protect=0
gpgcheck=0
```

The "baseurl" lines in awips2.repo tell yum where to find the repository and how to access it. In this case the URL starts with “http”, which tells yum to use the http protocol to access the repository.
The server host name in the “baseurl” line is px1f. That host name must appear in the /etc/hosts file on the ARD, or it must be resolvable to an IP address by some other mechanism. You could also use an IP address instead of the host name.

The web server running on px1f at the WFO must be able to see the repository. There must be symbolic links in the web server’s document root hierarchy pointing at the repository. Those links should already exist. If they do not, they can be created by the root user working on px1f:

```
TYPE: mkdir -p /data/fxa/A2_SysMonitor/tcrepo
TYPE: chmod a+rx /data/fxa/A2_SysMonitor/tcrepo
TYPE: ln -s /data/fxa /data/fxa/A2_SysMonitor/tcrepo/dataFx
```

**Example 2: Put Client Machine on WFO LAN**

If you are installing Thin Client on a laptop you may be able to put the laptop on the LAN of a WFO temporarily. Typically you would do that by editing /etc/sysconfig/network and /etc/sysconfig/network-scripts/ifcfg-eth0. (This assumes that the ethernet connection is made on device eth0.)

Make backup copies of those files first so you can restore them later. Inspect the corresponding files on an LX to determine what entries to use. Find an unused IP address or temporarily use the address of a shut-down workstation. Reboot the laptop after editing those files. Once you have put the laptop on the WFO LAN, follow Example 1.

**Example 3: Use a Local Repository**

Connect an external USB drive to a WFO with access to /data/fxa/INSTALL/awips2 and copy that directory to the external drive. Then you can mount the drive on your target client machine.

Create the awips2.repo file as described in Example 1, except that you should use the file protocol rather than http in the baseurl lines and adjust the path to correspond with where you copied awips2 and what your mount point is. For example:

```
baseurl=file:///mnt/sdc1/awips2/$basearch
baseurl=file:///mnt/sdc1/awips2/noarch
```

Note that there are three slashes after the colon in the baseurl lines. You could also make a tar ball of the repository and expand it on the target machine.

**26.3.1.6 Installing Thin Client**

Clean out the yum cache and verify that the repository is set up properly:

```
TYPE: yum clean all
TYPE: rm -rf /var/cache/yum
TYPE: yum grouplist | grep "AWIPS II"
```
The output from the “yum grouplist” command should be a list of AWIPS RPM groups, one of which is “AWIPS II Visualize.” Install that group, and capture the output to a file for later inspection. Respond with a “y” to the prompt asking whether you want to do the installation. Replace the “<hhmm>” with a time stamp to make the script file name unique.

```bash
script /tmp/script.<hhmm>
yum groupupdate "AWIPS II Visualize"
exit  # Exit the scripting session.
```

Check the script file for problems.

```bash
TYPE: grep -i fail /tmp/script.<hhmm>
```

Edit the file.

```bash
TYPE: /etc/xdg/autostart/awips2-alertviz.desktop
```

Change the line:

```bash
Exec=/bin/bash -i -c "/awips2/alertviz/alertviz.sh"
```

to

```bash
Exec=/bin/bash -i -c "/awips2/alertviz/alertviz.sh -component thinalertviz"
```

AlertViz will now start in Thin Client mode automatically when a user logs in.

### 26.3.1.7 Networking Configuration

The Thin Client must be given an IP address for the compression proxy server. This address may be provided via the “Communications Preferences” dialog when AlertViz starts, or it may be provided in a CAVE preferences dialog. Refer to Chapter 14 of the *AWIPS CAVE-D2D User’s Manual* for information on providing the IP address via a CAVE preferences dialog. In either case, the user can provide either an IP address or a domain name. If a domain name is used it must appear in the /etc/hosts file, unless the client machine uses some other mechanism for domain name resolution. In a CWSU ARD installation, for example, there should be an /etc/hosts entry for px1f.

If the Thin Client uses JMS messaging to receive notifications of data updates, the host name “cp1f” must be resolvable. So this name also should appear in /etc/hosts.

**Note:** This applies only to CWSU ARDs. Other cases of use are restricted to http communication with the servers where JMS messaging is not available.

The host name of the Thin Client machine itself must be resolvable to an IP address on which the client machine can be contacted.

For example, if the user executes the following command:

```bash
TYPE: hostname
```
And receives an output as:

```
laptop01
```

Then the user must be able to ping laptop01 and get a response.

This can become an issue if the user’s client machine gets its IP address via DHCP and the user wants to give it a host name other than the generic “localhost”. In that case the user might have to edit the HOSTNAME line in `/etc/sysconfig/network` to be

```
.....HOSTNAME=laptop01
```

The user has to be sure that the name laptop01 is resolved to the IP address of the machine. The user could do that by adding laptop01 to the list of aliases for the loopback IP address in `/etc/hosts`, so that `/etc/hosts` line would look like

```
127.0.0.1       localhost.localdomain localhost laptop01
```

If the host name is not resolvable, then AlertViz will fail to start and the user will find a string as below in the `~/.caveData/logs/consoleLogs/alertviz*_admin.log`.

```
Caused by: java.net.UnknownHostException: laptop01
```

### 26.3.1.8 Starting Thin Client

AlertViz must be running before CAVE (Thin Client) is started. Ordinarily AlertViz will start at user login. If it is not running, start it manually with the command line:

```
/awips2/alertviz/alertviz.sh -component thinalertviz
```

The command line to Thin Client is:

```
/awips2/cave/cave.sh -component thinclient
```

It may be convenient to create a launcher (panel or desktop icon) to run that command.

### 26.3.2 Installing the Win32 Client

Table 26.3.2-1 lists the hardware and software requirements for the Win32 Thin Client.

<table>
<thead>
<tr>
<th>Specification</th>
<th>Minimum</th>
<th>Recommended</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processor</td>
<td>2GHz Intel Core Duo</td>
<td>2.5 GHz Intel i7 or better</td>
</tr>
<tr>
<td>Memory</td>
<td>2GB</td>
<td>4GB(^1)</td>
</tr>
<tr>
<td>Disk Space</td>
<td>Approx. 1GB for program and data caches</td>
<td>n/a</td>
</tr>
<tr>
<td>Operating System</td>
<td>Windows XP(^2)</td>
<td>Vista 32-bit, Windows 7 32-bit, Windows 7 64-bit</td>
</tr>
<tr>
<td>Video Card</td>
<td>OpenGL 2.0 capable</td>
<td>NVidia Quadro F series</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NVidia GeForce 6 or better</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NVidia m200 or better</td>
</tr>
</tbody>
</table>

\(^1\)CAVE is a 32-bit application and can only access 2GB of memory.

\(^2\)Windows XP only supports OpenGL 2.0 or better if implemented in a third-party video driver.
The Win32 install package consists of one Microsoft Windows Installer Package (msi) and one Win32 Cabinet Self-Extractor Application (.exe):

1. AWIPS II Runtime Environment.msi: installs Java, JAI, JAI ImageIO, Python, several python site-packages, and the system DLLs needed for CAVE on Win32. This must be installed before AWIPS II CAVE.EXE.

2. AWIPS II CAVE.EXE: Installs CAVE and AlertViz and places shortcuts on the user desktop and in the Windows Start Menu under Programs -> AWIPS II. The shortcuts default to starting in Thin Client mode.

Administrator privileges are needed for installation. To install, just double click on the installers and follow the prompts. The AWIPS II Runtime Environment.msi installer will install Java and Python to C:\Program Files\Raytheon (C:\Program Files (x86)\Raytheon on a 64-bit machine). The Runtime Environment will not make any changes to the system or user environment. AWIPS II CAVE.EXE will install CAVE and AlertViz to C:\Program Files\Raytheon (C:\Program Files (x86)\Raytheon on a 64-bit machine) by default; however, the installation location for CAVE can be changed at install time by altering the location specified in the Custom Setup dialog.

Exhibits 26.3.2-1 through 26.3.2.7 illustrate the Runtime Environment Installation, and Exhibits 26.3.2-8 through 26.3.2-15 illustrate the AWIPS II CAVE Installation.
Chapter 26

Thin Client

Exhibit 26.3.2-2. End-User License Agreement

Exhibit 26.3.3. Install Screen for AWIPS II Runtime Environment Setup
[Note: At this point, this prompt appears: “Do you want to allow the following program from an unknown publisher to make changed to this Computer” Yes/No. Choose ‘Yes’ to start the Installation.]
Exhibit 26.3.2-6. Runtime Environment Setup (Installation Progress)

Exhibit 26.3.2-7. Runtime Environment Setup (Completion)
Exhibit 26.3.2-8. AWIPS II CAVE (Extraction)

Exhibit 26.3.2-9. AWIPS II CAVE Setup
Chapter 26

Thin Client

Exhibit 26.3.2-10. AWIPS II CAVE Setup (End-User License Agreement)

Exhibit 26.3.2-11. AWIPS II CAVE Setup (Custom Setup)
A user who attempts to run a new installation of CAVE or AlertViz may be prompted by the Windows Firewall to allow access for “alertviz.exe” and “cave.exe”. The user should answer that access is allowed for private networks only. [See Exhibit 26.3.2-12.] [Note: The user need not be an Admin to allow the access. The IMET laptop configurations that it has been tested against allow the user to enable access to either network class.]

Exhibit 26.3-12. Windows Firewall Alert Dialog

Exhibit 26.3-13. AWIPS II CAVE Setup (Ready to install)
[Note: At this point, this prompt appears: “Do you want to allow the following program from an unknown publisher to make changes to this Computer” Yes/No. Choose ‘Yes’ to start the Installation.]

Exhibit 26.3-14. AWIPS II CAVE Setup (Installation Screen)

Exhibit 26.3-15. AWIPS II CAVE Setup (Completion)
26.3.3 CAVE Features Not Supported by Thin Client

The following CAVE features are not supported:

- Non-D2D perspectives, i.e., GFE, Hydro, MPE, National Centers, Localization
- WarnGen
- AvnFPS
- 4D Storm Cell Investigator (FSI)
- LAPS Tools
- Radar Application and Radar Tools
- Data dissemination and upload.

26.3.4 Removing Configuration and Cached Files

To start fresh with CAVE and remove all configuration or cached files, you will need to perform the following steps:

1. Shut down CAVE and AlertViz.
2. Remove caveData folder from your home directory. (ex: Linux: /home/<username>/caveData, Windows: C:\users\<username>\caveData).

26.3.5 Other Known Limitations

The Win32 client has less memory available to it than the Linux client. Some operations involving multiple large data sets may cause an out-of-memory error.
Chapter 27
Collaboration
Chapter 27. Collaboration

Table of Contents

27.1 Introduction .............................................................................................................................1
27.2 Collaboration Configuration ...................................................................................................1

List of Exhibits

Exhibit 27.2-1. Collaboration Server Login Dialog ................................................................. 2
Exhibit 27.2-2. Collaboration Server Login Dialog ................................................................. 2
Exhibit 27.2-3. Collaboration Site Roles Dialog ........................................................................ 3
Exhibit 27.2.4. Example: Config.xml BASE File ................................................................. 4
27.1 Introduction

Collaboration is a component of CAVE. It offers two main functions, chatting and sharing displays. “Chat” allows users to chat with fellow forecasters and offices in a chat room or send them instant messages. “Sharing displays” adds to the chat room’s capabilities and allows the room's creator to show a CAVE map display to other participants in the room.

The Collaboration capabilities in AWIPS II as a whole are replacing the capabilities of 12Planet. 12Planet will likely no longer exist at some point. The “NWS Collaboration Room” tab is a chat room within Collaboration; it is replacing the nws_collaboration chat room found in the 12Planet application. The “NWS Collaboration Room” tab opens by default once you are logged into Collaboration (users who have deactivated the “Join Discussion On Login” option in the Preferences dialog can open the NWS Collaboration Room by clicking the Display Feed button). The nws_collaboration chat room is the main/default room that appears when loading 12Planet. In other words, Collaboration is to 12Planet as the NWS Collaboration Room is to nws_collaboration.

This chapter provides instructions for configuring Collaboration in CAVE for a specific site.

Note: Configuration only needs to be completed once; the site will pick it up in its settings.

27.2 Collaboration Configuration

Collaboration’s config.xml file allows for site customization of the feed and login options for a user. The “feed” refers to the chat room that replaces the nws_collaboration room in 12Planet. The BASE file in the example below is only included as an example; the feed should be configured at a site level.

1. In CAVE, open the Localization perspective.
2. In the File Browser, navigate to CAVE -> Collaboration -> config.xml
3. Right click on BASE, and select Copy To Site (XXX)
   - If user does not have permission, the userRoles.xml file will need to be modified.
4. Edit the file as follows:
   - For each server that you want to add to the Server dropdown menu, add a server hostname line under <siteConfigInformation>.

   For example: <server hostname="<Enter Server Name>"
   prettyName="XXX Cluster"></server>
For each server configured, this will add a server to the following Collaboration Server Login dialog shown in Exhibit 27.2-1.

Exhibit 27.2-1. Collaboration Server Login Dialog

For each site that you want to configure as a login site, add a value of `<config site="XXX">` under `<siteConfigInformation>`.

For each site configured, this will add a site Collaboration Server Login dialog as shown in Exhibit 27.2-2. \textbf{Note:} Your site should go first, followed by the sites that you back up.

Exhibit 27.2-2. Collaboration Server Login Dialog

For each site that you want to see in the feed when logged in as that site, add `<subscribedSites>XXX</subscribedSites>` under the previous tag.
- The feed will contain every site in the same room. However, only messages from sites that are defined using the notation above will be shown in the chat window.

- Also under the `<config>` tag, for each role that you want available when logging in as that site, add `<roles><SITE ROLE></roles>`.

- For each site configured, this will add a role to the Collaboration Site Roles dialog as shown in Exhibit 27.2-3.

"Roles" should be similar to or the same as the current usernames in 12Planet, minus any site information (e.g., Short Term Forecaster, Long Term Forecaster).

5. Save the file by clicking the File -> Save.

To be more specific, see the following example as captured in the BASE config.xml file in Exhibit 27.2-4. **Note:** This is an example and will vary depending on your site.

```xml
<siteConfigInformation> // top tag
<server hostname="dx1-oma.oma.us.ray.com" prettyName="Omaha Cluster"></server>
<config site="OAX"> // the site being configured, will be available for logins
<subscribedSites>OAX</subscribedSites> // users will see text in the feed from this site
<subscribedSites>DMX</subscribedSites> // users will see text in the feed from this site
<roles>Short Term Forecaster</roles> // users will have the option to login as this role
</siteConfigInformation> // end tag
```
Exhibit 27.2.4. Example: Config.xml BASE File
Chapter 28
WES-2 Bridge
Chapter 28. WES-2 Bridge

Table of Contents

28.1 Introduction ......................................................................................................................... 1
28.2 Installation .......................................................................................................................... 1
28.3 The EDEX-Environment Macro ......................................................................................... 2
28.4 EDEX Environment Configuration ..................................................................................... 3
28.5 EDEX Environment Management ...................................................................................... 4
28.6 Uninstalling AWIPS II EDEX Environment ..................................................................... 5
28.7 Archive Retention ............................................................................................................... 6
28.8 Archive Retention Dialog .................................................................................................. 6
28.9 Archive Retention Configuration through the Dialog ...................................................... 7
28.10 Localization Perspective Archive Directory ...................................................................... 7
28.11 Localization Perspective Archive Configuration File Viewing ...................................... 7

List of Exhibits

Exhibit 28.2-1. A Successful awips2-edex-environment Installation........................................ 2
Exhibit 28.3-1. The Default Output of the awips2-edex-environment Macro ............................ 2
Exhibit 28.8-1. Archive Retention ............................................................................................. 6
Exhibit 28.11-1. Localization Perspective Archive File Viewing .............................................. 8

List of Tables

Table 28.4-1. edex-environment Configuration File Field Names ............................................. 3
Table 28.4-2. Default Ports ......................................................................................................... 4
28.1 Introduction

The EDEX environment is used to create additional instances of EDEX (each instance of EDEX has its own database, its own file store, its own instance of Pypies, and QPID) based on the currently installed EDEX for the purpose of running a WES-2 Bridge test case. The EDEX environment macro should be used every time a new EDEX environment is needed to run a WES-2 Bridge test case.

You will need root access to run the EDEX environment macro, so that anyone at a WFO with root access will be able to utilize the macro. The EDEX environment macro will allow you to create, start, stop, and delete EDEX environment instances. A configuration file is needed to create an EDEX environment initially, and the same configuration file can be used to control the instance. The configuration file is described in this chapter.

Additionally, an EDEX environment can also be controlled by name. The EDEX environment macro should be used to configure a new EDEX environment for the WES-2 Bridge. Follow the procedures outlined in this chapter to prepare the EDEX environment to run a WES-2 Bridge test case.

Note: A standalone EDEX server will need to be running to allow archive data. The workstations will need to use this EDEX server. Workstations that are used to run the Archive Retention and/or Case Creation GUIs will need to mount the server’s /data_store and /archive directories.

28.2 Installation

Note: The awips2-edex-environment rpm must be present in the repository you will use for the installation.

To install awips2-edex-environment:

1. Log into a machine with a standalone installation of AWIPS II.
   
   ssh root@adam-wes2

2. Clear the yum cache.
   
   yum clean all

3. Install awips2-environment.
   
   yum install awips2-edex-environment

See Exhibit 28.2-1.
28.3 The EDEX-Environment Macro

1. Log out of the machine that you installed awips2-edex-environment on and log back in.
2. Verify that the edex-environment macro is available.

   edex-environment

See Exhibit 28.3-1.
28.4 EDEX Environment Configuration

The edex-environment configuration file is a simple xml document, with this layout:

```xml
<?xml version="1.0" encoding="ISO-8859-1" ?>
<Wes2BridgeCase>
  <name></name>
  <dataArchiveRoot></dataArchiveRoot>
  <databasePort></databasePort>
  <edexHttpPort></edexHttpPort>
  <qpidHttpPort></qpidHttpPort>
  <qpidJmxPort></qpidJmxPort>
  <jmsPort></jmsPort>
  <webPort></webPort>
  <confidentialPort></confidentialPort>
  <httpdPypiesPort></httpdPypiesPort>
  <pypiesLoggingPort></pypiesLoggingPort>
</Wes2BridgeCase>
```

Table 28.4-1 describes the field names used in the file; Table 28.4-2 describes the default ports used by the edex-environment and other AWIPS II components not configurable by edex-environment.

Table 28.4-1. edex-environment Configuration File Field Names

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>An alphanumeric character sequence used to name an edex environment. Each edex environment must have a unique name.</td>
</tr>
<tr>
<td>dataArchiveRoot</td>
<td>A directory path equivalent to the DATA_ARCHIVE_ROOT in setup.env.</td>
</tr>
<tr>
<td>databasePort</td>
<td>A numeric value representing the port that PostgreSQL should accept connections on for a particular edex environment – equivalent to the DB_PORT in setup.env. Each edex environment must have a unique databasePort.</td>
</tr>
<tr>
<td>edexHttpPort</td>
<td>A numeric value representing the port that the EDEX localization server will accept connections on – equivalent to the HTTP_PORT in setup.env. Each edex environment must have a unique edexHttpPort.</td>
</tr>
<tr>
<td>qpidHttpPort</td>
<td>A numeric value representing the QPID REST services port. Each edex environment must have a unique qpidHttpPort.</td>
</tr>
<tr>
<td>qpidJmxPort</td>
<td>A numeric value representing the port that the QPID Registry Server will accept connections on – equivalent to the port specified at the end of JMS_SERVER in setup.env. Each edex environment must have a unique qpidJmxPort.</td>
</tr>
<tr>
<td>jmsPort</td>
<td>A numeric value representing the port that QPID will accept connections on – equivalent to the port specified at the end of JMS_SERVER in setup.env. Each edex environment must have a unique jmsPort.</td>
</tr>
<tr>
<td>webPort</td>
<td>A numeric value representing the port on which embedded web components (example: jetty) will listen for a connection. Each edex environment must have a unique webPort.</td>
</tr>
<tr>
<td>confidentialPort</td>
<td>A numeric value representing an additional port that must be specified for the embedded web components (example: jetty). Each edex environment must have a unique confidentialPort.</td>
</tr>
<tr>
<td>httpdPypiesPort</td>
<td>A numeric value representing the port that httpd-pypies should listen for connections on. Each edex environment must have a unique httpdPypiesPort.</td>
</tr>
<tr>
<td>pypiesLoggingPort</td>
<td>A numeric value representing the port that the pypies logger should accept connections on. Each edex environment must have a unique pypiesLoggingPort.</td>
</tr>
</tbody>
</table>
Table 28.4-2. Default Ports

<table>
<thead>
<tr>
<th>Name</th>
<th>Default Ports Used by edex-environment</th>
</tr>
</thead>
<tbody>
<tr>
<td>databasePort</td>
<td>5432</td>
</tr>
<tr>
<td>edexHttpPort</td>
<td>9581</td>
</tr>
<tr>
<td>qpidHttpPort</td>
<td>8180</td>
</tr>
<tr>
<td>qpidJmxPort</td>
<td>8999, 9099</td>
</tr>
<tr>
<td>jmsPort</td>
<td>6572</td>
</tr>
<tr>
<td>webPort</td>
<td>8080</td>
</tr>
<tr>
<td>confidentialPort</td>
<td>8443</td>
</tr>
<tr>
<td>httpdPypiesPort</td>
<td>9582</td>
</tr>
<tr>
<td>pypiesLoggingPort</td>
<td>9020</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Default Ports Used by Other AWIPS II Components</th>
</tr>
</thead>
<tbody>
<tr>
<td>dataDelivery</td>
<td>9588 and 10144</td>
</tr>
<tr>
<td>ncfBandwithMgr</td>
<td>9590</td>
</tr>
</tbody>
</table>

28.5 **EDEX Environment Management**

Only a user with root privileges is able to manage EDEX environments.

In the example commands below:

- `${CONFIG}` refers to a configuration file on the filesystem.
- `${NAME}` refers to the name of an EDEX environment.

**Creating an EDEX Environment**

Before you can create an edex environment, you will need to create a configuration file (Refer to Section 28.3 of this chapter).

Run the following command to create an EDEX Environment:

```
edex-environment -create ${CONFIG}
```

**Starting an EDEX Environment**

There are two ways to start an EDEX environment. One way is to run the following command to start an EDEX environment based on the configuration file that was used to create it:

```
edex-environment -start ${CONFIG}
```

You can also, optionally, start an EDEX environment by name:

```
edex-environment -start -name ${NAME}
```

You cannot start an EDEX environment that does not exist.
**Stopping an EDEX Environment**

There are two ways to stop an EDEX environment. One way is to run the following command to stop an EDEX environment based on the configuration file that was used to create it:

```
edex-environment -stop ${CONFIG}
```

You can also, optionally, stop an EDEX environment by name:

```
edex-environment -stop -name ${NAME}
```

You cannot stop an EDEX environment that is not running, and you cannot stop an EDEX environment that does not exist.

**Deleting an EDEX Environment**

There are two ways to delete (remove) an EDEX environment. One way is to run the following command to delete an EDEX environment based on the configuration file used to create it:

```
edex-environment -remove ${CONFIG}
```

You can also, optionally, delete an EDEX environment by name:

```
edex-environment -remove -name ${NAME}
```

You cannot delete an EDEX environment that does not exist. Deleting an EDEX environment while it is still running is not recommended.

**Viewing All Available EDEX Environments**

To view a list of available EDEX environments, run the following command:

```
edex-environment –list
```

### 28.6 Uninstalling AWIPS II EDEX Environment

To uninstall awips2-edex-environment, complete the following steps:

1. Log into the machine that awips2-edex-environment has been installed on:
   
   ```
   ssh root@adam-wes2
   ```

2. Remove awips2-edex-environment
   
   ```
   yum remove awips2-edex-environment
   ```

3. Clean up any files that remain.
   
   ```
   rm –rf /awips2/edex-environment
   rm –rf /usr/local/edex-environment
   ```
28.7 Archive Retention

Archive Retention is the management of data deletion (purging) from the file system. It is defined by configuration files stored in the EDEX Localization subsystem. The configurations can be edited through the Archive Retention Dialog, which is accessible through the menu CAVE > Archive Retention..., or manually by editing the XML configuration file through the CAVE Localization Perspective. Manual edit is not recommended.

Permission to access Archive Retention and Case Creation is under the control of AWIPS User Administration. It is controlled under the component “Data Archiving.”

A quartz cron timer runs the Archive Purger. This timer is a configurable property in the /awips2/edex/conf/resources/com.raytheon.uf.edex.archive.cron.properties file.

    archive.purge.cron=0+5+0/3++*++?

This is the delivered baseline value, which configures it to run once every 3 hours.

To enable the purger in the same file,

    archive.purge.enable=true

28.8 Archive Retention Dialog

The Archive Retention dialog (shown in Exhibit 28.8-1) is the configuration GUI for visual editing of the data retention XML configuration files.

Exhibit 28.8-1. Archive Retention
28.9 **Archive Retention Configuration through the Dialog**

The purge times of Archive Data and Data Store files are configured by the Archive Retention Dialog. From top to bottom, the Archive Retention Configuration Dialog is divided into three sections. Use the top section of the dialog to select the Archive configuration, a specific **Category** for that Archive configuration, a **Minimum Retention** for the Archive, and an **Extended Retention** value for any table items selected in the Category element table (the middle section, described below). The Retention values can be entered and displayed in **Hours** or **Days** by selecting either value from the selection box. The retention time indicates how long a data file is retained before the Archive Purger deletes it.

The middle section contains a table that displays the possible **elements** of the currently selected Category in the Archive. The rows for each element are made up of a check box, a **Label**, and the **Current Size**. Checking the box next to a Label indicates that the Extended Retention value applies to the data related to that Label. Current size is populated by a background thread and may not be immediately available. If it is not yet calculated, then “????” is displayed instead.

The bottom section is for saving a configuration change for the current Archive. It has **Save**, **Close**, and **Recompute Sizes** buttons. The Save button is activated only if changes are made in the UI to the configuration. Changes are saved when the Save button is clicked. (The changes are saved to a SITE level file.) The dialog closes when the Close button is clicked. Computing the sizes is a slow process. Once the size is computed for a category’s label, it is not recalculated even if the underlying directory structure changes. Clicking on the **Recompute Size** button will reset all values to “unknown,” which will then restart a background job to redetermine the sizes for the labels. Basically it updates the label sizes in the Archive Case Creation dialog.

28.10 **Localization Perspective Archive Directory**

The data retention XML configuration files can be viewed through the CAVE Localization perspective in an Archive icon folder.

28.11 **Localization Perspective Archive Configuration File Viewing**

Archive configuration files can be viewed in the CAVE Localization Perspective under **Archive > Archive**. Each XML configuration file defined for Archive Retention is listed under the second Archive folder dropdown.

Two files are delivered with AWIPS II: **PROCESSED_DATA.xml** and **RAW_DATA.xml** (as shown in the middle screen of Exhibit 28.11-1). In the delivered configuration, these files are located in `/awips2/edex/data/archive`. The file **PROCESSED_DATA.xml** is the configuration for retention of Archiver output files. **RAW_DATA.xml** is for the ingest directory `/data_store`. Each of these delivered XML
configuration files has a header that explains how they are written and how a user can add elements to them.

Exhibit 28.11-1. Localization Perspective Archive File Viewing
Chapter 29
Data Delivery System
Administrator’s Guide
Chapter 29. Data Delivery System Administrator’s Guide

Table of Contents

29.1 General Data Delivery Administrative Functions ................................................................. 1
  29.1.1 Simple Start and Stop ............................................................................................... 1
  29.1.2 Configure a Proxy if Necessary ................................................................................ 1
  29.1.3 Registry Configuration ............................................................................................ 2

29.2 Configuration of Data Delivery .......................................................................................... 2
  29.2.1 Harvester Configuration .......................................................................................... 2
  29.2.2 OPeNDAP Service Configuration .......................................................................... 4
  29.2.3 Bandwidth Manager Configuration ....................................................................... 4
  29.2.4 Model/Level Lookup Configurations ..................................................................... 5
  29.2.5 Unit Configurations ............................................................................................... 7

29.3 Monitoring (Notification) and Statistics .......................................................................... 7
  29.3.1 Registry Statistics .................................................................................................... 8
  29.3.2 Subscription/Data Retrieval Statistics ..................................................................... 8

29.4 FAQ ...................................................................................................................................... 9

List of Tables

Table 29.2.1-1. Harvester Tunable Parameters and Purpose ......................................................... 3
Data Delivery has been implemented into the AWIPS II baseline for Initial Operating Capability (IOC) 0 to provide access to data that is independent of its location (that is, data that is not resident locally at a Weather Forecast Office, River Forecast Center, or National Center).

Data Delivery gives users the ability to create queries (One Time Requests) and subscriptions to data sets provided by the NOAA Operational Model Archive and Distribution System (NOMADS) data provider.

29.1 General Data Delivery Administrative Functions

29.1.1 Simple Start and Stop

In most cases Data Delivery can be started from the edex_camel init.d script. The command is as follows.

/etc/init.d/edex_camel start registry

In most cases Data Delivery can be stopped from the edex_camel init.d script. The command is as follows.

/etc/init.d/edex_camel stop registry

29.1.2 Configure a Proxy if Necessary

In the event your network is located behind a web proxy, a proxy configuration file is available for configuration.

This file is located in the following directory.

$EDEX_HOME/data/common_static/configured/$SITE/datadelivery

In the case of a Thin Client user, the configuration of the proxy is done from the EDEX server you are connecting to, not from your local network. No configuration on your side should be necessary.

Here is an example using the Omaha Raytheon network setup.

# Uncomment the lines below and enter the correct values to enable the use
# of a proxy

http.proxyHost=proxy.ext.ray.com

http.proxyPort=80

If your site is not located behind a proxy, comment the above lines out and no web proxy will be used in the HTTP connections for Data Delivery crawler and retrieval connections.

---

1 As of the release date of this edition of the System Manager’s Manual, the Data Delivery application was not yet operational.
29.1.3 Registry Configuration

For IOC 0, the Data Delivery Registry will not be federated in any way. Therefore, the configuration for the registry communication is quite minimal.

The registry is located in the registry.xml file located in the following directory.

$EDEX_HOME/data/common_static/base/datadelivery/registry/registry.xml

An example configuration follows.

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<registry>
  <lcmUrl>http://localhost:10144/lcm?WSDL</lcmUrl>
</registry>
```

If your Data Delivery registry runs on a different physical host than the standard setup, you will need to edit the “localhost” entry to whatever the FQDN of that host is in order for the Data Delivery clients to access the registry.

29.2 Configuration of Data Delivery

29.2.1 Harvester Configuration

The IOC 0 configuration of Data Delivery contains a web crawling harvester that finds potential data sets and submits them to the Registry for storage. There are many configurable pieces to this harvester, the first of which is the timing for when a crawler will be triggered to reach out to its provider and begin harvesting data. This is controlled by two crontab-like entries located in each harvester’s provider-specific configuration file. At install, these files by default are located in

$EDEX_HOME/data/common_static/base/datadelivery/harvester.

Under this directory, you will notice the NOMADS provider-specific file. The NOMADS file is the only operational provider. If this is the first time you have run Data Delivery or if you have done a complete fresh installation, these files will need to be updated. If you have run IOC 0 versions of Data Delivery before, you will need to edit the files in the $SITE localization level. They are the ones that are read in an operational state by Data Delivery. The base level files are merely used as starting points and are copied with local alterations to the $SITE localization after the first Seed Scan is performed. (It is generally after a day’s worth of operation.)

Open the file in an editor of your choice and look for the two entries that look like crontab.

```xml
<seedScan>0 0 0 * * ?</seedScan>
<mainScan>0 \12 * * * ?</mainScan>
```
A “Seed Scan” is a periodic total check of the provider site. It is done to check for new and/or updated data sets. The general rule is to set the Seed Scan to occur once daily – if possible, at a relatively quiet time of operation. If data on the provider changes very little over time, the scan can be set to occur less frequently (as little as once monthly if directed).

The “Main Scan” is the scan that finds and directs data sets to the registry for handling. In testing, it was determined that it takes, on average, less than 12 minutes to perform a complete scan of all data set collections on the NOMADS provider site. Hence, the scans are timed to happen 12 minutes apart, 3 times per hour by default. The frequency, however, is left to your discretion. One scan per hour may actually be sufficient. Most of the XML tags in this document are tunable, although some are not. The list of tunable parameters and what they do is available in Table 29.2.1-1.

Table 29.2.1-1. Harvester Tunable Parameters and Purpose

<table>
<thead>
<tr>
<th>Tunable Parameters</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;timeBetweenCrawlRequests&gt;500&lt;/timeBetweenCrawlRequests&gt;</code></td>
<td>Represents a delay between crawl requests in milliseconds. It is done as a courtesy to the provider so as not to act like a Distributed Denial of Service (DDoS) attack when browsing the website. It avoids locking up all available bandwidth both to the site and from the client.</td>
</tr>
<tr>
<td><code>&lt;postedFileDelay&gt;3 HOURS&lt;/postedFileDelay&gt;</code></td>
<td>A delay designed to allow temporal comparisons for files that traverse the day boundary.</td>
</tr>
<tr>
<td><code>&lt;ignore&gt;GDAS-BUFR&lt;/ignore&gt;</code></td>
<td>Used to designate data set collections on a provider site that you wish to ignore. If you place a collection title in as an “ignore,” then the crawler will not crawl anything containing that string in its URL. This is very useful for completely ignoring data sets that have no value to you. Find a string of characters that are unique to that data set and place it in an ignore tag and it will never be crawled again. Several examples of this in the active NOMADS configuration are the “rap_f” and “ruc” ignores. One ignores data sets that contain the character string “rap_f.” The other has the consequence of ignoring an entire model family, “ruc”.</td>
</tr>
<tr>
<td><code>&lt;useRobots&gt;true&lt;/useRobots&gt;</code></td>
<td>Signifies to the crawler agent that this provider site has a “Crawler configuration” available. This means that the site has a set of rules that allow the crawling agent to correctly traverse the site according to its rules. NOMADS has a robots.txt. You may encounter providers that do not have this feature.</td>
</tr>
<tr>
<td><code>&lt;collection projection=&quot;LatLon&quot; dataType=&quot;Grid&quot; ignore=&quot;false&quot; name=&quot;NCEP_GFS&quot; seedUrl&gt;NCEP_GFS&lt;/seedUrl&gt;</code></td>
<td>The Collections themselves have a couple of parameters that you can alter manually. The “ignore” attribute can be set to true if you wish to ignore this configured data set collection. You can also alter the “lastDate.” On occasions where out-of-cycle or irregular updates occur, it can sometimes make the difference between a data set being available and not being recognized. In normal running for the NOMADS provider, these can be left alone.</td>
</tr>
</tbody>
</table>
### Tunable Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;ingestNew&gt;true&lt;/ingestNew&gt;</code></td>
<td>If you recognize a new collection during a Seed Scan, ‘Should I set ignore to false and allow Main Scan's to pick up data sets for it?’ Default for a newly discovered collection is “true”. This means that a new collection discovered during a Seed Scan will automatically be added to the collection list in the Main Sequence Scan and scanned on the schedule of the Main Sequence Scan.</td>
</tr>
<tr>
<td><code>&lt;maxSeedDepth&gt;-1&lt;/maxSeedDepth&gt;</code></td>
<td>Sets the seed scan directory traversal depth. -1 means unlimited.</td>
</tr>
<tr>
<td><code>&lt;maxSeedPages&gt;-1&lt;/maxSeedPages&gt;</code></td>
<td>Sets the seed scan max number of pages traversed. -1 means unlimited.</td>
</tr>
<tr>
<td><code>&lt;maxMainDepth&gt;2&lt;/maxMainDepth&gt;</code></td>
<td>Sets the main scan directory traversal depth. -1 means unlimited.</td>
</tr>
<tr>
<td><code>&lt;maxMainPages&gt;1000&lt;/maxMainPages&gt;</code></td>
<td>Sets the main scan max number of pages traversed. -1 means unlimited.</td>
</tr>
</tbody>
</table>

### 29.2.2 OPeNDAP Service Configuration

The IOC 0 configuration of Data Delivery contains a configuration file for the OPeNDAP service. This file is located in the directory `$EDEX_HOME/data/common_static/base/datadelivery/harvester`. The file, which is named OPENDAPServiceConfig.xml, contains a list of GrADS Data Server (GDS, formerly known as GrADS-DODS server) constants and specific dataset overrides that are used by the harvester Metadata Handler in processing OPeNDAP derived gridded data. Some of the tags are specific to data set names, and others are more for the purpose of searching for a particular collection type. Most of the information contained in this file is specific to connecting to the NOMADS GraDS Data server. The constants in particular are for that purpose, and very little configuration is necessary for this file.

### 29.2.3 Bandwidth Manager Configuration

The AWIPS II Data Delivery server comes equipped with an engine for managing and parceling out bandwidth to subscription retrievals. Most of this system is transparent to the user. However, some portions are exposed to configuration. The main Bandwidth configuration file is the bandwidthmap.xml.

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!-- bandwidth is specified in kilobytes/second (kB/s) -->
<bandwidthMap>
  <route planDays="2" network="OPSNET" defaultBandwidth="768" bucketSizeMinutes="3"/>
</bandwidthMap>
```

The relevant features here are the “planDays,” which indicate how far out the Bandwidth Manager will schedule. The network, in this case, is OPSNET (the other possibility is “SBN”). The `<defaultBandwidth>` is the amount of bandwidth the manager will allocate.
for Data Delivery retrieval traffic. The <bucketSizeMinutes> is the size of the window used to compute the amount of traffic in kB/s for that given number of minutes. So subscriptions or ad hoc requests coming due in a window (bucket) are considered for sizing into those windows (buckets) it will take to complete the retrieval while staying below the default Bandwidth limit. Another consideration in this equation is the priority. Lower-priority Subscriptions will be shifted further into the future and considered after Higher-priority Subscriptions for a given “bucket.” The Bandwidth Manager will continue to shift subscriptions further into the future until they exceed the allowable Latency tolerance, the final consideration in the bandwidth usage. Latency can be specifically set up for that subscription, model, user, etc. Custom Latency rules are saved in the following file.

~common_static/site/$SITE/datadelivery/systemManagement/rules/latencyRules.xml

This is an example of a latency rule. Note that the latency is measured in minutes.

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<latencyRules>
  <rule xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="latencyRuleXML">
    <ruleName>GFS 1.0 latency rule</ruleName>
    <ruleField>gfs</ruleField>
    <ruleOperator>Like</ruleOperator>
    <ruleValue>GFS</ruleValue>
    <latency>30</latency>
  </rule>
</latencyRules>
```

When you create a new Subscription/Ad hoc request that exceeds the configured latency rules, you can elect to have the retrieval exceed the allowable rules and still execute. Then, if available bandwidth is available in some other window in the future, your retrieval will be executed. The administration and maintenance of the Latency and other Bandwidth functions are available through the Data Delivery System Management GUI interface. It is located under the Data Delivery header of the CAVE menu.

### 29.2.4 Model/Level Lookup Configurations

In the Data Delivery EDEX configuration, it was necessary to have Gridded parameter translations and level lookups to connect effectively to the NOMADS GrADs server. For IOC 0, a standard set of parameter lookup XML files is contained in the $EDEX_HOME/data/common_static/base/datadelivery/lookups directory. All of the model/groups currently hosted on the NOMADS site have a corresponding lookup file. In the event NOMADS adds model/group additions to NOMADS, new parameter configuration entries are auto-generated by the crawler upon discovery.

A sample model/group lookup follows. (This is an example from RTMA model.)

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ParameterLookup>
```
The default for a new parameter is to have the AWIPS and GrAD names the same. These are the GrADs name from the NOMADS server. Human (administrative) interaction is required to update a model/group configuration. You can edit the AWIPS naming to match the naming with an AWIPS II-recognized parameter definition. New or updated model/group parameter XML files are placed into the

$SEDEX_HOME/data/common_static/$SITE/datadelivery/lookups. The files in $SITE take precedence over base-level files like normal AWIPS II EDEX localization.

The other type of lookup configuration is the Level lookup. These lookup XML files normally do not require any human (administrative) interaction. They are auto-generated upon first discovery and placed into the

$SEDEX_HOME/data/common_static/$SITE/datadelivery/lookups. These files are created directly from queries to the NOMADS GrADs server so no editing should be necessary. Here is an example of a level lookup XML file (GFS levels).

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<LevelLookup>
  <Level>1000.0</Level>
  <Level>975.0</Level>
  <Level>950.0</Level>
  <Level>925.0</Level>
  <Level>900.0</Level>
  <Level>850.0</Level>
  <Level>800.0</Level>
  <Level>750.0</Level>
  <Level>700.0</Level>
  <Level>650.0</Level>
  <Level>600.0</Level>
  <Level>550.0</Level>
  <Level>500.0</Level>
  <Level>450.0</Level>
  <Level>400.0</Level>
  <Level>350.0</Level>
  <Level>300.0</Level>
  <Level>250.0</Level>
  <Level>200.0</Level>
  <Level>150.0</Level>
  <Level>100.0</Level>
</LevelLookup>
```
29.2.5 Unit Configurations

The Data Delivery EDEX has built-in accommodation for incorrect SI (International System of Units) unit designations that are sometimes accompanied by provider data set parameters. A list of discovered NOMADS unit inconsistencies and their corrections are contained within the UnitConfig.xml file. This file exists in the SEDEX_HOME/data/utility/common_static/$SITE/datadelivery/lookups directory. If you discover a parameter that may not be displaying correctly in CAVE, and the data is current and was successfully retrieved, it may very well have an incorrect unit designation that the AWIPS II unit converter does not recognize. The AWIPS II unit converter should recognize all SI standard units. NOMADS testers have created unit configuration overrides for non-SI units that they identified, and they are included in the standard file that you get with the install (displayed below). You are free to add to this list of overrides the correct SI unit designations that you discover on NOMADS and other future providers.

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!-- Correct any inconsistencies in the units -->
<unitLookup>
  <unitConfig name="K" providerName="k"/>
  <unitConfig name="Pa" providerName="pa"/>
  <unitConfig name="Pa" providerName="pascals"/>
  <unitConfig name="Pa/s" providerName="pa/s"/>
  <unitConfig name="J/kg" providerName="j/kg"/>
  <unitConfig name="W/m^2" providerName="w/m^2"/>
  <unitConfig name="m/s" providerName="meter/sec"/>
  <unitConfig name="N/m^2" providerName="newton/m2"/>
  <unitConfig name="N/m^2" providerName="n/m^2"/>
  <unitConfig name="C" providerName="degC"/>
  <unitConfig name="C-m/s" providerName="degC-m/s"/>
  <unitConfig name="kts/200ft" providerName="knots/200feet"/>
  <unitConfig name="km^2kg-s^-1" providerName="km2kg-1s-1"/>
  <unitConfig name="K/s" providerName="k/s"/>
  <unitConfig name="m^2/s" providerName="m2/s"/>
</unitLookup>
```

29.3 Monitoring (Notification) and Statistics

Within Data Delivery EDEX, there are two types of data that are being harvested for statistics. These are in addition to the standard ingest plugin statistics that are gathered by EDEX in general. These statistics are intended to be used to help tune the Data Delivery
EDEX system. They can be helpful in Bandwidth Management and Subscription Scheduling, and as a general guide to tracking the system for Data Delivery.

29.3.1 Registry Statistics

The registry statistics default configuration file is deployed to the $EDEX_HOME/data/utility/edex_static/base/stats directory. The file name is registryProcessStats.xml. The editable fields in this file are in the <statisticsGroup> tags. With the current file, all of the current groupings are listed, which means that statistics gathered for Data Delivery registry transactions will be grouped by Owner, Status, and Type of Transaction. Statistics for the registry are available for view in the CAVE menu. An example registry statistics file follows.

```
<statisticsConfig>
  <!-- Event Type should be fully qualified name of stat event -->
  <statisticsEvent type="com.raytheon.uf.common.registry.event.RegistryStatisticsEvent"
    displayName="Registry Statistics" category="Registry">
    <statisticsGroup name="owner" displayName="Transaction Owner" />
    <statisticsGroup name="status" displayName="Transaction Status" />
    <statisticsGroup name="type" displayName="Transaction Type" />
    <statisticsAggregate field="duration"
      displayName="Total Registry Duration" displayUnit="Minutes" />
  </statisticsEvent>
</statisticsConfig>
```

Important Note: The only things that are editable in the XML file are displayUnit and DisplayName. <statisticsGroups> can also be removed. However, be advised that you are removing fields that are currently being used to gather statistics and thus you will lose some data compared to the default settings. An EDEX restart is required for changes to appear in registry statistics.

29.3.2 Subscription/Data Retrieval Statistics

In Data Delivery the most important performance metrics for tuning Bandwidth Management are the Data Retrieval and overall Subscription Retrieval statistics. From these two Statistics groupings, you can determine four things:

1. Number of successful subscriptions vs. failed subscriptions.
2. Number of successful retrievals vs. failed retrievals.
3. Successful subscriptions that had data retrieval failures.
4. Failed subscriptions with some successful retrievals.

In this way, you can pinpoint the problems with the Bandwidth Management and adjust available bandwidth or latency accordingly. The retrieval/subscription statistics default configuration file is deployed to the $EDEX_HOME/data/utility/edex_static/base/stats directory. The file name is retrievalProcessStats.xml. An EDEX restart is required for changes to appear in retrieval/subscription statistics.
<statisticsConfig>
  <!-- Event Type should be fully qualified name of stat event -->
  <statisticsEvent
type="com.raytheon.uf.common.datadelivery.event.retrieval.SubscriptionRetrievalEvent"
displayName="Subscription Retrieval" category="Data Delivery">
    <statisticsGroup name="plugin" displayName="Data Type" />
    <statisticsGroup name="provider" displayName="Data Provider" />
    <statisticsGroup name="owner" displayName="Owner" />
    <statisticsGroup name="network" displayName="Network Route" />
    <statisticsGroup name="subscriptionType" displayName="Subscription Type" />
    <statisticsAggregate field="numFailed" displayName="Number of Failed Subscriptions" />
    <statisticsAggregate field="numComplete" displayName="Number of Completed Subscriptions" />
  </statisticsEvent>

  <statisticsEvent
type="com.raytheon.uf.common.datadelivery.event.retrieval.DataRetrievalEvent"
displayName="Data Retrieval" category="Data Delivery">
    <statisticsGroup name="plugin" displayName="Data Type" />
    <statisticsGroup name="provider" displayName="Data Provider" />
    <statisticsGroup name="owner" displayName="Owner" />
    <statisticsGroup name="network" displayName="Network Route" />
    <!-- Display unit options are KB, MB, GB -->
    <statisticsAggregate field="bytes" displayName="Amount of Data Downloaded" displayUnit="MB" />
    <statisticsAggregate field="numRecords" displayName="Number of Records Downloaded" />
  </statisticsEvent>
</statisticsConfig>

29.4 FAQ

My Data Delivery does not seem to have any metadata in the Data Set Discovery Browser. What could be wrong?

A combination of things could be wrong. Begin by checking to see if the crawler is indeed running. Go to the /awips2/edex/logs directory. Open the edex-registry-harvester-XXXXXXX.log with a text viewer of your choosing. Look for any indication that the “Main Sequence Scan” crawler is running in the file. If you see lots of errors or “couldn’t start,” go through the following checklist.

1. Is your proxy configured correctly? [If your proxy is not right, you will never be able to connect. If it is not correct, edit the proxy.properties file accordingly.]

2. Do you have the correct Provider root URL? Check the root URL in your NOMADS-harvester.xml. Try it in a web browser; does it work? If it does not work, you may have it incorrect or you have other (deeper) network configuration problems.
3. If your crawler is running and seems to be collecting metadata, you can prove this by looking in the logs for entries like this.

```
INFO 2013-01-23 00:00:04,029 [crawlerThreadPool-[DefaultQuartzScheduler_Worker-5]] Crawler: EDEX - Starting crawl [2/78], provider [NOMADS], collection [narre], model [narre], date [20130123], politeness delay [100]
INFO 2013-01-23 00:00:04,031 [crawlerThreadPool-[DefaultQuartzScheduler_Worker-5][pool-50-thread-1]] FileCommunicationStrategy: EDEX - Wrote linkStore: NOMADS : narre size: 89
```

This means that your Data Delivery EDEX is at least finding metadata and that the problem is further downstream than the crawler. Now open the edex-registry-harvester-XXXXXX.log. If you see a lot of errors in this log, it is indicative of a problem with your Data Delivery Registry. You should see entries like this in general.

```
INFO 2013-01-08 19:56:49,280 [harvesterThreadPool-2 LifecycleManagerImpl: EDEX - LifecycleManager receivedSubmitObjectsRequest
INFO 2013-01-08 19:56:49,280 [harvesterThreadPool-2 LifecycleManagerImpl: EDEX - 1 object submitted to registry
INFO 2013-01-08 19:56:49,280 [harvesterThreadPool-2 LifecycleManagerImpl: EDEX - Validating objects...
INFO 2013-01-08 19:56:49,280 [harvesterThreadPool-2 LifecycleManagerImpl: EDEX - Objects successfully validated! Submitting...
INFO 2013-01-08 19:56:49,362 [harvesterThreadPool-2 LifecycleManagerImpl: EDEX - Submit objects successful
INFO 2013-01-08 19:56:49,365 [harvesterThreadPool-2 LifecycleManagerImpl: EDEX - LifecycleManager submitObjects operation completed in 85 ms
```

Anything different would indicate a problem.

4. Check to see if the “MetaDataHandler” is finding the directories where metadata is deposited by the crawler. The $EDEX_HOME/data/utility/common_static/$SITE/datadelivery/harvester directory contains two subdirectories. The “links” directory is where the crawler deposits the data set links it has discovered during the
crawl. The “MetaDataHandler” reads this directory once a minute to check for new metadata. If the permissions on this directory are bad, or the files within have become corrupted, problems can ensue causing no metadata to be read and ingested into the registry. The files in this directory are XML files that contain the URL link for the new metadata. The other file to check is the ‘times’ directory. Generally, if the “MetaDataHandler” is running well, no times files will be contained in the times directory. If you notice any persistent files, that could indicate a problem. The best course of action would be to delete the conflicting files.

5. If the crawler is not running, check to see if a “lock” file exists in the $EDEX_HOME/data/utility/common_static/ $SITE/datadelivery/harvester directory. You will be looking for a file called NOMADS-main-crawl.lock (or NOMADS-seed-crawl.lock if you are debugging the Seed Scan process). The Crawler uses this file as a locking semaphore to prevent new crawlers from running if one is already in progress. Check the permissions on this file; they should be <awips2, fxalpha> (user, group). If not, delete them and wait for the next crawler to kick off based on your crontab setting in the harvester configuration file.

6. Did you look at your crontab settings in the harvester configuration file? Are they correct? If they are not in what would pass as “standard crontab syntax,” they will not work and your Crawler will never launch. Therefore, you will never get any metadata in the Discovery Browser. The web is a good resource for devising a good strategy for crontabs. Remember that, when doing your crontab, it is prudent to have the window large enough for the crawler to finish completely between scans. Otherwise, your scans will be tripping over each other and you may end up with gaps in your update schedule for data sets.
Chapter 30
User Administration
Chapter 30. User Administration

Table of Contents

30.1 Overview .............................................................................................................................. 1
30.2 Initial Configuration ............................................................................................................. 2
30.3 Configuration File Format .................................................................................................... 3
30.4 Use of the User Admin Dialog ............................................................................................. 4
  30.4.1 User Administration Users Tab .............................................................................. 6
  30.4.2 User Administration Roles Tab .............................................................................. 8

List of Exhibits

Exhibit 30.4-1. AWIPS User Administration - User Admin Dialog Box ('A User Administrator')................................................................................................................................. 4
Exhibit 30.4-2. AWIPS User Administration - Alert Message ('Not a User Administrator')...... 5
Exhibit 30.4-3. Permission-Controlled Components ................................................................. 5
Exhibit 30.4.1-1. User Admin GUI for User Administration Users ........................................... 6
Exhibit 30.4.1-2. Add New User Dialog Box ............................................................................. 7
Exhibit 30.4.1-3. Edit User (Assigned Roles and Assigned Permissions) Dialog Box .......... 7
Exhibit 30.4.2-1. User Admin GUI for User Administration Roles .......................................... 8
Exhibit 30.4.2-2. Add a New Role ............................................................................................ 9
Exhibit 30.4.2-3. Edit Role ....................................................................................................... 9
Exhibit 30.4.2-4. Assigned Roles ........................................................................................... 10
Exhibit 30.4.2-5. Assigned Permissions ............................................................................... 10
Exhibit 30.4.2-6. Assign Role to Users .................................................................................. 11
30.1 Overview

Some of the functions of certain components of CAVE are limited to users who have been granted a specific ‘permission’ or assigned a specific ‘role.’ A ‘permission’ gives a user access to a specific functionality; a ‘role’ authorizes a user to perform a designated group of functions.

Site User Administrators are responsible for issuing and managing ‘permissions’ and ‘roles,’ and for designating other functionalities as being open to “ALL” users. Their mechanism for managing permissions and roles is the AWIPS User Administration software, which is accessed through the CAVE->AWIPS User Administration... menu.

Currently, five components of CAVE are controlled by permissions and roles: 1) the User Administration software itself; 2) the Localization perspective; 3) Data Delivery; 4) Official User Product; and 5) Data Archiving. Only permissions may be granted for User Administration, Official User Product, Localization and Data Archiving; both permissions and roles have been defined for Data Delivery.

[Note: Data Delivery will only be seen under the Component dropdown menu for systems that can run Data Delivery. The CAVE sessions that do not contain a Data Delivery option are instances where CAVE was deployed/installed without the Data Delivery rpm or with it ‘turned off.’]

Descriptions of the five components follow.

- **User Administration.** Permissions are used to control which users have access to the User Admin dialog, and only one permission has been defined for this dialog: awips.user.admin. Users with this permission have user administrator rights.

  No roles have been defined for the User Admin dialog.

  By default, no users have permission to access this dialog at the time the system is installed. It is expected that each site will grant specific users the awips.user.admin permission. [See section 30.2 for details.]

- **Localization.** Permissions are used to control which users have access to which Localization files. For example, com.raytheon.localization.site/common_static/gfe is the permission that controls access to the GFE localization files. Any user who is assigned that permission has access to site-level gfe common_static localization files.

  The Localization component comes with permissions already set up for the special ALL user (by default). Localization also allows for the creation of new permissions.

  No roles have been defined for Localization.

- **Data Delivery.** The Permissions within Data Delivery control what individual users are allowed to do within the Data Delivery software, for example, create/edit/delete subscriptions, edit subscription groups, or manage system access. Two roles have been defined for Data Delivery. One is the subscription.user role, which packages the
subscription.view, subscription.create, subscription.edit, and subscription.delete permissions; this allows the user to view, create, edit, and delete his or her own subscriptions. The second role is subscription.admin, which grants a user all the permissions packaged in the subscription.user role, and adds the group.edit and subscription.approve.all permissions. A user who has been assigned the subscription.admin role can do everything a subscription.user can; he or she can also approve subscriptions and edit existing subscription groups.

- **Official User Product.** The Official User Product permission, awips.oup, determines which users are allowed to send things using the Official User Product framework. By default the permission is added to the “ALL” user so all users at the site can send products, as was done in the past. This permission is in place for users connecting from outside the NOAA firewall. Sites supporting users connecting from outside the firewall will need to lock down their OUP_Roles.xml file by removing the permission from the ALL user, adding individual users and granting them the awips.oup permission. This will limit the ability of outside users sending Official User Products.

- **Data Archiving.** The permissions within Archiving are used to control which individual users are allowed to run the Archive Case Creation and Retention software. For example, a selected user who has been granted archive.case creation Roles/Permissions is allowed to run the Archive Case Creation. Similarly, archive.retention Roles/Permissions allow a user to run the Archive Retention software. No roles have been defined for Data Archiving.

### 30.2 Initial Configuration

At the time of initial installation of the AWIPS II software, permissions for the User Administration dialog are locked down, so no one has access. It is up to the sites to determine who should have access to the User Admin software and then perform the User Admin configuration following completion of the initial installation of the AWIPS II software. [*Note:* Configuration is only completed once, following initial AWIPS II installation; the User Admin dialog does not need to be reconfigured after future installs.]

The steps for initial configuration follow.

1. On the EDEX request server,
   
   **TYPE:** `cd /awips2/edex/data/utility/common_static/base/roles`

   **TYPE:** `cp awipsUserAdminRoles.xml` 
   `/awips2/edex/data/utility/common_static/site/<SITE ID>/roles`

   **TYPE:** `cd /awips2/edex/data/utility/common_static/site/<SITE ID>/roles`

2. Using any text editor, edit awipsUserAdminRoles.xml.

   a. Locate the line that begins with “<user userId=“ and replace AuthorizedUserName with a valid user name.
b. Delete the two lines with the text DELETE THIS LINE.

3. Save the file.

4. Have the user try to access the AWIPS User Administration Dialog from within CAVE.

**Note:** Although the User Administration Configuration file has the userPermission and rolePermission defined, the Administrator is strictly required not to edit those fields and to use the options under the CAVE D2D to set up the userPermission and rolePermission.

### 30.3 Configuration File Format

The User Administration configuration file format is xml. Each file will have an application tag stating which application within AWIPS II uses these permissions.

A ‘**permission**’ is defined by a permission tag with an id attribute where the id is the permission string. Each permission tag can have an optional description tag to describe the permission. A ‘**role**’ is defined by a role tag with a roleId attribute where the roleId is the role string. Each role tag has a roleDescription for describing the role and one or more rolePermission tags to list the permissions belonging to the role.

User tags define the permissions and/or roles assigned to each user. The user tag has a userId attribute listing the user’s id. The userPermission tag houses the permissionId or roleId the user has been granted.

A sample file follows:

```xml
<nwsRoleData xmlns:ns2="group">
  <application>Data Delivery</application>
  <permission id="subscription.view">
    <description>
      Control Access to View Data Delivery Subscriptions
    </description>
  </permission>
  <permission id="subscription.dataset.browser">
    <description>Control access to the Dataset Discovery Browser</description>
  </permission>
  <role roleId="subscription.admin">
    <roleDescription>
      This role is a grouping of permissions, default subscription admin role
    </roleDescription>
    <rolePermission>subscription.approve.all</rolePermission>
</nwsRoleData>
```
30.4 Use of the User Admin Dialog

Users who are assigned the awips.user.admin permission described in Section 30.2 access the User Admin dialog via the Cave->AWIPS User Administration... menu item. To begin, they select the AWIPS User Administration option from the CAVE menu.

When an authorized User Administrator selects this option, the screen shown in Exhibit 30.4-1 will appear.

Exhibit 30.4-1. AWIPS User Administration - User Admin Dialog Box (‘A User Administrator’)

An unauthorized user will be blocked from entering the User Admin dialog, and the Alert Message shown in Exhibit 30.4-2 will appear instead.
Exhibit 30.4-2. AWIPS User Administration - Alert Message (‘Not a User Administrator’)

At the top of the User Admin dialog box is a Component dropdown box (editable) that lists the five components (User Administration, Data Archiving, Data Delivery, Localization, and Official User Product). See Exhibit 30.4-3.

Exhibit 30.4-3. Permission-Controlled Components
For each component there are two tabs, the User Administration **Users** tab and the User Administration **Roles** tab.

### 30.4.1 User Administration Users Tab

The Users tab includes a “Selected Users” window and a “Defined Roles/Permissions” window. They are shown in Exhibit 30.4.1-1 for all five components.

![Exhibit 30.4.1-1. User Admin GUI for User Administration Users](image-url)
**Note:** The center buttons on the User Admin dialog box, as well as the Localization and Data Delivery buttons (New, Edit, and Delete) perform different functions, depending on which tab is selected.

The Users tab lists all defined users and the roles/permissions assigned to each user.

- You can add new users to each component by clicking the **New** button.

Clicking the “New” button opens the **Add New User** dialog box shown in Exhibit 30.4.1-2, which is used to add new users.

![Add New User Dialog Box](image)

*Exhibit 30.4.1-2. Add New User Dialog Box*

Entering a new user in the **Add New User** dialog box (Exhibit 30.4.1-2) and clicking **OK** opens the **Edit User** dialog box (Assigned Roles and Assigned Permissions) (see Exhibit 30.4.1-3). This dialog box signifies whether roles or permissions are being granted.

![Edit User Dialog Box](image)

*Exhibit 30.4.1-3. Edit User (Assigned Roles and Assigned Permissions) Dialog Box.*
30.4.2 User Administration Roles Tab

The Roles tab includes a “Defined Roles” window and a “Roles/Permissions” window. Exhibit 30.4.2-1 illustrates the windows for all five components.

Exhibit 30.4.2-1. User Admin GUI for User Administration Roles
**Note:** The center buttons on the User Admin dialog box (New, Edit, and Delete) perform different functions, depending on which tab is selected.

- You can add new role dialogs to each component by clicking the **New...** button.

Clicking the **New...** button displays the Add New Role dialog as shown in Exhibit 30.4.2-2.

![Exhibit 30.4.2-2. Add a New Role](image)

Once you enter a role name and an optional description and click **OK** to continue, the **Edit Role** dialog is displayed (see Exhibit 30.4.2-3).

![Exhibit 30.4.2-3. Edit Role](image)
As shown in Exhibit 30.4.2-3, the **Edit Role** dialog (combo box) allows other roles to be assigned to this new role, allows permissions to be assigned to this new role, and allows the user to assign this role to specific users. These actions are illustrated individually in Exhibits 30.4.2-4, 30.4.2-5, and 30.4.2-6.

---

**Exhibit 30.4.2-4. Assigned Roles**

**Exhibit 30.4.2-5. Assigned Permissions**
Exhibit 30.4.2-6. Assign Role to Users

All changes are stored in memory until the user clicks the Save button (see Exhibit 30.4-1). No changes are persisted until the Save button is clicked. To cancel changes, click the Close button.
Appendix A
Abbreviations and Acronyms
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>μEngine</td>
<td>MicroEngine</td>
</tr>
<tr>
<td>A</td>
<td></td>
</tr>
<tr>
<td>AAP</td>
<td>Alert Adaptation Parameter</td>
</tr>
<tr>
<td>ACARS</td>
<td>Aircraft Communications Addressing and Reporting System</td>
</tr>
<tr>
<td>ACN</td>
<td>AWIPS Communications Network</td>
</tr>
<tr>
<td>ACT</td>
<td>Attribute Color Threshold</td>
</tr>
<tr>
<td>ADAPPT</td>
<td>AWIPS Decision Assistance Production Preparation Tools</td>
</tr>
<tr>
<td>ADE</td>
<td>AWIPS Development Environment</td>
</tr>
<tr>
<td>AFD</td>
<td>Area Forecast Discussion</td>
</tr>
<tr>
<td>AFOS</td>
<td>Automation of Field Operations and Services</td>
</tr>
<tr>
<td>AGMG</td>
<td>AWIPS GIS Mapping Group</td>
</tr>
<tr>
<td>AHPS</td>
<td>Advanced Hydrologic Prediction Service</td>
</tr>
<tr>
<td>ALERT</td>
<td>automated local evaluation in real time</td>
</tr>
<tr>
<td>AM</td>
<td>Alert Message</td>
</tr>
<tr>
<td>AMQP</td>
<td>Advanced Message Queuing Protocol</td>
</tr>
<tr>
<td>ANCF</td>
<td>AWIPS Network Control Facility (Silver Spring, MD)</td>
</tr>
<tr>
<td>ANSI</td>
<td>American National Standards Institute</td>
</tr>
<tr>
<td>ANT</td>
<td>Another Neat Tool, a Java-oriented “build” tool</td>
</tr>
<tr>
<td>API</td>
<td>application program interface</td>
</tr>
<tr>
<td>APS</td>
<td>Asynchronous Product Scheduler</td>
</tr>
<tr>
<td>ASCII</td>
<td>American Standard Code for Information Interchange</td>
</tr>
<tr>
<td>ASOS</td>
<td>Automated Surface Observing System</td>
</tr>
<tr>
<td>AVN</td>
<td>Aviation model</td>
</tr>
<tr>
<td>AvnFPS</td>
<td>Aviation Forecast Preparation System</td>
</tr>
<tr>
<td>AWC</td>
<td>Aviation Weather Center</td>
</tr>
<tr>
<td>AWIPS</td>
<td>Advanced Weather Interactive Processing System</td>
</tr>
<tr>
<td>AX</td>
<td>Archive Server</td>
</tr>
<tr>
<td>B</td>
<td></td>
</tr>
<tr>
<td>BBS</td>
<td>Bulletin Board Service</td>
</tr>
<tr>
<td>BMGS</td>
<td>Backup Master Ground Station</td>
</tr>
<tr>
<td>BNCF</td>
<td>Backup NCF (Fairmont, West Virginia)</td>
</tr>
<tr>
<td>BSD</td>
<td>Berkeley Software Distribution</td>
</tr>
<tr>
<td>BUFR</td>
<td>Binary Universal Form for data Representation</td>
</tr>
<tr>
<td>C</td>
<td></td>
</tr>
<tr>
<td>CAVE</td>
<td>Common AWIPS Visualization Environment</td>
</tr>
<tr>
<td>CCF</td>
<td>NWS Central Collection Facility</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>CDD</td>
<td>cooling degree days</td>
</tr>
<tr>
<td>CFC</td>
<td>Clutter Filter Control</td>
</tr>
<tr>
<td>CGI</td>
<td>Common Gateway Interface</td>
</tr>
<tr>
<td>CLI</td>
<td>command line interface</td>
</tr>
<tr>
<td>CM</td>
<td>Configuration Management</td>
</tr>
<tr>
<td>CO</td>
<td>Communications Server</td>
</tr>
<tr>
<td>COMT</td>
<td>COMET (WFO Configuration)</td>
</tr>
<tr>
<td>CONUS</td>
<td>conterminous United States</td>
</tr>
<tr>
<td>COOP</td>
<td>Continuity of Operations Planning</td>
</tr>
<tr>
<td>COTS</td>
<td>commercial off-the-shelf</td>
</tr>
<tr>
<td>CP</td>
<td>communications processor</td>
</tr>
<tr>
<td>CPSBN</td>
<td>Communications Processor - Satellite Broadcast Network</td>
</tr>
<tr>
<td>CPU</td>
<td>central processing unit</td>
</tr>
<tr>
<td>CPSYNC</td>
<td>Synchronous Communication Processor</td>
</tr>
<tr>
<td>CRS</td>
<td>Console Replacement System</td>
</tr>
<tr>
<td>CS</td>
<td>Call System</td>
</tr>
<tr>
<td>CSCI</td>
<td>Computer Software Configuration Item</td>
</tr>
<tr>
<td>CSU</td>
<td>channel service unit</td>
</tr>
<tr>
<td>CSVText</td>
<td>Comma Separated Values Text</td>
</tr>
<tr>
<td>CWA</td>
<td>county warning area</td>
</tr>
<tr>
<td>CWSU</td>
<td>Center Weather Service Unit</td>
</tr>
<tr>
<td>CY</td>
<td>calendar year</td>
</tr>
<tr>
<td>CZ</td>
<td>Composite Reflectivity</td>
</tr>
<tr>
<td>CZC</td>
<td>Composite Reflectivity Contour</td>
</tr>
<tr>
<td>D2D</td>
<td>Display 2-Dimensional</td>
</tr>
<tr>
<td>DAT</td>
<td>digital audiotape</td>
</tr>
<tr>
<td>DBMS</td>
<td>database management system</td>
</tr>
<tr>
<td>DCP</td>
<td>Data Collection Platform</td>
</tr>
<tr>
<td>DDR</td>
<td>Double Data Rate</td>
</tr>
<tr>
<td>DGEX</td>
<td>Downscaled GFS and Eta Extension Grids</td>
</tr>
<tr>
<td>DHR</td>
<td>Digital Hybrid Scan Reflectivity</td>
</tr>
<tr>
<td>DMD</td>
<td>Digital Mesocyclone Display</td>
</tr>
<tr>
<td>DMS</td>
<td>Data Monitoring System</td>
</tr>
<tr>
<td>dpi</td>
<td>dots per inch</td>
</tr>
<tr>
<td>DS</td>
<td>HP-UX Data Server</td>
</tr>
<tr>
<td>DSM</td>
<td>Daily Summary Message</td>
</tr>
<tr>
<td>DSU</td>
<td>digital service unit</td>
</tr>
<tr>
<td>DTG</td>
<td>Date Time Group</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>DTMF</td>
<td>dual tone multifrequency</td>
</tr>
<tr>
<td>DTS</td>
<td>dedicated transmission service</td>
</tr>
<tr>
<td>DVB</td>
<td>Digital Video Broadcast</td>
</tr>
<tr>
<td>DVD</td>
<td>Digital Video Display</td>
</tr>
<tr>
<td>DVD-R</td>
<td>Digital Video Display (Read Only)</td>
</tr>
<tr>
<td>DX</td>
<td>Linux Data Server</td>
</tr>
<tr>
<td>ECMWF</td>
<td>European Centre for Medium-Range Weather Forecasts</td>
</tr>
<tr>
<td>EDEX</td>
<td>Environmental Data EXchange</td>
</tr>
<tr>
<td>EET</td>
<td>Extreme Echo Tops</td>
</tr>
<tr>
<td>EHB</td>
<td>Engineering Handbook</td>
</tr>
<tr>
<td>EIA</td>
<td>Electronics Industry Association</td>
</tr>
<tr>
<td>EMDS</td>
<td>Emergency Manager Decision Support</td>
</tr>
<tr>
<td>EMDS</td>
<td>Emergency Managers Dissemination Service</td>
</tr>
<tr>
<td>EMM</td>
<td>Enclosure Management Module</td>
</tr>
<tr>
<td>EMRS</td>
<td>Engineering Management Reporting System</td>
</tr>
<tr>
<td>ENP</td>
<td>Eastern North Pacific</td>
</tr>
<tr>
<td>EOF</td>
<td>end of file</td>
</tr>
<tr>
<td>EOR</td>
<td>end of report</td>
</tr>
<tr>
<td>EPSS</td>
<td>Enhanced Packet-Switched Service (an FTS2000 service)</td>
</tr>
<tr>
<td>ERA</td>
<td>Embedded Remote Access</td>
</tr>
<tr>
<td>ERL</td>
<td>Environmental Research Laboratory</td>
</tr>
<tr>
<td>ESA</td>
<td>electronic systems analyst</td>
</tr>
<tr>
<td>ESRL</td>
<td>Earth System Research Laboratory</td>
</tr>
<tr>
<td>ET</td>
<td>echo top</td>
</tr>
<tr>
<td>ETC</td>
<td>echo top contour</td>
</tr>
<tr>
<td>FAA</td>
<td>Federal Aviation Administration</td>
</tr>
<tr>
<td>FAC</td>
<td>File Access Controller</td>
</tr>
<tr>
<td>FC</td>
<td>Fibre Channel</td>
</tr>
<tr>
<td>FDDI</td>
<td>Fiber Distributed Data Interface</td>
</tr>
<tr>
<td>FDMA</td>
<td>Frequency Division Multiple Access</td>
</tr>
<tr>
<td>FFMP</td>
<td>Flash Flood Monitoring and Prediction</td>
</tr>
<tr>
<td>FFG</td>
<td>Flash Flood Guidance</td>
</tr>
<tr>
<td>FFGG</td>
<td>Forced Flash Flood Guidance</td>
</tr>
<tr>
<td>FFTI</td>
<td>Flash Flood Threat Index</td>
</tr>
<tr>
<td>FFW</td>
<td>Flash Flood Warning</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>FMH</td>
<td>Federal Meteorological Handbook</td>
</tr>
<tr>
<td>FOSS</td>
<td>Free and Open Source software</td>
</tr>
<tr>
<td>FR</td>
<td>frame relay</td>
</tr>
<tr>
<td>FSI</td>
<td>Four-dimensional Stormcell Investigator</td>
</tr>
<tr>
<td>FSL</td>
<td>Forecast Systems Laboratory</td>
</tr>
<tr>
<td>FSR</td>
<td>Field Site Representative</td>
</tr>
<tr>
<td>FTM</td>
<td>Free Text Message</td>
</tr>
<tr>
<td>GB</td>
<td>gigabyte</td>
</tr>
<tr>
<td>GCI</td>
<td>General Communication, Inc.</td>
</tr>
<tr>
<td>GELT</td>
<td>Geographic Entity Lookup Table</td>
</tr>
<tr>
<td>GFE</td>
<td>Graphical Forecast Editor</td>
</tr>
<tr>
<td>GFS</td>
<td>Global Forecast System</td>
</tr>
<tr>
<td>GHz</td>
<td>Gigahertz</td>
</tr>
<tr>
<td>GIF</td>
<td>Graphics Interchange Format</td>
</tr>
<tr>
<td>GINI</td>
<td>GOES Ingest NOAAPORT Interface</td>
</tr>
<tr>
<td>GIS</td>
<td>Geographic Information Systems</td>
</tr>
<tr>
<td>GMT</td>
<td>Greenwich Mean Time</td>
</tr>
<tr>
<td>GNOME</td>
<td>GNU Network Object Model Environment</td>
</tr>
<tr>
<td>GOES</td>
<td>Geostationary Operational Environmental Satellite</td>
</tr>
<tr>
<td>GRE</td>
<td>Generic Routing Encapsulation</td>
</tr>
<tr>
<td>GRIB</td>
<td>gridded binary (data format)</td>
</tr>
<tr>
<td>GSD</td>
<td>Global Systems Division</td>
</tr>
<tr>
<td>GSI</td>
<td>Globecom Systems Inc.</td>
</tr>
<tr>
<td>GSM</td>
<td>general status message</td>
</tr>
<tr>
<td>GUARDIAN</td>
<td>General User Alert Display Panel</td>
</tr>
<tr>
<td>GUI</td>
<td>graphical user interface</td>
</tr>
<tr>
<td>GWW</td>
<td>Global Wind and Wave</td>
</tr>
<tr>
<td>HDD</td>
<td>Hard Disk Drive</td>
</tr>
<tr>
<td>HDLC</td>
<td>High-Level Data Link Control</td>
</tr>
<tr>
<td>HDF</td>
<td>Hierarchical Data Format</td>
</tr>
<tr>
<td>HDW</td>
<td>High Density Winds</td>
</tr>
<tr>
<td>HDSS</td>
<td>Hierarchical Data Storage System</td>
</tr>
<tr>
<td>HFS</td>
<td>Hierarchical File System</td>
</tr>
<tr>
<td>HFS</td>
<td>Hydrologic Forecast System</td>
</tr>
<tr>
<td>HP</td>
<td>Hewlett-Packard</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>---------------------------------------------------------------</td>
</tr>
<tr>
<td>HPC</td>
<td>Hydrometeorological Prediction Center</td>
</tr>
<tr>
<td>HPE</td>
<td>High-resolution Precipitation Estimator</td>
</tr>
<tr>
<td>HRAP</td>
<td>Hydrologic Rainfall Analysis Project</td>
</tr>
<tr>
<td>HSR</td>
<td>Hybrid Scan Reflectivity</td>
</tr>
<tr>
<td>HWO</td>
<td>Hazardous Weather Outlook (replaces or augments the SPS)</td>
</tr>
<tr>
<td>HWR</td>
<td>Hourly Weather Roundup</td>
</tr>
<tr>
<td>I/O</td>
<td>input/output</td>
</tr>
<tr>
<td>ICAO</td>
<td>International Civil Aviation Organization</td>
</tr>
<tr>
<td>ICP</td>
<td>Intelligent Communications Processor</td>
</tr>
<tr>
<td>ICWF</td>
<td>Interactive Computer Worded Forecast</td>
</tr>
<tr>
<td>IOC</td>
<td>Initial Operating Capability</td>
</tr>
<tr>
<td>IEEE</td>
<td>Institute of Electrical and Electronics Engineers, Inc.</td>
</tr>
<tr>
<td>IFL</td>
<td>interfacility link</td>
</tr>
<tr>
<td>IFLOWS</td>
<td>Integrated Flood Observing and Warning System</td>
</tr>
<tr>
<td>IFPS</td>
<td>Interactive Forecast Preparation System</td>
</tr>
<tr>
<td>IGC</td>
<td>Inter-Gateway Communication</td>
</tr>
<tr>
<td>IHFS</td>
<td>Integrated Hydrologic Forecast System</td>
</tr>
<tr>
<td>IMET</td>
<td>Incident METeorologist</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>IPC</td>
<td>Interprocess Communication</td>
</tr>
<tr>
<td>IPVS</td>
<td>IP Virtual Server</td>
</tr>
<tr>
<td>ISDN</td>
<td>Integrated Services Digital Network</td>
</tr>
<tr>
<td>ISS</td>
<td>Incident Support Specialist</td>
</tr>
<tr>
<td>IT/O</td>
<td>HP OpenView Operations (old acronym)</td>
</tr>
<tr>
<td>IV&amp;V</td>
<td>Independent Verification and Validation</td>
</tr>
<tr>
<td>JAI</td>
<td>Java Advanced Imaging</td>
</tr>
<tr>
<td>JAI Imageio</td>
<td>Java Advanced Imaging Image I/O tools API</td>
</tr>
<tr>
<td>JMS</td>
<td>Java Messaging System</td>
</tr>
<tr>
<td>JRE</td>
<td>Java Runtime Environment</td>
</tr>
<tr>
<td>JSO</td>
<td>Java Serialized Object</td>
</tr>
<tr>
<td>JVM</td>
<td>Java Virtual Machine</td>
</tr>
<tr>
<td>LAC</td>
<td>Listening Area Code</td>
</tr>
<tr>
<td>LAMP</td>
<td>Local AWIPS MOS Program</td>
</tr>
</tbody>
</table>

---

AWP.MAN.SMMA2

– Hard copies uncontrolled. Verify effective date prior to use.–
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LAN</td>
<td>Local Area Network</td>
</tr>
<tr>
<td>LAPD</td>
<td>Link Access Procedure-D</td>
</tr>
<tr>
<td>LAPS</td>
<td>Local Analysis and Prediction System</td>
</tr>
<tr>
<td>LARC</td>
<td>local automatic remote collector</td>
</tr>
<tr>
<td>LCD</td>
<td>Liquid Crystal Display</td>
</tr>
<tr>
<td>LDAD</td>
<td>local data acquisition and dissemination</td>
</tr>
<tr>
<td>LDM</td>
<td>Local Data Manager</td>
</tr>
<tr>
<td>LED</td>
<td>light-emitting diode</td>
</tr>
<tr>
<td>LDM(D)</td>
<td>Local Data Manager (Downstream)</td>
</tr>
<tr>
<td>LIIT</td>
<td>Long Island International Teleport</td>
</tr>
<tr>
<td>LNB</td>
<td>low-noise block</td>
</tr>
<tr>
<td>LS</td>
<td>LDAD server</td>
</tr>
<tr>
<td>LST</td>
<td>local standard time</td>
</tr>
<tr>
<td>LTO</td>
<td>Linear Tape Open (a technology)</td>
</tr>
<tr>
<td>LV</td>
<td>logical volume</td>
</tr>
<tr>
<td>LX</td>
<td>Linux workstation</td>
</tr>
</tbody>
</table>

**M**

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mb</td>
<td>millibar</td>
</tr>
<tr>
<td>MB</td>
<td>megabyte</td>
</tr>
<tr>
<td>MDL</td>
<td>Meteorological Development Laboratory (formerly TDL)</td>
</tr>
<tr>
<td>MDP</td>
<td>Mesocyclone Display Parameters</td>
</tr>
<tr>
<td>METAR</td>
<td>Aviation Routine Weather Report</td>
</tr>
<tr>
<td>MGS</td>
<td>Master Ground Station</td>
</tr>
<tr>
<td>MHS</td>
<td>Message Handling System</td>
</tr>
<tr>
<td>MicroART</td>
<td>Microcomputer Automated Radio Theodolite</td>
</tr>
<tr>
<td>MIDDSS</td>
<td>Meteorological Interactive Data Display System</td>
</tr>
<tr>
<td>MIPS</td>
<td>million instructions per second</td>
</tr>
<tr>
<td>MOS</td>
<td>model output statistics</td>
</tr>
<tr>
<td>MPE</td>
<td>Multisensor Precipitation Estimator</td>
</tr>
<tr>
<td>MPLS</td>
<td>Multi-Protocol Label Switching</td>
</tr>
<tr>
<td>MSAS</td>
<td>MAPS Surface Analysis System</td>
</tr>
<tr>
<td>MSM</td>
<td>Monthly Summary Message</td>
</tr>
<tr>
<td>MTA</td>
<td>Message Transfer Agent</td>
</tr>
<tr>
<td>MTP</td>
<td>multipoint-to-point</td>
</tr>
<tr>
<td>MTSATHDW</td>
<td>Japanese High Density Winds from MTSAT</td>
</tr>
</tbody>
</table>

**N**

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAS</td>
<td>Network Attached Storage</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>NAT</td>
<td>Network Address Translation</td>
</tr>
<tr>
<td>NC</td>
<td>National Center</td>
</tr>
<tr>
<td>NCDC</td>
<td>National Climatic Data Center</td>
</tr>
<tr>
<td>NCEP</td>
<td>National Centers for Environmental Prediction</td>
</tr>
<tr>
<td>NCF</td>
<td>Network Control Facility</td>
</tr>
<tr>
<td>NCWF</td>
<td>National Convective Weather Forecast</td>
</tr>
<tr>
<td>NDFD</td>
<td>National Digital Forecast Database</td>
</tr>
<tr>
<td>NDM</td>
<td>National Data Sets Maintenance</td>
</tr>
<tr>
<td>NESDIS</td>
<td>National Environmental Satellite, Data, and Information Service</td>
</tr>
<tr>
<td>netCDF</td>
<td>network common data format</td>
</tr>
<tr>
<td>NFS</td>
<td>network file system</td>
</tr>
<tr>
<td>NGM</td>
<td>Nested Grid Model</td>
</tr>
<tr>
<td>NH</td>
<td>Northern Hemisphere</td>
</tr>
<tr>
<td>NIS</td>
<td>Network Information Services</td>
</tr>
<tr>
<td>NIST</td>
<td>National Institute of Standards and Technology</td>
</tr>
<tr>
<td>NLDN</td>
<td>National Lightning Detection Network</td>
</tr>
<tr>
<td>NMT</td>
<td>National Weather Service Modernization and Test Integration System (a place, like NCF)</td>
</tr>
<tr>
<td>NOAA</td>
<td>National Oceanic and Atmospheric Administration</td>
</tr>
<tr>
<td>NOAAPORT</td>
<td>A broadband communications system designed to distribute information products produced or gathered by NOAA organizations</td>
</tr>
<tr>
<td>NOC</td>
<td>Network Operations Center</td>
</tr>
<tr>
<td>NOGAPS</td>
<td>Navy Operational Global Atmospheric Prediction System</td>
</tr>
<tr>
<td>NOHRSC</td>
<td>National Operational Hydrologic Remote Sensing Center</td>
</tr>
<tr>
<td>NOMADS</td>
<td>NOAA Operational Model Archive and Distribution System</td>
</tr>
<tr>
<td>NRC</td>
<td>National Reconditioning Center</td>
</tr>
<tr>
<td>NRP</td>
<td>NOAAPORT Receive Processor</td>
</tr>
<tr>
<td>NRS</td>
<td>NOAAPORT Receive System</td>
</tr>
<tr>
<td>NSM</td>
<td>Netscreen Security Manager</td>
</tr>
<tr>
<td>NTP</td>
<td>Network Time Protocol</td>
</tr>
<tr>
<td>NUSM</td>
<td>NEXRAD Unit Status Message</td>
</tr>
<tr>
<td>NWR</td>
<td>NOAA Weather Radio</td>
</tr>
<tr>
<td>NWRE</td>
<td>NOAA Weather Radio Editor</td>
</tr>
<tr>
<td>NWRS</td>
<td>NOAA Weather Radio Service</td>
</tr>
<tr>
<td>NWRWAVES</td>
<td>NOAA Weather Radio With All-Hazards VTEC Enhanced Software</td>
</tr>
<tr>
<td>NWS</td>
<td>National Weather Service</td>
</tr>
<tr>
<td>NWSRFS</td>
<td>NWS River Forecast System</td>
</tr>
<tr>
<td>NWSTG</td>
<td>NWS Telecommunications Gateway</td>
</tr>
<tr>
<td>NWWS</td>
<td>NOAA Weather Wire Service</td>
</tr>
</tbody>
</table>
### Appendix A

#### Abbreviations and Acronyms

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWP.MAN.SMM.A2</td>
<td>– Hard copies uncontrolled. Verify effective date prior to use.—</td>
</tr>
<tr>
<td>OAR</td>
<td>Oceanic and Atmospheric Research</td>
</tr>
<tr>
<td>OB</td>
<td>Operational Build</td>
</tr>
<tr>
<td>ODS</td>
<td>Online Dynamic Server</td>
</tr>
<tr>
<td>OH</td>
<td>Office of Hydrology</td>
</tr>
<tr>
<td>OHD</td>
<td>Office of Hydrologic Development</td>
</tr>
<tr>
<td>OM</td>
<td>Office of Meteorology</td>
</tr>
<tr>
<td>ORDA</td>
<td>Open Radar Data Acquisition</td>
</tr>
<tr>
<td>ORPG</td>
<td>Open Radar Product Generator</td>
</tr>
<tr>
<td>OS</td>
<td>operating system</td>
</tr>
<tr>
<td>OS</td>
<td>operational site</td>
</tr>
<tr>
<td>OSF</td>
<td>Open Software Foundation</td>
</tr>
<tr>
<td>OSFW</td>
<td>NEXRAD Operational Support Facility (WFO)</td>
</tr>
<tr>
<td>OT&amp;E</td>
<td>operational testing and evaluation</td>
</tr>
<tr>
<td>OTR</td>
<td>one-time request</td>
</tr>
<tr>
<td>OUP</td>
<td>Official User Product</td>
</tr>
<tr>
<td>OVO</td>
<td>OpenView Operations (replaced IT/O)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PA</td>
<td>Project Authorization</td>
</tr>
<tr>
<td>PC</td>
<td>Personal Computer</td>
</tr>
<tr>
<td>PID</td>
<td>process identification</td>
</tr>
<tr>
<td>PIL</td>
<td>product inventory list</td>
</tr>
<tr>
<td>POES</td>
<td>Polar Orbiting Environmental Satellite</td>
</tr>
<tr>
<td>PPI</td>
<td>planned product improvement; plan position indicator</td>
</tr>
<tr>
<td>PPID</td>
<td>Parent Data Controller ID</td>
</tr>
<tr>
<td>ppm</td>
<td>pages per minute</td>
</tr>
<tr>
<td>PRR</td>
<td>Product Request Response</td>
</tr>
<tr>
<td>PSS</td>
<td>packet-switching service</td>
</tr>
<tr>
<td>PTL</td>
<td>Product List</td>
</tr>
<tr>
<td>PTM</td>
<td>point-to-multipoint</td>
</tr>
<tr>
<td>PTSU</td>
<td>Pacific Tsunami Warning Center</td>
</tr>
<tr>
<td>PTP</td>
<td>point-to-point</td>
</tr>
<tr>
<td>PUP</td>
<td>principal user processor</td>
</tr>
<tr>
<td>PV</td>
<td>PowerVault</td>
</tr>
<tr>
<td>PVC</td>
<td>permanent virtual circuit</td>
</tr>
<tr>
<td>PX</td>
<td>Linux Preprocessor</td>
</tr>
<tr>
<td>PYRO</td>
<td>Python Remote Objects</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>QC</td>
<td>quality control</td>
</tr>
<tr>
<td>QCMS</td>
<td>Quality Control and Monitoring System</td>
</tr>
<tr>
<td>QPE</td>
<td>Quantitative Precipitation Estimate</td>
</tr>
<tr>
<td>QPF</td>
<td>Quantitative Precipitation Forecast</td>
</tr>
<tr>
<td>QPID</td>
<td>Queue Processor Interface Daemon</td>
</tr>
<tr>
<td>QuikSCAT</td>
<td>NASA’s Quick Scatterometer</td>
</tr>
<tr>
<td>RAID</td>
<td>Redundant Array of Independent Disks</td>
</tr>
<tr>
<td>RAM</td>
<td>random access memory</td>
</tr>
<tr>
<td>RAMOS</td>
<td>Remote Automatic Meteorological Observing System</td>
</tr>
<tr>
<td>RAP</td>
<td>Rapid Refresh</td>
</tr>
<tr>
<td>RAX</td>
<td>RFC Archive Server</td>
</tr>
<tr>
<td>RCM</td>
<td>radar-coded message</td>
</tr>
<tr>
<td>RCM</td>
<td>Radar Configuration Manager</td>
</tr>
<tr>
<td>RCP</td>
<td>Rich Client Platform</td>
</tr>
<tr>
<td>RDA</td>
<td>radar data acquisition</td>
</tr>
<tr>
<td>RDBMS</td>
<td>relational database management system</td>
</tr>
<tr>
<td>REP</td>
<td>River Ensemble Processor</td>
</tr>
<tr>
<td>RHEL</td>
<td>Red Hat Enterprise Linux</td>
</tr>
<tr>
<td>RFC</td>
<td>River Forecast Center</td>
</tr>
<tr>
<td>RFC GW</td>
<td>RFC Gateway</td>
</tr>
<tr>
<td>RGG</td>
<td>Red Green Blue</td>
</tr>
<tr>
<td>RH</td>
<td>relative humidity</td>
</tr>
<tr>
<td>RHQ</td>
<td>Regional Headquarters</td>
</tr>
<tr>
<td>RMI</td>
<td>Remote Method Invocation</td>
</tr>
<tr>
<td>RISC</td>
<td>reduced instruction set computer (workstation)</td>
</tr>
<tr>
<td>RMR</td>
<td>radar multiple request</td>
</tr>
<tr>
<td>ROC</td>
<td>WSR-88D Radar Operations Center</td>
</tr>
<tr>
<td>ROSA</td>
<td>remote observing system automation</td>
</tr>
<tr>
<td>RP</td>
<td>REP CPU</td>
</tr>
<tr>
<td>RPG</td>
<td>radar product generator</td>
</tr>
<tr>
<td>RPS</td>
<td>routine product set</td>
</tr>
<tr>
<td>RRS</td>
<td>Radiosonde Replacement System</td>
</tr>
<tr>
<td>RTGSST</td>
<td>Real Time High Resolution Global Sea Surface Temperature grids</td>
</tr>
<tr>
<td>SAFESEAS</td>
<td>System on AWIPS for Forecasting and Evaluation of Seas and Lakes</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>SAM</td>
<td>System Administration Manager</td>
</tr>
<tr>
<td>SBN</td>
<td>Satellite Broadcast Network</td>
</tr>
<tr>
<td>SBP</td>
<td>Satellite Broadcast Processor</td>
</tr>
<tr>
<td>SCAN</td>
<td>System for Convection Analysis and Nowcasting</td>
</tr>
<tr>
<td>SCD</td>
<td>Supplementary Climate Data</td>
</tr>
<tr>
<td>SCID</td>
<td>Storm Cell Identification Display</td>
</tr>
<tr>
<td>SCP</td>
<td>Satellite Cloud Product</td>
</tr>
<tr>
<td>SCSI</td>
<td>small computer system interface</td>
</tr>
<tr>
<td>SCTI</td>
<td>SCAN Convective Threat Index</td>
</tr>
<tr>
<td>SDK</td>
<td>Software Developers Kit</td>
</tr>
<tr>
<td>SDRAM</td>
<td>Synchronous Dynamic Random Access Memory</td>
</tr>
<tr>
<td>SDS</td>
<td>software disk striping; switched data service</td>
</tr>
<tr>
<td>SEC</td>
<td>Systems Engineering Center</td>
</tr>
<tr>
<td>SEDA</td>
<td>Serial Event Driven Architecture</td>
</tr>
<tr>
<td>SFMG</td>
<td>Spaceflight Meteorology Group’s (SMG) WFO system identifier</td>
</tr>
<tr>
<td>SGS</td>
<td>Site Ground Station</td>
</tr>
<tr>
<td>SHEF</td>
<td>Standard Hydrometeorological Exchange Format</td>
</tr>
<tr>
<td>SMM</td>
<td>AWIPS System Manager’s Manual</td>
</tr>
<tr>
<td>SLDFD</td>
<td>site-level data flow diagram</td>
</tr>
<tr>
<td>SNMP</td>
<td>Simple Network Management Protocol</td>
</tr>
<tr>
<td>SNOW</td>
<td>System for Nowcasting of Winter Weather</td>
</tr>
<tr>
<td>SOA</td>
<td>Service Oriented Architecture</td>
</tr>
<tr>
<td>SPG</td>
<td>Supplemental Product Generator</td>
</tr>
<tr>
<td>SPS</td>
<td>Special Weather Statement</td>
</tr>
<tr>
<td>SSH</td>
<td>secure shell</td>
</tr>
<tr>
<td>SSMI (/I)</td>
<td>Special Sensor Microwave Imager (both SSMI and SSM/I are used but SSM/I is correct)</td>
</tr>
<tr>
<td>SST</td>
<td>sea surface temperature</td>
</tr>
<tr>
<td>SST</td>
<td>Site Support Team</td>
</tr>
<tr>
<td>STD</td>
<td>season to date</td>
</tr>
<tr>
<td>STI</td>
<td>Storm Track Information</td>
</tr>
<tr>
<td>SVS</td>
<td>switched voice service</td>
</tr>
<tr>
<td>SW</td>
<td>Sky Weather</td>
</tr>
<tr>
<td>SWP</td>
<td>spectrum width/software</td>
</tr>
<tr>
<td>SWP</td>
<td>Severe WX Problem</td>
</tr>
</tbody>
</table>

**T**

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TAF</td>
<td>Terminal Aerodrome Forecast (international code)</td>
</tr>
<tr>
<td>TB</td>
<td>terabyte</td>
</tr>
<tr>
<td>TC</td>
<td>temperatures in degrees Celsius</td>
</tr>
</tbody>
</table>
Tcl | tool command language
---|---
TCP | Transport Control Protocol
TDWR | Terminal Doppler Weather Radar
TF | temperature in degrees Fahrenheit
THP | Total Hourly Precipitation
TIU | terminal interface unit
Tk | tool kit
TPG | Toy Parser Generator
TVS | Tornadic Vortex Signature
TWEB | Transcribed Weather Enroute Broadcast

U

UAM | User Alert Message
UGC | universal geographic code
UI | User Interface
UID | user ID number
ULR | User Selectable Reflectivity
URI | Uniform Resource Identifier
URL | Universal Resource Locator
USP | User Select Precipitation
UTC | Universal Time Coordinated
UW | u-component of wind

V

V | Velocity
VCP | Volume Coverage Pattern
VIL | Vertically Integrated Liquid
VIR | visible infrared
VRP | Voice-Ready Product
VSAT | Very Small Aperture Terminal

W

WAN | wide area network
WAX | WFO Archive Server
WBC | Watch By County
WCL | Watch County List
WCN | Weather Watch County Notification
WES | Weather Event Simulator
WES-2 Bridge | Weather Event Simulator-2 Bridge (provides training and simulation
### Abbreviations and Acronyms

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WFO</td>
<td>Weather Forecast Office</td>
</tr>
<tr>
<td>WIN32</td>
<td>Common 32-bit Microsoft Windows Platform</td>
</tr>
<tr>
<td>WMO</td>
<td>World Meteorological Organization (UN)</td>
</tr>
<tr>
<td>WSH</td>
<td>Weather Service Headquarters</td>
</tr>
<tr>
<td>WOU</td>
<td>Watch Outline Update Message</td>
</tr>
<tr>
<td>WSO</td>
<td>Weather Service Office</td>
</tr>
<tr>
<td>WSR</td>
<td>Weather Surveillance Radar</td>
</tr>
<tr>
<td>WSR-88D</td>
<td>WSR-1988 Doppler Weather Radar</td>
</tr>
<tr>
<td>WWV</td>
<td>time-signal broadcast call letters</td>
</tr>
<tr>
<td>WW3</td>
<td>WAVEWATCH III</td>
</tr>
</tbody>
</table>

**X, Y, Z**

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XML</td>
<td>eXtensible Markup Language</td>
</tr>
<tr>
<td>YTD</td>
<td>Year to Date</td>
</tr>
<tr>
<td>YUM</td>
<td>Yellowdog Updater Modified</td>
</tr>
</tbody>
</table>
Appendix B
Decoding and Storage Data Flow Exhibits
Appendix B. Decoding and Storage Data Flow Exhibits

Table of Contents

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>B.1</td>
</tr>
<tr>
<td>B.2</td>
</tr>
</tbody>
</table>

List of Exhibits

- Exhibit B.2-1. AWIPS II Servers Overview ................................................................. 4
- Exhibit B.2-2. Overall AWIPS II Servers Processes ......................................................... 5
- Exhibit B.2-3. SBN Data Flow .......................................................................................... 6
- Exhibit B.2-4. Radar Data Flow ......................................................................................... 7
- Exhibit B.2-5. Acquire and Ingest of Satellite Products .................................................... 8
- Exhibit B.2-6. Acquire and Ingest of NWSTG Products ..................................................... 9
- Exhibit B.2-7. Acquire and Ingest of Radar Products ......................................................... 10
- Exhibit B.2-8. Acquire and Ingest of Asynchronous Products ............................................ 11
- Exhibit B.2-9. Acquire and Ingest of Lightning Products .................................................. 12
- Exhibit B.2-10. Acquire and Ingest of METAR Products ................................................... 13
- Exhibit B.2-11. Acquire and Ingest of MOS Products ....................................................... 14
- Exhibit B.2-12. Acquire and Ingest of BUFR Products ....................................................... 15
- Exhibit B.2-13. Acquire and Ingest of Synoptic Products ................................................... 16
- Exhibit B.2-14. Acquire and Ingest of SSM/I Products ....................................................... 17
- Exhibit B.2-15. Generic Acquire and Decode LDAD Product (Page 1 of 2) ....................... 18
- Exhibit B.2-15. Generic Acquire and Decode LDAD Product (Page 2 of 2) ....................... 19
- Exhibit B.2-16. Acquire LARC/Handar 555 Data (for LDAD) ............................................ 20
- Exhibit B.2-17. Decode LARC/Handar 555 Data (for LDAD) (Page 1 of 2) ....................... 21
- Exhibit B.2-17. Decode LARC/Handar 555 Data (for LDAD) (Page 2 of 2) ....................... 22
- Exhibit B.2-18. Acquire Campbell Data (for LDAD) ........................................................... 23
- Exhibit B.2-19. Decode Campbell Data (for LDAD) (Page 1 of 2) ...................................... 24
- Exhibit B.2-19. Decode Campbell Data (for LDAD) (Page 2 of 2) ...................................... 25
- Exhibit B.2-20. Acquire Sutron Data (for LDAD) ............................................................... 26
- Exhibit B.2-21. Decode Sutron Data (for LDAD) (Page 1 of 2) .......................................... 27
- Exhibit B.2-21. Decode Sutron Data (for LDAD) (Page 2 of 2) .......................................... 28
- Exhibit B.2-22. Acquire and Decode Comma Separated Variable Mesonet Data (for LDAD) (Page 1 of 2) ................................................................. 29
- Exhibit B.2-22. Acquire and Decode Comma Separated Variable Mesonet Data (for LDAD)
List of Tables

Table B.2-1. AWIPS Data Flow Diagrams ................................................................. 2
Table B.2-2. AWIPS Data Flow Diagrams by Location ........................................... 3
B.1 Introduction

This section is composed of a series of exhibits that depict a high-level process and data flow from ingest through the routers to the data controllers and to the appropriate decoder. Graphic conventions used in the exhibits are as follows:

- Processes are shown in boxes.
- Servers are shown in shadowed boxes and are labeled as:
  - “CP” for communications processor;
  - “LS” for the LDAD Server;
  - “DX1” for Linux Data Server 1;
  - “DX2” for Linux Data Server 2;
  - “DX3” for Linux Data Server 3;
  - “DX4” for Linux Data Server 4;
  - “PX1” for Linux Preprocessor 1;
  - “PX2” for Linux Preprocessor 2; or
  - “WAX” for WFO Archive Server and RAX for RFC Archive Server.
- The arrows depict primary data flow.

Log files are depicted on the server where the logging process resides. Generated data files and accessed configuration files are all depicted. All files are shown with a full relative path designation.

B.2 AWIPS Data Flow Diagrams

All of the exhibits included in Appendix B also appear in the body of the System Manager’s Manual. Table B.2-1 cross-references the data flow diagrams in this appendix to their locations in the body of the document.
## Table B.2-1. AWIPS Data Flow Diagrams

<table>
<thead>
<tr>
<th>Appendix B Exhibit No.</th>
<th>Title</th>
<th>Chapter Exhibit No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>B.2-1</td>
<td>AWIPS II Servers Overview</td>
<td>2.7-3</td>
</tr>
<tr>
<td>B.2-2</td>
<td>Overall AWIPS II Server Processes</td>
<td>2.7-2</td>
</tr>
<tr>
<td>B.2-3</td>
<td>LDM Data Flow</td>
<td>4.4-1</td>
</tr>
<tr>
<td>B.2-4</td>
<td>Radar Data Flow</td>
<td>4.4-2</td>
</tr>
<tr>
<td>B.2-5</td>
<td>Acquire and Ingest of Satellite Products</td>
<td>5.2.1-1</td>
</tr>
<tr>
<td>B.2-6</td>
<td>Acquire and Ingest of NWSTG Products</td>
<td>6.3.3-1</td>
</tr>
<tr>
<td>B.2-7</td>
<td>Acquire and Ingest of Radar Products</td>
<td>7.2.1-1</td>
</tr>
<tr>
<td>B.2-8</td>
<td>Acquire and Ingest of Asynchronous Products</td>
<td>10.2-1</td>
</tr>
<tr>
<td>B.2-9</td>
<td>Acquire and Ingest of Lightning Products</td>
<td>6.4-1</td>
</tr>
<tr>
<td>B.2-10</td>
<td>Acquire and Ingest of METAR Products</td>
<td>6.5-1</td>
</tr>
<tr>
<td>B.2-11</td>
<td>Acquire and Ingest of MOS Products</td>
<td>6.6-1</td>
</tr>
<tr>
<td>B.2-12</td>
<td>Acquire and Ingest of BUFR Products</td>
<td>6.7-1</td>
</tr>
<tr>
<td>B.2-13</td>
<td>Acquire and Ingest of Synoptic Products</td>
<td>6.8-1</td>
</tr>
<tr>
<td>B.2-14</td>
<td>Acquire and Ingest of SSM/I Products</td>
<td>6.9-1</td>
</tr>
<tr>
<td>B.2-15</td>
<td>Generic Acquire and Decode LDAD Product</td>
<td>8.3.1-1</td>
</tr>
<tr>
<td>B.2-16</td>
<td>Acquire LARC/Handar 555 Data (for LDAD)</td>
<td>8.7-1</td>
</tr>
<tr>
<td>B.2-17</td>
<td>Decode LARC/Handar 555 Data (for LDAD)</td>
<td>8.8-1</td>
</tr>
<tr>
<td>B.2-18</td>
<td>Acquire Campbell Data (for LDAD)</td>
<td>8.9-1</td>
</tr>
<tr>
<td>B.2-19</td>
<td>Decode Campbell Data (for LDAD)</td>
<td>8.10-1</td>
</tr>
<tr>
<td>B.2-20</td>
<td>Acquire Sutron Data (for LDAD)</td>
<td>8.11-1</td>
</tr>
<tr>
<td>B.2-21</td>
<td>Decode Sutron Data (for LDAD)</td>
<td>8.12-1</td>
</tr>
<tr>
<td>B.2-22</td>
<td>Acquire and Decode Comma Separated Variable Mesonet Data (for LDAD)</td>
<td>8.13-1</td>
</tr>
<tr>
<td>B.2-23</td>
<td>Acquire and Process ASOS/MicroART Data (for LDAD)</td>
<td>8.14-1</td>
</tr>
<tr>
<td>B.2-24</td>
<td>Decode Alaska Profiler Data</td>
<td>8.16-1</td>
</tr>
<tr>
<td>B.2-25</td>
<td>Acquire and Process RRS Data (for LDAD)</td>
<td>8.17-1</td>
</tr>
<tr>
<td>B.2-26</td>
<td>Climatological Reports Formatter for NWR and NWWS (Daily)</td>
<td>9.1.3-1</td>
</tr>
<tr>
<td>B.2-27</td>
<td>Climatological Reports Formatter for NWR and NWWS (Monthly/Seasonal/Annual)</td>
<td>9.1.3-2</td>
</tr>
<tr>
<td>B.2-28</td>
<td>Record Climate Data Flow</td>
<td>9.2.2-1</td>
</tr>
<tr>
<td>B.2-29</td>
<td>Hourly Weather Roundup Formatter for NOAA Weather Wire Service</td>
<td>9.3.2-1</td>
</tr>
<tr>
<td>B.2-30</td>
<td>Hourly Weather Roundup Formatter for NOAA Weather Radio</td>
<td>9.4.2-1</td>
</tr>
</tbody>
</table>
Table B.2-2 provides an inverse mapping to Table B.2-1, listing the data flow diagrams by their locations in the body of the System Manager’s Manual and cross-referencing them to their locations in Appendix B.

### Table B.2-2. AWIPS Data Flow Diagrams by Location

<table>
<thead>
<tr>
<th>Chapter Exhibit No.</th>
<th>Title</th>
<th>Appendix B Exhibit No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.7-3</td>
<td>AWIPS II Servers Overview</td>
<td>B.2-1</td>
</tr>
<tr>
<td>2.7-2</td>
<td>Overall AWIPS II Server Processes</td>
<td>B.2-2</td>
</tr>
<tr>
<td>4.4-1</td>
<td>LDM Data Flow</td>
<td>B.2-3</td>
</tr>
<tr>
<td>4.4-2</td>
<td>Radar Data Flow</td>
<td>B.2-4</td>
</tr>
<tr>
<td>5.2.1-1</td>
<td>Acquire and Ingest of Satellite Products</td>
<td>B.2-5</td>
</tr>
<tr>
<td>6.3.3-1</td>
<td>Acquire and Ingest of NWSTG Products</td>
<td>B.2-6</td>
</tr>
<tr>
<td>7.2.1-1</td>
<td>Acquire and Ingest of Radar Products</td>
<td>B.2-7</td>
</tr>
<tr>
<td>10.2-1</td>
<td>Acquire and Ingest of Asynchronous Products</td>
<td>B.2-8</td>
</tr>
<tr>
<td>6.4-1</td>
<td>Acquire and Ingest of Lightning Products</td>
<td>B.2-9</td>
</tr>
<tr>
<td>6.5-1</td>
<td>Acquire and Ingest of METAR Products</td>
<td>B.2-10</td>
</tr>
<tr>
<td>6.6-1</td>
<td>Acquire and Ingest of MOS Products</td>
<td>B.2-11</td>
</tr>
<tr>
<td>6.7-1</td>
<td>Acquire and Ingest of BUFR Products</td>
<td>B.2-12</td>
</tr>
<tr>
<td>6.8-1</td>
<td>Acquire and Ingest of Synoptic Products</td>
<td>B.2-13</td>
</tr>
<tr>
<td>6.9-1</td>
<td>Acquire and Ingest of SSM/I Products</td>
<td>B.2-14</td>
</tr>
<tr>
<td>8.3.1-1</td>
<td>Generic Acquire and Decode LDAD Product</td>
<td>B.2-15</td>
</tr>
<tr>
<td>8.7-1</td>
<td>Acquire LARC/Handar 555 Data (for LDAD)</td>
<td>B.2-16</td>
</tr>
<tr>
<td>8.8-1</td>
<td>Decode LARC/Handar 555 Data (for LDAD)</td>
<td>B.2-17</td>
</tr>
<tr>
<td>8.9-1</td>
<td>Acquire Campbell Data (for LDAD)</td>
<td>B.2-18</td>
</tr>
<tr>
<td>8.10-1</td>
<td>Decode Campbell Data (for LDAD)</td>
<td>B.2-19</td>
</tr>
<tr>
<td>8.11-1</td>
<td>Acquire Sutron Data (for LDAD)</td>
<td>B.2-20</td>
</tr>
<tr>
<td>8.12-1</td>
<td>Decode Sutron Data (for LDAD)</td>
<td>B.2-21</td>
</tr>
<tr>
<td>8.13-1</td>
<td>Acquire and Decode Comma Separated Variable Mesonet Data (for LDAD)</td>
<td>B.2-22</td>
</tr>
<tr>
<td>8.14-1</td>
<td>Acquire and Process ASOS/MicroART Data (for LDAD)</td>
<td>B.2-23</td>
</tr>
<tr>
<td>8.16-1</td>
<td>Decode Alaska Profiler Data</td>
<td>B.2-24</td>
</tr>
<tr>
<td>8.17-1</td>
<td>Acquire and Process RRS Data (for LDAD)</td>
<td>B.2-25</td>
</tr>
<tr>
<td>9.1.3-1</td>
<td>Climatological Reports Formatter for NWR and NWWS (Daily)</td>
<td>B.2-26</td>
</tr>
<tr>
<td>9.1.3-2</td>
<td>Climatological Reports Formatter for NWR and NWWS (Monthly/Seasonal/Annual)</td>
<td>B.2-27</td>
</tr>
<tr>
<td>9.2.2-1</td>
<td>Record Climate Data Flow</td>
<td>B.2-28</td>
</tr>
<tr>
<td>9.3.2-1</td>
<td>Hourly Weather Roundup Formatter for NOAA Weather Wire Service</td>
<td>B.2-29</td>
</tr>
<tr>
<td>9.4.2-1</td>
<td>Hourly Weather Roundup Formatter for NOAA Weather Radio</td>
<td>B.2-30</td>
</tr>
</tbody>
</table>
Exhibit B.2-1. AWIPS II Servers Overview
Exhibit B.2-2. Overall AWIPS II Servers Processes
Exhibit B.2-3. SBN Data Flow
Exhibit B.2-4. Radar Data Flow
Exhibit B.2-5. Acquire and Ingest of Satellite Products
Exhibit B.2-6. Acquire and Ingest of NWSTG Products
Exhibit B.2-7. Acquire and Ingest of Radar Products
Exhibit B.2-8. Acquire and Ingest of Asynchronous Products
Exhibit B.2-9. Acquire and Ingest of Lightning Products

Configuration Files:
- LDM
  - noaaportIngest
  - EDEX Bridge
  - SBN CP (cp1f)

Log Files:
- nwstg.log
- ldmd.log
- edexBridge.log

Data Files:
- /data_store/binlightning/YYYYMMDD/HH/edexBridge.log
- /data_store/binlightning/YYYYMMDD/HH/edex-ingest-yyyyymmdd.err

Databases:
- /data_store/binlightning/YYYYMMDD/HH/edexBridge.db
- /awips2/edex/hdf5/binlightning/binlightning-yyyy-mm-dd-hh\.h5

- Hard copies uncontrolled. Verify effective date prior to use. –
Exhibit B.2-10. Acquire and Ingest of METAR Products
Exhibit B.2-11. Acquire and Ingest of MOS Products
Exhibit B.2-12. Acquire and Ingest of BUFR Products
Exhibit B.2-13. Acquire and Ingest of Synoptic Products

-- Hard copies uncontrolled. Verify effective date prior to use. --
Appendix B Decoding and Storage Data Flow Exhibits

Exhibit B.2-14. Acquire and Ingest of SSM/I Products
Exhibit B.2-15. Generic Acquire and Decode LDAD Product (Page 1 of 2)
Exhibit B.2-15. Generic Acquire and Decode LDAD Product (Page 2 of 2)

AWP.MAN.SMM.A2

Hard copies uncontrolled. Verify effective date prior to use.
Exhibit B.2-16. Acquire LARC/Handar 555 Data (for LDAD)
Exhibit B.2-17. Decode LARC/Handar 555 Data (for LDAD) (Page 1 of 2)
Appendix B

Decoding and Storage Data Flow Exhibits

Exhibit B.2-17. Decode LARC/Handar 555 Data (for LDAD) (Page 2 of 2)

– Hard copies uncontrolled. Verify effective date prior to use. –
Exhibit B.2-18. Acquire Campbell Data (for LDAD)
Exhibit B.2-19. Decode Campbell Data (for LDAD) (Page 1 of 2)
Appendix B

Decoding and Storage Data Flow Exhibits

Exhibit B.2-19. Decode Campbell Data (for LDAD) (Page 2 of 2)

- Hard copies uncontrolled. Verify effective date prior to use. -
Exhibit B.2-20. Acquire Sutron Data (for LDAD)
Exhibit B.2-21. Decode Sutron Data (for LDAD) (Page 1 of 2)
Exhibit B.2-21. Decode Sutron Data (for LDAD) (Page 2 of 2)
Exhibit B.2-22. Acquire and Decode Comma Separated Variable Mesonet Data (for LDAD) (Page 1 of 2)
Exhibit B.2-22. Acquire and Decode Comma Separated Variable Mesonet Data (for LDAD) (Page 2 of 2)
Exhibit B.2-23. Acquire and Process ASOS/MicroART Data (for LDAD) (Page 1 of 2)
Config Files:

$FXA_DATA/workFiles/
wanMsgHandling/
rcv_action2codes.txt
$FXA_DATA/workFiles/
wanMsgHandling/
siteCommission.txt

Log Files:

$LDAD_EXTERNAL_LOGDIR/
<yyyymmdd>/distributeProduct<logid>

Data Files:

Databases:

Exhibit B.2-23. Acquire and Process ASOS/MicroART Data (for LDAD) (Page 2 of 2)
Appendix B
Decoding and Storage Data Flow Exhibits

Exhibit B.2-24. Decode Alaska Profiler Data

-- Hard copies uncontrolled. Verify effective date prior to use. --
Exhibit B.2-25. Acquire and Process RRS Data (for LDAD) (Page 1 of 2)
Config Files:
$FXA_DATA/workFiles/wanMsgHandling/rcv_action2codes.txt
$FXA_DATA/workFiles/wanMsgHandling/siteCommission.txt

Log Files:
$LOG_DIR/
<yyymmdmdd>/
distributeProduct
<logid>

Data Files:
/data/fxa/mhs/tmp

Databases:

Exhibit B.2-25. Acquire and Process RRS Data (for LDAD) (Page 2 of 2)
Exhibit B.2-26. Climatological Reports Formatter for NWR and NWWS (Daily)
Exhibit B.2-27. Climatological Reports Formatter for NWR and NWWS (Monthly/Seasonal/Annual)

Appendix B
Decoding and Storage Data Flow Exhibits

– Hard copies uncontrolled. Verify effective date prior to use. –
Exhibit B.2-28. Record Climate Data Flow
Exhibit B.2-29. Hourly Weather Roundup Formatter for NOAA Weather Wire Service
Appendix C
Directories for /awips2/edex/data and /data/fxa
Appendix C.  
Directories for /awips2/edex/data and /data/fxa

Table of Contents

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>C.1</td>
</tr>
<tr>
<td>C.2</td>
</tr>
</tbody>
</table>
This appendix includes two sections. Section C.1 contains the /awips2/edex/data directory, and Section C.2 contains the outline for the /data/fxa directory tree.

C.1 Data Directories in /awips2/edex/data

/awips2/edex/data

/hdf5

/aviation Contains decoded aviation products in HDF5 format plus some static data.
Example: /awips2/edex/data/hdf5/aviation/KECG.01.nc

/binlightning Contains decoded binlightning products in HDF5 format.
/awips2/edex/data/hdf5/binlightning/<model>-<yyyy>-<mm>-<dd>-<hh>.h5 in HDF% format.
Example: /awips2/edex/data/hdf5/binlightning/binlightning-2013-02-06-10.h5

/bufrscat Contains decoded bufrscat products in HDF5 format.
/awips2/edex/data/hdf5/bufrscat/<model>-<yyyy>-<mm>-<dd>-<hh>.h5 in HDF% format.

/bufrhdw Contains decoded bufrhdw products in HDF5 format.
/awips2/edex/data/hdf5/bufrhdw/<model>-<yyyy>-<mm>-<dd>-<hh>.h5 in HDF% format.

/bufrmthdw Contains decoded bufrmthdw products in HDF5 format.
/awips2/edex/data/hdf5/bufrmthdw/<model>-<yyyy>-<mm>-<dd>-<hh>.h5 in HDF% format.

/bufrmosGFS All decoded bufrmosGFS data are under
/awips2/edex/data/hdf5/bufrmosGFS/<model>-<yyyy>-<mm>-<dd>-<hh>.h5 in HDF% format.
Appendix C

Directories for /awips2/edex/data and /data/fxa


/bufrmosHPC All decoded bufrmosHPC data are under /awips2/edex/data/hdf5/bufrmosHPC/<model>-<yyyy>-<mm>-<dd>-<hh>.h5 in HDF5 format.

Example: /awips2/edex/data/hdf5/bufrmosHPC/bufrmos-HPC-2011-10-19-00.h5

/bufrmosLAMP All decoded bufrmosLAMP data are under /awips2/edex/data/hdf5/bufrmosLAMP/<model>-<yyyy>-<mm>-<dd>-<hh>.h5 in HDF5 format.


/bufrmosAVN All decoded bufrmosAVN data are under /awips2/edex/data/hdf5/bufrmosAVN/<model>-<yyyy>-<mm>-<dd>-<hh>.h5 in HDF5 format.


/bufrmosMRF All decoded bufrmosMRF data are under /awips2/edex/data/hdf5/bufrmosMRF/<model>-<yyyy>-<mm>-<dd>-<hh>.h5 in HDF5 format.


/bufrmosETA All decoded bufrmosETA data are under /awips2/edex/data/hdf5/bufrmosETA/<model>-<yyyy>-<mm>-<dd>-<hh>.h5 in HDF5 format.

Example: e.g.: /awips2/edex/data/hdf5/bufrmosETA/bufrmos-ETA-2011-10-23-11.h5

/bufrcnwf Contains decoded bufrcnwf products in HDF5 format.

awips2/edex/data/hdf5/bufrcnwf/<model>-<yyyy>-<mm>-<dd>-<hh>.h5 in HDF5 format.

Appendix C

Directories for /awips2/edex/data and /data/fxa

/bufrsigwx
Contains decoded bufrsigwx products in HDF5 format.
There are three directories under /bufrsigwx
SWBOTH, SWH, SWM

SWBOTH: sigwxVTS-2011-07-04-18.h5


/bufrssmi
Contains decoded bufrssmi products in HDF5 format.

awips2/edex/data/hdf5/bufrssmi/<model>-<yyyy>-<mm>-<dd>-<hh>.h5 in HDF% format.

Example: /awips2/edex/data/hdf5/bufrssmi/ssmi-2011-10-26-16.h5

/bufrua
Contains decoded bufrua products in HDF5 format.

awips2/edex/data/hdf5/bufrua/<model>-<yyyy>-<mm>-<dd>-<hh>.h5 in HDF% format.

Example: /awips2/edex/data/hdf5/bufrua/bufrua-2011-10-25-06.h5

/cwa
Contains decoded cwa products in HDF5 format.

awips2/edex/data/hdf5/cwa/<model>-<yyyy>-<mm>-<dd>-<hh>.h5 in HDF% format.

Example: /awips2/edex/data/hdf5/cwa/cwa-2013-02-05-02-16.h5

/cwat
Contains decoded cwat products in HDF5 format.

Example: /awips2/edex/data/hdf5/cwat/cwat-2011-11-01-02-02.h5

/ffmp
Contains decoded ffmp products in HDF5 format.

Example: /awips2/edex/data/hdf5/ffmp/ffmp-2011-11-01-02-02.h5
/fog
Contains decoded fog products in HDF5 format.
Example: /awips2/edex/data/hdf5/fog/fog-2011-10-13-16-02.h5

fssobs
Contains decoded obs products in HDF5 format.
Example: /awips2/edex/data/hdf5/fssobs/fssobs-2012-07-09-16.h5

/gfe
Contains forecast grids created by GFE and Smart Init
/awips2/edex/data/hdf5/gfe/BOX
(Note: The contents of this directory are based on the 13.3.1 – Build 13 @ tbdw)
DGEX GFS80 HPCGuide LAPS NAM12 Restore RUC13 Topo Fcest HIRESWarw HPCQPF MOSGuide NamDNG5 RFCQPF SAT GFS40 HIRESWnmm ISC MSAS Official RTMA SREF

/goessounding
Contains decoded goessounding products in HDF5 format.

/grid
Contains decoded grib products in HDF5 format.

GFSGuide MPE-Mosaic-TAR
## Appendix C

### Directories for /awips2/edex/data and /data/fixa

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>ECMF3</td>
<td>HiResW-ARW-GU</td>
<td>QPE-RFC-RSA</td>
</tr>
<tr>
<td>ECMF4</td>
<td>HiResW-ARW-HI</td>
<td>QPE-RFC-STR</td>
</tr>
<tr>
<td>ECMF5</td>
<td>HiResW-ARW-SJU</td>
<td>QPE-RHA</td>
</tr>
<tr>
<td>ECMF6</td>
<td>HiResW-ARW-West</td>
<td>QPE-TAR</td>
</tr>
<tr>
<td>ECMF7</td>
<td>HiResW-NMM-AK</td>
<td>QPE-TIR</td>
</tr>
<tr>
<td>ECMF8</td>
<td>HiResW-NMM-East</td>
<td>QPE-TUA</td>
</tr>
<tr>
<td>ECMF-NorthernHemisphere</td>
<td>HiResW-NMM-GU</td>
<td>QPE-XNAV-ALR</td>
</tr>
<tr>
<td>ECMF-Tropical</td>
<td>HiResW-NMM-HI</td>
<td>QPE-XNAV-FWR</td>
</tr>
<tr>
<td>ECMWF-HiRes</td>
<td>HiResW-NMM-SJU</td>
<td>QPE-XNAV-KRF</td>
</tr>
<tr>
<td>ENPWave253</td>
<td>HiResW-NMM-West</td>
<td>QPE-XNAV-MSR</td>
</tr>
<tr>
<td>ENSEMBLE</td>
<td>HI-RTMA</td>
<td>QPE-XNAV-ORN</td>
</tr>
<tr>
<td>ENSEMBLE38</td>
<td>HPCGuide</td>
<td>QPE-XNAV-RHA</td>
</tr>
<tr>
<td>ENSEMBLE39</td>
<td>HPCGuide-AK</td>
<td>QPE-XNAV-TAR</td>
</tr>
<tr>
<td>ENSEMBLE40</td>
<td>HPCqpf</td>
<td>QPE-XNAV-TIR</td>
</tr>
<tr>
<td>EPwave10</td>
<td>HPCqpfNDFD</td>
<td>QPE-XNAV-TUA</td>
</tr>
<tr>
<td>ETA</td>
<td>mesoEta212</td>
<td>RCM</td>
</tr>
<tr>
<td>ETA207</td>
<td>mesoEta215</td>
<td>RFCqpf</td>
</tr>
<tr>
<td>ETA212</td>
<td>mesoEta216</td>
<td>RTGSST</td>
</tr>
<tr>
<td>ETA218</td>
<td>mesoEta217</td>
<td>RTGSSTHR</td>
</tr>
<tr>
<td>ETA242</td>
<td>mesoEta237</td>
<td>RTMA</td>
</tr>
<tr>
<td>FFG-ALR</td>
<td>MOSGuide</td>
<td>rtma-test</td>
</tr>
<tr>
<td>FFG-FWR</td>
<td>MOSGuide-AK</td>
<td>RUC130</td>
</tr>
<tr>
<td>FFG-KRF</td>
<td>MPE-Local</td>
<td>RUC236</td>
</tr>
<tr>
<td>FFG-MSR</td>
<td>MPE-Local-ALR</td>
<td>SeaIce</td>
</tr>
<tr>
<td>FFG-ORN</td>
<td>MPE-Local-FWR</td>
<td>HPCGuide</td>
</tr>
<tr>
<td>FFG-PTR</td>
<td>MPE-Local-MSR</td>
<td>SREF212</td>
</tr>
<tr>
<td>FFG-RHA</td>
<td>MPE-Local-ORN</td>
<td>SREF216</td>
</tr>
<tr>
<td>FFG-RSA</td>
<td>MPE-Local-RHA</td>
<td>SREF243</td>
</tr>
<tr>
<td>FFG-STR</td>
<td>MPE-Local-RSA</td>
<td>TPCWindProb</td>
</tr>
<tr>
<td>FFG-TAR</td>
<td>MPE-Local-STR</td>
<td>UKMET37</td>
</tr>
<tr>
<td>FFG-TIR</td>
<td>MPE-Local-TAR</td>
<td>UKMET38</td>
</tr>
<tr>
<td>FFG-TUA</td>
<td>MPE-Local-TIR</td>
<td>UKMET39</td>
</tr>
<tr>
<td>GFS160</td>
<td>MPE-Local-TUA</td>
<td>UKMET40</td>
</tr>
<tr>
<td>GFS161</td>
<td>MPE-Mosaic</td>
<td>UKMET-NorthernHemisphere</td>
</tr>
<tr>
<td>GFS199</td>
<td>MPE-Mosaic-ALR</td>
<td>WCwave10</td>
</tr>
<tr>
<td>GFS201</td>
<td>MPE-Mosaic-FWR</td>
<td>WCwave4</td>
</tr>
<tr>
<td>GFS212</td>
<td>MPE-Mosaic-MSR</td>
<td>WNAwave10</td>
</tr>
<tr>
<td>GFS213</td>
<td>MPE-Mosaic-ORN</td>
<td>WNAwave238</td>
</tr>
<tr>
<td>GFS254</td>
<td>MPE-Mosaic-RHA</td>
<td>WNAwave4</td>
</tr>
</tbody>
</table>

**ldadhydro**

Contains decoded ldadhydro products in HDF5 format.

Appendix C  Directories for /awips2/edex/data and /data/fxa

/ldadmesonet Contains decoded ldadmesonet products in HDF5 format.
Example: /awips2/edex/data/hdf5/ldadmesonet/ldadmesonet-2013-02-06-23.h5

/lsr Contains decoded lsr products in HDF5 format.
Example: /awips2/edex/data/hdf5/lsr/lsr-2011-10-22-23.h5

/modelsounding Contains decoded modelsounding products in HDF5 format.

/mosaic Contains mosaic products in HDF5 format.

/ntrans Contains gfs ntrans products in HDF5 format.
Example: /awips2/edex/data/hdf5/ntrans/gfs/gfs-20130924-12-us

/ncairep, ncapafm, ncepirep, ncsat, nctaf, ncuair Contains National Center products in HDF5 format.

/nucaps Contains nucaps products in HDF5 format.

/pgen Contains pgen products in HDF5 format.

/obs Contains decoded obs (metar) products in HDF5 format.
Example: /awips2/edex/data/hdf5/obs/metar-2011-10-23-06.h5

/poessounding Contains decoded poessounding products in HDF5 format.

/preciprate Contains preciprate products in HDF5 format
Example: /awips2/edex/data/hdf5/preciprate/preciprate-2012-07-02-17-21.h5

/profiler Contains decoded profiler products in HDF5 format.
Example:
ApPENDIX C  Directories for /awips2/edex/data and /data/fxa

/awips2/edex/data/hdf5/profiler/profiler-2011-10-31-17.h5

/qpf Contains decoded qpf products in HDF5 format.
Example: /awips2/edex/data/hdf5/qpf/qpf-2011-11-01-03-11.h5

/QC Contains ldadmesonet and msas products

/radar Contains radar products in HDF5 format under each site.
Example: /awips2/edex/data/hdf5/radar/kabr/1.5/radar-kabr-2011-11-01-03.h5

/redbook Contains decoded redbook products in HDF5 format.
Example: /awips2/edex/data/hdf5/redbook/PEWA34/redbook-2013-02-06-12-2.h5

/satellite Contains decoded satellite products in HDF5 format.
Under each satellite directory, you will find the SOURCE directory:

Alaska National
Alaska Regional
East CONUS
Hawaii National
Hawaii Regional
NH Composite - Meteosat-GOES E-GOES W-GMS
Northern Hemisphere Composite
Puerto Rico National
Puerto Rico Regional
Supernational
West CONUS
101MODS
101RGB
111MODS
111RGB
281MODS

Under each SOURCE directory, you will find the RESOLUTION directory. For example, under East CONUS, you will find the following directories:

Imager 11 micron IR
Appendix C

Directories for /awips2/edex/data and /data/fxa

Under each RESOLUTION directory, you will find the satellite products. For example, under Imager Visible, you will find the following satellite products:

```
satellite-2013-02-08-01.h5  satellite-2013-02-08-11.h5
satellite-2013-02-08-02.h5  satellite-2013-02-08-12.h5
satellite-2013-02-08-03.h5  satellite-2013-02-08-13.h5
satellite-2013-02-08-04.h5  satellite-2013-02-08-14.h5
satellite-2013-02-08-05.h5  satellite-2013-02-08-15.h5
satellite-2013-02-08-10.h5  satellite-2013-02-08-16.h5
```

/scan    Contains decoded scan products in HDF5 format.
Example: /awips2/edex/data/hdf5(scan/scan-2011-10-31-19-04.h5

/sfcobs Contains decoded sfcobs products in HDF5 format.
Under sfcobs you will find the following directories:
1003 1004 1005 1006 1007
Under each directory, you will find the corresponding sfcobs products.
Example:
awips2/edex/data/hdf5/sfcobs/1003/sfcobs-2011-10-04-12.h5

/solarimage Contains decoded solarimage products in HDF5 format.

/svrwx Contains decoded svrwx products in HDF5 format.
Appendix C

Directories for /awips2/edex/data and /data/fxa

/c tg
Contains decoded tcg products in HDF5 format.

/topo
Contains topo files
  akTopo.dat.gz
  hlsTopo
  modelStaticTopo.h5
  srtm30.hdf
  usTopo.dat.gz
  caribTopo.dat.gz licenses
  licenses
  pacTopo.dat.gz
  staticTopo.h5
  worldTopo.dat.gz

/vil
Contains decoded vil products in HDF5 format.

Note: vil is a radar-based product that shows liquid amounts. VIL stands for “Vertically Integrated Liquid.” Also in this group are Digital Vertically Integrated Liquid (DVIL) and Enhanced Vertically Integrated Liquid (EDVIL). Each HDF5 file contains all three of these products, which are stored as displayable data blobs (two dimensional grids). For VIL, the grid size is 4KM; the other two are 1 km grids.

/utility
Placeholder for cave_config, cave_static, common_static, edex_static files.

C.2 Directory Tree for /data/fxa

<table>
<thead>
<tr>
<th>Directory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/afos</td>
<td>NOT IN USE</td>
</tr>
<tr>
<td>/aiv</td>
<td>NOT IN USE</td>
</tr>
<tr>
<td>/archive</td>
<td>NOT IN USE</td>
</tr>
<tr>
<td>/ax_refresh</td>
<td>NOT IN USE</td>
</tr>
<tr>
<td>/badRadar</td>
<td>NOT IN USE</td>
</tr>
<tr>
<td>/badSatellite</td>
<td>NOT IN USE</td>
</tr>
<tr>
<td>/badText</td>
<td>NOT IN USE</td>
</tr>
<tr>
<td>/customFiles</td>
<td>NOT IN USE</td>
</tr>
<tr>
<td>/CP_PARTITION</td>
<td>CP Partition</td>
</tr>
<tr>
<td>/data</td>
<td>data</td>
</tr>
</tbody>
</table>
Directories for /awips2/edex/data and /data/fxa

/dgm
/DRT
/elt
/eLog
/flatText
/freeware
/freeway
/Grid
/hazCollect
/img
/ispan
/INSTALL
/install_root
/j24
/laps
/laps_data
/laps_domain
/laps_geog
/LDAD
/ldadScheduler
/mhs
/nationalData
/nwr
/nwws
/point
/pgdata
/radar
/redbook
/rps-lists
/sat
/siteConfig
/textWSwork
/tmp
/trigger
/tstorm
/userSkewTs
/userPrefs
/verification
/vgf
/workFiles

NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE

LAPS
LAPS data
LAPS domain
LAPS geography
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE
NOT IN USE

AWIPS user preferences files
Appendix D
Glossary
ADE  
AWIPS Development Environment

AlertViz  
The AWIPS II version of the AWIPS I Guardian application. AlertViz is designed to present various notifications, error messages, and alarms to the user. AlertViz can be executed either independently or from CAVE itself.

Camel  
An open-source project from Apache that is used to implement the AWIPS II Service Oriented Architecture (SOA).

CAVE  
Common AWIPS Visualization Environment, the unified display client for AWIPS II. CAVE is implemented using the Eclipse framework and consists of multiple display perspectives that can run concurrently.

caveData  
A directory created by CAVE in each user's home directory. caveData stores a local copy of customization and configuration settings. When CAVE starts, in theory, it refreshes caveData with the current settings from the EDEX server.

Collaboration  
A component of CAVE, which offers two main functions, chatting and sharing displays. “Chat” allows users to chat with fellow forecasters and offices in a chat room or send them instant messages. “Sharing displays” adds to the chat room’s capabilities and allows the room’s creator to show a CAVE map display to other participants in the room.

Eclipse  
An open-source development environment that supports many programming languages, including JAVA and Python. Eclipse is also a rich client platform (RCP) upon which other applications can be developed. Although both CAVE and EDEX are developed and compiled using the Eclipse development environment, CAVE itself is an application developed using the Eclipse RCP, meaning that the user-interface elements and windowing mechanisms are provided by the Eclipse environment. Eclipse also implements a plug-in concept that allows CAVE and other RCP applications to be loosely coupled; this means that many components can be added and deleted simply by adding and removing plug-ins and not necessarily by recompiling the entire app.

Focus  
A state of the system that indicates which component receives keyboard events. A component is said to have the focus if keyboard events are sent to that component.

EDEX  
Environmental Data Exchange, the server for AWIPS II. EDEX ingests, decodes, stores, catalogs, and archives all the data used by AWIPS II. It also feeds data to the various CAVE clients.
endpoint  
Refers (when used within the context of a Service Oriented Architecture) to the beginning and destination of a particular service. In AWIPS II, “endpoint” commonly refers to the file endpoints that the various EDEX data decoders can use to read data. One way (though not necessarily the most preferred way) to feed data into EDEX is simply to have another program put a file in the proper file endpoint. For example, putting a text file that contains a metar in the endpoint directory of the obs decoder (/awips2/edex/data/SBN/metar) causes the obs decoder to execute using the input text file.

HDF  
Hierarchical Data Format, designed by the University of Illinois at Urbana-Champaign as an alternative to netCDF developed by Unidata. HDF5 is the primary data storage format used by AWIPS II. HDF5 supports multiple types of data within a single file. For example, a single HDF5 file of radar data could contain many volume scans of base reflectivity, base velocity plus any associated derived products like composite reflectivity, vertically integrated liquid, and precipitation accumulations, as well as algorithm output. The file may also contain data from multiple radars.

Hibernate  
An open-source project used in AWIPS II to provide connectivity between the PostgreSQL database for metadata and the JAVA code itself. Hibernate provides several methodologies to map database fields to variables and complex data objects in Java using either XML configuration files or special annotations in the JAVA code.

LDM  
Local Data Manager software, written and supported by Unidata. Many meteorological entities use LDM to manage the movement of data. LDM often connects data that comes from a NOAAPORT or AWIPS Satellite Broadcast Network (SBN) receiver to any number of downstream clients. These downstream clients might store the data, decode the data, or perform other functions on the data in an event-driven manner as the data arrive. In AWIPS II, the LDM provides the link between the AWIPS SBN, which receives raw data via satellite, and EDEX, which decodes and stores the data. In the normal AWIPS II, the LDM writes data directly into an archive and sends a message to EDEX using Qpid. The message tells EDEX that the particular data file is available for processing and uses a file header to determine what kind of file it is.
metadata

Generally defined as “data about data.” In AWIPS II, metadata refers either to either a specific PostgreSQL database or to the data contained within it. AWIPS II queries this metadata database to keep track of what data is currently available. In AWIPS I, the system would search through a system of files to find a particular piece or slice of data, such as radar, surface, model, or satellite data. In AWIPS II, particular pieces of data have a unique identifier, known as a dataURI, which contains enough information to specify the data. For example, metadata for a radar product might contain the radar id, the volume scan time, the elevation angle, and the product type.

NUCAPS

NUCAPS, short for ‘NOAA Unique Cross-track Infrared Sounder (CrIS) and Advanced Technology Microwave Sounder (ATMS) Product System,’ is under development at the NOAA/NESDIS Center for Satellite Applications and Research. The system will produce thinned apodized CrIS radiances, principal components of CrIS radiances, cloud-cleared CrIS radiances, and trace gas profile products for Numerical Weather Prediction (NWP) customers in near real-time.

Python

The main scripting language used in AWIPS II.

Qpid

Queue Processor Interface Daemon, the messaging system used in AWIPS II. An open-source project, Qpid is an implementation of the Advanced Message Queuing Protocol. When the LDM receives a data file to be processed, it sends EDEX a message using Qpid. When EDEX has finished decoding the file, it sends CAVE and any other listeners a message that says the particular piece of data (referenced by a dataURI) is available for display or further processing.

Re-engineered application

Some applications in AWIPS II were completely rewritten. These re-engineered applications include GFE, D2D, and FFMP.

Re-hosted application

The localization and configuration of these applications are the same as for AWIPS I. Examples of Re-hosted applications include Climate, NWRWAVES, HazCollect, and NOAA Weather Wire Service (NWWS) Interface.

SOA

Service Oriented Architecture, the design philosophy behind AWIPS II. In SOA, major system functions are services, and they are loosely coupled, rather than having large monolithic programs. In AWIPS II, raw data are ingested by an IngestSrv service, and are stored using a PersistSrv service. Camel “wires” or connects the services together (i.e., connecting the endpoints of successive services together) using XML configuration files, and the services can be modified without significantly affecting one another.
Appendix D

Glossary

**Tear-Away Menu**
A menu that can be moved around the workstation screen, away from the location on the screen where it first opened, by grabbing it in the title area and repositioning it on the screen.

**Thin Client**
A component that implements support for NWS enterprise requirements for remote access to baseline AWIPS II capabilities. The Thin Client will take advantage of new and enhanced AWIPS II capabilities in a seamless manner as they are deployed to the baseline.

**utility store**
A collection of utility data. The utility store is part of EDEX, and is typically located at /awips2/edex/data/utility/.

**Velocity**
An open-source template language. AWIPS II uses Velocity to construct WarnGen templates. Certain variables from the WarnGen JAVA code and data fields from the PostgreSQL database have been exposed to the Velocity templates, meaning that this data is available for use by Velocity. Because Velocity is an actual language, it has the ability to perform calculations and comparisons without having to recompile the WarnGen code itself.

**Urban Boundaries**
An application that allows more accurate and detailed city information in the WarnGen third and fourth bullets.

**WES-2 Bridge**
Weather Event Simulator bridge that provides training and simulation capabilities for AWIPS II platform.

**XML**
Extensible Markup Language. XML is not a programming language; rather, it is a common and structured way of formatting data, especially data that contains settings and preferences. In AWIPS II, most of the configuration files are stored in XML-formatted files.
Appendix E
Mass Storage Design
### Appendix E. Mass Storage Design

#### Table of Contents

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>E.1</td>
<td>LS Drive Allocations</td>
<td>1</td>
</tr>
<tr>
<td>E.2</td>
<td>Linux LX-Drive Allocation</td>
<td>1</td>
</tr>
<tr>
<td>E.3</td>
<td>Linux XT-Drive Allocation</td>
<td>2</td>
</tr>
<tr>
<td>E.4</td>
<td>PX Drive Allocation</td>
<td>2</td>
</tr>
<tr>
<td>E.5</td>
<td>Linux DX Drive Allocation and Linux DX Shared Database Allocation</td>
<td>2</td>
</tr>
</tbody>
</table>
This appendix provides current AWIPS system disk allocations.

### E.1 LS Drive Allocations

**Volume Group:**

<table>
<thead>
<tr>
<th>VG Name</th>
<th>Size</th>
<th>PE Size</th>
<th>Total PE</th>
<th>Alloc PE / Size</th>
<th>Free PE / Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>vg00</td>
<td>232.42 GiB</td>
<td>4.00 MiB</td>
<td>59499</td>
<td>54016 / 211.00 GiB</td>
<td>5483 / 21.42 GiB</td>
</tr>
<tr>
<td>vg01</td>
<td>232.42 GiB</td>
<td>4.00 MiB</td>
<td>59499</td>
<td>54016 / 211.00 GiB</td>
<td>5483 / 21.42 GiB</td>
</tr>
<tr>
<td>vg02</td>
<td>465.62 GiB</td>
<td>4.00 MiB</td>
<td>119199</td>
<td>119040 / 465.00 GiB</td>
<td>159 / 636.00 MiB</td>
</tr>
</tbody>
</table>

### E.2 Linux LX-Drive Allocation

**Volume Group:**

<table>
<thead>
<tr>
<th>VG Name</th>
<th>Size</th>
<th>PE Size</th>
<th>Total PE</th>
<th>Alloc PE / Size</th>
<th>Free PE / Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>vg00</td>
<td>68.00 GiB</td>
<td>4.00 MiB</td>
<td>17407</td>
<td>8950 / 34.96 GiB</td>
<td>8457 / 33.04 GiB</td>
</tr>
<tr>
<td>vg01</td>
<td>68.00 GiB</td>
<td>4.00 MiB</td>
<td>17407</td>
<td>8950 / 34.96 GiB</td>
<td>8457 / 33.04 GiB</td>
</tr>
</tbody>
</table>
E.3  Linux XT-Drive Allocation

Volume Group:

<table>
<thead>
<tr>
<th>VG Name</th>
<th>vg00</th>
</tr>
</thead>
<tbody>
<tr>
<td>VG Size</td>
<td>68.00 GiB</td>
</tr>
<tr>
<td>PE Size</td>
<td>4.00 MiB</td>
</tr>
<tr>
<td>Total PE</td>
<td>17407</td>
</tr>
<tr>
<td>Alloc PE / Size</td>
<td>8700 / 33.98 GiB</td>
</tr>
<tr>
<td>Free PE / Size</td>
<td>8707 / 34.01 GiB</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>VG Name</th>
<th>vg01</th>
</tr>
</thead>
<tbody>
<tr>
<td>VG Size</td>
<td>68.00 GiB</td>
</tr>
<tr>
<td>PE Size</td>
<td>4.00 MiB</td>
</tr>
<tr>
<td>Total PE</td>
<td>17407</td>
</tr>
<tr>
<td>Alloc PE / Size</td>
<td>8700 / 33.98 GiB</td>
</tr>
<tr>
<td>Free PE / Size</td>
<td>8707 / 34.01 GiB</td>
</tr>
</tbody>
</table>

E.4  PX Drive Allocation

Volume Group:

<table>
<thead>
<tr>
<th>VG Name</th>
<th>vg00</th>
</tr>
</thead>
<tbody>
<tr>
<td>VG Size</td>
<td>67.50 GiB</td>
</tr>
<tr>
<td>PE Size</td>
<td>4.00 MiB</td>
</tr>
<tr>
<td>Total PE</td>
<td>17279</td>
</tr>
<tr>
<td>Alloc PE / Size</td>
<td>8075 / 31.54 GiB</td>
</tr>
<tr>
<td>Free PE / Size</td>
<td>9204 / 35.95 GiB</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>VG Name</th>
<th>vg01</th>
</tr>
</thead>
<tbody>
<tr>
<td>VG Size</td>
<td>67.50 GiB</td>
</tr>
<tr>
<td>PE Size</td>
<td>4.00 MiB</td>
</tr>
<tr>
<td>Total PE</td>
<td>17279</td>
</tr>
<tr>
<td>Alloc PE / Size</td>
<td>10635 / 41.54 GiB</td>
</tr>
<tr>
<td>Free PE / Size</td>
<td>6644 / 25.95 GiB</td>
</tr>
</tbody>
</table>

E.5  Linux DX Drive Allocation and Linux DX Shared Database Allocation

Volume Group:

<table>
<thead>
<tr>
<th>VG Name</th>
<th>vg00</th>
</tr>
</thead>
<tbody>
<tr>
<td>VG Size</td>
<td>67.50 GiB</td>
</tr>
<tr>
<td>PE Size</td>
<td>4.00 MiB</td>
</tr>
<tr>
<td>Total PE</td>
<td>17279</td>
</tr>
<tr>
<td>Alloc PE / Size</td>
<td>16914 / 66.07 GiB</td>
</tr>
<tr>
<td>Free PE / Size</td>
<td>365 / 1.43 GiB</td>
</tr>
<tr>
<td>VG Name</td>
<td>vg01</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>VG Size</td>
<td>67.50 GiB</td>
</tr>
<tr>
<td>PE Size</td>
<td>4.00 MiB</td>
</tr>
<tr>
<td>Total PE</td>
<td>17279</td>
</tr>
<tr>
<td>Alloc PE / Size</td>
<td>16914 / 66.07 GiB</td>
</tr>
<tr>
<td>Free PE / Size</td>
<td>365 / 1.43 GiB</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>VG Name</th>
<th>vg_aiihdf5</th>
</tr>
</thead>
<tbody>
<tr>
<td>VG Size</td>
<td>400.02 GiB</td>
</tr>
<tr>
<td>PE Size</td>
<td>4.00 MiB</td>
</tr>
<tr>
<td>Total PE</td>
<td>102406</td>
</tr>
<tr>
<td>Alloc PE / Size</td>
<td>102144 / 399.00 GiB</td>
</tr>
<tr>
<td>Free PE / Size</td>
<td>262 / 1.02 GiB</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>VG Name</th>
<th>vg_aiildm</th>
</tr>
</thead>
<tbody>
<tr>
<td>VG Size</td>
<td>168.01 GiB</td>
</tr>
<tr>
<td>PE Size</td>
<td>4.00 MiB</td>
</tr>
<tr>
<td>Total PE</td>
<td>43010</td>
</tr>
<tr>
<td>Alloc PE / Size</td>
<td>43008 / 168.00 GiB</td>
</tr>
<tr>
<td>Free PE / Size</td>
<td>2 / 8.00 MiB</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>VG Name</th>
<th>vg_db</th>
</tr>
</thead>
<tbody>
<tr>
<td>VG Size</td>
<td>149.01 GiB</td>
</tr>
<tr>
<td>PE Size</td>
<td>4.00 MiB</td>
</tr>
<tr>
<td>Total PE</td>
<td>38147</td>
</tr>
<tr>
<td>Alloc PE / Size</td>
<td>1679 / 6.56 GiB</td>
</tr>
<tr>
<td>Free PE / Size</td>
<td>36468 / 142.45 GiB</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>VG Name</th>
<th>vg_aiidb</th>
</tr>
</thead>
<tbody>
<tr>
<td>VG Size</td>
<td>249.00 GiB</td>
</tr>
<tr>
<td>PE Size</td>
<td>4.00 MiB</td>
</tr>
<tr>
<td>Total PE</td>
<td>63745</td>
</tr>
<tr>
<td>Alloc PE / Size</td>
<td>38400 / 150.00 GiB</td>
</tr>
<tr>
<td>Free PE / Size</td>
<td>25345 / 99.00 GiB</td>
</tr>
</tbody>
</table>

**DX3/DX4**

**Volume Group:**

<table>
<thead>
<tr>
<th>VG Name</th>
<th>vg00</th>
</tr>
</thead>
<tbody>
<tr>
<td>VG Size</td>
<td>65.00 GiB</td>
</tr>
<tr>
<td>PE Size</td>
<td>4.00 MiB</td>
</tr>
<tr>
<td>Total PE</td>
<td>16639</td>
</tr>
<tr>
<td>Alloc PE / Size</td>
<td>16508 / 64.48 GiB</td>
</tr>
<tr>
<td>Free PE / Size</td>
<td>131 / 524.00 MiB</td>
</tr>
<tr>
<td>VG Name</td>
<td>vg01</td>
</tr>
<tr>
<td>---------</td>
<td>--------------</td>
</tr>
<tr>
<td>VG Size</td>
<td>65.00 GiB</td>
</tr>
<tr>
<td>PE Size</td>
<td>4.00 MiB</td>
</tr>
<tr>
<td>Total PE</td>
<td>16639</td>
</tr>
<tr>
<td>Alloc PE / Size</td>
<td>16508 / 64.48 GiB</td>
</tr>
<tr>
<td>Free PE / Size</td>
<td>131 / 524.00 MiB</td>
</tr>
</tbody>
</table>
Appendix F
NWS/AWIPS Security Policy
# Appendix F. NWS/AWIPS Security Policy

## Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>F.0</td>
<td>Introduction</td>
<td>1</td>
</tr>
<tr>
<td>F.0.1</td>
<td>Purpose</td>
<td>1</td>
</tr>
<tr>
<td>F.0.2</td>
<td>Scope</td>
<td>2</td>
</tr>
<tr>
<td>F.0.3</td>
<td>Applicable Documents</td>
<td>2</td>
</tr>
<tr>
<td>F.0.4</td>
<td>Changes to AWIPS IT Security Policy</td>
<td>3</td>
</tr>
<tr>
<td>F.1</td>
<td>Policy</td>
<td>3</td>
</tr>
<tr>
<td>F.2</td>
<td>Responsibilities</td>
<td>4</td>
</tr>
<tr>
<td>F.2.1</td>
<td>IT System Owner</td>
<td>4</td>
</tr>
<tr>
<td>F.2.2</td>
<td>AWIPS ISSO</td>
<td>4</td>
</tr>
<tr>
<td>F.2.3</td>
<td>NWS SISSO</td>
<td>5</td>
</tr>
<tr>
<td>F.2.4</td>
<td>Site AWIPS ISSO</td>
<td>6</td>
</tr>
<tr>
<td>F.2.5</td>
<td>AWIPS System Administrator</td>
<td>7</td>
</tr>
<tr>
<td>F.2.6</td>
<td>Operational Shift Team Leaders</td>
<td>7</td>
</tr>
<tr>
<td>F.2.7</td>
<td>NCF</td>
<td>8</td>
</tr>
<tr>
<td>F.2.8</td>
<td>NWSTC</td>
<td>8</td>
</tr>
<tr>
<td>F.2.9</td>
<td>NWS Employees, Non-NWS Government Employees, Non-Government Contractor/Vendor Personnel</td>
<td>8</td>
</tr>
<tr>
<td>F.3</td>
<td>Allowed Use of AWIPS</td>
<td>8</td>
</tr>
<tr>
<td>F.3.1</td>
<td>Access Control</td>
<td>9</td>
</tr>
<tr>
<td>F.3.1.1</td>
<td>Discretionary Access Control (DAC) and Access Control Lists (ACL)</td>
<td>9</td>
</tr>
<tr>
<td>F.3.1.2</td>
<td>Awareness and Password Administration</td>
<td>9</td>
</tr>
<tr>
<td>F.3.1.2.1</td>
<td>Awareness</td>
<td>10</td>
</tr>
<tr>
<td>F.3.1.2.2</td>
<td>AWIPS Password Administration</td>
<td>10</td>
</tr>
<tr>
<td>F.3.2</td>
<td>Use of Non-AWIPS Software on AWIPS Equipment</td>
<td>11</td>
</tr>
<tr>
<td>F.3.3</td>
<td>AWIPS Network Security</td>
<td>11</td>
</tr>
<tr>
<td>F.3.3.1</td>
<td>Network Operation and Routing Policy</td>
<td>11</td>
</tr>
<tr>
<td>F.3.3.2</td>
<td>Interfaces with Untrusted Networks</td>
<td>12</td>
</tr>
<tr>
<td>F.3.3.3</td>
<td>Services</td>
<td>13</td>
</tr>
<tr>
<td>F.3.3.4</td>
<td>Firewall Rule Set</td>
<td>13</td>
</tr>
<tr>
<td>F.3.3.5</td>
<td>AWIPS LAN Connections to Non-AWIPS Systems</td>
<td>14</td>
</tr>
<tr>
<td>F.3.3.6</td>
<td>AWIPS IP Address Space Management</td>
<td>15</td>
</tr>
<tr>
<td>F.3.3.7</td>
<td>Dial-in and Dial-out Access</td>
<td>15</td>
</tr>
<tr>
<td>F.3.3.7.1</td>
<td>Restricted LDAD Access</td>
<td>15</td>
</tr>
</tbody>
</table>
F.3.3.7.2 Restricted Dial-in Access
F.3.3.7.3 Hardware Controls for Restricted Dial-in Access
F.3.3.7.4 Restricted Dial-out Access

F.3.4 Processing and Handling Classified Information in AWIPS

F.3.5 Handling and Storage of AWIPS Magnetic or Storage Media

F.3.6 Copyrighted AWIPS Software Control

F.3.7 AWIPS Site Telecommunications

F.4 Implementation of the AWIPS Security Policy

F.4.1 Responding to a Security Incident

F.4.2 Policy Interpretation and Revision

F.4.3 Policy Distribution

F.5 Security Patch Management
This appendix is in accordance with the NWS/AWIPS NOAA8107 System Security Plan (SSP) (dated May 12, 2010), NWS Information Technology Directive (http://www.nws.noaa.gov/directives/sym/pd06007002curr.pdf), and NOAA Common Controls (https://www.csp.noaa.gov/noaa/security-program/CandA/NOAAACC-201004/Common Controls Documentation/NOAA_Common Controls_v4_1_2010_04_07.doc). The SSP was developed in accordance with NIST SP 800-18, Rev-1 and NIST SP 800-53, Rev-3, the NOAA Information Technology Security Manual (ITSM) and Department of Commerce (DOC) IT Security Program Policy (ITSPP), which are web-accessible: https://www.csp.noaa.gov/.

Any updates or changes to the policy and procedures trigger a notification that is sent to the person or persons with security responsibility throughout NOAA via hardcopy distribution as outlined in NOAA Administrative Order (NAO) 212-13 Section 6.04. All policy and procedures have been reviewed and are consistent with all applicable laws, Executive Orders, directives, policies, regulations, standards, and guidance.

The SSP is a living document that is changed as required to reflect system, operational, or organizational changes. The information in this appendix represents the most current policy received to date from the NWS by the Contractor. No changes will be made by the Contractor except for format purposes and the Contractor will only update this section as directed by the Government.

F.0 Introduction

The DOC, NOAA, and NWS are strongly committed to Information Technology (IT) Security. Senior leaders recognize that security must be inherent in our business processes and IT systems. Managers systematically recognize and address information security risks and take steps to understand and manage these risks. Information Technology (IT) systems security for AWIPS presents a constant challenge which must be met at every level of use and management. While various external threats exist, such as: hacker break-ins, terrorist attacks, and malicious intentions, the true risks to security come from poorly configured controls of software and hardware and users/administrators of the system not following the rules set forth in policy.

Use of technology alone, e.g., firewalls and robust operating system software, will not eliminate vulnerabilities in system security. In fact, firewalls can give users a false sense of security leading them to wrongly believe that they no longer need to be personally responsible for following security guidelines. The security of the entire system depends on each user’s individual awareness of and commitment to security policy and procedures. Users and administrators must always be aware of the implications of their actions to the security of the system.

F.0.1 Purpose

This document refers to a comprehensive IT Systems policy for the AWIPS. The high-level security policy delineates the allowed use of AWIPS and the general approach for applying the AWIPS security policy. It provides guidance to all personnel operating or servicing AWIPS hardware, software, or networks, including contract personnel, and specifies services and access controls for all AWIPS equipment. In addition, this
document references guidance for the implementation, maintenance, and monitoring of
the AWIPS Security Policy.

F.0.2 Scope

This document references the controls, rules, roles, and responsibilities necessary for a
viable IT systems security program. It does not specify in detail the procedures,
technologies, or specific systems needed to implement the policy. These details are
provided in separate documents. The scope of this policy includes:

- All of AWIPS;
- Other systems and equipment interfaced to AWIPS, as covered by this policy;
- All National Weather Service (NWS) and contractor personnel who use, operate, or
  service AWIPS hardware, software, or networks.

F.0.3 Applicable Documents

- E Government Act (Public Law 107-347), Title III, Federal Information Security
  Management Act (FISMA) dated December 2002;
- Office of Management and Budget (OMB) Circular A-130 Appendix III, Security of
  Federal Automated Information Resources dated November 2000;
- FIPS PUB 199, Standards for Security Categorization of Federal Information and
  Information Systems dated February 2004;
- FIPS PUB 200, Minimum Security Controls for Federal Systems dated March 2006;
- FIPS PUB 201-1, Personal Identity Verification for Federal Employees and
  Contractors dated March 2006;
- NIST SP 800-16, Information Technology Security Training Requirements: A Role
  and Performance Based Model dated April 1998;
- NIST SP 800-18 Revision 1, Guide for Developing Security Plans for Federal
  Information Systems dated February 2006;
- NIST SP 800-26 Revision 1, Guide for Information Security Program Assessments
  and System Reporting Form dated August 2005;
- NIST SP 800-30, Risk Management Guide for Information Technology Systems
  dated July 2002;
- NIST SP 800-34, Contingency Planning Guide for Information Technology Systems
  dated June 2002;
- NIST SP 800-37, Guide for the Security Certification and Accreditation of Federal
  Information Systems dated May 2004;
- NIST SP 800-47, Security Guide for Interconnecting Information Technology
  Systems dated August 2002;
- NIST SP 800-50, Building an Information Technology Security Awareness and
  Training Program dated October 2003;
Appendix F  

NWS/AWIPS Security Policy

- NIST SP 800-53 Revision 3, Recommended Security Controls for Federal Information Systems dated July 2009;
- NIST SP 800-64 Revision 1, Security Considerations in the Information System Development Life Cycle dated June 2004;
- Department of Commerce (DOC) IT Security Program Policy and Minimum Implementation Standards (ITSPP), Revised January 2009;
- Department of Commerce (DOC) IT Security Program Policy and Minimum Implementation Standards, Revised June 30, 2005
- National Oceanic and Atmospheric Administration (NOAA) Common Controls Version 2.0, March 1, 2009

F.0.4 Changes to AWIPS IT Security Policy

Proposed changes to the AWIPS IT Security Policy may be submitted to the Weather Service Headquarters (WSH) AWIPS Information Technology System Security Officer (ISSO). The AWIPS ISSO will disseminate the proposed change to the AWIPS IT Security Focal Points for review and approval. If approved, it will be submitted to AWIPS management for final review and approval. When approved by Management, the change will be incorporated into the policy.

F.1 Policy

AWIPS is critical to the ability of the NWS to accomplish its primary mission of protecting life and property. AWIPS systems will have levels of control consistent with DOC, NOAA, and NWS policy. The objectives of this policy are to:

- Prevent denial of service. Unauthorized access to the AWIPS computers or networks can cause degradation in or loss of the system’s operational availability or performance. Tampering with software or hardware can result in loss of data, compromised product integrity, or decreased performance of the entire AWIPS network.
- Preserve the integrity of the AWIPS system and data. Unauthorized changes to data or system configurations must be prevented in order to maintain the integrity of: weather data used to prepare forecast products, the forecast products themselves, and the effective dissemination of these products to the users.
- Protect confidentiality. AWIPS contains sensitive information that requires protection from unauthorized disclosure or dissemination. Examples of such information include, but are not limited to: proprietary data (e.g., lightning reports), NWS
products, sensitive system information, proprietary software, and information enabling access to internal/external computer systems.

F.2 Responsibilities

Responsibilities for NWS IT security start at the Office of the Assistant Administrator, and flows down through management of all NWS organizations to individual users. AWIPS IT security responsibilities will comply with all provisions of the NWS Information Technology Security Policy. Responsibilities are defined by a role based approach and are derived from ownership as defined below.

F.2.1 IT System Owner

Ownership of information and/or processing IT resources may be assigned to an organization, NWS, or to a functional element within an organization, such as an installation, a position, or a specific individual such as an IT installation manager at the regional, office, center, or laboratory level. When ownership is assigned to an organization, the designated head of the organization shall be the IT system owner. The originator or creator of data or information who has the greatest functional interest and is the physical possessor of the IT resource shall be considered in the determination of IT system ownership. Each IT system owner shall:

- Determine the sensitivity of the IT resources for which there is a responsibility. (The IT system owner shall periodically reevaluate previously specified levels of sensitivity and protection.)
- Determine the level of security which is consistent with appropriate directives and ensure that this level of protection is maintained on those IT resources.
- Complete all required certification actions.
- Designate an individual to serve as the Information System Security Officer (ISSO) with responsibility to develop, implement and manage the security of the IT system. (At the regional, office, center, or laboratory levels, there is a local Information System Security Officer that may serve as the local focal point for ISSO issues.)

F.2.2 AWIPS ISSO

The AWIPS ISSO is responsible for the direct management of the AWIPS IT systems security program. The AWIPS ISSO requires adequate support staff and budget to accomplish all AWIPS specific security tasks. The AWIPS ISSO will be responsible for ensuring AWIPS compliance with requirements of higher level policies and will:

- Manage IT security for the AWIPS Network Control Facility (NCF) and the AWIPS terrestrial Wide Area Network (WAN).
- Approve security policy, plans, procedures, technologies, mechanisms, and devices for AWIPS.
- Manage Security scans and ensure that patches are implemented for the next OB release to fix identified vulnerabilities in the software/hardware.
• Manage all AWIPS Internet Protocol (IP) address space.
• Approve security alerts, patches, and bug fixes for AWIPS systems.
• Manage and track vulnerabilities identified during Accreditation and Continuous Monitoring of system security controls.
• Manage all WSH AWIPS firewall systems.
• Research and recommend security technologies, mechanisms, devices, and systems for the AWIPS program.
• Assign and approve IP addresses for AWIPS IP address space requests, as appropriate, and maintain the AWIPS IP address data base.
• Maintain liaison with incident response teams from other security organizations.
• Review and implement as directed approved security alerts, patches, and bug fixes required for AWIPS.
• Monitor and ensure the security of all interfaces between AWIPS and external systems.
• Manage the development of AWIPS system-wide security plans, policies, and procedures.
• Develop a review mechanism for AWIPS audit trails which ensures the timely detection of and response to AWIPS security incidents.
• Coordinate development of IT systems security and awareness training for AWIPS System Administrators and users with the NWS Training Center (NWSTC).
• Provide assistance to AWIPS System Administrators and ISSOs on AWIPS IT systems security matters.
• Develop security validation and inspection criteria.
• Review and coordinate physical security requirements for all AWIPS sites.

F.2.3 NWS SISSO

Each Regional ISSO will be responsible for AWIPS IT systems security throughout the region. [The Information Technology Security Officer is the departmental level security officer, i.e., Jeremiah Dewey is the ITSO for OST.] The Regional ISSO will be responsible for all requirements of this and higher level security policies and will:

• Ensure the security of all interfaces between AWIPS and external systems in the region.
• Develop regional security policy, plans, and procedures.
• Evaluate and review security technologies, mechanisms, devices, and systems for all regional sites and recommend changes to the AWIPS ISSO. [The NWS security officer is the Senior Information System Security Officer (SISSO) Robert McKinney.]
• Review, respond to, and report security incidents and maintain logs and records associated with these incidents.
• Maintain liaison with security incident response teams in other regions.
Appendix F

NWS/AWIPS Security Policy

- Issue approved security alerts, patches, and bug fixes to the AWIPS site ISSO’s and monitor completion of the same.
- Develop and coordinate regional IT systems security training and awareness programs for AWIPS Administrators and users, and provide assistance to AWIPS field sites on IT systems security matters.
- Manage the AWIPS Local Data Acquisition and Dissemination (LDAD) firewalls consistent with this and higher level policy and procedures. Any requested changes to the configuration of the LDAD firewall will be approved by the Regional ISSO and reported to the AWIPS ISSO. Current records of configuration, including changes, for all firewalls will be maintained.
- Notify the AWIPS ISSO of unusual system activity reported by field sites and actions/support required to mitigate any attempted or successful breach of security.
- Monitor and review physical security policy, practices, and procedures for all regional AWIPS sites.

F.2.4 Site AWIPS ISSO

The overall security of AWIPS at each site is the responsibility of that site’s manager or system owner as described above. Contractor sites on the autonomous network will be the responsibility of the WSH AWIPS ISSO and prime contractor consistent with this policy and will apply retroactively from the date of this policy. All future contracts for AWIPS will contain the provisions of this policy.

The day-to-day security operations at each AWIPS site are critical to, and provide the basis for the success of the IT systems security program. The name of the ISSO appointed at each site will be provided to the NWS AWIPS ISSO through the Regional ISSO, if applicable. When possible, the Site ISSO should be someone other than the System Administrator. The Site ISSO is responsible for the local management and oversight for the system security. The Site ISSO will ensure compliance with this and higher level policies [The system level security is now the Information System Security Officer (ISSO)] and will:

- Develop local policy, plans and procedures, commensurate with guidance, or required local enhancements to existing policy, plans and procedures.
- Perform special AWIPS IT system reviews after a security incident, and archive all logs and records associated with the event.
- Notify the site manager and the Regional ISSO of any unusual or unexplained activity on the AWIPS system, and prepare incident reports when required (Local procedures will provide after-hours guidance for shift team leaders to respond to security incidents).
- Ensure the implementation of alerts, patches, and bug fixes.
- Ensure only approved software and hardware are integrated into AWIPS.
- Ensure local procedures and station duty manuals provide instructions for incident reporting, site access requirements, personal security-related information, and
information on the release of site-specific sensitive information (e.g., what it is, what the release policy is, who has access, etc.).

- Brief new employees on IT security awareness and training as part of the Entry-on-Duty process.
- Evaluate all IT and automated data processing (ADP) positions for sensitivity level regardless of classified data access.

**F.2.5 AWIPS System Administrator**

The individual site System Administrators (SAs) perform the majority of the administrative work supporting the local security managers. A strong link exists between the individual responsible for security and the program manager. The site System Administrator will:

- Configure site AWIPS resources commensurate with policies and procedures including: adding, configuring, and removing user accounts; managing system resources; and ensuring compliance with security policy, plans and procedures.
- Perform regularly scheduled analysis of system generated logs as well as special reviews after an unanticipated event and provide, as required, reports to the site ISSO and manager.
- Assist Site ISSO in the development of local policy, plans, and procedures.
- Notify the site manager, Site ISSO and Regional ISSO of any unusual activity on the system and prepare incident reports when required.
- Apply alerts, patches, and bug fixes as required.
- Implement approved software and hardware changes to AWIPS.
- Maintain system logon warning banners.
- Immediately notify the AWIPS NCF of breaches in security and ensure mitigation of security problems.

**F.2.6 Operational Shift Team Leaders**

Shift team leaders at operational sites may be required to respond to a security incident. The team leaders, in the absence of the local ISSO or System Administrator, will be responsible for reporting the incident to the NCF. The following guidance will be followed.

- Team leaders will report unexplained, significantly decreased performance, access by unknown persons or sites, corruption of data and/or products, unexplained changes to AWIPS software or hardware configuration, and any other perceived breach of security.
- During regular office work hours (0800 - 1600), all incidents will be reported to the site ISSO or Systems Administrator.
- All after-hours incidents will be reported to the NCF for processing and resolution.
F.2.7 NCF

The support mission of the NCF requires that it respond to all reported AWIPS security events. Should a site experience and/or report a perceived breach of security the NCF will, in coordination with the AWIPS Security Manager, take all required steps to mitigate the problem and return the site to a secure, fully operational state. Notifications and incident reports after hours, including archival of records and logs, is the responsibility of the NCF.

F.2.8 NWSTC

The NWSTC will be responsible for developing security training modules. They will provide security awareness training to AWIPS users during operational training and AWIPS System Security training to System Administrators and ISSOs, as required.

F.2.9 NWS Employees, Non-NWS Government Employees, Non-Government Contractor/Vendor Personnel

Awareness of and concern for security by all AWIPS users provides the baseline from which security can be maintained. Efforts to secure a system will fail without full cooperation and compliance from users. Each AWIPS user will:

- Be responsible for the security of his or her account. Users must protect their account access information and will not share their access information with any other user.
- Read this policy and acknowledge via signature they have read and understand the policy.
- Ensure their actions do not violate the security policy regardless of the security mechanisms in place.
- Report to the AWIPS Site ISSO or the local team leader any perceived breach of security, misuse of AWIPS systems or critical support equipment and systems, unusual requests for access to AWIPS systems accounts and or physical equipment, and any unexplained system difficulties.

F.3 Allowed Use of AWIPS

AWIPS may be used by authorized personnel to support the NWS mission, including forecast and warning operations, training, and/or research and development. User awareness and training, combined with stringent password controls minimize most security risks. Unauthorized user actions leading to disruption or degradation of AWIPS service are prohibited and may be punishable by fine, imprisonment, separation, or administrative consequences. Such actions include, but are not limited to:

- Unauthorized changes to system hardware or software configuration.
- Unauthorized access to AWIPS IT systems or support systems.
- Physical security breaches.
• Actions causing impairment of the AWIPS Network (This type of action may cause an affected site to be disconnected from the WAN pending determination of source and mitigation of problem.)

F.3.1 Access Control

In security vernacular, Discretionary Access Control (DAC) is the level of control applied to files and resources users may access on a system at a given time. These controls provide internal protection of data at the file level and control how, who, and what type of access given files or applications have at a given site. Access Control Lists (ACL) are a means of restricting access to; a network segment, protocol, resource or node. ACL’s can determine who is allowed to see the network, what protocol the network will pass to what device, or enhanced access control to DAC’s for servers.

F.3.1.1 Discretionary Access Control (DAC) and Access Control Lists (ACL)

DAC’s and ACL’s provide layered protection for sensitive data and resources in AWIPS. DACs in AWIPS have design implications. These controls when changed or enhanced at a site will not affect operation of the system as designed. These controls utilized beyond AWIPS baseline will be used at the discretion of system owners at all sites based on need. Department of Defense Instruction 8510.01 provides a guide for ISSO’s use in determining local requirements. Each individual site will consider the totality of the IT systems located, used and controlled specifically at that site as a trusted system or resource for the members of that organization. Minimum security levels will be prescribed by the site ISSO and Manager and implemented by the systems administrator.

ACL’s may be used and implemented at the local level as required and are embedded in the established baseline configuration for AWIPS. Primary ACL’s are derived from network design and security requirements and are implemented by the Network Control Facility. DAC/ACL security mechanisms provide enhanced security to hardware and software resources used in many different ways in AWIPS. Ultimately however, the end goal is the protection of various types of data. Data security for the NWS is required for proprietary data, personnel data, personal data, military/DOD mission support data/products, operational support data/products, copyrighted software and other data peculiar to a site or organization. All of these data are considered sensitive and will meet the requirements established for data security in the National Weather Service, Information Technology Security Policy.

F.3.1.2 Awareness and Password Administration

AWIPS operational readiness and a site’s capability to provide the public with timely warning and forecast products requires continual awareness of system vulnerabilities. Vulnerabilities to a site are multifaceted and may include: physical security vulnerabilities (e.g., unsecured access to communications or environmental controls and hardware, natural phenomena such as hurricanes), collateral equipment vulnerabilities (e.g., unsecured access to computer controlled phone systems, backup power equipment or other critical non-AWIPS mission support equipment), or system specific vulnerabilities (e.g., security hole in the hardware networking infrastructure). Most
vulnerabilities can be controlled through a consistently-applied awareness of security requirements and compliance with published policy.

F.3.1.2.1 Awareness

Site managers, ISSOs and systems administrators will designate which local data is sensitive, determine site physical security and significant collateral equipment security requirements, and review other site-specific security requirements. These local security requirements will be assessed and, as required, policy enhancements will be attached to the AWIPS IT Security Policy. All sites will require authorized AWIPS users to sign a statement of agreement stipulating security policies and procedures have been read and are understood by the employee prior to activation of individual accounts or use of AWIPS assets. Attachment 4 provides a sample form that may be used.

F.3.1.2.2 AWIPS Password Administration

Passwords provide the foundation for AWIPS security. The minimum rules for password use in AWIPS systems follow.

- Root and other system accounts, including the system administrator accounts, are considered sensitive.
- Immediately after completion of site installation activities and departure of the installation team, the local System Administrator will change all awipsusr and textdemo account passwords.
- Access to accounts on all servers and workstations will be strictly controlled at each site. The AWIPS System Administrator and local office Manager or their designee will maintain passwords for the sensitive local accounts in a secure place on station. All requirements for site accounts or access to sensitive accounts are subject to owner approval and configuration. For 24 hour maintenance support the Network Control Facility will be advised of any and all changes to root account passwords. Any and all access to AWIPS servers and workstations can only be granted by the owner or the maintaining unit.
- Password lists to all accounts on every piece of password-protected equipment associated in any way with AWIPS will be kept in a secure place and strictly controlled.

NOTE: A protected file on ADP equipment is not considered secure for this purpose.

- Any unauthorized access to password lists or compromise of passwords on the list will require an immediate change of all affected or compromised passwords.
- The NCF and AWIPS Security Manager will be responsible for, and hold complete control of the passwords for all network specific systems such as routers, hubs, communications processors and ancillary password-protected equipment.
- Passwords for restricted accounts will be set to expire every 6 months. All other requirements specified in Engineering Handbook (EHB)-13 shall be adhered to.
Password expiration for all other users should follow the guidelines established in EHB 13.

- Users are responsible for maintaining and securing their individual account passwords consistent with this policy and local guidance.
- Accounts of users who no longer require access to an AWIPS system on the AWIPS network shall be removed as soon as possible. Accounts of Government or contractor employees who have been reassigned, transferred, or terminated, shall be removed at the earliest practical time. Use of account expiration dates is required to ensure a regular review of the continued need for access.
- All individual users will be assigned a unique user ID and password; sharing of accounts is not permitted. The use of “generic” guest or service accounts to gain host access is not allowed. Except where needed to run specific software packages (e.g., the `awipsusr` or `fxa` account), group accounts are not authorized.

**F.3.2 Use of Non-AWIPS Software on AWIPS Equipment**

All sites will have an applications program manager for locally developed AWIPS applications. The applications manager will provide oversight to all local applications development, including full documentation and functional testing of such applications prior to installation. Extreme caution shall be exerted by local administrators when testing locally developed software on operational resources.

Operational degradation of local or national systems due to non-standard or poorly designed software being installed on operational AWIPS assets will constitute a security violation and must be reported in accordance with this policy.

Non-AWIPS software obtained from commercial sources, other Government agencies, Government contractors, or allied research institutions and locally-developed software may only be installed by the site AWIPS System Administrator or designee. Use of such software on AWIPS equipment must meet all of the following criteria:

- It is required to support the NWS mission.
- It does not interfere with the intended operational use of the system, or networking environment.
- The required approval for installation has been obtained.

**F.3.3 AWIPS Network Security**

The risk of Internet intrusion to AWIPS is significant. This policy supports the operational mission of the NWS field sites while providing for reasonable network isolation from the Internet through firewalls and authentication mechanisms.

**F.3.3.1 Network Operation and Routing Policy**

In this policy the term “trusted network” refers to the AWIPS system and associated equipment that uses the designated AWIPS IP address space. Under no circumstance shall this address space be advertised to the Internet. Bridges, gateways or devices
connected to the Internet will not be connected to AWIPS except through the firewall. The trusted network has been designated as a unique “Autonomous System.” The flow and routing of information and data to different segments of the autonomous system, such as those managed by NWS National Centers, will be managed by the NCF as follows:

- For security purposes, only AWIPS-provided routers will be connected to the AWIPS site Local Area Networks (LANs). The AWIPS routers and hubs will be password protected. Router configuration will be centrally managed and controlled by the NCF and AWIPS Security Manager.
- No other IP network or sub-network shall be connected to the trusted network except by way of an approved firewall system as described in Chapter 4.3.2.
- TCP/IP is the standard protocol suite for the AWIPS network. Since LAN bandwidth is a critical factor in the AWIPS design, it must not be compromised by the use of “chatty,” non-standard, PC LAN protocol suites. Network devices that run other protocol suites, such as Novell SPX/IPX, AppleTalk, DECNET, NETbios, XNS, etc., shall not be connected to the AWIPS site LAN. The AWIPS routers will not route or bridge any data communication protocol suites other than TCP/IP over the AWIPS terrestrial WAN.
- The Message Handling System will be the primary vehicle for moving data and products over the WAN. Priority schemes ensuring this traffic receives top network resource priority will be used.

F.3.3.2 Interfaces with Untrusted Networks

The trusted network may be connected to untrusted networks (i.e., external IP networks or sub-networks including those connected to the Internet) through the approved AWIPS firewalls with prior written approval from the NWS AWIPS ISSO for all WSH and National Center firewalls or regional ISSO for field LDAD firewalls. The following are the conditions for connection to and requirements for firewalls in AWIPS:

- These systems will provide an authentication system for all outside-in connections to AWIPS.
- Any requested connection must support NWS operations, research, and/or training needs.
- The requested connection must be accomplished in a manner that protects the AWIPS network from Internet intrusion.
- Required connections to the untrusted networks will be from the inside-out. Exceptions to this will only be as authorized using Attachment 1 or 2 by the appropriate NWS AWIPS ISSO. Inconvenience will not be considered justification for outside in connections.
- All firewall systems will implement the monitoring, authentication, and access control mechanisms necessary to accommodate the AWIPS allowed-use policy.
- The WSH AWIPS firewall system will be administered by the AWIPS Security Manager, supporting staff, and other personnel approved by the NWS AWIPS ISSO.
• The AWIPS LDAD firewall systems will be configured by the site with the guidance of the Regional ISSO, and will be monitored and managed by the Regional ISSO and supporting staff. Other AWIPS LDAD firewall systems, such as those at NWSTC and National Centers, will be configured and managed by the site ISSO.
• Only the ISSO and AWIPS site System Administrators will have local accounts on the AWIPS firewall systems.
• Traveling lap-top and other home computers will not be used to access the AWIPS autonomous network unless explicitly allowed in writing by the NWS AWIPS ISSO. Lap-top or other computers used on the autonomous network will comply with all other requirements of this policy.

F.3.3.3 Services

The following paragraphs describe the services authorized to be configured between the untrusted networks and the trusted network.

• Transfer of data between trusted and untrusted networks will be by means of File Transfer Protocol (FTP) or Remote Copy (RCP) as restricted by the rule set in Chapter 4.3.4 for all AWIPS firewalls.
• Administrative connectivity through the AWIPS firewalls shall be through Telnet only. X or Windows-related connectivity will not be passed across the firewall for any reason.
• Simple Network Management Protocol (SNMP) will be used from the inside-out to the extent required by the NCF to monitor LDAD functionality.
• Remote Shell (RSH) will be provided from the autonomous network to the LDAD server (inside-out) only as required to support the LDAD functionality.

F.3.3.4 Firewall Rule Set

All AWIPS firewalls will comply with the following rule set. Deviation from these rules will only be via approval of the appropriate ISSO. All approved deviations will be reported to the NWS AWIPS ISSO. Networking configuration for the firewall will support the following AWIPS firewall rules.

• FTP and Telnet Access to the Trusted Network from Untrusted Networks.
  Firewalls shall be initially configured to disallow any connectivity from untrusted (outside-in) networks. Users requiring outside-in connectivity will request it using Attachments 1 or 2. Every effort will be made by approving ISSOs to facilitate user needs via inside-out connections. Users authorized by the appropriate authority, may have access to host computers on the trusted network from untrusted networks provided:
  o Such authorization is granted to individual users, not to organizations, networks or hosts.
  o The authorization is restricted to allow access to specific AWIPS host IP’s from specific untrusted IP’s.
The firewall user authentication mechanism (proxy service) approved by the appropriate authority is implemented as part of the access process through the AWIPS firewall.

Users are only allowed to use the firewall software’s FTP and Telnet proxy services.

- **Firewall proxy services.** Use of the firewall proxy services is subject to restrictions imposed by the authorizing official and deemed necessary for the security of AWIPS.

- **RCP and RSH Access Between the Trusted Network and Untrusted Networks.** Firewalls shall be configured to allow RCP/RSH from the trusted network to the LDAD server only. This access is limited to the functionality provided by the AWIPS software utilization of the LDAD server. These services will not be configured for outside-in access nor will they be provided to any other system.

- **FTP and Telnet Access to the Untrusted Network from Trusted Networks.** Firewalls shall be configured to allow connectivity from the trusted networks (inside-out) to untrusted network resources. Inside-out connectivity shall be used for official purposes only. Any breach of this policy, such as the download of inappropriate data, constitutes a security violation and will be reported as a security incident.

- **SNMP Access to the Untrusted Network.** SNMP shall be configured and allowed to traverse the AWIPS firewall to the extent required by and supported in AWIPS for monitoring the LDAD server operation. The LDAD server will be the only authorized point for SNMP traffic outside the firewall.

- **All Other Access Between Trusted and Untrusted Networks.** In no case shall HTTP or X-Windows related traffic or access be allowed across any AWIPS firewall. These and other protocols may be approved by the NWS AWIPS ISSO on a case-by-case basis, provided that they are shown to be necessary to support the NWS mission or operations.

**F.3.3.5 AWIPS LAN Connections to Non-AWIPS Systems**

Non-AWIPS, NWS mission-support “trusted” IT systems with TCP/IP LAN interfaces (e.g., the NWS’s Console Replacement System) may be interfaced to the AWIPS site LAN and become part of the trusted network, provided:

- They utilize AWIPS IP addresses designated for the trusted network.
- Their use complies with the AWIPS Security Policy and all applicable procedures and guidelines established by the AWIPS IT Security Policy.
- Their interface to the AWIPS LAN has been approved through normal NWS procedures.
- If a major system or new technology, an Interface Control Document for the interface between AWIPS and the trusted system has been written and approved prior to connecting the interface.
F.3.3.6 AWIPS IP Address Space Management

All AWIPS IP addresses will be managed and allocated by the AWIPS Security Manager and may be requested by filling out the IP Address Request Form (Attachment 3) and routing it, through appropriate levels, to the NWS AWIPS ISSO. Any single system required for administrative use or research and training may be installed in the trusted network space with approval of the NWS AWIPS ISSO (Attachment 3). This equipment will have the latest version of NWS supported virus software and all required security patches, updates, and bug fixes installed and will not use active daemons or hardware to allow other untrusted devices to connect to or through them to AWIPS. The local office will be responsible for ensuring protection of AWIPS resources from security vulnerabilities inherent to these types of systems.

F.3.3.7 Dial-in and Dial-out Access

Improperly configured dial-in or dial-out capabilities can create security vulnerabilities by providing unintended “back-door” access to the AWIPS network. For this reason dial-in and dialout access will be restricted to the provisions described in the following paragraphs.

F.3.3.7.1 Restricted LDAD Access

AWIPS Site Managers may authorize individuals (such as Cooperative Users) and external computer systems and sensors to access the AWIPS LDAD servers. These “LDAD users” will not be allowed to access any AWIPS trusted network host computers, other AWIPS equipment, or facilities except those explicitly designated for LDAD use. All LDAD sites allowing login or BBS types of access to LDAD will ensure the appropriate Unix security actions are taken prior to these authorizations.

F.3.3.7.2 Restricted Dial-in Access

Selected AWIPS users may be authorized by the NWS AWIPS ISSO or the applicable regional ISSO to have dial-in access to AWIPS networks, provided that such access is necessary to support AWIPS for NWS operations. Such authorization will be granted to individual users, not to organizations, computer systems, or networks. The authorization will be restricted to access specific AWIPS hosts and to specific time periods the ISSO deems necessary for the security of AWIPS. Such users may dial into the AWIPS networks using only the telephone numbers, modems, terminal servers, associated security equipment, and procedures approved by the respective ISSO.

F.3.3.7.3 Hardware Controls for Restricted Dial-in Access

Each site manager will ensure modems used to perform maintenance and monitor and control functions during emergencies or any other modem, (except WAN backup circuits) with access or connection to the trusted AWIPS networks or equipment has a locally-installed hardware device which provides a disconnect for the phone lines to those modems when not in use. Procedures for connecting the lines via this device for emergency purposes shall be added to the local station duty manual and all personnel
shall be trained in its use and location. Prior to connecting this device by request from outside sources, steps shall be taken to validate the source of the request (e.g., call back authentication).

**F.3.3.7.4 Restricted Dial-out Access**

All AWIPS dial-out services and equipment must conform to the guidelines established by the AWIPS System Manager’s Manual. Extreme care must be taken to prevent the misuse of dial-out modems that would permit unauthorized access into AWIPS from external networks. Use of dial-out modems to establish IP connections (e.g., using SLIP or PPP protocols) with external networks is not permitted.

**F.3.4 Processing and Handling Classified Information in AWIPS**

The protection and transmission of classified information has varying degrees of requirements depending on the classification of the data. Classified information must not be processed on IT systems which do not have DOC/NOAA approval to do so. The NWS AWIPS ISSO has additional information concerning the processing and handling of classified information on NOAA IT systems.

**F.3.5 Handling and Storage of AWIPS Magnetic or Storage Media**

Sensitive or classified data stored on portable media must be protected in a manner similar to the hard-copy equivalent. This will require the storage of backup or archived portable media in secure containers on and off site. Requirements for hard copy information security are explained in Department Administrative Order (DAO) 207-1, Security Programs, dated 24 June 1991. The NWS AWIPS ISSO has additional information concerning the handling and storage of portable storage media.

**F.3.6 Copyrighted AWIPS Software Control**

Public Law 102-561 amends the Copyright Act, Title 18, United States Code, with respect to the criminal penalties for software copyright infringement. This law now makes certain acts Federal felonies with fines, imprisonment, or both as penalties. Criminal liability results when within a 180-day period at least 10 copies of a copyrighted work with a retail value in excess of $2,500 have been reproduced or distributed. The act must also have been willful, for commercial advantage, or for private financial gain. Lesser acts of infringement remain a misdemeanor which is punishable by a fine. DOC policy against acts of copyright violation for software purchased by the DOC and its operating units is provided in Chapter 10.11 of the U.S. Department of Commerce Information Technology Management Handbook. All provision of this policy will apply to AWIPS and any software procured for AWIPS systems. Software is automatically protected by Federal copyright law from the moment of its creation. Under the Copyright Act, software developers have exclusive rights to reproduce their copyrighted work. Persons who purchase a copy of software only have the right to:

- Copy or install the software onto a single computer.
- Make another copy for archival purposes only.

When upgrades to software are purchased, the old version must either be destroyed or returned to the software developer. It is illegal to use the old software version on another computer. Upgraded software is usually purchased at a reduced cost and is considered a continuation of the original license and not an additional license. Each AWIPS site will ensure compliance with licensing agreements issued by the manufacturer for software purchased by the Government. Copies of original software, upgrades, and documentation shall be accounted for and retained.

**F.3.7 AWIPS Site Telecommunications**

The NWS AWIPS ISSO is responsible for recommending procedures and providing advice regarding adequate protection for the communications of sensitive information in accordance with the U.S. Department of Commerce Information Technology Security Manual. National security or classified information **must not** be discussed over or otherwise transmitted or processed by any form of telecommunications unless approved measures are taken to protect the information. Policies and specifications dealing with the security of Federal telecommunications are developed and issued under the purview of the National Security Council. These policies are stated in the U.S. Department of Commerce Information Technology Security Manual and the U.S. Department of Commerce National Security Information Manual.

**F.4 Implementation of the AWIPS Security Policy**

AWIPS Security is implemented in accordance with the SSP.

**F.4.1 Responding to a Security Incident**

A security incident is any event that could pose a threat to the integrity, availability, or confidentiality of AWIPS equipment, applications, or data. It is the responsibility of an affected site to report suspected security incidents to the NCF, AWIPS Security Manager and local or regional NWS AWIPS ISSO who in turn will report them as required by the National Weather Service, Information Technology Security Policy to the appropriate authority. The latest “ADP Security Incident Report” form, available at the NOAA security web site (www.rdc.noaa.gov/~security/) shall be filled out by the System Administrator at the affected site and submitted. The NCF and AWIPS Security Manager, if required, will coordinate with the NOAA Telecommunications and ADP Security Branch and the affected site’s local or regional ISSO to determine the necessary course of action and provide higher-level reporting. Depending on the severity of the incident and the level of threat, the Federal Bureau of Investigation may become involved and actions may be taken leading to the apprehension and eventual prosecution of a violator. Since the availability of AWIPS is critical to the NWS mission to protect life and property, the highest possible response priority will be given to the restoration of service and the protection of the system from further attack.
F.4.2 Policy Interpretation and Revision

Threats to computer security are constantly evolving. AWIPS also is evolving with the development of new software, hardware, techniques, operational procedures, and observation and dissemination systems. Because of these evolutionary changes, it is essential to regularly review and update the AWIPS Security Policy and related procedures. The NWS AWIPS ISSO will ensure that the policy is reviewed at least once a year and updated as needed.

F.4.3 Policy Distribution

The AWIPS Security Policy will be distributed to all AWIPS system owners and support organizations. All AWIPS users will review the policy annually and sign a statement acknowledging their understanding of and agreement to abide by the AWIPS Security Policy. This agreement can be locally produced and will be maintained at each site.

F.5 Security Patch Management

Attachment 1
Request for User Id and Password on
The AWIPS WSH Firewall System

Please fill out this form up through “Supervisor’s Approval”. Then send or fax the form to: Fax 301-713-0173
AWIPS Security Manager, NOAA NWS SSMC2, Rm 15404, 1325 East West Hwy, Silver Spring, MD 20910.

Note: This request assumes that the requester already has obtained a valid user account on an AWIPS System and
needs to access his/her account from a system that is not connected to the AWIPS LAN or WAN (e.g., access from a
workstation connected to an untrusted network such as a Local LAN/WAN or INTERNET).

Name of Requester, organization, and code:

Phone Number & ccmail or email address:

Justification, use and destination: State why and where if not on ds1 at your site you need to access your account
on AWIPS from a system that is not connected to the AWIPS LAN or WAN and describe your use, e.g., transfer
files via FTP, run Telnet session for ...., etc.).

Indicate the IP address and domain name of the workstation or PC on INTERNET that you wish to use to
communicate with the AWIPS WSH Firewall System and a preferred username of six characters.

I have read the AWIPS Security Policy and agree to abide by that policy.
I will not disclose my user ID and password to other persons.

Requested by: ___________________________  Signed _____________  Date _____________

Supervisor’s approval: ___________________________  Signed _____________  Date _____________

Security Officer Approval to establish user access to AWIPS firewall system: ___________________________  Signed _____________  Date _____________

User access to AWIPS firewall system established: ___________________________  Signed _____________  Date _____________
## Attachment 2
### Request for User Id and Password in the AWIPS LDAD Firewall System

Please fill out this form up through “Supervisor’s Approval”. Then send or fax the form to: Regional ISSO, with a cc: to: AWIPS Security Manager, NOAA NWS SSMC2, Rm 15404, 1325 East West Hwy, Silver Spring, MD 20910.

**Note:** This request assumes that the requester already has obtained a valid user account on an AWIPS System and needs to access his/her account from a system that is not connected to the AWIPS LAN or WAN. (e.g., access from a workstation connected to an untrusted network such as a local LAN or INTERNET.)

### Name of Requester, organization, and code:

### Phone Number & ccmail or email address:

**Justification and use:** State why you need to access your account on AWIPS via FTP or Telnet from a system that is not connected to the AWIPS LAN or WAN and describe your use, e.g., transfer files via FTP, run Telnet session to access data, etc.

### Indicate the IP addresses of each workstation or PC on INTERNET that you wish to use to communicate with the AWIPS LDAD Firewall System:

I have read the AWIPS Security Policy and agree to abide by that policy.
I will not disclose my user ID and password to other persons.

<table>
<thead>
<tr>
<th>Requested By:</th>
<th>Signed</th>
<th>Date</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Supervisor’s approval:</th>
<th>Signed</th>
<th>Date</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Regional Approval to establish user access to AWIPS LDAD firewall system:</th>
<th>Signed</th>
<th>Date</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>User access to LDAD firewall system established:</th>
<th>Signed</th>
<th>Date</th>
</tr>
</thead>
</table>
## Attachment 3

### Request for AWIPS IP Address

Please fill out this form up through “Supervisor’s Approval”. Then send or fax the form to: Fax 301-713-0173 or AWIPS NWS ISSO, NOAA NWS SSMC2 Rm 15426, 1325 East West Hwy, Silver Spring MD 20910, via the regional ISSO, with a copy to the NCF.

<table>
<thead>
<tr>
<th>Name of Requester:</th>
<th>__________________________________________________</th>
</tr>
</thead>
<tbody>
<tr>
<td>Organization:</td>
<td>___________________________________________________</td>
</tr>
<tr>
<td>Telephone Number:</td>
<td>__________________________________________________</td>
</tr>
<tr>
<td>Proposed Workstation/PC name:</td>
<td>________________________________________</td>
</tr>
<tr>
<td>Machine Type/Operating System:</td>
<td>________________________________________</td>
</tr>
<tr>
<td>System Administrator:</td>
<td>________________________________________________</td>
</tr>
</tbody>
</table>

Justification (Please state what work will be performed on this workstation/PC, and how it relates to the AWIPS system):

Primary Network Applications To Be Utilized (e.g., TELNET, FTP, etc):

I certify that the above stated workstation/PC complies with applicable NOAA UNIX security recommendations, is compliant with AWIPS security policy, and poses minimal risk to the security of the AWIPS system. I understand that the NCF reserves the right to filter any IP traffic that is generated by the above-named device which may be degrading the operational site AWIPS equipment or degrading the operation of the AWIPS Terrestrial Network (ATN).

Requested by: | Signed | Date
--- | --- | ---
NWS Supervisor’s Approval: | Signed | Date
Regional Approval: | Signed | Date
AWIPS NWS ISSO: | Signed | Date

ASSIGNED IP ADDRESS:
Attachment 4
User Acknowledgment

As a user of the AWIPS system, I acknowledge my responsibilities to conform to the following requirements and conditions established by the AWIPS Security Policy.

1. I understand that failure to sign this acknowledgment will result in denial of access to the AWIPS system.

2. I understand the need to protect system and individual passwords. I will not share my accounts or passwords. I understand that I am responsible for all actions taken using my accounts. I understand my responsibility to appropriately protect all output generated by me on AWIPS (including printed output, magnetic tapes, floppy diskettes, and downloaded hard disk files) and to properly label all media.

3. I understand my responsibility to report any/all systems or network problems to the system administrator. I will not install, modify, or remove any hardware or software without the permission of the system administrator.

4. I will not induce or process data which the network is not specifically authorized to handle. I acknowledge my responsibility to use the AWIPS system/network only for official NWS business.

5. I understand my use of AWIPS systems/networks is subject to monitoring to ensure proper functioning, to protect against improper or unauthorized use or access, and to verify the presence or performance of applicable security features or procedures. By using the AWIPS system, I grant consent to such monitoring.

6. I will protect my equipment by keeping food, drink, and electrical appliances away from AWIPS resources I use or maintain.

7. I will protect my work area by politely challenging and assisting people who do not belong in the area.

8. I understand my responsibility to backup data as required by local policy and procedures.

9. I acknowledge my responsibility to conform to the requirements of this document and the AWIPS Security Policy when using or maintaining any AWIPS information systems or networks. I also acknowledge that failure to comply with these requirements and conditions may constitute a security violation resulting in denial of access to AWIPS information systems, networks, or facilities and further action as deemed appropriate.

Signature_________________ Date ___________
Appendix G
AWIPS Password Management Policy

To view the criteria for setting up acceptable passwords, see Section 3.1, General Guidance on Individual User Accounts, of this System Manager’s Manual.
Appendix H
SBN Products for WAN Transmission
During an SBN Failure
Appendix H. SBN Products for WAN
Transmission During an SBN Failure

Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>H.1 Transmitted SBN Products</td>
<td>1</td>
</tr>
<tr>
<td>H.2 Current Address Lists</td>
<td>1</td>
</tr>
</tbody>
</table>
Certain SBN products will be transmitted over the WAN in the event of an SBN failure. The current list of those SBN products is provided in this appendix.

### H.1 Transmitted SBN Products

The file, `/awips/ops/data/wanbu_prodtbl.nmc`, resides on dx1. It contains the default address lists, which consist of the product IDs and the default attributes, in the following format:

```
"<prodID>|<sitelist>|<priority>|<valid_time>
```

**Note**: `<prodID>` must be in quotes, as shown.

Example:

```
"^YH[IJ][A-K](99|85|70|40|30|20|15|10) KWBC"|ALL
```

where

- `<prodID>` is "^YH[IJ][A-K](99|85|70|40|30|20|15|10) KWBC"
- `<sitelist>` is `ALL`
- `<priority>` and `<valid_time>` are not specified for these products

### H.2 Current Address Lists

The current file default address lists follow. Note that `<prodID>` and `<sitelist>` (`= ALL`) are the only attributes currently specified for any of these products.

```
"^AB.* PGTW"|ALL
"^TP.* KGWC"|ALL
"^FZ.* KNHC"|ALL
"^FZ.* KNHC"|ALL
"^WS"|ALL
"^AX.* KNHC"|ALL
"^WT.* RJTD"|ALL
"^AB.* KNHC"|ALL
"^NOUS4. KNHC"|ALL
"^UR.* KNHC"|ALL
"^UZ.* KNHC"|ALL
"^ACPN.. PHFO"|ALL
"^ACP.. PHFO"|ALL
"^TXUS20| "|ALL
"^TCUS"|ALL
"^SE"|ALL
"^WE"|ALL
"^FY"|ALL
"^W"|ALL
"^FWUS"|ALL
"^KQXX"|ALL
"^NOUS.. KWNO"|ALL
"^NOUS.. KWBC"|ALL
```

# NC NWP Text
```
"^FOUS.. KWNO"|ALL
"^FOUE.. KWNO"|ALL
```

AWP.MAN.SMM.A2 H-1

– Hard copies uncontrolled. Verify effective date prior to use. –
acqparms.uplink                build1.cat
acqparms.uplink_h0             cp_filter.cfg
acqparms.uplink_h0h1           datemask.dat
acqparms.uplink_msg             DateTemplate.fil
acqparms.vrh                  dwb_startall.ini
acq_send_parms.sbn             dwb.sym
acq_wmo_file_parms.hazcollect  dwb.sym.err
acq_wmo_file_parms.nwstg        dwb
acq_wmo_file_parms.nwws         linux_install.txt
acq_wmo_file_parms.sbn          mc
acq_wmo_parms.acr              mcdb
acq_wmo_parms.afc              mhs
acq_wmo_parms.afg              nesdis_files.east.sbn
acq_wmo_parms.ajk              nesdis_files.west.sbn
acq_wmo_parms.gum              radar_cccc.txt
acq_wmo_parms.hfo              radarInfo.txt
acq_wmo_parms.ml1              radar_subcat.txt
acq_wmo_parms.ml2              retrans_prodtbl.goes
acq_wmo_parms.ml3              retrans_prodtbl.goes_east
acq_wmo_parms.ml4              retrans_prodtbl.goes_west
acq_wmo_parms.sbn              retrans_prodtbl.nmc
acq_wmo_parms.sbn.east         retrans_prodtbl.nmc2
acq_wmo_parms.sbn.local        retrans_prodtbl.nmc3
acq_wmo_parms.sbn.radar        retrans_prodtbl.noaaport_opt
acq_wmo_parms.sbn west         rpt_monitor
acq_wmo_parms.sju              siteid_to_cccc.txt
acq_wmo_parms.sto              TargetsList
acq_wmo_parms.tbw4             wanbu_prodtbl.nmc
acq_wmo_parms.vrh              wanbu_prodtbl.nmc

#----------------------------------------------------------------------
# NC NWP Text
#----------------------------------------------------------------------

"AB.* PGTW"|ALL
"TP.* KGWC"|ALL
"FZ.* KNHC"|ALL
"FZ.* KNHC"|ALL
"WS"|ALL
"AX.* KNHC"|ALL
"WT.* RJTD"|ALL
"AB.* KNHC"|ALL
"NOUS4. KNHC"|ALL
"UR.* KNHC"|ALL
"UZ.* KNHC"|ALL
"ACP2. PHFO"|ALL
"ACP3. PHFO"|ALL
"TXUS20"|ALL
"TCUS"|ALL
"SE"|ALL
"WE"|ALL
"FW"|ALL
"NN"|ALL
"FWUS"|ALL
"NOXX"|ALL
"NOUS. KWNO"|ALL
"NOUS. KWBC"|ALL

#----------------------------------------------------------------------
# Hard copies uncontrolled. Verify effective date prior to use.
Appendix I
Administrative Tips for Monitoring
System Performance
Appendix I. Administrative Tips for Monitoring System Performance

Table of Contents

I.1 Check to See Where the Packages Are Running .............................................................. 1
I.2 Check for Mandatory Processes on DX1F ................................................................. 1
I.3 Check for Mandatory Processes on DX2F ................................................................. 8
I.4 Check for Mandatory Processes on DX3/DX4 ............................................................. 11
I.5 Check for Abnormal Processes on CP1F ................................................................. 14
I.6 Check for Abnormal Processes on CPSBN1/CPSBN2 .............................................. 16
I.7 Check for Abnormal Processes on AX ................................................................. 17
I.8 Check for Mandatory Processes on PX1 ................................................................. 17
I.9 Check for Mandatory Processes on PX2 ................................................................. 18
I.10 Check for Mandatory Processes on REP Servers (RFCs only) .................................. 21
I.11 Checklist for Servers and Workstations ................................................................. 22
   I.11.1 Servers ................................................................................................................ 22
   I.11.2 Workstations ...................................................................................................... 31
   I.11.3 All Machines ...................................................................................................... 32
To monitor system performance, complete the following procedures in sequential order at least ONCE per day.

I.1 Check to See Where the Packages Are Running

To verify that the Red Hat Cluster Manager swap packages have been enabled for the px1apps, px2apps, dx1apps, dx2apps swap packages, as user root on the respective servers,

**TYPE:** \texttt{hb\_stat}

An example of dx1apps follows.

```
[root@dx1-tbw3 ~]# hb_stat
Heartbeat Status Monitor       Jul 09 18:39:56

Member Status IP address
---------- ---------- ---------------
dx1-tbw3 Up 165.92.24.1

Service Status
------------- --------------- ---------- ------------------
Service IPaddr Cronfile Owner Start Time
---------- --------------- ---------- ------------------
a2dx1apps 165.92.24.65 a2dx1cron,a2SIT dx1-tbw3 2012-05-29 19:35:31
```

I.2 Check for Mandatory Processes on DX1F

Log in to the Linux data server running the dx1apps package (normally DX1F).

1. Check what is mounted on the server. Verify that the NAS is mounted.

**TYPE:** \texttt{df -H}

- Look for /data/\texttt{fxa} and the percentage used. Percentages larger than 70 percent could indicate a problem. Notify the Network Control Facility if this is the case.
- Also look for any directories that appear to be filling up (88 percent or higher).

An example follows for RFC.

```
Filesystem Size Used Avail Use% Mounted on
/dev/mapper/vg01-lvol01 985M 499M 436M 54% /
tmpfs 5.9G 0 5.9G 0% /dev/shm
/dev/mapper/vg01-lvol_fxa 4.9G 3.9G 703M 85% /awips/fxa
/dev/mapper/vg01-lvol_ifps 1.3G 79M 1.2G 7% /awips/ifps
```
Appendix I

Administrative Tips for Monitoring System Performance

/dev/mapper/vg01-lvol_ldad 985M 87M 848M 10% /awips/ldad
/dev/mapper/vg01-lvol_ops 291M 68M 209M 25% /awips/ops
/dev/mapper/vg01-lvol_awips2 32G 3.0G 27G 11% /awips2
/dev/mapper/vg01-lvol_logs 9.9G 153M 9.2G 2% /data/logs
/dev/mapper/vg01-lvol_tmp 985M 27M 908M 3% /tmp
/dev/mapper/vg01-lvol_usr 9.7G 8.3G 939M 90% /usr
/dev/mapper/vg01-lvol_var 2.9G 1.7G 1.1G 60% /var
nas1:/awipsADAPT 1.0G 164M 860M 17% /awips/adapt
nas1:/awipsDEV 4.0G 1.1M 4.0G 1% /awips/dev
nas1:/awipsGFESuit 100G 24G 77G 24% /awips/GFESuite
nas1:/awipsHYDRO 100G 50G 51G 50% /awips/hydroapps
nas1:/dataLOCAL 150G 20G 131G 14% /data/local
nas1:/dataX400 600M 12M 589M 2% /data/x400
nas1:/DSshared 1.5G 207M 1.3G 14% /DS_shared
nas1:/dataADAPT 3.0G 0 3.0G 0% /data/adapt
nas1:/dataGFE 20M 0 20M 0% /data/GFE
nas1:/awipsHOME 110G 54G 57G 49% /home
nas1:/dataFXA 500G 338G 163G 68% /data/fxa
nas1:/aiidata/utility 500G 14G 487G 3% /awips2/edex/data/utility
nas1:/localapps 50G 20M 50G 1% /localapps
nas1:/dataSTORE 500G 47G 454G 10% /data_store
nas1:/storage 500G 73G 428G 15% /awips/storage
/dev/sda3 97M 64M 29M 70% /boot
/dev/mapper/vg_aiidb-awipsiidb 148G 17G 125G 12% /awips2/data
nas1:/GFESuite2 10G 116M 9.9G 2% /awips2/GFESuite

/dev/mapper/vg_aiihdf5-awipsiihdf5 393G 56G 318G 15% /awips2/edex/data/hdf5

An example follows for WFO.

Filesystem Size Used Avail Use% Mounted on
/dev/mapper/vg01-lvol01 985M 426M 509M 46% /
tmpfs 5.9G 0 5.9G 0% /dev/shm
/dev/mapper/vg01-lvol_fxa 4.9G 1.7G 2.9G 38% /awips/fxa
/dev/mapper/vg01-lvol_ifps 1.3G 79M 1.2G 7% /awips/ifps
/dev/mapper/vg01-lvol_ldad 985M 80M 856M 9% /awips/ldad
/dev/mapper/vg01-lvol_ops 291M 190M 86M 69% /awips/ops
/dev/mapper/vg01-lvol_awips2 32G 2.2G 28G 8% /awips2
/dev/mapper/mpathap3 97M 64M 29M 70% /boot
/dev/mapper/vg01-lvol_logs 9.9G 152M 9.2G 2% /data/logs
/dev/mapper/vg01-lvol_tmp 985M 20M 916M 3% /tmp
/dev/mapper/vg01-lvol_usr 9.7G 7.8G 1.5G 85% /usr
/dev/mapper/vg01-lvol_var 2.9G 768M 2.0G 28% /var
nas1:/awipsADAPT 1.0G 669M 356M 66% /awips/adapt
nas1:/awipsDEV 4.0G 128K 4.0G 1% /awips/dev
nas1:/awipsGFESuit 100G 17G 84G 17% /awips/GFESuite
nas1:/awipsHYDRO 100G 770M 100G 1% /awips/hydroapps
Verify that the fxa processes listed below are running.

### TYPE:

```
ps -wef | grep fxa
```

An example follows.

```
awips     2245 12599  0 13:39 ?        00:00:12 postgres: awips
fxatext 165.92.24.3(53451) idle
awips     3159 12599 20 15:20 ?        00:01:19 postgres: awips
fxatext 165.92.24.4(44340) idle
awips     3762 12599  0 15:21 ?        00:00:00 postgres: awips
fxatext 165.92.24.4(44403) idle
awips     4532 12599  0 15:23 ?        00:00:00 postgres: awips
fxatext 165.92.24.4(44488) idle
awips     4662 12599  0 15:23 ?        00:00:00 postgres: awips
fxatext 165.92.24.4(44488) idle
awips     4726 12599  0 15:23 ?        00:00:00 postgres: awips
fxatext 165.92.24.3(56394) idle
awips     7113  5128  0 15:26 pts/0    00:00:00 grep fxa
awips    11105 12599  4 14:43 ?        00:01:51 postgres: awips
fxatext 165.92.24.4(42423) DELETE
awips    11215 12599  0 14:44 ?        00:00:04 postgres: awips
fxatext 165.92.24.3(55144) idle
awips    11586 12599  0 14:44 ?        00:00:04 postgres: awips
fxatext 165.92.24.3(55157) idle
awips     12728     1  0 Mar14 ?        00:50:14 /awips2/java/bin/java
               -cp /awips2/rcm/data/config/res:/awips2/rcm/lib/activation-
               1.1.jar:/awips2/rcm/lib/activeio-core-
               3.1.2.jar:/awips2/rcm/lib/activemq-camel-
               5.3.0.jar:/awips2/rcm/lib/activemq-console-
               5.3.0.jar:/awips2/rcm/lib/activemq-core-
               5.3.0.jar:/awips2/rcm/lib/activemq-jaas-
               5.3.0.jar:/awips2/rcm/lib/activemq-pool-
               5.3.0.jar:/awips2/rcm/lib/activemq-web-
```

Verify that the hard copies are uncontrolled. Verify effective date prior to use.
Appendix I  Administrative Tips for Monitoring System Performance

5.3.0.jar:/awips2/rcm/lib/backport-util-concurrent-
2.1.jar:/awips2/rcm/lib/commons-beanutils-
1.8.3.jar:/awips2/rcm/lib/commons-codec-
1.4.jar:/awips2/rcm/lib/commons-collections-
3.2.jar:/awips2/rcm/lib/commons-lang-2.3.jar:/awips2/rcm/lib/commons-
logging-1.1.2.jar:/awips2/rcm/lib/geronimo-j2ee-management_1.0_spec-
1.0.jar:/awips2/rcm/lib/geronimo-jms_1.1_spec-
1.1.1.jar:/awips2/rcm/lib/geronimo-jta_1.0.1B_spec-
1.0.1.jar:/awips2/rcm/lib/jbzip2-0.9.1.jar:/awips2/rcm/lib/jcl-over-
slf4j-1.7.5.jar:/awips2/rcm/lib/jencks-amqpool-
2.0.jar:/awips2/rcm/lib/jul-to-slf4j-1.7.5.jar:/awips2/rcm/lib/log4j-
1.2.16.jar:/awips2/rcm/lib/log4jextras-
1.0.jar:/awips2/rcm/lib/log4j-over-slf4j-1.7.5.jar:
1.7.5.jar:/awips2/rcm/lib/mina-core-1.1.7.jar:/awips2/rcm/lib/mina-
filter-codec-netty-1.1.7.jar:/awips2/rcm/lib/mina-filter-compression-
1.1.7.jar:/awips2/rcm/lib/mina-filter-ssl-
1.1.7.jar:/awips2/rcm/lib/mina-integration-jmx-
1.1.7.jar:/awips2/rcm/lib/mina-integration-spring-
1.1.7.jar:/awips2/rcm/lib/org.springframework.beans-
3.1.4.RELEASE.jar:/awips2/rcm/lib/org.springframework.beans-sources-
3.1.4.RELEASE.jar:/awips2/rcm/lib/org.springframework.context-
3.1.4.RELEASE.jar:/awips2/rcm/lib/org.springframework.context.sur-
3.1.4.RELEASE.jar:/awips2/rcm/lib/org.springframework.context.support-
3.1.4.RELEASE.jar:/awips2/rcm/lib/qpid-client-0.18.jar:/awips2/rcm/lib/qpid-client-0.18-
sources.jar:/awips2/rcm/lib/qpid-common-
0.18.jar:/awips2/rcm/lib/qpid-common-0.18-
sources.jar:/awips2/rcm/lib/radarserver-mq.jar:/awips2/rcm/lib/slf4j-
api-1.7.5.jar:/awips2/rcm/lib/slf4j-api-1.7.5-
sources.jar:/awips2/rcm/lib/slf4j-log4j12-1.7.5.jar -
Dcom.raytheon.rcm.configDir=/awips2/rcm/data/config -
Dcom.raytheon.rcm.awips1.decompressRadarProducts=yes -
Dcom.raytheon.rcm.logDir=/awips2/rcm/data/logs -
Dlog4j.configuration=log4j.properties -Dqpid.dest_syntax=BURL -
Dawsips2_fxa=/awsips2/fxa -
Dcom.raytheon.rcm.edexRadarEndpoint=/data_store/radar -
com.raytheon.rcm.mqsrvr.MQServer

fxa 12989  1  0 Mar14 ? 00:00:01
/awsips2/fxa/bin/notificationServer

fxa 13751  1  0 Mar14 ? 00:00:00
/awsips2/fxa/bin/MhsServer

fxa 14243  1  0 Mar14 ? 00:00:00
/awsips2/fxa/bin/NWWSProduct

fxa 16074  1  0 Mar14 ? 00:00:34 /usr/bin/perl
/awsips2/fxa/bin/nwrTrans.pl

awips 19887 12599  5 14:57 ? 00:01:40 postgres: awips
/fxatext 165.92.24.4(43194) idle
awips 23171 12599  0 15:00 ? 00:00:03 postgres: awips
/fxatext 165.92.24.3(55694) BEGIN
awips 23472 12599  4 15:00 ? 00:01:13 postgres: awips
/fxatext 165.92.24.4(43381) idle
awips 29080 12599  7 15:10 ? 00:01:09 postgres: awips
2. Look for abnormal CPU usage. Greater than 75 percent resourced by an individual process may cause problems.

TYPE: `top`

```
top - 18:37:10 up 114 days, 4:14, 5 users, load average: 3.38, 3.56, 3.87
Tasks: 555 total, 2 running, 553 sleeping, 0 stopped, 0 zombie
Cpu(s):  6.8%us, 1.7%sy, 0.0%ni, 81.4%id, 9.9%wa, 0.0%hi, 0.2%si, 0.0%st
Mem: 12464784k total, 12050292k used, 414492k free, 116852k buffers
Swap: 2199544k total, 3312k used, 2196232k free, 9863172k cached
```

<table>
<thead>
<tr>
<th>PID</th>
<th>USER</th>
<th>PR</th>
<th>NI</th>
<th>VIRT</th>
<th>RES</th>
<th>SHR</th>
<th>S</th>
<th>%CPU</th>
<th>%MEM</th>
<th>TIME+</th>
<th>COMMAND</th>
</tr>
</thead>
<tbody>
<tr>
<td>24115</td>
<td>awips</td>
<td>15</td>
<td>0</td>
<td>59320</td>
<td>41m</td>
<td>7372R</td>
<td>39.7</td>
<td>0.3</td>
<td>0:15.96</td>
<td>httpd</td>
<td></td>
</tr>
<tr>
<td>15542</td>
<td>awips</td>
<td>16</td>
<td>0</td>
<td>75172</td>
<td>57m</td>
<td>7388S</td>
<td>27.2</td>
<td>0.5</td>
<td>0:36.76</td>
<td>httpd</td>
<td></td>
</tr>
<tr>
<td>31819</td>
<td>awips</td>
<td>15</td>
<td>0</td>
<td>40740</td>
<td>22m</td>
<td>7336S</td>
<td>13.6</td>
<td>0.2</td>
<td>0:04.90</td>
<td>httpd</td>
<td></td>
</tr>
<tr>
<td>30560</td>
<td>awips</td>
<td>16</td>
<td>0</td>
<td>75248</td>
<td>57m</td>
<td>7380S</td>
<td>12.6</td>
<td>0.5</td>
<td>0:08.62</td>
<td>httpd</td>
<td></td>
</tr>
<tr>
<td>1188</td>
<td>awips</td>
<td>15</td>
<td>0</td>
<td>1623m</td>
<td>170m</td>
<td>149mS</td>
<td>8.3</td>
<td>1.4</td>
<td>0:00.70</td>
<td></td>
<td></td>
</tr>
<tr>
<td>30505</td>
<td>awips</td>
<td>15</td>
<td>0</td>
<td>45676</td>
<td>27m</td>
<td>7336S</td>
<td>7.0</td>
<td>0.2</td>
<td>0:14.44</td>
<td>httpd</td>
<td></td>
</tr>
<tr>
<td>7581</td>
<td>awips</td>
<td>15</td>
<td>0</td>
<td>1601m</td>
<td>356m</td>
<td>352mS</td>
<td>4.0</td>
<td>2.9</td>
<td>0:28.86</td>
<td></td>
<td></td>
</tr>
<tr>
<td>22184</td>
<td>awips</td>
<td>16</td>
<td>0</td>
<td>1599m</td>
<td>368m</td>
<td>365mD</td>
<td>4.0</td>
<td>3.0</td>
<td>0:54.67</td>
<td></td>
<td></td>
</tr>
<tr>
<td>29542</td>
<td>awips</td>
<td>15</td>
<td>0</td>
<td>1599m</td>
<td>180m</td>
<td>178mS</td>
<td>4.0</td>
<td>1.5</td>
<td>0:12.75</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7544</td>
<td>awips</td>
<td>15</td>
<td>0</td>
<td>1599m</td>
<td>175m</td>
<td>172mS</td>
<td>3.6</td>
<td>1.4</td>
<td>0:11.40</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1182</td>
<td>awips</td>
<td>16</td>
<td>0</td>
<td>1619m</td>
<td>107m</td>
<td>92mD</td>
<td>3.3</td>
<td>0.9</td>
<td>0:00.40</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4857</td>
<td>awips</td>
<td>17</td>
<td>0</td>
<td>63172</td>
<td>45m</td>
<td>7416S</td>
<td>3.3</td>
<td>0.4</td>
<td>0:38.65</td>
<td>httpd</td>
<td></td>
</tr>
<tr>
<td>939</td>
<td>root</td>
<td>15</td>
<td>0</td>
<td>2712</td>
<td>1344</td>
<td>812R</td>
<td>0.7</td>
<td>0.0</td>
<td>0:00.21</td>
<td>top</td>
<td></td>
</tr>
<tr>
<td>20122</td>
<td>awips</td>
<td>15</td>
<td>0</td>
<td>1596m</td>
<td>60m</td>
<td>58mS</td>
<td>0.7</td>
<td>0.5</td>
<td>0:00.69</td>
<td></td>
<td></td>
</tr>
<tr>
<td>31154</td>
<td>awips</td>
<td>15</td>
<td>0</td>
<td>1601m</td>
<td>777m</td>
<td>773mS</td>
<td>0.7</td>
<td>6.4</td>
<td>0:48.09</td>
<td></td>
<td></td>
</tr>
<tr>
<td>418</td>
<td>root</td>
<td>10</td>
<td>-5</td>
<td>0</td>
<td>0</td>
<td>0S</td>
<td>0.3</td>
<td>0.0</td>
<td>72:02.59</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4345</td>
<td>awips</td>
<td>15</td>
<td>0</td>
<td>333m</td>
<td>7520</td>
<td>2836S</td>
<td>0.3</td>
<td>0.1</td>
<td>167:46.69</td>
<td>python</td>
<td></td>
</tr>
</tbody>
</table>
Appendix I  Administrative Tips for Monitoring System Performance

- Check the idle time. Values less than 10 percent for the duration of the command may indicate a resource problem. Contact the NCF and report your findings.
- A <CTRL>c will break out of the top command.

3. Check the status of message handling.

TYPE:  
dwb_smon

<table>
<thead>
<tr>
<th>Server</th>
<th>PID</th>
<th>User ID</th>
<th>Group ID</th>
<th>Length</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>msgreq_svr</td>
<td>14341</td>
<td>root</td>
<td>root</td>
<td>0</td>
<td>UP @ Tue May 29</td>
</tr>
<tr>
<td>(nfe/11030)</td>
<td>14343</td>
<td></td>
<td></td>
<td></td>
<td>UP @ Tue May 29</td>
</tr>
<tr>
<td>(worker)</td>
<td>14344</td>
<td></td>
<td></td>
<td></td>
<td>UP @ Tue May 29</td>
</tr>
<tr>
<td>(worker)</td>
<td>14452</td>
<td></td>
<td></td>
<td></td>
<td>UP @ Tue May 29</td>
</tr>
<tr>
<td>(worker)</td>
<td>14655</td>
<td></td>
<td></td>
<td></td>
<td>UP @ Tue May 29</td>
</tr>
<tr>
<td>(worker)</td>
<td>14896</td>
<td></td>
<td></td>
<td></td>
<td>UP @ Tue May 29</td>
</tr>
</tbody>
</table>

| msgrcv_svr   | 14432| root    | root     | 0      | UP @ Tue May 29 |
| (nfe/11040)  | 14433|         |          |        | UP @ Tue May 29 |
| (worker)     | 13810|         |          |        | UP @ Mon Jul  9 |
| (worker)     | 14450|         |          |        | UP @ Mon Jul  9 |

If there are msgqueues, notify the NCF of your findings. If nothing is returned, MHS is not running and needs to be started.

4. Ensure that the a2dx1 apps package is running.

TYPE:  
hb_stat

5. Ensure that the /awips2/data partition is mounted.

TYPE:  
df /awips2/data

Look for results similar to the following:

<table>
<thead>
<tr>
<th>Filesystem</th>
<th>1K-blocks</th>
<th>Used</th>
<th>Available</th>
<th>Use%</th>
<th>Mounted on</th>
</tr>
</thead>
<tbody>
<tr>
<td>/dev/mapper/vg_aiidb-awipsiidb</td>
<td>154818540</td>
<td>16722836</td>
<td>130231384</td>
<td>12%</td>
<td>/awips2/data</td>
</tr>
</tbody>
</table>
6. Ensure that the /awips/edex/data/hdf5 partition is mounted locally.

   **TYPE:**  
   df /awips2/edex/data/hdf5

   Look for results similar to the following:

   Filesystem           1K-blocks   Used     Available   Use%  Mounted on
   /dev/mapper/vg00-lvol_awips2 41284928 12287476 26900300 32%   /awips2

7. Ensure that postgres is running.

   **TYPE:**  
   ps -wef|grep postmaster

   awips 30249    0  0 Jul05 ? 00:00:58 /awips2/postgresql/bin/postmaster -D /awips2/data -p 5432

8. Look at the most recent log in /awips2/data/pg_log.

   **TYPE:**  
   ls -ltr /awips2/data/pg_log

   **TYPE:**  
   tail -f /awips2/data/pg_log/<LOGFILE>

9. Ensure that RCM (Radar Server) is running.

   **TYPE:**  
   ps -wef|grep rcm

   **TYPE:**  
   tail -f /awips2/rcm/data/logs/radarserver.log

10. Check for System Issues – Full partitions, load average, etc.

    **TYPE:**  
    df -hP

    **TYPE:**  
    df /data_store

11. Ensure that the directory is mounted from dx2f, as shown below, unless a2dx2apps is running on the same server as a2dx1apps.

    [root@dx1-tbw3 ~]# df -hP /data_store

      Filesystem       Size Used  Available  Use%  Mounted on
      nas2:/dataSTORE  2.0T   55G    2.0T    3%  /data_store

12. Ensure that the partition is mounted from the nas1:/GFESuite2 share.

    **TYPE:**  
    df /awips2/GFESuite

      Filesystem           1K-blocks Used     Available   Use%  Mounted on
      nas1:/GFESuite2 10485760 538432   9947328   6%   /awips2/GFESuite

13. Ensure that the partition is mounted from the nas1:/aiidata/utility share.

    **TYPE:**  
    df /awips2/edex/data/utility

      Filesystem           1K-blocks Used     Available   Use%  Mounted on
      nas1:/aiidata/utility 524288000 18008000 506280000  4%   /awips2/edex

      /data/utility

14. Ensure that MHS is running properly.
I.3 Check for Mandatory Processes on DX2F

Log in to the Linux data server running the dx2apps package (normally DX2).

1. Check what is mounted on the server. Verify that the NAS is mounted.
   TYPE:  
   TYPE: df
   o  Look for /data/fxa and the percentage used. Percentages larger than 70 percent could indicate a problem. Notify the NCF if this is the case.
   o  Also look for any directories that appear to be filling up (88 percent or higher).

2. Look for abnormal CPU usage. Greater than 75 percent resourced by an individual process may cause problems.
   TYPE:  
   TYPE: top
   o  Check the idle time. Values less than 10 percent for the duration of the command may indicate a resource problem. Contact the NCF and report your findings.
   o  A <CTRL> c will break out of the top command.

3. Ensure that a2dx2apps is running.
   TYPE:  
   TYPE: hb_stat

4. Ensure that /data_store is mounted locally from the NAS [NAS2].
   TYPE:  
   TYPE: df /data_store
   Filesystem 1K-blocks Used Available Use% Mounted on
   nas2:/dataSTORE 2147483648 57418624 2090065024 3% /data_store

5. Check for System Issues – Full partitions, load average, etc.
   TYPE:  
   TYPE: df -hP
   TYPE: top

6. Ensure that the partition is mounted from the nas1:/GFESuite2 share.
   TYPE:  
   TYPE: df /awips2/GFESuite
7. Ensure that the partition is mounted from the nas1:/aiidata/utility share.

**TYPE:** \texttt{df /awips2/edex/data/utility}

<table>
<thead>
<tr>
<th>Filesystem</th>
<th>1K-blocks Used</th>
<th>Available Use%</th>
<th>Mounted on</th>
</tr>
</thead>
<tbody>
<tr>
<td>nas1:/aiidata/utility</td>
<td>524288000</td>
<td>5316672</td>
<td>518971328 2% /awips2/edex</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>data/utility</td>
</tr>
</tbody>
</table>

8. Ensure that the /awips2/edex/data/hdf5 partition is mounted locally.

**TYPE:** \texttt{df /awips2/edex/data/hdf5}

Look for results similar to the following:

<table>
<thead>
<tr>
<th>Filesystem</th>
<th>1K-blocks Used</th>
<th>Available Use%</th>
<th>Mounted on</th>
</tr>
</thead>
<tbody>
<tr>
<td>/dev/mapper/vg_aiihdf5-awipsiihdf</td>
<td>411817200</td>
<td>103372372</td>
<td>287525740 27% /awips2/edex</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>data/hdf5</td>
</tr>
</tbody>
</table>

9. Ensure that Pypies is running.

**TYPE:** \texttt{ps -wef|grep pypies}

```
[root@dx2-tbdw ~]# ps -wef|grep pypies
awips 445 13238 2 15:31 ? 00:00:00
 /awips2/httpd_pypies/usr/sbin/httpd -f
 /awips2/httpd_pypies/etc/httpd/conf/httpd.conf
awips 446 13238 1 15:31 ? 00:00:00
 /awips2/httpd_pypies/usr/sbin/httpd -f
 /awips2/httpd_pypies/etc/httpd/conf/httpd.conf
awips 447 13238 2 15:31 ? 00:00:00
 /awips2/httpd_pypies/usr/sbin/httpd -f
 /awips2/httpd_pypies/etc/httpd/conf/httpd.conf
sridharb 651 31175 0 15:32 pts/0 00:00:00 grep pypies
root 13227 1 0 Mar14 ? 00:00:00 su awips -c
 /awips2/python/bin/python -u /awips2/python/lib/python2.7/site-packages/pypies/logging/logProcess.py > /tmp/pypiesLoggingService.log 2>&1
awips 13233 13227 0 Mar14 ? 00:00:00 bash -c
 /awips2/python/bin/python -u /awips2/python/lib/python2.7/site-packages/pypies/logging/logProcess.py > /tmp/pypiesLoggingService.log 2>&1
awips 13234 13233 0 Mar14 ? 00:13:25
 /awips2/python/bin/python -u /awips2/python/lib/python2.7/site-packages/pypies/logging/logProcess.py
root 13238 1 0 Mar14 ? 00:00:09
 /awips2/httpd_pypies/usr/sbin/httpd -f
 /awips2/httpd_pypies/etc/httpd/conf/httpd.conf
root 13240 13238 0 Mar14 ? 00:00:00
 /awips2/httpd_pypies/usr/sbin/rotatelogs
 /awips2/httpd_pypies/var/log/httpd/error_log.%Y.%m.%d 86400
```
Appendix I  Administrative Tips for Monitoring System Performance

root  13241 13238  0 Mar14 ?  00:04:01
/aws2/httpd_pypies/usr/sbin/rotatelogs
/aws2/httpd_pypies/var/log/httpd/access_log.%Y.%m.%d 86400

awips  19843 13238  1 14:53 ?  00:00:32
/aws2/httpd_pypies/usr/sbin/httpd -f
/aws2/httpd_pypies/etc/httpd/conf/httpd.conf

awips  20054 13238  2 14:54 ?  00:00:45
/aws2/httpd_pypies/usr/sbin/httpd -f
/aws2/httpd_pypies/etc/httpd/conf/httpd.conf

awips  20071 13238  1 14:54 ?  00:00:38
/aws2/httpd_pypies/usr/sbin/httpd -f
/aws2/httpd_pypies/etc/httpd/conf/httpd.conf

awips  20546 13238  1 14:56 ?  00:00:28
/aws2/httpd_pypies/usr/sbin/httpd -f
/aws2/httpd_pypies/etc/httpd/conf/httpd.conf

awips  20549 13238  1 14:56 ?  00:00:29
/aws2/httpd_pypies/usr/sbin/httpd -f
/aws2/httpd_pypies/etc/httpd/conf/httpd.conf

awips  20550 13238  0 14:56 ?  00:00:20
/aws2/httpd_pypies/usr/sbin/httpd -f
/aws2/httpd_pypies/etc/httpd/conf/httpd.conf

awips  20552 13238  1 14:56 ?  00:00:27
/aws2/httpd_pypies/usr/sbin/httpd -f
/aws2/httpd_pypies/etc/httpd/conf/httpd.conf

awips  20553 13238  0 14:56 ?  00:00:20
/aws2/httpd_pypies/usr/sbin/httpd -f
/aws2/httpd_pypies/etc/httpd/conf/httpd.conf

awips  20918 13238  1 14:58 ?  00:00:24
/aws2/httpd_pypies/usr/sbin/httpd -f
/aws2/httpd_pypies/etc/httpd/conf/httpd.conf

awips  22103 13238  1 15:00 ?  00:00:21
/aws2/httpd_pypies/usr/sbin/httpd -f
/aws2/httpd_pypies/etc/httpd/conf/httpd.conf

awips  22708 13238  0 15:01 ?  00:00:16
/aws2/httpd_pypies/usr/sbin/httpd -f
/aws2/httpd_pypies/etc/httpd/conf/httpd.conf

awips  23536 13238  1 15:05 ?  00:00:17
/aws2/httpd_pypies/usr/sbin/httpd -f
/aws2/httpd_pypies/etc/httpd/conf/httpd.conf

awips  23537 13238  1 15:05 ?  00:00:21
/aws2/httpd_pypies/usr/sbin/httpd -f
/aws2/httpd_pypies/etc/httpd/conf/httpd.conf

awips  27259 13238  1 15:18 ?  00:00:12
/aws2/httpd_pypies/usr/sbin/httpd -f
/aws2/httpd_pypies/etc/httpd/conf/httpd.conf

awips  27261 13238  0 15:18 ?  00:00:07
/aws2/httpd_pypies/usr/sbin/httpd -f
/aws2/httpd_pypies/etc/httpd/conf/httpd.conf

awips  27262 13238  1 15:18 ?  00:00:09
/aws2/httpd_pypies/usr/sbin/httpd -f
/aws2/httpd_pypies/etc/httpd/conf/httpd.conf

– Hard copies uncontrolled. Verify effective date prior to use. –
I.4 Check for Mandatory Processes on DX3/DX4

Log in to the Linux data server running the dx3apps package (normally DX3).

1. Check what is mounted on the server. Verify that the NAS is mounted.

   TYPE: \texttt{df -H}

   - Look for /\texttt{data/fxa} and the percentage used. Percentages larger than 70 percent could indicate a problem. Notify the NCF if this is the case.
   - Also look for any directories that appear to be filling up (88 percent or higher).

2. Please make sure that the EDEX CAMEL is up and running.

   TYPE: \texttt{service edex_camel status}

   EDEX Camel (ingest) is running (wrapper PID 31622)
   EDEX Camel (ingest) is running (java PID 31624)
   EDEX Camel (ingestGrib) is running (wrapper PID 31732)
   EDEX Camel (ingestGrib) is running (java PID 31734)
   EDEX Camel (ingestDat) is running (wrapper PID 31835)
   EDEX Camel (ingestDat) is running (java PID 31839)
   EDEX Camel (request) is running (wrapper PID 31960)
   EDEX Camel (request) is running (java PID 31962)

   TYPE: \texttt{top}

   - Check the idle time. Values less than 10 percent for the duration of the command may indicate a resource problem. Contact the NCF and report your findings.
   - A <CTRL> c will break out of the top command.

3. Ensure that there are four EDEX processes running, one each of the following: ingest, ingestDat, ingestGrib, and request:
4. Ensure that the ingest logs are all updating in /awips2/edex/logs and have low processing and latency times.

```bash
[root@dx3-tbw3 ~]# ps -fu awips
```

```
<table>
<thead>
<tr>
<th>UID</th>
<th>PID</th>
<th>PPID</th>
<th>C</th>
<th>STIME</th>
<th>TTY</th>
<th>TIME CMD</th>
</tr>
</thead>
<tbody>
<tr>
<td>awips</td>
<td>3909</td>
<td>0</td>
<td>Dec20</td>
<td></td>
<td>/bin/ksh</td>
<td></td>
</tr>
<tr>
<td></td>
<td>/awips2/edex/data/share</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>awips</td>
<td>3947</td>
<td>3909</td>
<td>99</td>
<td>Dec20</td>
<td>9-22:26:17</td>
<td>runso</td>
</tr>
<tr>
<td></td>
<td>rary.ohd.proc hpe_field</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>awips</td>
<td>18828</td>
<td>18826</td>
<td>0</td>
<td>Nov22</td>
<td>00:00:23</td>
<td>sshd: awips@notty</td>
</tr>
<tr>
<td></td>
<td>/awips2/edex/bin/start</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>awips</td>
<td>28279</td>
<td>28257</td>
<td>0</td>
<td>Dec23</td>
<td>00:03:23</td>
<td>java -Xmx32m -</td>
</tr>
<tr>
<td></td>
<td>/awips2/edex/data/share</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>awips</td>
<td>28279</td>
<td>28257</td>
<td>0</td>
<td>Dec23</td>
<td>04:23:33</td>
<td>/awips2/java/bin/java -Dedex.run</td>
</tr>
<tr>
<td></td>
<td>/awips2/edex/bin/start</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>awips</td>
<td>28343</td>
<td>1</td>
<td>Dec23</td>
<td></td>
<td>/bin/bash</td>
<td></td>
</tr>
<tr>
<td></td>
<td>/awips2/edex/bin/start</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>awips</td>
<td>28404</td>
<td>28343</td>
<td>0</td>
<td>Dec23</td>
<td>00:04:30</td>
<td>java -Xmx32m -</td>
</tr>
<tr>
<td></td>
<td>/awips2/edex/bin/start</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>awips</td>
<td>28490</td>
<td>1</td>
<td>Dec23</td>
<td></td>
<td>/bin/bash</td>
<td></td>
</tr>
<tr>
<td></td>
<td>/awips2/edex/bin/start</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>awips</td>
<td>28551</td>
<td>28490</td>
<td>0</td>
<td>Dec23</td>
<td>00:03:41</td>
<td>java -Xmx32m -</td>
</tr>
<tr>
<td></td>
<td>/awips2/edex/bin/start</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>awips</td>
<td>29965</td>
<td>28404</td>
<td>34</td>
<td>Dec28</td>
<td>19:24:13</td>
<td>/awips2/java/bin/java -Dedex.run</td>
</tr>
</tbody>
</table>
```

```
TYPE:         tail -f edex-ingest-$($(date +%Y%m%d)).log
TYPE:         tail -f edex-ingestGrib-$($(date +%Y%m%d)).log
TYPE:         tail -f edex-ingestDat-$($(date +%Y%m%d)).log
TYPE:         tail -f edex-request-$($(date +%Y%m%d)).log
TYPE:         tail -f edex-ingest-satellite-$($(date +%Y%m%d)).log
TYPE:         tail -f edex-ingest-radar-$($(date +%Y%m%d)).log
TYPE:         tail -f edex-ingest-shef-$($(date +%Y%m%d)).log
TYPE:         tail -f edex-ingest-text-$($(date +%Y%m%d)).log
TYPE:         tail -f edex-ingest-purge-$($(date +%Y%m%d)).log
TYPE:         tail -f edex-ingest-trigger-$($(date +%Y%m%d)).log
```
Appendix I Administrative Tips for Monitoring System Performance

5. Check for System Issues – Full partitions, load average, etc.

   TYPE:  
   df  -H

<table>
<thead>
<tr>
<th>Filesystem</th>
<th>Size</th>
<th>Used</th>
<th>Avail</th>
<th>Use%</th>
<th>Mounted on</th>
</tr>
</thead>
<tbody>
<tr>
<td>/dev/mapper/vg01-lvol101</td>
<td>1.2G</td>
<td>529M</td>
<td>546M</td>
<td>50%</td>
<td>/</td>
</tr>
<tr>
<td>tmpfs</td>
<td>4.2G</td>
<td>0</td>
<td>4.2G</td>
<td>0%</td>
<td>/dev/shm</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_fxa</td>
<td>5.5G</td>
<td>1.3G</td>
<td>4.0G</td>
<td>24%</td>
<td>/awips/fxa</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_ifps</td>
<td>1.4G</td>
<td>82M</td>
<td>1.2G</td>
<td>7%</td>
<td>/awips/ifps</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_lad</td>
<td>1.1G</td>
<td>144M</td>
<td>837M</td>
<td>15%</td>
<td>/awips/ladad</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_ops</td>
<td>285M</td>
<td>136M</td>
<td>135M</td>
<td>51%</td>
<td>/awips/ops</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_awips2</td>
<td>41G</td>
<td>13G</td>
<td>26G</td>
<td>33%</td>
<td>/awips2</td>
</tr>
<tr>
<td>/dev/sda3</td>
<td>102M</td>
<td>65M</td>
<td>32M</td>
<td>68%</td>
<td>/boot</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_logs</td>
<td>2.2G</td>
<td>71M</td>
<td>2.0G</td>
<td>4%</td>
<td>/data/logs</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_tmp</td>
<td>1.1G</td>
<td>19M</td>
<td>961M</td>
<td>2%</td>
<td>/tmp</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_usr</td>
<td>11G</td>
<td>7.6G</td>
<td>2.3G</td>
<td>78%</td>
<td>/usr</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_var</td>
<td>3.1G</td>
<td>331M</td>
<td>2.7G</td>
<td>12%</td>
<td>/var</td>
</tr>
<tr>
<td>nas1:/awips/ADAPT</td>
<td>1.1G</td>
<td>701M</td>
<td>373M</td>
<td>66%</td>
<td>/awips/adapter</td>
</tr>
<tr>
<td>nas1:/awips/DEV</td>
<td>4.3G</td>
<td>132k</td>
<td>4.3G</td>
<td>1%</td>
<td>/awips/dev</td>
</tr>
</tbody>
</table>
Appendix I  Administrative Tips for Monitoring System Performance

**TYPE:**

```
df /data_store
```

<table>
<thead>
<tr>
<th>Filesystem</th>
<th>1K-blocks</th>
<th>Used</th>
<th>Available</th>
<th>Use%</th>
<th>Mounted on</th>
</tr>
</thead>
<tbody>
<tr>
<td>/data_STORE</td>
<td>524288000</td>
<td>49797568</td>
<td>474490432</td>
<td>10%</td>
<td>/data_store</td>
</tr>
</tbody>
</table>

6. Ensure that the directory is mounted from nas1:/data_store

```
[root@dx3-tbw3 ~]# df -hP /data_store
```

<table>
<thead>
<tr>
<th>Filesystem</th>
<th>Size</th>
<th>Used</th>
<th>Available</th>
<th>Use%</th>
<th>Mounted on</th>
</tr>
</thead>
<tbody>
<tr>
<td>/data_STORE</td>
<td>2.0T</td>
<td>56G</td>
<td>2.0T</td>
<td>3%</td>
<td>/data_store</td>
</tr>
</tbody>
</table>

7. Ensure that the partition is mounted from the nas1:/GFESuite2 share.

**TYPE:**

```
df /awips2/GFESuite
```

8. Ensure that the partition is mounted from the nas1:/aiidata share.

**TYPE:**

```
df /awips2/edex/data
```

9. Ensure that the clustering loopback device is up (ifconfig lo:0).

**TYPE:**

```
ifconfig lo:0
```

```
lo:0   Link encap:Local Loopback
       inet addr:165.92.24.61  Mask:255.255.255.255
       UP LOOPBACK RUNNING  MTU:16436  Metric:1
```

10. Ensure that awips2ARCHIVE is mounted on nas2.

Repeat steps 1 – 10 for DX4.

_I.5 Check for Abnormal Processes on CP1F_

1. Ensure that a2cp1apps is running.
Appendix I

Administrative Tips for Monitoring System Performance

2. Ensure that QPID is running and processing.

   TYPE: \texttt{ps -wef\|grep qpid}
   TYPE: \texttt{qpid-stat -q -S msg -L10}

   Make sure that queues are not high or lingering (the numbers in the msg column drop if they are high).

<table>
<thead>
<tr>
<th>queue</th>
<th>excl</th>
<th>msg</th>
<th>msgIn</th>
<th>msgOut</th>
<th>bytes</th>
<th>bytesIn</th>
<th>bytesOut</th>
<th>cons</th>
<th>bind</th>
</tr>
</thead>
<tbody>
<tr>
<td>_edex.alert-edex_com@amq.topic_6e404bec-1528-4423-b661-b2c2dabe9eb3</td>
<td>51</td>
<td>881k</td>
<td>881k</td>
<td>126k</td>
<td>6.17g</td>
<td>6.17g</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td><a href="mailto:_edex.alert-edex_com@amq.topic_2eaf956-34fb-4d3d-8949-0ac6ce3f530e">_edex.alert-edex_com@amq.topic_2eaf956-34fb-4d3d-8949-0ac6ce3f530e</a></td>
<td>51</td>
<td>2.04m</td>
<td>2.04m</td>
<td>126k</td>
<td>15.1g</td>
<td>15.1g</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Ingest.lsr</td>
<td>0</td>
<td>10.3k</td>
<td>10.3k</td>
<td>0</td>
<td>679k</td>
<td>679k</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>_edex.alerts@amq.topic_623a5687-6f8c-4590-833c-3bfb75190a06</td>
<td>0</td>
<td>219k</td>
<td>219k</td>
<td>0</td>
<td>1.52g</td>
<td>1.52g</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td><a href="mailto:_pluginPurged@amq.topic_36bf2e7f-29dd-4e2d-954b-1c3e986916ee">_pluginPurged@amq.topic_36bf2e7f-29dd-4e2d-954b-1c3e986916ee</a></td>
<td>0</td>
<td>8.50k</td>
<td>8.50k</td>
<td>0</td>
<td>56.4k</td>
<td>56.4k</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Ingest.bufrssmi</td>
<td>0</td>
<td>28.0k</td>
<td>28.0k</td>
<td>0</td>
<td>2.13m</td>
<td>2.13m</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Ingest.nucaps</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Ingest.regionalsat</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>93</td>
<td>93</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>_rebuildD2DCache@amq.topic_d216b37f-1894-453c-b200-73b3c6112f46</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>mpeLightSrvScheduledWork</td>
<td>0</td>
<td>1.73k</td>
<td>1.73k</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

3. Ensure that Pulse/IPVS is running and has edex-request JVMs connected.

   TYPE: \texttt{ps -wef\|grep pulse}

   root 18244 1 0 Feb21 ? 00:00:00 pulse

   TYPE: \texttt{ipvsadm --list}

   IP Virtual Server version 1.2.1 (size=4096)
   Prot LocalAddress:Port Scheduler Flags
   -> RemoteAddress:Port Forward Weight ActiveConn InActConn
   TCP edexcluster-tbw3:9581 lc
   -> dx4-tbw3:9581 Route 2 1 91
   -> dx3-tbw3:9581 Route 1 3 0
4. Ensure that /awips2/qpid/messageStore is mounted from the nas1:qpid share.
   TYPE:  
   df /awips2/qpid/messageStore

   [root@cpsbn1-tbw3 ~]# df /awips2/qpid/messageStore
   Filesystem 1K-blocks Used Avail Use% Mounted on
   nas1:/qpid 10485760 26688 10459072 1% /awips2/qpid/messageStore

5. Ensure that data is coming in, and that there are no ERROR messages.
   TYPE:  
   tail -f ~ldm/logs/ldmd.log

I.6 Check for Abnormal Processes on CPSBN1/CPSBN2

1. Ensure that LDM is running.
   TYPE:  
   ps -fu ldm

   You should see one ldmd processes, a pqact process and an edexBridge process.

   UID   PID  PPID  C  STIME  TTY          TIME  CMD
   ldm   13780 13779  0 01:28 pts/0  00:00:00 -bash
   ldm   14343 13780  0 01:31 pts/0  00:00:00 ps -fu ldm
   ldm   31149 1  0 Jul19 ?  00:00:00 ldmd -I 0.0.0.0 -P 388 -M 256 -m 3600 -o 3600 -q /usr/local/ldm/data/l
   ldm   31150 31149  0 Jul19 ?  00:47:14 pqact -e
   ldm   31152 31149  0 Jul19 ?  00:10:09 edexBridge -vxl /usr/local/ldm/logs/edexBridge.log -s cp1f

2. Ensure that five noaaportIngest processes are running.
   TYPE:  
   ps -wef|grep noaaportIngest

   root  31153 31149 2 Jul19 ?  02:31:01 noaaportIngestester -m 224.0.1.1 -n -u 3 -t mhs -r 1 -s NMC
   root  31154 31149 0 Jul19 ?  00:08:17 noaaportIngestester -m 224.0.1.2 -n -u 4 -t mhs -r 1 -s GOES
   root  31155 31149 1 Jul19 ?  01:13:30 noaaportIngestester -m 224.0.1.3 -n -u 5 -t mhs -r 1 -s NMC2
   root  31156 31149 0 Jul19 ?  00:00:02 noaaportIngestester -m 224.0.1.4 -n -u 6 -t mhs -r 1 -s NOAAPORT_OPT
   root  31157 31149 0 Jul19 ?  00:05:47 noaaportIngestester -m 224.0.1.5 -n -u 7 -t mhs -r 1 -s NMC3

3. Check for System Issues – Full partitions, load average, etc.
   TYPE:  
   top
I.7 Check for Abnormal Processes on AX

Log in to the AX server.

1. Check what is mounted on the AX. Look for any directories that appear to be filling up (88 percent or higher). Verify that the /home, /data/fxa, and /data/local directories are mounted from the NAS.

   TYPE: `df -H`

2. Look for abnormal CPU usage. Any individual process using greater than 75 percent of the CPU may cause system problems.

   TYPE: `top`
   
   o Check the idle time. Values less than 10 percent for the duration of the command may indicate a resource problem. Contact the NCF and report your findings.
   
   o A `<CTRL> c` will break out of the `top` command.

I.8 Check for Mandatory Processes on PX1

Log in to the server running the `px1apps` package (normally PX1).

1. Check what is mounted on PX1. Verify that the NAS is mounted. Look for any directories that appear to be filling up (88 percent or higher).

   TYPE: `df -H`

2. `ps -wef |grep fxa`

   Verify that the `fxa` processes are running:

3. Look for abnormal CPU usage. Any individual process using greater than 75 percent of the CPU may cause system problems.

   TYPE: `top`
   
   o Check the idle time. Values less than 10 percent for the duration of the command may indicate a resource problem. Contact the NCF and report your findings.
   
   o A `<CTRL> c` will break out of the `top` command.

4. Ensure that `a2px1apps` is running.

   TYPE: `hb_stat`

5. Ensure that re-hosted applications are running.

   TYPE: `ps -fu fxa`
The following can be seen running on the server: The asyncScheduler and NWWS might vary in use from site to site.

- asyncPilMsgServer
- asyncScheduler
- asyncPollData.pl
- hmMonitorServer
- NWWSSchedule

**Note:** The awips2_radarIngest.pl, laps_driver.pl, and remap_polar_netcdf.exe are not constant processes. They may or may not show. The user must verify that the log files are updating and that LAPS data is current in the system.

### 6. Check for System Issues – Full partitions, load average, etc.

**TYPE:**

- `df -H`
- `top`
- `df /data_store`

### 7. Ensure that the directory is mounted from nas2:/dataSTORE

```
[root@dx1-hgx ~]# df -hP /data_store

Filesystem               Size  Used  Available  Use% Mounted on
nas2:/dataSTORE       2.0T   56G   2.0T     3%  /data_store
```

### 9. Ensure that the directory is mounted from the nas1:aiidata/utility as shown below:

**TYPE:**

- `df /awips2/edex/data/manual`

```
Filesystem   1K-blocks  Used   Available  Use% Mounted on
nas1:/aiidata   524288000   16934272  507353728   4%    /awips2/edex/data
```

### I.9 Check for Mandatory Processes on PX2

Log in to the server running the px2apps package (normally PX2).

1. Check what is mounted on PX1. Verify that the NAS is mounted. Look for any directories that appear to be filling up (88 percent or higher).

**TYPE:**

- `df -H`

2. **TYPE:**

- `ps -wef |grep fxa`

Verify that the fxa processes listed below are running.

```
root      2405  2280  0 15:07 pts/0  00:00:00 grep fxa
fxa      15304     1  0  2013 ?        00:00:00 /awips/fxa/bin/ldadServer
```
Appendix I Administrative Tips for Monitoring System Performance

1. Look for abnormal CPU usage. Any individual process using greater than 75 percent of the CPU may cause system problems.

   TYPE: top

   o Check the idle time. Values less than 10 percent for the duration of the command may indicate a resource problem. Contact the NCF and report your findings.
   o A <CTRL> c will break out of the top command.

2. Ensure that a2px2apps is running.

   TYPE: hb_stat

3. Ensure that rehosted applications are running.

   TYPE: ps -fu ldad

   The following should be running on the server:

<table>
<thead>
<tr>
<th>UID</th>
<th>PID</th>
<th>PPID</th>
<th>C</th>
<th>STIME</th>
<th>TTY</th>
<th>TIME</th>
<th>CMD</th>
</tr>
</thead>
<tbody>
<tr>
<td>ldad</td>
<td>2952</td>
<td>14831</td>
<td>0</td>
<td>00:00</td>
<td>?</td>
<td>00:00:01</td>
<td>/usr/bin/perl /awips/ldad/bin/pollForData.pl Start /</td>
</tr>
<tr>
<td>ldad</td>
<td>14374</td>
<td>1</td>
<td>0</td>
<td>Feb05</td>
<td>?</td>
<td>00:00:06</td>
<td>/awips/ldad/bin/listener</td>
</tr>
</tbody>
</table>
6. **Enter:** `ps -fu fxa`

Look for the following processes running:

- ldadServer
- fsiWatchdog.sh
- FSIprocessorEDEX
- rssd
7. Check for System Issues – Full partitions, load average, etc.

   TYPE: df -hP
   TYPE: top
   TYPE: df /data_store

8. Ensure that the directory is mounted from nas2:/dataSTORE

   TYPE: df -hP /data_store
   Filesystem      Size   Used  Available  Use%  Mounted on
   nas2:/dataSTORE 2.0T    56G   2.0T     3%  /data_store

9. Ensure that the directory is mounted from the nas1:aiidata/utility as shown below:

   TYPE: df /awips2/edex/data/manual
   Filesystem 1K-blocks Used Available Use% Mounted on
   nas1:/aiidata 524288000 16936448 507351552 4% /awips2/edex/data

10. Ensure that the directory is mounted from the nas1:aiidata/share as shown below.

    TYPE: df /awips2/edex/data/share
    Filesystem 1K-blocks Used Available Use% Mounted on
    nas1:/aiidata 524288000 16936448 507351552 4% /awips2/edex/data

1.0  Check for Mandatory Processes on REP Servers (RFCs only)

    Log in to rp1-<siteID> as user root.

   1. Check what is mounted on RP1. Look for any directories that appear to be filling up
      (88 percent or higher). Verify that the REP NAS is mounted.

      TYPE: df -H

   2. Look for abnormal CPU usage. Any individual process using greater than 75
      percent of the CPU may cause system problems.

      TYPE: top
      o Check the idle time. Values less than 10 percent for the duration of the
        command may indicate a resource problem. Contact the NCF and report your
        findings.
      o A <CTRL> c will break out of the top command.

   3. Repeat steps 1 and 2 on RP2.
I.11 Checklist for Servers and Workstations

I.11.1 Servers

DX1

1. Check to see whether postgres is running.

   TYPE: ps -ef | grep postgres

   awips 1364 15282 0 18:52 ? 00:00:00 postgres: awips
   metadata 165.92.24.3(53649) idle
   awips 1556 15282 0 18:52 ? 00:00:00 postgres: awips
   fxatext 165.92.24.4(50940) idle
   awips 1617 15282 3 18:52 ? 00:00:55 postgres: awips maps
   165.92.24.3(53667) idle
   awips 2958 15282 0 18:54 ? 00:00:12 postgres: awips
   fxatext 165.92.24.3(53748) idle
   awips 4407 15282 0 18:14 ? 00:00:05 postgres: awips
   metadata 165.92.24.3(44624) idle
   awips 4409 15282 0 18:14 ? 00:00:04 postgres: awips
   metadata 165.92.24.3(44626) idle
   awips 4410 15282 0 18:14 ? 00:00:05 postgres: awips
   metadata 165.92.24.3(44627) idle
   awips 7601 15282 0 18:59 ? 00:00:00 postgres: awips
   metadata 165.92.24.3(39837) idle
   awips 15404 15282 0 Jan21 ? 00:00:01 postgres: logger
   process
   awips 15455 15282 0 Jan21 ? 00:00:01 postgres: writer
   process
   awips 15456 15282 0 Jan21 ? 00:00:00 postgres: wal writer
   process
   awips 15457 15282 0 Jan21 ? 00:00:00 postgres: autovacuum
   launcher process
   awips 15458 15282 0 Jan21 ? 00:00:42 postgres: stats
   collector process
   awips 15952 15282 0 19:09 ? 00:00:00 postgres: awips
   hd_ob92oax 165.92.24.3(57044) idle
   awips 15953 15282 0 19:09 ? 00:00:00 postgres: awips
   hd_ob92oax 165.92.24.3(57045) idle
   awips 16061 15282 0 18:29 ? 00:00:01 postgres: awips
   metadata 165.92.24.3(43691) idle

Note: Check the logs at /awips2/data/pg_log.
Note: Check the logs at /awips2/rcm/data/logs.
2. Check to see whether any partitions are full.

   **TYPE:**   \texttt{df \ -H}

<table>
<thead>
<tr>
<th>Filesystem</th>
<th>Size</th>
<th>Used</th>
<th>Avail</th>
<th>Use%</th>
<th>Mounted on</th>
</tr>
</thead>
<tbody>
<tr>
<td>/dev/mapper/vg01-lvol01</td>
<td>1.1G</td>
<td>447M</td>
<td>534M</td>
<td>46%</td>
<td>/</td>
</tr>
<tr>
<td>tmpfs</td>
<td>6.3G</td>
<td>0</td>
<td>6.3G</td>
<td>0%</td>
<td>/dev/shm</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_fxa</td>
<td>5.2G</td>
<td>1.9G</td>
<td>3.1G</td>
<td>38%</td>
<td>/awips/fxa</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_ifps</td>
<td>1.4G</td>
<td>82M</td>
<td>1.2G</td>
<td>7%</td>
<td>/awips/ifps</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_ldad</td>
<td>1.1G</td>
<td>84M</td>
<td>897M</td>
<td>9%</td>
<td>/awips/ldad</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_ops</td>
<td>305M</td>
<td>200M</td>
<td>90M</td>
<td>69%</td>
<td>/awips/ops</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_awips2</td>
<td>34G</td>
<td>2.4G</td>
<td>30G</td>
<td>8%</td>
<td>/awips2</td>
</tr>
<tr>
<td>/dev/mapper/mpathap3</td>
<td>102M</td>
<td>67M</td>
<td>35M</td>
<td>70%</td>
<td>/boot</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_logs</td>
<td>11G</td>
<td>159M</td>
<td>9.9G</td>
<td>2%</td>
<td>/data/logs</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_tmp</td>
<td>1.1G</td>
<td>21M</td>
<td>960M</td>
<td>3%</td>
<td>/tmp</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_ifps</td>
<td>1.4G</td>
<td>82M</td>
<td>1.2G</td>
<td>7%</td>
<td>/awips/ifps</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_ldad</td>
<td>1.1G</td>
<td>84M</td>
<td>897M</td>
<td>9%</td>
<td>/awips/ldad</td>
</tr>
</tbody>
</table>

**Note:** Check to see if everything is mounted (/awips2, /awips2/db, /awips2/GFESuite).

3. Check to see whether the Heartbeat, a2dx1apps, and dx1f up (ping dx1f) are running.

4. Check to see whether MHS is running.

   **TYPE:**   \texttt{ps \ -ef \ | \ grep mhs}

```
root       627 31786  0   16:07 pts/0    00:00:00 grep mhs
root      13644  1  0   Nov11 ?    00:00:00 archive_msgreq -p/data/mhs/archive/req_pipe -o/data/mhs/archive/req/processed
root      13647  1  0   Nov11 ?    00:00:00 archive_msgresp -p/data/mhs/archive/resp_pipe -o/data/mhs/archive/resp/processed
root      32211  1  0   Nov21 ?    00:00:01 mdp_daemon -k 7789 -f /awips/ops/data/mhs/mdp-smtp.cfg
awipsmhs 32213 32211  0   Nov21 ?    00:01:08 smtp_send -eb -cz -k7789 -n0 -p/tmp/smtp0_send_fifo
```
Appendix I  Administrative Tips for Monitoring System Performance

awipsmhs 32365  1  0 Nov21 ? 00:00:18 smtp_recv -n0 -m0 -o/data/mhs/smtp_recv/output -b/var/spool/mail/awipsmhs

**TYPE:**  ypcat hosts | grep mhs

165.92.33.65  dxlf-box cpCfgHost mhs  siteTime mhs  dxlf
165.92.33.65  dxlf-box cpCfgHost mhs  siteTime mhs  dxlf
165.92.33.65  dxlf-box cpCfgHost mhs  siteTime mhs  dxlf
165.92.33.65  dxlf-box cpCfgHost mhs  siteTime mhs  dxlf
165.92.33.65  dxlf-box cpCfgHost mhs  siteTime mhs  dxlf

5. Check to see whether sendmail is running.

**TYPE:**  ps -ef | grep sendmail

fxa       7935  6688  0 13:50 ? 00:00:00 /usr/sbin/sendmail -F CronDaemon -i -odi -oem -oi -t
sridharb 22665 14878  0 14:00 pts/15 00:00:00 grep sendmail
root     27298     1  0 Mar28 ? 00:06:37 sendmail: accepting connections
smmsp    27306     1  0 Mar28 ? 00:00:03 sendmail: Queue runner@00:00:10 for /var/spool/clientmqueue

**TYPE:**  msg_send –atbdr –c0

TBW3-75458

**DX2**

Check to see whether any partitions are full.

**TYPE:**  df -H

```
Filesystem Size   Used Avail Use% Mounted on
/dev/mapper/vg01-lvol01  1.1G  420M   561M  43% /dev/shm
/tmpfs                   6.3G  0  6.3G   0% /dev/shm
/dev/mapper/vg01-lvol_fxa  5.2G  1.5G   3.5G  30% /awips/fxa
/dev/mapper/vg01-lvol_ifps  1.4G  82M   1.2G   7% /awips/ifps
/dev/mapper/vg01-lvol_1dad  1.1G  85M   896M   9% /awips/1dad
/dev/mapper/vg01-lvol_ops  305M  122M  183M  42% /awips/ops
/dev/mapper/vg01-lvol_awips2  34G  2.3G  30G   8% /awips2
/dev/mapper/mpathap3  102M  67M   35M  70% /boot
/dev/mapper/vg01-lvol_logs  11G  166M  9.9G   2% /data/logs
/dev/mapper/vg01-lvol_tmp  1.1G  19M  962M   2% /tmp
/dev/mapper/vg01-lvol_usr  11G  8.5G  10G  87% /usr
/dev/mapper/vg01-lvol_var  3.1G  356M  2.6G  13% /var
nas1:/awipsADAPT  1.1G  701M  373M  66% /awips/adapt
nas1:/awipsDEV  4.3G  132k  4.2G   1% /awips/dev
nas1:/awipsGFESuit  108G  18G  89G  17% /awips
                  /GFESuite
nas1:/awipsHYDRO  108G  807M  107G   1% /awips
                   /hydroapps
nas1:/dataLOCAL  162G  46G  116G  29% /data/local
nas1:/dataX400  630M  228M  402M  37% /data/x400
nas1:/DSshared  1.1G  413M  662M  39% /DS_shared
nas1:/dataADAPT  3.3G  0  3.3G   0% /data/adapt
nas1:/dataGFE  21M   0  21M   0% /data/GFE
nas1:/awipsHOME  119G  94G  25G  80% /home
```
Appendix I  Administrative Tips for Monitoring System Performance

nas1:/dataFXA  537G  388G  150G  73% /data/fxa
nas1:/aiidata/utility  537G  11G  526G  3% /awips2/edex
nas1:/GFESuite2  11G  144M  11G  2% /awips2
nas1:/localapps  54G  40M  11G  1% /localapps
nas2:/dataSTORE  2.2T  49G  2.2T  3% /data_store
nas1:/GFESuite  11G  144M  11G  2% /awips2/GFESuite
nas1:/localapps  54G  40M  11G  1% /localapps
nas2:/dataSTORE  2.2T  49G  2.2T  3% /data_store
nas1:/GFESuite2  11G  144M  11G  2% /awips2/GFESuite

Note: Check to see if everything is mounted (/awips2, /awips2/db, /awips2/GFESuite).

DX3/4
1. Check to see whether edex ingest, ingestGrib, and request are running.

   TYPE:  ps -ef | grep edex

   Note: Make sure they are logging and ingesting data in /awips2/edex/logs.

2. Check to see whether any partitions are full.

   TYPE:  df -H

<table>
<thead>
<tr>
<th>Filesystem</th>
<th>Size</th>
<th>Used</th>
<th>Avail</th>
<th>Use%</th>
<th>Mounted on</th>
</tr>
</thead>
<tbody>
<tr>
<td>/dev/mapper/vg01-lvol01</td>
<td>1.2G</td>
<td>529M</td>
<td>546M</td>
<td>50%</td>
<td>/</td>
</tr>
<tr>
<td>tmpfs</td>
<td>4.2G</td>
<td>0</td>
<td>4.2G</td>
<td>0%</td>
<td>/dev/shm</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_fxa</td>
<td>5.5G</td>
<td>1.3G</td>
<td>4.0G</td>
<td>24%</td>
<td>/awips/fxa</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_ifps</td>
<td>1.4G</td>
<td>82M</td>
<td>1.2G</td>
<td>7%</td>
<td>/awips/ifps</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_ldad</td>
<td>1.1G</td>
<td>144M</td>
<td>837M</td>
<td>15%</td>
<td>/awips/ldad</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_ops</td>
<td>285M</td>
<td>136M</td>
<td>133M</td>
<td>51%</td>
<td>/awips/ops</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_awipsv2</td>
<td>41G</td>
<td>13G</td>
<td>26G</td>
<td>33%</td>
<td>/awips2</td>
</tr>
<tr>
<td>/dev/sda3</td>
<td>102M</td>
<td>65M</td>
<td>37M</td>
<td>63%</td>
<td>/boot</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_logs</td>
<td>2.2G</td>
<td>71M</td>
<td>2.0G</td>
<td>4%</td>
<td>/data/logs</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_tmp</td>
<td>1.1G</td>
<td>19M</td>
<td>961M</td>
<td>2%</td>
<td>/tmp</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_usr</td>
<td>11G</td>
<td>7.6G</td>
<td>1.2G</td>
<td>78%</td>
<td>/usr</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_var</td>
<td>3.1G</td>
<td>331M</td>
<td>2.7G</td>
<td>12%</td>
<td>/var</td>
</tr>
<tr>
<td>nas1:/awipsADAPT</td>
<td>1.1G</td>
<td>701M</td>
<td>373M</td>
<td>66%</td>
<td>/awips/adapt</td>
</tr>
<tr>
<td>nas1:/awipsDEV</td>
<td>4.3G</td>
<td>132k</td>
<td>4.3G</td>
<td>1%</td>
<td>/awips/dev</td>
</tr>
<tr>
<td>nas1:/awipsGFESuit</td>
<td>108G</td>
<td>18G</td>
<td>90G</td>
<td>17%</td>
<td>/awips/GFESuite</td>
</tr>
<tr>
<td>nas1:/awipsHYDRO</td>
<td>108G</td>
<td>807M</td>
<td>107G</td>
<td>1%</td>
<td>/awips/hydroapps</td>
</tr>
<tr>
<td>nas1:/dataLOCAL</td>
<td>162G</td>
<td>46G</td>
<td>116G</td>
<td>29%</td>
<td>/data/local</td>
</tr>
<tr>
<td>nas1:/dataX400</td>
<td>630M</td>
<td>228M</td>
<td>402M</td>
<td>37%</td>
<td>/data/x400</td>
</tr>
<tr>
<td>nas1:/DSshared</td>
<td>1.1G</td>
<td>413M</td>
<td>662M</td>
<td>39%</td>
<td>/DS_shared</td>
</tr>
<tr>
<td>nas1:/dataADAPT</td>
<td>3.3G</td>
<td>0</td>
<td>3.3G</td>
<td>0%</td>
<td>/data/adapt</td>
</tr>
<tr>
<td>nas1:/dataGFE</td>
<td>21M</td>
<td>0</td>
<td>21M</td>
<td>0%</td>
<td>/data/GFE</td>
</tr>
<tr>
<td>nas1:/awipsHOME</td>
<td>119G</td>
<td>94G</td>
<td>25G</td>
<td>80%</td>
<td>/home</td>
</tr>
<tr>
<td>nas1:/dataFXA</td>
<td>537G</td>
<td>388G</td>
<td>150G</td>
<td>73%</td>
<td>/data/fxa</td>
</tr>
<tr>
<td>nas1:/aiidata</td>
<td>537G</td>
<td>11G</td>
<td>526G</td>
<td>3%</td>
<td>/awips/edex/data</td>
</tr>
<tr>
<td>nas1:/GFESuite2</td>
<td>11G</td>
<td>144M</td>
<td>11G</td>
<td>2%</td>
<td>/awips2/GFESuite</td>
</tr>
<tr>
<td>nas1:/localapps</td>
<td>54G</td>
<td>40M</td>
<td>14G</td>
<td>1%</td>
<td>/localapps</td>
</tr>
<tr>
<td>nas2:/dataSTORE</td>
<td>2.2T</td>
<td>50G</td>
<td>2.2T</td>
<td>3%</td>
<td>/data_store</td>
</tr>
<tr>
<td>nas1:/awips2ARCHIVE</td>
<td>3.3T</td>
<td>8.2G</td>
<td>3.3T</td>
<td>1%</td>
<td>/archive</td>
</tr>
</tbody>
</table>

Note: Check to see if everything is mounted (/awips2, /awips2/db, /awips2/GFESuite)
Appendix I  
Administrative Tips for Monitoring System Performance

3. Check to see whether the clustering loopback device is up.

   TYPE:  
   ifconfig lo:0

   lo:0   Link encap:Local Loopback
           inet addr:165.92.24.61  Mask:255.255.255.255
           UP LOOPBACK RUNNING  MTU:16436  Metric:1

CPSBN

1. Check to see whether ldm is running on cpsbn1 and cpsbn2.

   TYPE:  
   ps -wef | grep ldm

   ldm 14708 13780 0 01:36 pts/0 00:00:00 grep ldm
   ldm 31149 1 0 Jul19 ? 00:00:00 ldmd -I 0.0.0.0 -P
   388 -M 256 -m 3600 -o 3600 -q /usr/local/ldm/data/ldm.pq
   /usr/local/ldm/etc/ldmd.conf
   ldm 31150 31149 0 Jul19 ? 00:47:15 pqact -e
   ldm 31152 31149 0 Jul19 ? 00:10:09 edexBridge -vxl
   /usr/local/ldm/logs/edexBridge.log -s cplf

2. Check to see whether qpid is running on cpsbn1

   TYPE:  
   ps -ef | grep qpid

   root 10750 10636 0 19:52 pts/0 00:00:00 grep qpid
   root 32392 1 0 2013 ? 00:00:00 su awips -c /bin/bash
   /awips2/qpid/bin/qpid-wrapper
   awips 32394 32392 0 2013 ? 00:00:00 /bin/bash
   /awips2/qpid/bin/qpid-wrapper
   awips 32397 32394 0 2013 ? 00:01:40 /awips2/java/bin/java
   -Xmx32m -XX:MaxPermSize=12m -XX:ReservedCodeCacheSize=4m -jar
   /awips2/qpid/bin/yajsw/wrapper.jar -c /awips2/qpid/conf/wrapper.conf
   awips 32424 32397 9 2013 ? 3-15:53:27
   /awips2/java/bin/java -classpath
   /awips2/qpid/bin/yajsw/.wrapperApp.jar:/awips2/qpid/bin/yajsw/wrapper
   r.jar:/awips2/qpid/lib/opt/*:/awips2/qpid/lib/opt/qpid-deps/* -XX:+
   UseConcMarkSweepGC -XX:+CMSIncrementalMode -XX:NewSize=400m -XX:
   MaxNewSize=400m -XX:SurvivorRatio=4 -XX:+HeapDumpOnOutOfMemoryError
   -XX:HeapDumpPath=/data/fxa/qpid-server -DQPID_HOME=/awips2/qpid
   -DQPID_WORK=/awips2/qpid -Damqj.logging.level=info -DQPID
   HOME=/awips2/qpid -DQPID_WORK=/awips2/qpid -
   DAMqj.read_write_pool_size=32 -Dqpid.broker.exceptionHandler.
   continue=true -Xmx2048m -Dwrrapper.port=15003 -Dwrrapper.key=
   -7261080382435685649 -Dwrrapper.tmp.path=/tmp -Djna_tmpdir=/tmp
   -Dwrrapper.config=/awips2/qpid/conf/wrapper.conf
   org.rzo.yajsw.app.WrapperJVMMain

3. Check to see whether IPVS running.

   TYPE:  
   ps -ef|grep pulse
Appendix I  Administrative Tips for Monitoring System Performance

root    29652     1    0 Apr06 ?    00:00:00 pulse

TYPE:  ps -ef|grep ipvsadm

root    18398 18389 0 Feb21 ? 00:09:28 /usr/sbin/nanny -c -h
165.92.24.3 -p 9581 -s GET / HTTP/1.0\r\n\r\n -x HTTP -a 15 -I
/sbin/ipvsadm -t 6 -w 1 -V 165.92.24.61 -M g -U none --lvs

root    18399 18389 0 Feb21 ? 00:09:59 /usr/sbin/nanny -c -h
165.92.24.4 -p 9581 -s GET / HTTP/1.0\r\n\r\n -x HTTP -a 15 -I
/sbin/ipvsadm -t 6 -w 2 -V 165.92.24.61 -M g -U none --lvs

4. Check to see whether the partitions are full.

TYPE:  df -H

<table>
<thead>
<tr>
<th>Filesystem</th>
<th>Size</th>
<th>Used</th>
<th>Avail</th>
<th>Use%</th>
<th>Mounted on</th>
</tr>
</thead>
<tbody>
<tr>
<td>/dev/mapper/vg01-lvol01</td>
<td>1.1G</td>
<td>416M</td>
<td>565M</td>
<td>43%</td>
<td>/dev/shm</td>
</tr>
<tr>
<td>tmpfs</td>
<td>3.1G</td>
<td>0</td>
<td>3.1G</td>
<td>0%</td>
<td>/</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_awips</td>
<td>529M</td>
<td>140M</td>
<td>382M</td>
<td>28%</td>
<td>/awips</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_awips2</td>
<td>5.2G</td>
<td>1.3G</td>
<td>3.7G</td>
<td>26%</td>
<td>/awips2</td>
</tr>
<tr>
<td>/dev/sda3</td>
<td>204M</td>
<td>65M</td>
<td>139M</td>
<td>34%</td>
<td>/boot</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_home</td>
<td>260M</td>
<td>12M</td>
<td>248M</td>
<td>5%</td>
<td>/home</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_tmp</td>
<td>529M</td>
<td>18M</td>
<td>465M</td>
<td>4%</td>
<td>/tmp</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_usr</td>
<td>16G</td>
<td>6.2G</td>
<td>9.5G</td>
<td>37%</td>
<td>/usr</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_var</td>
<td>3.1G</td>
<td>323M</td>
<td>1.2G</td>
<td>11%</td>
<td>/var</td>
</tr>
<tr>
<td>/dev/mapper/vg00-lvol_data</td>
<td>87G</td>
<td>2.5G</td>
<td>80G</td>
<td>4%</td>
<td>/data</td>
</tr>
<tr>
<td>nas1:/dataFXA</td>
<td>537G</td>
<td>388G</td>
<td>150G</td>
<td>73%</td>
<td>/data FXA</td>
</tr>
<tr>
<td>nas1:/aiidata</td>
<td>537G</td>
<td>11G</td>
<td>526G</td>
<td>3%</td>
<td>/aiidata</td>
</tr>
<tr>
<td>nas2:/dataSTORE</td>
<td>2.2T</td>
<td>50G</td>
<td>2.2T</td>
<td>3%</td>
<td>/data STORE</td>
</tr>
<tr>
<td>nas2:/awips2ARCHIVE</td>
<td>3.3T</td>
<td>8.2G</td>
<td>3.3T</td>
<td>1%</td>
<td>/archive</td>
</tr>
<tr>
<td>nas1:/qpid</td>
<td>11G</td>
<td>27M</td>
<td>11G</td>
<td>1%</td>
<td>/awips2/qpid</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/messageStore</td>
</tr>
</tbody>
</table>

PX1

1. Check to see whether the partitions are full.

TYPE:  df -H

<table>
<thead>
<tr>
<th>Filesystem</th>
<th>Size</th>
<th>Used</th>
<th>Avail</th>
<th>Use%</th>
<th>Mounted on</th>
</tr>
</thead>
<tbody>
<tr>
<td>/dev/mapper/vg01-lvol01</td>
<td>1.1G</td>
<td>444M</td>
<td>537M</td>
<td>46%</td>
<td>/dev/shm</td>
</tr>
<tr>
<td>tmpfs</td>
<td>6.3G</td>
<td>0</td>
<td>6.3G</td>
<td>0%</td>
<td>/</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_fxa</td>
<td>5.2G</td>
<td>1.6G</td>
<td>3.4G</td>
<td>33%</td>
<td>/awips/fxa</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_ifps</td>
<td>1.6G</td>
<td>36M</td>
<td>1.2G</td>
<td>3%</td>
<td>/awips/ifps</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_laps</td>
<td>1.1G</td>
<td>144M</td>
<td>837M</td>
<td>15%</td>
<td>/awips/laps</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_ldad</td>
<td>1.1G</td>
<td>179M</td>
<td>802M</td>
<td>19%</td>
<td>/awips/ldad</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_ops</td>
<td>305M</td>
<td>85M</td>
<td>210M</td>
<td>30%</td>
<td>/awips/ops</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_awips2</td>
<td>11G</td>
<td>1.9G</td>
<td>8.2G</td>
<td>19%</td>
<td>/awips2</td>
</tr>
<tr>
<td>/dev/sda3</td>
<td>102M</td>
<td>65M</td>
<td>37M</td>
<td>67%</td>
<td>/boot</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_logs</td>
<td>2.1G</td>
<td>548M</td>
<td>1.5G</td>
<td>28%</td>
<td>/data/logs</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_tmp</td>
<td>508M</td>
<td>11M</td>
<td>497M</td>
<td>3%</td>
<td>/tmp</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_usr</td>
<td>16G</td>
<td>7.6G</td>
<td>7.2G</td>
<td>52%</td>
<td>/usr</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_var</td>
<td>3.1G</td>
<td>358M</td>
<td>2.6G</td>
<td>13%</td>
<td>/var</td>
</tr>
<tr>
<td>nas1:/awipsADAPT</td>
<td>1.1G</td>
<td>701M</td>
<td>373M</td>
<td>66%</td>
<td>/awips/adapt</td>
</tr>
<tr>
<td>nas1:/awipsDEV</td>
<td>4.3G</td>
<td>173k</td>
<td>4.3G</td>
<td>1%</td>
<td>/awips/dev</td>
</tr>
<tr>
<td>nas1:/awipsGFESuit/ws</td>
<td>108G</td>
<td>18G</td>
<td>90G</td>
<td>17%</td>
<td>/awips/GFESuite</td>
</tr>
<tr>
<td>nas1:/awipsHOME</td>
<td>119G</td>
<td>94G</td>
<td>25G</td>
<td>80%</td>
<td>/home</td>
</tr>
</tbody>
</table>
Appendix I  Administrative Tips for Monitoring System Performance

nas1:/awipsHYDRO  108G  807M  107G  1% /awips /hydroapps
nas1:/dataADAPT    3.3G  0  3.3G  0% /data/adapt
nas1:/dataFXA      537G  388G  150G  73% /data/fxa
nas1:/dataGFE      21M   0  21M  0% /data/GFE
nas1:/dataLOCAL    162G  46G  116G  29% /data/local
nas1:/verify       2.2T  3.1G  2.2T  1% /data/verify
nas1:/aiidata      537G  11G  526G  3% /awips2/edex/data
nas1:/localapps    54G  40M  54G  1% /localapps
nas1:/GFESuite2    11G  144M  11G  2% /awips2/GFESuite
nas2:/dataSTORE    2.2T  50G  2.2T  1% /data_store
nas2:/awips2ARCHIVE 3.3T  8.2G  3.3T  1% /archive

2. Check to see whether everything is mounted on /awips2, /data_store.

3. Check to see whether the heartbeat is running, if a2px1apps is running, and if px1f is up (ping px1f).

4. Check to see whether the Rehosted applications are running.

    TYPE:    ps -ef | grep fxa

    fxa  17841     1  0 Oct10 ?        00:17:56
    /awips/fxa/bin/purgeProcess -commit
    fxa  23433  8715  0 17:42 ?        00:00:00 crond
    fxa  23434 23433  0 17:42 ?        00:00:00 /bin/sh -c /bin/csh -c "/usr/bin/perl /awips/laps/etc/awips2_satIngest.pl /awips/laps/data/fxa/laps' > /dev/null 2>&1; /bin/csh -c "/usr/bin/perl
    /awips/laps/etc/laps_driver.pl lvd_sat_ingest.exe /awips/laps/data/fxa/laps' > /dev/null 2>&1
    fxa  23435 23434  1 17:42 ?        00:00:00 /bin/csh -c
    /usr/bin/perl /awips/laps/etc/awips2_satIngest.pl /awips/laps/data/fxa/laps
    fxa  23602 23435  0 17:42 ?        00:00:00 /usr/bin/perl
    /awips/laps/etc/awips2_satIngest.pl /awips/laps/data/fxa/laps
    fxa  23604 23602  1 17:42 ?        00:00:00 /bin/csh
    /awips/fxa/bin/convertSat.csh 20111029_1730
    fxa  24605 23604  0 17:42 ?        00:00:00 sleep 3
    root  24610 24116  0 17:42 pts/0    00:00:00 grep fxa
    fxa  31697     1  0 Oct13 ?        00:00:00
    /awips/fxa/bin/asyncPilMsgServer
    fxa  31700     1  0 Oct13 ?        00:12:11
    /awips/fxa/bin/asyncScheduler
    fxa  31930 31700  0 Oct13 ?        00:00:08 testl 23 20 1 test1
    /dev/ttyn1a 9600 8 NONE 1 NONE
    fxa  31954     1  0 Oct13 ?        00:00:02 /usr/bin/perl
    /awips/fxa/bin/asyncPollData.pl
    fxa  31962     1  0 Oct13 ?        00:00:00
    /awips/fxa/bin/hmMonitorServer
    fxa  32101     1  0 Oct13 ?        00:00:01
    /awips/fxa/bin/NWWSSchedule
    /awips/fxa/bin/ingProcMon.pl -c PX1
PX2

1. Check to see whether any partitions are full.

   TYPE:  
   
   df -H

<table>
<thead>
<tr>
<th>Filesystem</th>
<th>Size</th>
<th>Used</th>
<th>Avail</th>
<th>Use%</th>
<th>Mounted on</th>
</tr>
</thead>
<tbody>
<tr>
<td>/dev/mapper/vg01-lvol01</td>
<td>1.1G</td>
<td>444M</td>
<td>537M</td>
<td>46%</td>
<td>/</td>
</tr>
<tr>
<td>tmpfs</td>
<td>6.3G</td>
<td>0</td>
<td>6.3G</td>
<td>0%</td>
<td>/dev/shm</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_fxa</td>
<td>5.2G</td>
<td>2.0G</td>
<td>3.0G</td>
<td>41%</td>
<td>/awips/fxa</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_ifps</td>
<td>1.6G</td>
<td>36M</td>
<td>1.5G</td>
<td>3%</td>
<td>/awips/ifps</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_laps</td>
<td>1.1G</td>
<td>145M</td>
<td>836M</td>
<td>15%</td>
<td>/awips/laps</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_ldad</td>
<td>1.1G</td>
<td>251M</td>
<td>730M</td>
<td>26%</td>
<td>/awips/ldad</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_ops</td>
<td>305M</td>
<td>87M</td>
<td>203M</td>
<td>30%</td>
<td>/awips/ops</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_awips2</td>
<td>11G</td>
<td>1.9G</td>
<td>8.2G</td>
<td>19%</td>
<td>/awips2</td>
</tr>
<tr>
<td>/dev/sda3</td>
<td>102M</td>
<td>65M</td>
<td>33M</td>
<td>67%</td>
<td>/boot</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_logs</td>
<td>2.1G</td>
<td>38M</td>
<td>2.0G</td>
<td>2%</td>
<td>/data/logs</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_tmp</td>
<td>508M</td>
<td>12M</td>
<td>471M</td>
<td>3%</td>
<td>/tmp</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_usr</td>
<td>16G</td>
<td>7.6G</td>
<td>7.2G</td>
<td>52%</td>
<td>/usr</td>
</tr>
<tr>
<td>/dev/mapper/vg01-lvol_var</td>
<td>3.1G</td>
<td>359M</td>
<td>2.6G</td>
<td>13%</td>
<td>/var</td>
</tr>
<tr>
<td>nas1:/awipsADAPT</td>
<td>1.1G</td>
<td>701M</td>
<td>373M</td>
<td>66%</td>
<td>/awips/adapt</td>
</tr>
<tr>
<td>nas1:/awipsDEV</td>
<td>4.3G</td>
<td>173k</td>
<td>4.3G</td>
<td>1%</td>
<td>/awips/dev</td>
</tr>
<tr>
<td>nas1:/awipsGFESuit/ws</td>
<td>108G</td>
<td>18G</td>
<td>90G</td>
<td>17%</td>
<td>/awips/GFESuite</td>
</tr>
<tr>
<td>nas1:/awipsHOME</td>
<td>119G</td>
<td>94G</td>
<td>25G</td>
<td>80%</td>
<td>/home</td>
</tr>
<tr>
<td>nas1:/awipsHYDRO</td>
<td>108G</td>
<td>807M</td>
<td>107G</td>
<td>1%</td>
<td>/awips/hydroapps</td>
</tr>
<tr>
<td>nas1:/dataADAPT</td>
<td>3.3G</td>
<td>0</td>
<td>3.3G</td>
<td>0%</td>
<td>/data/adt</td>
</tr>
<tr>
<td>nas1:/dataFXA</td>
<td>537G</td>
<td>388G</td>
<td>150G</td>
<td>73%</td>
<td>/data/fxa</td>
</tr>
<tr>
<td>nas1:/dataGFE</td>
<td>21M</td>
<td>0</td>
<td>21M</td>
<td>0%</td>
<td>/data/GFE</td>
</tr>
<tr>
<td>nas1:/dataLOCAL</td>
<td>162G</td>
<td>46G</td>
<td>116G</td>
<td>29%</td>
<td>/data/local</td>
</tr>
<tr>
<td>nas1:/verify</td>
<td>2.3T</td>
<td>3.1G</td>
<td>2.2T</td>
<td>1%</td>
<td>/data/verify</td>
</tr>
<tr>
<td>nas1:/aiidata</td>
<td>537G</td>
<td>11G</td>
<td>526G</td>
<td>3%</td>
<td>/awips2/exed/data</td>
</tr>
<tr>
<td>nas1:/localapps</td>
<td>54G</td>
<td>40M</td>
<td>54G</td>
<td>1%</td>
<td>/localapps</td>
</tr>
<tr>
<td>nas1:/GFESuite2</td>
<td>11G</td>
<td>144M</td>
<td>11G</td>
<td>2%</td>
<td>/awips2/GFESuite</td>
</tr>
<tr>
<td>nas2:/dataSTORE</td>
<td>2.2T</td>
<td>50G</td>
<td>2.2T</td>
<td>3%</td>
<td>/data_store</td>
</tr>
<tr>
<td>nas2:/awips2ARCHIVE</td>
<td>3.3T</td>
<td>8.2G</td>
<td>3.3T</td>
<td>1%</td>
<td>/archive</td>
</tr>
</tbody>
</table>

   Note: Check to see if everything is mounted (/awips2, /data_store, /awips2/exed/data/hdf5)

2. Check to see whether the Heartbeat, a2px2apps, and px2f up (ping px2f) are running.

3. Check to see whether the Rehosted applications are running.

   TYPE:  
   
   ps -ef | grep ldad

   ldad       760     1  0 Oct13 ?        00:00:05
   /awips/fxa/bin/CommsRouter LDAD_ROUTER
   ldad       768     1  0 Oct13 ?        00:00:05
   /awips/fxa/bin/DataController LDAD_ROUTER LdadController.config
   ldad       771   768  0 Oct13 ?        00:00:13
   /awips/fxa/bin/routerLdadDecoder px2-tbw3/56457/768 136938768
   ldad       772   768  0 Oct13 ?        00:00:31
   /awips/fxa/bin/routerStoreNetcdf px2-tbw3/56457/768 136965408
   ldad       773   768  0 Oct13 ?        00:00:19
Appendix I
Administrative Tips for Monitoring System Performance

/awips/fxa/bin/routerStoreEDEX px2-tbw3/56457/768 136974328
ldad 774 768 0 Oct13 ?   00:00:42
/awips/fxa/bin/routerShefEncoderEDEX px2-tbw3/56457/768 136974552
ldad 775 768 0 Oct13 ?   00:01:07
/awips/fxa/bin/routerStoreTextEDEX px2-tbw3/56457/768 136974856
ldad 1207 1 0 Oct13 ?   00:00:37 /bin/sh
/awips/ldad/bin/watchDogInternal.sh
ldad 26284 1207 0 17:43 ?   00:00:00 sleep 30
root 26286 26159 0 17:43 pts/1   00:00:00 grep ldad
perl /awips/ldad/bin/pollForData.pl Start
/awips/ldad/data/pollForData.conf
fxa 31767 1 0 Oct13 ?   00:00:00
/awips/fxa/bin/ldadServer
ldad 32762 1 0 Oct13 ?   00:00:01
/awips/ldad/bin/listener

**TYPE:** ps -ef|grep fxa

fxa 469 1 0 Oct30 ?   00:00:00 /bin/sh
/awips/fxa/FSIedex/bin/fsiWatchdog.sh
fxa 471 469 0 Oct30 ?   00:11:49
/awips/fxa/FSIedex/bin/FSIprocessorEDEX
fxa 808 1 0 Oct30 ?   00:00:00
/awips/fxa/fsi/bin/rssd
ldad 2419 1 0 Oct30 ?   00:00:17
/awips/fxa/bin/CommsRouter LDAD_ROUTER
ldad 2427 1 0 Oct30 ?   00:00:07
/awips/fxa/bin/DataController LDAD_ROUTER LdadController.config
ldad 2430 2427 0 Oct30 ?   00:01:13
/awips/fxa/bin/routerLdadDecoder px2-hgx/42989/2427 167269768
ldad 2431 2427 0 Oct30 ?   00:10:51
/awips/fxa/bin/routerStoreNetcdf px2-hgx/42989/2427 1672696408
ldad 2432 2427 0 Oct30 ?   00:01:50
/awips/fxa/bin/routerStoreEDEX px2-hgx/42989/2427 167305328
ldad 2433 2427 0 Oct30 ?   00:00:00
/awips/fxa/bin/routerShefEncoderEDEX px2-hgx/42989/2427 167305552
ldad 2434 2427 0 Oct30 ?   00:00:00
/awips/fxa/bin/routerStoreTextEDEX px2-hgx/42989/2427 167305856
root 14089 13693 0 15:42 pts/1   00:00:00 grep fxa
fxa 26795 1 0 Oct16 ?   00:00:00
/awips/fxa/bin/ldadServer
fxa 29944 1 0 Oct22 ?   00:00:00 avnpython
/awips/adapt/avnfps/OB9.2/py/avninit.py px2f
fxa 30092 29944 0 Oct22 ?   00:01:20 avnpython
/awips/adapt/avnfps/OB9.2/py/avnserver.py -d -n px2f
fxa 30101 29944 0 Oct22 ?   00:00:01 avnpython
/awips/adapt/avnfps/OB9.2/py/avndrs.py -d -n px2f
fxa 30125 29944 0 Oct22 ?   00:01:51 avnpython
/awips/adapt/avnfps/OB9.2/py/avndis.py -d -n px2f
fxa 30131 29944 0 Oct22 ?   00:00:19 avnpython
/awips/adapt/avnfps/OB9.2/py/avnxs.py -d -n px2f
fxa 30145 1 0 Oct22 ?   00:49:51
/awips/adapt/avnfps/gamin-0.1.10/libexec/gam_server
Appendix I

Administrative Tips for Monitoring System Performance

4. To monitor the MHS queues at the site:

<table>
<thead>
<tr>
<th>TYPE</th>
<th>dwb_smon</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server</td>
<td>PID</td>
</tr>
<tr>
<td></td>
<td>---------</td>
</tr>
<tr>
<td>msgreq_svr</td>
<td>24495</td>
</tr>
<tr>
<td>13:05:29 2011</td>
<td></td>
</tr>
<tr>
<td>(nfe/11030)</td>
<td>24496</td>
</tr>
<tr>
<td>13:05:29 2011</td>
<td></td>
</tr>
<tr>
<td>(worker)</td>
<td>24498</td>
</tr>
<tr>
<td>13:05:29 2011</td>
<td></td>
</tr>
<tr>
<td>(worker)</td>
<td>24741</td>
</tr>
<tr>
<td>13:05:30 2011</td>
<td></td>
</tr>
<tr>
<td>(worker)</td>
<td>24872</td>
</tr>
<tr>
<td>13:05:31 2011</td>
<td></td>
</tr>
<tr>
<td>(worker)</td>
<td>25164</td>
</tr>
<tr>
<td>13:05:32 2011</td>
<td></td>
</tr>
<tr>
<td>msgrcv_svr</td>
<td>24511</td>
</tr>
<tr>
<td>13:05:29 2011</td>
<td></td>
</tr>
<tr>
<td>(nfe/11040)</td>
<td>24514</td>
</tr>
<tr>
<td>13:05:29 2011</td>
<td></td>
</tr>
<tr>
<td>(worker)</td>
<td>23741</td>
</tr>
<tr>
<td>13:05:30 2011</td>
<td></td>
</tr>
<tr>
<td>(worker)</td>
<td>29667</td>
</tr>
<tr>
<td>17:28:03 2011</td>
<td></td>
</tr>
</tbody>
</table>

I.11.2 Workstations

1. Check for System Issues – Full partitions, load average, etc.

   TYPE: df –H
   TYPE: top
   TYPE: df /awips2/edex/data/share

2. Ensure that the directory is mounted from the nas1:aidata/share as shown below:

   [root@px1-tbw3 ~]# df /awips2/edex/data/share
   Filesystem    1K-blocks    Used   Available  Use% Mounted on
   nas1:/aidata/share     527556704  4756032  522800672   1% /awips2/edex/data/share

   TYPE: df /data/verify
3. Ensure that the directory is mounted from the nas1:verify as shown below:

```
[root@pxl-tbw3 ~]# df /data/verify
Filesystem 1K-blocks Used Available Use% Mounted on
nas1:/verify 1006233440 519968 1005713472 1% /data/verify
```

4. Ensure that AlertViz launches and does not ask for localization server.

5. Ensure that CAVE launches and products load.

6. Ensure that GFE launches within CAVE and that grids are available.

7. Ensure that WarnGen launches and that templates are available.

8. Ensure that AvnFPS launches and that data is available.

9. Ensure that Text Workstation launches on the XT.

10. Ensure that you can issue a textdb –r command from a terminal window.

### I.11.3 All Machines

1. Ensure that DNS is working.
   
   **TYPE:** `dig as3-ancf`

2. Ensure that NIS is working.

   **TYPE:** `ypcat hosts`

3. Ensure that NTP working.

   **TYPE:** `ntpq`

   **TYPE:** `pe`

```
ntpq> pe
remote refid st t when poll reach delay offset jitter
===========================================================================
*165.92.30.82 .CDMA. 1 u 138 1024 377 17.392 0.057 0.548
+165.92.180.82 .CDMA. 1 u 168 1024 377 32.069 -0.142 3.829
LOCAL(0) .LOCL. 10 l 13 64 377 0.000 0.000 0.001
```
Appendix J
WarnGen Templates
Appendix J. WarnGen Templates

Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>J.1 WarnGen Templates: AWIPS I / AWIPS II Equivalents</td>
<td>1</td>
</tr>
<tr>
<td>J.2 Auxiliary WarnGen Template Files</td>
<td>4</td>
</tr>
<tr>
<td>J.3 Template Language and Documentation</td>
<td>5</td>
</tr>
<tr>
<td>J.3.1 Troubleshooting WarnGen Startup</td>
<td>5</td>
</tr>
</tbody>
</table>

List of Tables

Table J.1-1. WarnGen Severe Weather Product Templates                   | 1    |
Table J.1-2. WarnGen Marine Weather Product Templates                   | 2    |
Table J.1-3. WarnGen Hydrologic Product Templates                       | 2    |
Table J.1-4. Non-Warning Product Templates                              | 3    |
Table J.1-5. WarnGen Impact-Based Warning (IBW) Templates               | 3    |
The templates described in this appendix will streamline WarnGen operations and simplify WarnGen template maintenance. Template Team members are continually updating detailed information about WarnGen templates on the AWIPS Migration wiki at the following URL:


General WarnGen configuration is now set in the config.xml (ConfigInfo) file, found in the $EDEX_HOME/data/utility/common_static/site/XXX/warngen directory on dx3/dx4, where $EDEX_HOME is typically /awips2/edex and XXX is the three-letter ID of the WFO (e.g., LWX or CTP). The localization concept via a script is no longer required. File edits are now done via the Localization Perspective within CAVE.

To edit config.xml:
1. Open the localization perspective in CAVE.
2. Under the D2D and then the warngen tree folders, right click on the BASE config.xml and do a copy to site (XXX) where XXX is your active site ID.
3. You should now see a new copy of config.xml with a SITE (XXX) tag to it. A baseline copy of config.xml is copied to the local site folder described above.
4. Double click this file to edit it in the editor.
5. An example config.xml is contained within the baseline file and should be very useful for customizing the WarnGen GUI as well as the primary/backup WFO nomenclature and IDs. Save your edits when done. [Note: You will need to restart CAVE and D2D to enact any changes made to config.xml.]

Note: The AWIPS II templates, although largely complete, are still in development and may contain minor issues.

### J.1 WarnGen Templates: AWIPS I / AWIPS II Equivalents

Tables J.1-1 – J.1-5 list the templates for AWIPS I and their equivalents in AWIPS II.

<table>
<thead>
<tr>
<th>Warning</th>
<th>AWIPS I Templates</th>
<th>AWIPS II Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tornado Warning - PIL: TOR, e.g. STLTOREREAX</td>
<td>wwa_tor.preWWA</td>
<td>tornadoWarning.vm and tornadoWarning.xml</td>
</tr>
<tr>
<td>Severe Thunderstorm Warning - PIL: SVR, e.g., STLSVREAX</td>
<td>wwa_svr.preWWA</td>
<td>severeThunderstormWarning.vm and severeThunderstormWarning.xml</td>
</tr>
<tr>
<td>Severe Weather Statement, follow-up to TOR or SVR - PIL: SVS, e.g., STLSVSEAX</td>
<td>wwa_svrwx_ssta_county.pre WWA</td>
<td>severeWeatherStatement.vm and severeWeatherStatement.xml</td>
</tr>
<tr>
<td>Extreme Wind Warning - PIL: EWW, e.g., MIAEWWWMFL</td>
<td>wwa_eww.preWWA</td>
<td>extremeWindWarning.vm and extremeWindWarning.xml</td>
</tr>
<tr>
<td>Severe Weather Statement, follow-up to EWW - PIL: SVS, e.g., MIASVSMFL</td>
<td>wwa_eww_svs.preWWA</td>
<td>extremeWindWarningFollowup.vm and extremeWindWarningFollowup.xml</td>
</tr>
</tbody>
</table>
### Appendix J WarnGen Templates

#### Table J.1-2. WarnGen Marine Weather Product Templates

<table>
<thead>
<tr>
<th>Warning</th>
<th>AWIPS I Templates</th>
<th>AWIPS II Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Special Marine Warning - PIL: SMW, e.g., BOSSMWBOX</td>
<td>wwa_specmarine.preWWA</td>
<td>specialMarineWarning.vm and specialMarineWarning.xml</td>
</tr>
<tr>
<td>Marine Weather Statement, follow-up to SMW - PIL: MWS, e.g., BOSMWSBOX</td>
<td>wwa_mar_wx_sta.preWWA</td>
<td>specialMarineWarningFollowup.vm and specialMarineWarningFollowup.xml</td>
</tr>
</tbody>
</table>

#### Table J.1-3. WarnGen Hydrologic Product Templates

<table>
<thead>
<tr>
<th>Warning</th>
<th>AWIPS I Templates</th>
<th>AWIPS II Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flash Flood Warning (convective) - PIL: FFW, e.g., STLFFWSGF</td>
<td>wwa_ffw.preWWA</td>
<td>flashFloodWarning.vm and flashFloodWarning.xml</td>
</tr>
<tr>
<td>Flash Flood Statement (follow-up to convective FFW) - PIL: FFS, e.g., STLFFSSGF</td>
<td>wwa_flflood_sta_county.preWWA</td>
<td>flashFloodWarningFollowup.vm and flashFloodWarningFollowup.xml</td>
</tr>
<tr>
<td>Non-Convective Flash Flood Warning (incl. Dam Break) - PIL: FFW, e.g., OMAFFWOAX</td>
<td>wwa_dam_break.preWWA</td>
<td>nonConvectiveFlashFloodWarning.vm and nonConvectiveFlashFloodWarning.xml</td>
</tr>
<tr>
<td>Non-Convective Flash Flood Statement (incl. Dam Break) - PIL: FFS, e.g., OMAFFSOAX</td>
<td>wwa_flflood_sta.preWWA</td>
<td>nonConvectiveFlashFloodWarningFollowup.vm and nonConvectiveFlashFloodWarningFollowup.xml</td>
</tr>
<tr>
<td>Areal Flood Warning - PIL: FLW, e.g., WBCFLWLWX</td>
<td>wwa_flood_wrn.preWWA</td>
<td>arealFloodWarning.vm and arealFloodWarning.xml</td>
</tr>
</tbody>
</table>

---

AWP.MAN.SMM.A2

– Hard copies uncontrolled. Verify effective date prior to use. –
### Appendix J WarnGen Templates

<table>
<thead>
<tr>
<th>Warning</th>
<th>AWIPS I Templates</th>
<th>AWIPS II Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Areal Flood Statement (follow-up to FLW) - PIL: FLS, e.g., WBCFLSLLWX</td>
<td>wwa_flood_stata.preWWA</td>
<td>arealFloodWarningFollowup.vm and arealFloodWarningFollowup.xml</td>
</tr>
<tr>
<td>Areal Flood Advisory (events below severe criteria) - PIL: FLS, e.g., STLFLSSGF</td>
<td>wwa_flood_adv.preWWA</td>
<td>arealFloodAdvisory.vm and arealFloodAdvisory.xml</td>
</tr>
<tr>
<td>Areal Flood Advisory Statement (follow-up to areal flood advisory) - PIL: FLS, e.g., STLFLSSGF</td>
<td>wwa_flood_adv_stata.preWWA</td>
<td>arealFloodAdvisoryFollowup.vm and arealFloodAdvisoryFollowup.xml</td>
</tr>
</tbody>
</table>

### Table J.1-4. WarnGen Non-Warning Product Templates

<table>
<thead>
<tr>
<th>Non-Warning</th>
<th>AWIPS I Templates</th>
<th>AWIPS II Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Significant Weather Advisory - PIL: SPS, e.g., STLSPEAX</td>
<td>wwa_swa.preWWA (in /data/fxa/customFiles or /awips/fxa/data/localization/XXX)</td>
<td>significantWeatherAdvisory.vm and significantWeatherAdvisory.xml</td>
</tr>
<tr>
<td>Short Term Forecast - PIL: NOW, e.g., WBCNOWLWX</td>
<td>wwa_shorttermfcst.preWWA</td>
<td>shortTermForecast.vm and shortTermForecast.xml</td>
</tr>
<tr>
<td>Special Weather Statement for near-severe Thunder and Snow Squalls - PIL: SPS, e.g., PHLSPSCTP</td>
<td>N/A</td>
<td>erSpecialWeatherStatement.vm and erSpecialWeatherStatement.xml</td>
</tr>
<tr>
<td>Special Weather Statement - PIL: SPS, e.g., WBCSPSLWX</td>
<td>wwa_sws.preWWA</td>
<td>specialWeatherStatement.vm and specialWeatherStatement.xml</td>
</tr>
</tbody>
</table>

### Table J.1-5. WarnGen Impact-Based Warning (IBW) Templates

<table>
<thead>
<tr>
<th>Warning</th>
<th>AWIPS I Templates</th>
<th>AWIPS II Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tornado Warning - PIL: TOR, e.g., STLTORAX</td>
<td>wwa_impact_tor.preWWA</td>
<td>impactTornadoWarning.vm and impactTornadoWarning.xml</td>
</tr>
<tr>
<td>Severe Thunderstorm Warning - PIL: SVR, e.g., STLSVREAX</td>
<td>wwa_impact_svr.preWWA</td>
<td>impactSevereThunderstormWarning.vm and impactSevereThunderstormWarning.xml</td>
</tr>
<tr>
<td>Severe Weather Statement, follow-up to TOR or SVR - PIL: SVS, e.g., STLSVSEAX</td>
<td>wwa_impact_svs.preWWA</td>
<td>impactSevereWeatherStatement.vm and impactSevereWeatherStatement.xml</td>
</tr>
<tr>
<td>PIL: SMW, e.g., ARBSMWDTX</td>
<td>wwa_impact_smw.preWWA</td>
<td>impactSpecialMarineWarning.vm and impactSpecialMarineWarning.xml</td>
</tr>
<tr>
<td>Marine Weather Statement, follow-up to SMW - PIL: MWS, e.g., ARBWMWSDTOX</td>
<td>wwa_impact_mws.preWWA</td>
<td>impactSpecialMarineWarningFollowup.vm and impactSpecialMarineWarningFollowup.xml</td>
</tr>
</tbody>
</table>
### J.2 Auxiliary WarnGen Template Files

You will likely note additional files in the AWIPS II WarnGen directory, some of which may be edited. Others are part of the baseline.

1. The baseline files (found in \$EDEX_HOME/data/utility/common_static/base/warngen) are:
   - **countyTypes.txt.** A translational table to create proper county/zone/independent city wording.
   - **states.txt.** A translational table of state abbreviations.
   - **immediateCauses.txt.** A translational table of hydro VTEC immediate causes.
   - **VM_global_library.** A Velocity template repository houses functions that are used repeatedly through the WarnGen templates. If you find a #functionname(args) like syntax and cannot figure out what it does, chances are that the referenced function is contained here.

2. The editable files (found in \$EDEX_HOME/data/utility/common_static/site/XXX/warngen) are:
   - **dupCounties.vm.** An import file to assist those sites that have same-named counties across two states. Similar to a patch placed in AWIPS I to produce correct text output.
   - **damInfo.vm.** Created to assist users in creating local dam information for the non-convective Flash Flood Warning/Statement?.
   - **damInfoBullet.xml.** Same as damInfo.vm, except that it is used to create bullets that are displayed in the WarnGen GUI through the accompanying non-convective Flash Flood Warning/Statement? XML files.
   - **damInfoBulletName.xml.** Same as above, but used to create bullets that are displayed in the WarnGen GUI through the accompanying non-convective Flash Flood Warning/Statement? XML files.
   - **mileMarkers.xml.** An XML file that creates WarnGen objects from the mile markers tables in the AWIPS II database. The WarnGen documentation describes the utilization of the importMarkersInfo.sh script to get AWIPS I mile markers into AWIPS II. This XML file contains instructions and an example for making the mile marker information available for use in the templates. This XML file is referenced via an "include" statement in each WarnGen template's XML configuration file. Simply uncomment the include section that has already been inserted by the developers.
   - **mileMarkers.vm.** The "guts" that actually generate the mile marker or road output. Here you will reference the mile marker objects created in the XML file, and create "lead-ins" that will create proper phrasing for your mile marker output. This auxiliary vm file is then referenced via a "parse" statement in each WarnGen template's Velocity .vm file, instructing WarnGen to generate the mile marker
Appendix J WarnGen Templates

output. Simply uncomment the appropriate section in each WarnGen template's vm file that has already been inserted by the developers.

- **forecasterName.vm.** vm code that will help assign a warning forecaster name to warnings automatically. Must be customized to create local user's desired moniker (name, initials, number, etc.). This auxiliary vm file is then referenced (if desired) via a "parse" statement added to each WarnGen template's Velocity .vm file.

### J.3 Template Language and Documentation

WarnGen templates are created using velocity template language (VTL) with supporting configurations in an xml formatted file with an .xml extension. More information on this can be found at: [http://svn.apache.org/repos/asf/velocity/engine/tags/V_1_0_1/docs/vtl-reference-guide.html](http://svn.apache.org/repos/asf/velocity/engine/tags/V_1_0_1/docs/vtl-reference-guide.html) and [http://click.sourceforge.net/docs/velocity/vtl-reference-guide.html](http://click.sourceforge.net/docs/velocity/vtl-reference-guide.html). Refer to the following wiki page for more information on WarnGen documentation:

[https://collaborate.nws.noaa.gov/trac/siteconfig/attachment/wiki/WarnGen/AWIPS_2_WarnGen_Documentation_1.3.1.doc](https://collaborate.nws.noaa.gov/trac/siteconfig/attachment/wiki/WarnGen/AWIPS_2_WarnGen_Documentation_1.3.1.doc)

### J.3.1 Troubleshooting WarnGen Startup

This exercise is designed to familiarize you with troubleshooting CAVE applications by launching via the command line and searching through logs.

**Exercise Procedure**

AWIPS II has reengineered the WarnGen application into CAVE. While GUI launches the GUI in the same as launches in AWIPS I, the mechanics behind the processes are different. There are configuration files for the GUI, as well as the templates. This exercise walks through the process of identifying and correcting a problem in site data configuration.

1. Log into your AWIPS workstation. AlertViz should start automatically.

2. Use the left mouse button to open the AWIPS menu, and select ‘Start CAVE (LX) / TextWS (XT).”

![Image](Start CAVE (lx) / TextWS (xt) - AWIPS C Ave-TextWS.png)
3. Change scales by selecting ‘WFO’ from the “Map Scale” pulldown menu.

![Map Scale Menu]

4. Start WarnGen by selecting the “Yellow” button in the upper right corner of the CAVE D2D Perspective.

5. You should have received an AlertViz popup after depressing the “WarnGen” button. The error inside the message should be “Failed to initialized map tool”.

![Alert Visualization Popup Message Dialog]

6. Select the ‘Show Log’ button so that the message log appears below the AlertViz popup and double click on the line with the failure message shown in the screen capture in step 5.

7. You should see an additional window appear on the AlertViz popup with the following text. Review the java exception error on your screen.
Failed to initialize a Tool Resource


at com.raytheon.viz.warnngen.gui.WarnngenAction getResource(WarnngenAction.java:82)

at com.raytheon.viz.warnngen.gui.WarnngenAction getResource(WarnngenAction.java:1)


at org.eclipse.ui.internal.handlers.HandlerProxy execute(HandlerProxy.java:293)

at org.eclipse.core.commands.Command executeWithChecks(Command.java:476)

at org.eclipse.core.commands.ParameterizedCommand executeWithChecks(ParameterizedCommand.java:508)

at org.eclipse.ui.internal.handlers.HandlerService executeCommand(HandlerService.java:169)

at org.eclipse.ui.internal.handlers.SlaveHandlerService executeCommand(SlaveHandlerService.java:241)

at org.eclipse.ui.menus.CommandContributionItem handleWidgetSelection(CommandContributionItem.java:820)

at org.eclipse.ui.menus.CommandContributionItem$5 handleEvent(CommandContributionItem.java:796)

at org.eclipse.swt.widgets.EventTable sendEvent(EventTable.java:84)

at org.eclipse.swt.widgets.Widget.sendEvent(Widget.java:1258)

at org.eclipse.swt.widgets.Display runDeferredEvents(Display.java:3540)

at org.eclipse.swt.widgets.Display readAndDispatch(Display.java:3161)

at org.eclipse.ui.internal.Workbench.runEventLoop(Workbench.java:2640)

at org.eclipse.ui.internal.Workbench$7 run(Workbench.java:671)

at org.eclipse.core.databinding.observable.Realm runWithDefault(Realm.java:332)

at org.eclipse.ui.internal.Workbench createAndRunWorkbench(Workbench.java:664)

at org.eclipse.ui.PlatformUI createAndRunWorkbench(PlatformUI.java:149)

at com.raytheon.viz.ui.personalities.awips.VizApplication start(VizApplication.java:233)

at org.eclipse.equinox.internal.app.EclipseAppHandle run(EclipseAppHandle.java:196)

at org.eclipse.core.runtime.internal.adaptor.EclipseAppLauncher.runApplication(EclipseAppLauncher.java:110)

at org.eclipse.core.runtime.internal.adaptor.EclipseAppLauncher start(EclipseAppLauncher.java:124)
8. Click the ‘Acknowledge All’ button, and minimize CAVE. The WarnGen GUI is configured through site data configuration with an XML file. Perhaps something went wrong? The file would have been transferred to your caveData when you localized, so take a look there first. Open a terminal window by left clicking and selecting ‘terminal’, and, from the prompt, change into the following directory.

Enter: cd caveData/configuration/site/LLL/com.raytheon.viz.warngen

Where LLL is the SITE ID of the system you are using for the exercise.

9. Note the path of the directory you changed to in step 8. It is in a configuration directory, at a site level, and has to do with “viz” and “warngen”. Take a look at the config.xml file that is in this directory.

Enter: cat config.xml

10. XML syntax is a series of tags that start with an opening tag and, at some point, have a closing tag. The closing tag is different from the opening tag only in that it has a “/” character before the name of the tag. For example:

<name> Kevin Johnson </name>
Scour the text on your screen now and see if you can find any errors in the XML.

11. You should have noted that the `<siteNode>` XML tag has no closing `</siteNode>` tag. You will need to fix this in order for WarnGen to start. Use the *vi* editor.

   **Enter:** `cp -a config.xml BAD-config.xml`
   **Enter:** `vi config.xml`

   Navigate to the line that starts `<siteNode>` and add the string `</siteNode>`, as shown below, after the three letter site node. Now, save the file.

   `<siteNode>OAX</siteNode>`

   **Enter:** `Esc :wq!`

12. Now, maximize your CAVE by selecting it from your task bar list. Once maximized, try to start WarnGen once more.

13. Success! WarnGen will now start! That was the problem. However, you just edited a file in caveData to test the change. You know that the file originates in the EDEX Localization Store. It needs to be fixed there in order to handle the site data configuration change properly.

14. Change focus to the terminal window that should be behind the CAVE display. Secure shell into the DX3 or DX4 as user *root* and change into the Localization Store.

   **Enter:** `ssh root@dx3`
   **Enter:** `cd /awips2/edex/data/utility/cave_config/`

15. The problem file is actually in `site/LLL/com.raytheon.viz.warngen`; however, for the purposes of this exercise you are going to create a user-level file to fix the problem. In a normal situation, a user would not have his, or her, own version of this file. It is important to note that this user version file is only for the purposes of this exercise.

   **Enter:** `mkdir -p user/YOURUSERNAME/com.raytheon.viz.warngen`
   **Enter:** `chown -R awips:awips user`
   **Enter:** `cp -a site/LLL/com.raytheon.viz.warngen/config.xml user/YOURUSERNAME/com.raytheon.viz.warngen/`

   **Enter:** `cd user/YOURUSERNAME/com.raytheon.viz.warngen`

   **Note:** Substitute your user name for YOURUSERNAME above. For example: `mkdir -p user/kevinj/com.raytheon.viz.warngen`.

   Notice the similarities to the path in caveData that you originally looked into for the file.

16. Use the *vi* editor to edit the config.xml and fix the `<siteNode>` tag as done above.

17. Once the file has been fixed,

   **Enter:** `exit`
18. You need to make sure that the file gets downloaded properly now, so remove the config.xml in your caveData and re-launch CAVE to see if WarnGen will start. In your terminal window, change into the user-level directory to which your file will be downloaded.

   Enter: cd
   ~/caveData/configuration/user/YOURUSERNAME/com.raytheon.viz.warngen

19. Exit CAVE by selecting ‘CAVE → Exit…’.

   The reason that CAVE must be restarted is that it has already read the corrected file from when you made the local change in step 11. By restarting, you are ensuring a proper file is re-downloaded from EDEX Localization Store.

20. Re-launch CAVE through your left click menu, and once again change scales and start WarnGen via the button. (Reference steps 3 and 4 above.)

   WarnGen should have started successfully. In your terminal window, check to see if the file was downloaded.

   Enter: ls -l

   You should see that the file has now been downloaded from the EDEX Localization Store. If this was not an exercise, and the site-level file had actually been changed, it would have been downloaded correctly to the site-level directory to ensure that the next forecaster to launch WarnGen would not see the error.
Appendix K
Reporting Problems to the NCF:
Tips on Completing Trouble Ticket
Worksheets and Describing Slow
Workstation Performance
# Appendix K. Reporting Problems to the NCF:
Tips on Completing Trouble Ticket Worksheet and 
Describing Slow Workstation Performance

## Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>K.1 NCF Trouble Ticket Worksheet</td>
<td>1</td>
</tr>
<tr>
<td>K.2 AWIPS Workstation Performance (Slowness) During Severe Weather</td>
<td>4</td>
</tr>
<tr>
<td>K.2.1 AWIPS Tips for Improved System Performance</td>
<td>4</td>
</tr>
</tbody>
</table>
From time to time, you will need to contact the Network Control Facility (NCF) in Silver Spring, Maryland, to report a problem at your site. This appendix is provided to help you give the NCF engineers the information needed to resolve your issue.

The appendix has two sections:

- K.1, which includes a copy of the NCF Trouble Ticket Worksheet and instructions for its use; and

- K.2, which provides additional guidelines for working with the NCF to resolve issues during severe weather. This section also includes tips for avoiding problems during severe weather.

### K.1 NCF Trouble Ticket Worksheet

The NCF Trouble Ticket Worksheet is a tool designed for your use in reporting a problem at your site. Please keep copies of the worksheet on hand and have the information requested on the worksheet available when you call the NCF to report your problem. Missing or incomplete information can have a negative effect on the resolution process, and increase the time it takes to diagnose and resolve the problem.

A copy of the NCF Trouble Ticket Worksheet and instructions for completing the worksheet appear on the following two pages.
### NCF Trouble Ticket Worksheet

**Note:** The NCF will determine the Severity Code depending on site supplied weather situation (#3) and the problem type encountered (#10)

1. Date:___________  Time:___________  Trouble Ticket Number:___________

2. Site Identification:________________________ (e.g., TSA for WFO Tulsa, OK)
   (Use ONLY standard NWS office IDs, Emphasize RFC or WFO.)

3. Weather Situation: Critical/Watches/Warnings: ____  Busy: ____  Quiet/Benign:____

4. NWS Operational Impact Assessment:
   Immediate Fix:________  Routine Fix:________
   Do Not Fix UNTIL NWS Calls Back:________

5. NWS Person Calling:

6. NWS Senior Forecaster/Senior Hydrologist on Duty:

7. NCF Contact (Name of NCF Person on Telephone):

8. NWS Call Back Telephone Number:

9. NWS Brief Summary of Problem/Symptoms:_________________________________
   _______________________________________________________________________
   _______________________________________________________________________

10. Problem Type:

    Data Quality or Availability
    
    | A. Radar | B. Model | C. Satellite | D. LDAD |
    |----------|----------|--------------|---------|
    | E. ALL   | F. NWSTG | G. Text products | H. Other |
    
    Software
    
    | A. CAVE/D2D | B. Text Editing | C. OH application | D. CAVE/GFE |
    | E. SCAN     | F. WHFS       | G. EDEXdecoder   | H. WarnGen  |
    | I. AVNFP   | J. LDAD      | K. CLIMATE       | L. HWR  |
    | M. APS      | N. Localization | O. Archive  | P. Other |
    
    Hardware
    
    | A. Monitor | B. Workstations | C. Multiple Workstations | D. Printer |
    | E. Terminal Server | F. Tape Drive | G. LDAD Servers | H. CPs |
    | I. PXs | J. DXs | K. Other | |

Site Cannot

| A. Prepare/issue watch/warning/forecasts and/or advisories |
| B. Collect/distribute mission critical observations |

Network Issues

| A. Impacts AWIPS Functionality | B. Doesn’t result in “blackout” |
| C. Other |

Other Issues

| A. Site customization problem, such as localization or product modification. |
| B. Customization issue, such as problems arising from development outside the AWIPS baseline |
| C. Administrative support (e.g., Government-furnished software usage questions; routine reports; ad hoc reports). |
| D. Questions on how to use software. |

11. Problem Frequency: ___ First Time Occurrence  ___ Intermittent - Occurred _____ times over _____ Hours  ___ Constant - Cannot use to do work

12. NCF Estimated Time to Diagnose/Troubleshoot/Resolve: _______

13. NCF Actual Time to Restore: _______

14. NWS Initial customer service rating: 1 2 3 4 5

15. Rating after NCF resolved problem/restored system: 1 2 3 4 5
   (NCF rating scale: 1-poor, 2-fair, 3-average, 4-good, 5-excellent.)
NCF Trouble Ticket Worksheet Instructions

1. Dates and times should be in universal time coordinate (UTC)/Z.

2. Always ask for a Trouble Ticket number.

3. Use the standard NWS office IDs and emphasize whether you are calling from an RFC or WFO.

4. Indicate your office work situation due to weather. An office may
   - Have benign/quiet weather;
   - Be involved with critical weather, warnings, watches, and related issues; or
   - Be very busy because of past or anticipated significant weather.

   **Note:** The NCF will determine the Severity Code depending on site supplied weather situation (No. 3) and the problem type encountered (No. 10).

5. Your impact assessment indicates a priority of attention required. For example, some problems SHOULD NOT be worked on by the NCF until the ESA, SOO, DOH, AWIPS Focal Point or office manager has been contacted or is present. In this case, the initial call is to alert the NCF that there is a problem that is pending.

6. The NWS caller’s name.

7. The name of the senior person on duty—lead forecaster, senior forecaster, or senior hydrologist. This is especially critical during hours when the ESA, SOO, or AWIPS Focal Point are not available. This references the office POC during nonstandard hours.

8. Name of NCF engineer handling the call. You or the senior duty person may receive a call from another NCF person—please enter the second name.

9. The NCF will verify the NWS number for return calls. Please give the NCF the appropriate number to ensure communication between the NCF and your office is not delayed.

10. Please briefly comment about the problem, describing the symptoms.

11. Problem type should be checked or circled so it can be read to the NCF engineer.
K.2  *AWIPS Workstation Performance (Slowness) During Severe Weather*

Your responses to questions as to the performance (slowness) of your CAVE/D2D perspective or your workstation will help the NCF resolve performance problems, especially during severe weather or emergency situations. Please be ready to answer the following questions:

- Is the slowness occurring with a specific display in D2D, or with the workstation as a whole?
- Did your site run through a pre-severe weather checklist?

If the problem is affecting just one display:

- Do you have the Local & Regional Warning Display loaded?
- Do you have the Local Warning Display loaded?
- About what time did the slowness start?
- On which monitor is the pane loaded?
- Does anyone else have this loaded on another CAVE/D2D? If so, are they having problems?
- Is the slowness mainly with restoring a pane that has been put into a side-panel?

If the entire workstation is affected:

- Do you have FSI running? If so, does stopping it help?

K.2.1  *AWIPS Tips for Improved System Performance*

With help from the NCF, you may be able to improve system performance during severe weather. To the best of your abilities, several hours before severe weather initiation is anticipated, take the following actions:

1. Call the NCF and request "an edex restart to prepare for a severe weather event." Remind the NCF to restart edex one at a time. [Important: No grid editing in GFE should be done while the NCF is restarting the edex processes.]

2. Quite likely while on the same call, ask the NCF to open a ‘severe weather’ trouble ticket for your site. Opening the ticket helps put the NCF on notice that you expect to be busy, and this will help the NCF remain alert to system monitoring duties.

   Note: Actions 1 and 2 should not take long, and usually the NCF is able to start, and maybe even complete, these actions while still on the phone.

3. Notify the NCF by telephone that you are going to reboot your workstations. Then, as timing and duties permit, reboot all workstations, even the ones that may
not currently have been in use. This will ensure all workstations are ready if/when they need to be staffed up.

4. Have extra workstations logged in. In case you need to quickly move to another workstation due to unforeseen problems, you can quickly move to the extra workstation and restart CAVE, procedures, etc.

5. Call or chat with your service backup sites. Give them a friendly reminder that you anticipate severe weather, and that you are on AWIPS II and in OT&E status. You hope you will not need their assistance, but just in case you do, they will not be caught totally caught off guard.

Note: The NCF has a “critical weather monitoring” script that it runs after SPC issues a Severe Thunderstorm Watch or Tornado Watch. This action is taken automatically at the NCF as soon as the watch box is issued. This script does not take any site action, but it gives the NCF a good snapshot of how your system is performing. If the NCF needs to take any actions as a result of the script, someone from the NCF will call BEFORE initiating action.
Appendix L
LDAD Configuration Samples and
Firewall Architecture
Appendix L. LDAD Configuration
Samples and Firewall Architecture

Table of Contents

L.1 LDAD Wiring Lists .............................................................................................................. 1
L.2 LDAD Terminal Server Port Configuration ......................................................................... 2
L.3 LDAD Modem Configurations ............................................................................................ 3
L.4 Environment Variables ......................................................................................................... 6
L.5 AWIPS Firewall Architecture and Configuration ................................................................. 7
   L.5.1 AWIPS Firewall Architecture .................................................................................... 8
   L.5.2 Firewall Configuration ............................................................................................... 8
   L.5.3 LDAD LAN (DMZ) Configuration ............................................................................. 9
   L.5.4 AWIPS LAN Reconfiguration .................................................................................... 9
   L.5.5 Interconnectivity Between the Firewall Zones and Security Policies
       (Firewall Rules) ........................................................................................................... 10
   L.5.6 LDAD FAX Troubleshooting .................................................................................. 13
   L.5.7 LDAD Firewall Site Configuration Management .................................................... 13
   L.5.8 Csportd: Configuration and Setup ........................................................................... 13

List of Tables

Table L.1-1. LDAD Physical Wiring List ...................................................................................... 1
Table L.1-2. LDAD Firewall Wiring List .................................................................................... 2
Table L.2-1. Port Assignments ..................................................................................................... 2
Table L.3-1. LDAD Modem Configurations ............................................................................. 3
Table L.4-1. Environment Variables ............................................................................................ 7
Table L.5.5-1. Default Firewall Ruleset .................................................................................... 10
### L.1 LDAD Wiring Lists

Table L.1-1 provides the LDAD physical wiring list. Table L.1-2 provides the AWIPS LDAD firewall wiring list.

#### Table L.1-1. LDAD Physical Wiring List

<table>
<thead>
<tr>
<th>Category</th>
<th>Cable Type</th>
<th>End Point 1</th>
<th>End Point 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network</td>
<td>CAT5 UTP</td>
<td>Terminal Server Ethernet 10/100 Base T</td>
<td>SMC 8024 Port 3</td>
</tr>
<tr>
<td>Network</td>
<td>CAT5 UTP</td>
<td>External Gateway Ethernet Plaintree</td>
<td>1 018 Port 4</td>
</tr>
<tr>
<td>Network</td>
<td>SMC 1080 Port 8</td>
<td>Firewall exp1</td>
<td></td>
</tr>
<tr>
<td>Serial</td>
<td>RJ45-DB25 Male RS232 DTE-DCE</td>
<td>Terminal Server Port 1</td>
<td>Modem Chassis Modem x</td>
</tr>
<tr>
<td>Serial</td>
<td>RJ45-DB25 Male RS232 DTE-DCE</td>
<td>Terminal Server Port 2</td>
<td>Modem Chassis Modem x</td>
</tr>
<tr>
<td>Serial</td>
<td>RJ45-DB25 Male RS232 DTE-DCE</td>
<td>Terminal Server Port 3</td>
<td>Modem Chassis Modem x</td>
</tr>
<tr>
<td>Serial</td>
<td>RJ45-DB25 Male RS232 DTE-DCE</td>
<td>Terminal Server Port 4</td>
<td>Modem Chassis Modem x</td>
</tr>
<tr>
<td>Serial</td>
<td>RJ45-DB25 Male RS232 DTE-DCE</td>
<td>Terminal Server Port 5</td>
<td>Modem Chassis Modem x</td>
</tr>
<tr>
<td>Serial</td>
<td>RJ45-DB25 Male RS232 DTE-DCE</td>
<td>Terminal Server Port 6</td>
<td>Modem Chassis Modem x</td>
</tr>
<tr>
<td>Serial</td>
<td>RJ45-DB25 Male RS232 DTE-DCE</td>
<td>Terminal Server Port 7</td>
<td>Modem Chassis Modem x</td>
</tr>
<tr>
<td>Serial</td>
<td>RJ45-DB25 Male RS232 DTE-DCE</td>
<td>Terminal Server Port 8</td>
<td>Modem Chassis Modem x</td>
</tr>
<tr>
<td>Serial</td>
<td>RJ45-DB25 Male RS232 DTE-DCE</td>
<td>Terminal Server Port 9</td>
<td>Modem Chassis Modem x</td>
</tr>
<tr>
<td>Serial</td>
<td>RJ45-DB25 Male RS232 DTE-DCE</td>
<td>Terminal Server Port 10</td>
<td>Modem Chassis Modem x</td>
</tr>
<tr>
<td>Serial</td>
<td>RJ45-DB25 Male RS232 DTE-DCE</td>
<td>Terminal Server Port 19</td>
<td>DTMF Converter 1</td>
</tr>
<tr>
<td>Serial</td>
<td>RJ45-DB25 Male RS232 DTE-DCE</td>
<td>Terminal Server Port 20</td>
<td>DTMF Converter 2</td>
</tr>
<tr>
<td>Serial</td>
<td>RJ45-DB25 Male RS232 DTE-DCE</td>
<td>Terminal Server Port 21</td>
<td>Modem Chassis Modem x</td>
</tr>
<tr>
<td>Serial</td>
<td>RJ45-DB25 Male RS232 DTE-DCE</td>
<td>Terminal Server Port 22</td>
<td>Modem Chassis Modem x</td>
</tr>
<tr>
<td>Serial</td>
<td>RJ45-DB25 Male RS232 DTE-DCE</td>
<td>Terminal Server Port 23</td>
<td>Modem Chassis Modem x</td>
</tr>
<tr>
<td>Serial</td>
<td>RJ45-DB25 Male RS232 DTE-DCE</td>
<td>Terminal Server Port 24</td>
<td>Modem Chassis Modem x</td>
</tr>
<tr>
<td>Serial</td>
<td>RJ45-DB25 Male RS232 DTE-DCE</td>
<td>Terminal Server Port 30</td>
<td>LDAD FAX Modem</td>
</tr>
<tr>
<td>Serial</td>
<td>RJ45-DB25 Male RS232 DTE-DCE</td>
<td>Terminal Server Port 31</td>
<td>LDAD LS2 Console Port</td>
</tr>
<tr>
<td>Serial</td>
<td>RJ45-DB25 Male RS232 DTE-DCE</td>
<td>Terminal Server Port 32</td>
<td>LDAD LS3 Console Port</td>
</tr>
<tr>
<td>Serial</td>
<td>RJ45-DB25 Male RS232 DTE-DCE</td>
<td>Terminal Server Port 29</td>
<td>LDAD LAN Switch Console Port</td>
</tr>
</tbody>
</table>
Table L.1-2. LDAD Firewall Wiring List

<table>
<thead>
<tr>
<th>Source Device</th>
<th>Source Identifier</th>
<th>Destination Device</th>
<th>Destination Identifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>FW 1</td>
<td>LAN 1</td>
<td>FW/SW1</td>
<td>Port 4</td>
</tr>
<tr>
<td>FW 2</td>
<td>LAN 3</td>
<td>LDAD LSW</td>
<td>Port 5</td>
</tr>
<tr>
<td>FW/SW 1</td>
<td>Port 1</td>
<td>AWIPS HSW 1</td>
<td>Port 21</td>
</tr>
<tr>
<td>FW/SW 2</td>
<td>Port 2</td>
<td>AWIPS HSW 2</td>
<td>Port 21</td>
</tr>
<tr>
<td>FW/SW 2</td>
<td>Port 1</td>
<td>AWIPS HSW 1</td>
<td>Port 22</td>
</tr>
<tr>
<td>FW 1</td>
<td>Console</td>
<td>M &amp; C</td>
<td>Port 38</td>
</tr>
<tr>
<td>FW 2</td>
<td>Console</td>
<td>M &amp; C</td>
<td>Port 29</td>
</tr>
<tr>
<td>Terminal Server</td>
<td>Port 19</td>
<td>DTMF 1</td>
<td>Port RS232</td>
</tr>
<tr>
<td>Terminal Server</td>
<td>Port 20</td>
<td>DTMF 2</td>
<td>Port RS232</td>
</tr>
<tr>
<td>FW 1</td>
<td>LAN 4</td>
<td>FW 2</td>
<td>Lan 4</td>
</tr>
</tbody>
</table>

L.2  LDAD Terminal Server Port Configuration

The LDAD Terminal Server is a Cyclades Alterpath ACS32, which provides an Ethernet interface for connection to the LDAD LAN Switch and 32 serial ports for connections to communication devices such as the modems in the Modem Nest, and Console Management ports such as the LS2/3 Servers. Each port can be configured to a maximum speed of 56 K. The terminal server in an average configuration is used to connect 10 dial-in/dial-out modems (including MicroART, RRS, and ASOS), four dedicated modems, a fax modem, and various console connections.

Port assignments are shown in Table L.2-1.

NOTE: The LDAD D270 server and the LDAD LAN Switch are not included in the table.

Table L.2-1. Port Assignments

<table>
<thead>
<tr>
<th>Port</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Used to dial out to devices configured for 7 bits, even parity</td>
</tr>
<tr>
<td>2</td>
<td>Used to dial out to devices configured for 8 bits, no parity</td>
</tr>
<tr>
<td>3</td>
<td>First dial-in line, 8 bits, no parity</td>
</tr>
<tr>
<td>4</td>
<td>Second dial-in line, 8 bits, no parity</td>
</tr>
<tr>
<td>5</td>
<td>Third dial-in line, 8 bits, no parity</td>
</tr>
<tr>
<td>6</td>
<td>Fourth dial-in line, 8 bits, no parity</td>
</tr>
<tr>
<td>7</td>
<td>Fifth dial-in line, 8 bits, no parity</td>
</tr>
<tr>
<td>8</td>
<td>Sixth dial-in line, 8 bits, no parity</td>
</tr>
<tr>
<td>9</td>
<td>MicroART Modem</td>
</tr>
<tr>
<td>10</td>
<td>ASOS Modem</td>
</tr>
<tr>
<td>21</td>
<td>First Dedicated Modem</td>
</tr>
<tr>
<td>22</td>
<td>Second Dedicated Modem</td>
</tr>
<tr>
<td>23</td>
<td>Third Dedicated Modem</td>
</tr>
<tr>
<td>24</td>
<td>Fourth Dedicated Modem</td>
</tr>
<tr>
<td>30</td>
<td>Fax Modem</td>
</tr>
</tbody>
</table>
The LDAD terminal server is configured with a minimum of 15 IP addresses that must be assigned in the address space in which the site LDAD resides. First, a single IP address is used to address the terminal server as a whole. Next, an individual and unique IP address is used to address each individual port from Port 1 through Port 10 and from Port 21 through Port 24. An additional IP address is required for the LDAD rack WaveSwitch.

For Port 1, the IP address is named “7e1out” within the LDAD site and is used to access a dial-out port configured properly for communications with older devices that utilize 7-bit even parity communications configurations. LARC gauge communications is one example of this type of communications.

The IP address assigned to Port 2 within the LDAD site is named “8n1out” and is used in much the same way as the Port 1 address. In this case, the modem connected to Port 2 is configured for 8-bit no parity communications. Current modem communications equipment is usually configured in this manner. One example is Campbell gauge communications.

The IP addresses assigned to Ports 3 through 10 are used by dial-in users who wish to establish direct TCP communications via PPP to the LDAD LAN.

**L.3 LDAD Modem Configurations**

LDAD modem configurations are presented in Table L.3-1.

<table>
<thead>
<tr>
<th>Category</th>
<th>Option</th>
<th>Default Setting</th>
<th>7e1out</th>
<th>8n1out</th>
<th>cpport</th>
<th>PPP/Interactive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test</td>
<td>End Test</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td>Test</td>
<td>Accept Redial</td>
<td>On</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>Test</td>
<td>LAL Busy Out</td>
<td>Off</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
</tr>
<tr>
<td>Test</td>
<td>Line Select</td>
<td>Dial</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td>Test</td>
<td>Mode</td>
<td>V.34 Auto</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td>Test</td>
<td>Auto Type</td>
<td>CCITT</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td>Test</td>
<td>Line Speed</td>
<td>Bell</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td>Test</td>
<td>Modulation</td>
<td>33.6</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td>Modulation</td>
<td>Min Rate</td>
<td>300</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td>Modulation</td>
<td>Asymmetric Rate</td>
<td>On</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td>Modulation</td>
<td>Fast Call</td>
<td>Off</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td>Modulation</td>
<td>Adaptive Rate</td>
<td>On</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td>Modulation</td>
<td>Mode Select</td>
<td>Originate</td>
<td>Default</td>
<td>Default</td>
<td>Answer</td>
<td>Answer</td>
</tr>
<tr>
<td>Modulation</td>
<td>Clock Select</td>
<td>Internal</td>
<td>Default</td>
<td>Default</td>
<td>Answer</td>
<td>Answer</td>
</tr>
<tr>
<td>Modulation</td>
<td>Retrain</td>
<td>High BER</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td>Modulation</td>
<td>Longspace Disconnect</td>
<td>Off</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td>Modulation</td>
<td>PSTN Disconnect</td>
<td>On</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td>Modulation</td>
<td>Guard Tone</td>
<td>Off</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td>Category</td>
<td>Option</td>
<td>Default Setting</td>
<td>7e1out</td>
<td>8n1out</td>
<td>csportd</td>
<td>PPP/Interactive</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------------------</td>
<td>----------------</td>
<td>--------</td>
<td>--------</td>
<td>---------</td>
<td>---------------</td>
</tr>
<tr>
<td>Restoral</td>
<td>Hold Dial Line</td>
<td>Off</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>Data Transfer Mode</td>
<td>Spd Auto Reliable</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>EC Buffer</td>
<td>Regular</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>Error Correction</td>
<td>V.42</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>Data Compression</td>
<td>Enabled</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td>EC/DC</td>
<td>Break</td>
<td>Destruct</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>Modem Flow</td>
<td>On</td>
<td>Default</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
</tr>
<tr>
<td></td>
<td>Disconnect Buffer Delay</td>
<td>Off</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>Error Correction Identifier</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>Select</td>
<td>AT</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>At Data Format</td>
<td>Async</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>V25 Format</td>
<td>Bitsync</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>NoACU Format</td>
<td>Async</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>Default Dial</td>
<td>Off</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>Manual/ Auto Answer</td>
<td>Using S0</td>
<td>Default</td>
<td>Ring #1</td>
<td>Ring #1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Async Echo</td>
<td>On</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>Char Length</td>
<td>10</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>ACU</td>
<td>V25 Char</td>
<td>ASCII</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>Sync Idle</td>
<td>Char</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>V25 Response</td>
<td>V25bis</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>Parity</td>
<td>V25</td>
<td>Even</td>
<td>Space</td>
<td>Space</td>
<td>Space</td>
</tr>
<tr>
<td></td>
<td>AT Msg</td>
<td>Before CD</td>
<td>After CD</td>
<td>After CD</td>
<td>After CD</td>
<td>After CD</td>
</tr>
<tr>
<td></td>
<td>Result Code</td>
<td>Enabled</td>
<td>Orig</td>
<td>Orig</td>
<td>Disable</td>
<td>Disable</td>
</tr>
<tr>
<td></td>
<td>Result Code Format</td>
<td>Verbose</td>
<td>Verbose</td>
<td>Verbose</td>
<td>Verbose</td>
<td>Verbose</td>
</tr>
<tr>
<td></td>
<td>DCE/DTE Rate Display</td>
<td>DTE rate</td>
<td>comm</td>
<td>msg</td>
<td>comm</td>
<td>msg</td>
</tr>
<tr>
<td></td>
<td>EC Reliable Msg</td>
<td>Off</td>
<td>Long</td>
<td>Long</td>
<td>Long</td>
<td>Long</td>
</tr>
<tr>
<td></td>
<td>LPDA2 Address</td>
<td>ff</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>LPDA2 ID</td>
<td>326x</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>LPDA2 Detection</td>
<td>ff</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>Call Progress Verbosity</td>
<td>4</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>DTE rate</td>
<td>auto</td>
<td>57.6</td>
<td>57.6</td>
<td>300/57600</td>
<td>57.6</td>
</tr>
<tr>
<td></td>
<td>Flow Control</td>
<td>XON/XOFF</td>
<td>RTS/CTS</td>
<td>RTS/CTS</td>
<td>RTS/CTS</td>
<td>RTS/CTS</td>
</tr>
<tr>
<td></td>
<td>Speed Conversion</td>
<td>On</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>DTR Control</td>
<td>High</td>
<td>Reset</td>
<td>Reset</td>
<td>Reset</td>
<td>Reset</td>
</tr>
<tr>
<td></td>
<td>RTS Control</td>
<td>High</td>
<td>Normal</td>
<td>Normal</td>
<td>Normal</td>
<td>Normal</td>
</tr>
<tr>
<td></td>
<td>CTS Control</td>
<td>Async</td>
<td>High</td>
<td>High</td>
<td>Normal</td>
<td>Normal</td>
</tr>
<tr>
<td></td>
<td>RTS/CTS Delay</td>
<td>0</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>DCD Control</td>
<td>High</td>
<td>Wink</td>
<td>Wink</td>
<td>Wink</td>
<td>Normal</td>
</tr>
<tr>
<td></td>
<td>RTS/DCD Remote Signaling</td>
<td>Codex</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>DCD Loss</td>
<td>S10</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td>Category</td>
<td>Option</td>
<td>Default Setting</td>
<td>7e1out</td>
<td>8n1out</td>
<td>csportd</td>
<td>PPP/Interactive</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------------------------</td>
<td>------------------</td>
<td>------------</td>
<td>------------</td>
<td>-------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>Disconnect</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DSR</td>
<td>High</td>
<td>Drop on disconnect</td>
<td>Drop on disconnect</td>
<td>Drop on disconnect</td>
<td>Drop on disconnect</td>
<td></td>
</tr>
<tr>
<td>Overspeed</td>
<td>1%</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>DTR delay</td>
<td>S25</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>Circuit 140 Remote Digital Loopback</td>
<td>Off</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>Circuit 141 Local Analog Loopback</td>
<td>Off</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>DTE pin 25</td>
<td>Busy</td>
<td>Test</td>
<td>Test</td>
<td>Test</td>
<td>Test</td>
<td></td>
</tr>
<tr>
<td>External Option Set Select</td>
<td>Off</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>External Control</td>
<td>Pin 14</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>Inactivity Timeout</td>
<td>S30</td>
<td>10 Mins</td>
<td>10 Mins</td>
<td>10 Mins</td>
<td>10 Mins</td>
<td></td>
</tr>
<tr>
<td>Jack Type</td>
<td>RJ11C</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>DL Tx Level</td>
<td>0</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td></td>
</tr>
<tr>
<td>LL Tx Level</td>
<td>0</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td></td>
</tr>
<tr>
<td>Telco</td>
<td>Line Compensation</td>
<td>Off</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
</tr>
<tr>
<td>Speaker Control</td>
<td>Dialing</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>Speaker Volume</td>
<td>Medium</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>Network Compensation</td>
<td>Off</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>Pause Delay</td>
<td>3 Seconds</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>Dial Wait</td>
<td>S7</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>Dial Control</td>
<td>Tone</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>Dialing</td>
<td>Call Timeout</td>
<td>60</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>Blind Dial</td>
<td>S6</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>Pulse Cycle</td>
<td>40%</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>Tone Length</td>
<td>72 msecs</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>Front Panel Security</td>
<td>Password</td>
<td>Disable</td>
<td>Default</td>
<td>Disable</td>
<td>Disable</td>
<td>Disable</td>
</tr>
<tr>
<td></td>
<td>Password Verify</td>
<td>Disable</td>
<td>Default</td>
<td>Default</td>
<td>Disable</td>
<td>Disable</td>
</tr>
<tr>
<td></td>
<td>Callback</td>
<td>Off</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Remote Num Required</td>
<td>Off</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>Access Security</td>
<td>Group Password</td>
<td>Disable</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>Tone</td>
<td>None</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Sim Ringback</td>
<td>Disable</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Dial Restricted</td>
<td>Off</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Override mode</td>
<td>Off</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NC Address</td>
<td>000</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>Network Control</td>
<td>NC Port Rate</td>
<td>75</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Pass Through</td>
<td>option1</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NC Line Disconnect</td>
<td>Off</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>Remote Configuration</td>
<td>Remote Access</td>
<td>Enabled</td>
<td>Disable</td>
<td>Disable</td>
<td>Disable</td>
<td>Disable</td>
</tr>
<tr>
<td></td>
<td>Dial Restricted</td>
<td>Off</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>Category</td>
<td>Option</td>
<td>Default Setting</td>
<td>7e1out</td>
<td>8n1out</td>
<td>csportd</td>
<td>PPP/Interactive</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------------------------------------</td>
<td>-----------------</td>
<td>--------</td>
<td>--------</td>
<td>---------</td>
<td>----------------</td>
</tr>
<tr>
<td></td>
<td>Ring Count to Answer On</td>
<td>0:0</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>Ring Count</td>
<td>1:0</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>Escape Code Character</td>
<td>2:43</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>Carriage Return Character</td>
<td>3:13</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>Line Feed Character</td>
<td>4:10</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>Back Space Character</td>
<td>5:8</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>Wait for Dialtone</td>
<td>6:2</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>Wait for Data Carrier</td>
<td>7:30</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>S Register</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Pause Delay Value</td>
<td>8:2</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>Carrier Loss Disconnect Delay</td>
<td>10:15</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>DTMF Tone Duration</td>
<td>11:72</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>Escape Code Guard Time</td>
<td>12:50</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>Test Timer</td>
<td>18:0</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>DTR Before Looking Fort DTR</td>
<td>25:5</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>RTS/CTS Delay</td>
<td>26:1</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>DTE Inactivity Disconnect</td>
<td>30:0</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>Disconnect Buffer Delay</td>
<td>38:5</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>Access Security Tone Duration</td>
<td>45:5</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>Access Security Lead Digit Delay Timeout</td>
<td>46:12</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>AC Detect</td>
<td>98:0</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>V.32 Training Time</td>
<td>99:0</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
<td>Default</td>
</tr>
</tbody>
</table>

### L.4 Environment Variables

The LDAD environment variable values in Table L.4-1 use the Honolulu WFO for site-specific variables such as DEFAULT_WFO and are valid for user ldad on PX2.
### Table L.4-1. Environment Variables

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>APPS_DEFAULTS_USER</td>
<td>/awips/hydroapps/.Apps_defaults</td>
</tr>
<tr>
<td>CLASSPATH</td>
<td>/awips/ldad/classes</td>
</tr>
<tr>
<td>DEFAULT RFC</td>
<td>ANC</td>
</tr>
<tr>
<td>DEFAULT_SITE</td>
<td>HNL</td>
</tr>
<tr>
<td>DEFAULT_STATE</td>
<td>HI</td>
</tr>
<tr>
<td>FILE_SERVER_DEFAULT_PATHS</td>
<td>+/-awips/ldad/data/</td>
</tr>
<tr>
<td>FXA_DATA</td>
<td>/data/fxa</td>
</tr>
<tr>
<td>FXA_HOME</td>
<td>/awips/fxa</td>
</tr>
<tr>
<td>FXA_LOCAL_SITE</td>
<td>HFO</td>
</tr>
<tr>
<td>FXA_NATL_CONFIG_DATA</td>
<td>/awips/fxa/data/localization</td>
</tr>
<tr>
<td>LDAD_DATA</td>
<td>/data/fxa/LDAD</td>
</tr>
<tr>
<td>LDAD_DECODED_DATA</td>
<td>/data/fxa/LDAD/decoded</td>
</tr>
<tr>
<td>LDAD_DECODER_SOURCE_DIR</td>
<td>/awips/ldad/src/decoder</td>
</tr>
<tr>
<td>LDAD_EXTERNAL_BIN</td>
<td>/ldad/bin</td>
</tr>
<tr>
<td>LDAD_EXTERNAL_DATA</td>
<td>/data/ldad</td>
</tr>
<tr>
<td>LDAD_EXTERNAL_DATA_REPOSITORY</td>
<td>/data/ldad/hmIngest</td>
</tr>
<tr>
<td>LDAD_EXTERNAL_HOME</td>
<td>/ldad</td>
</tr>
<tr>
<td>LDAD_EXTERNAL_HOST</td>
<td>ls1</td>
</tr>
<tr>
<td>LDAD_EXTERNAL_LOGDIR</td>
<td>/data/logs/ldad</td>
</tr>
<tr>
<td>LDAD_EXTERNAL_PUBLIC</td>
<td>/data/ldad/public</td>
</tr>
<tr>
<td>LDAD_GATEWAY_SOURCE_DIR</td>
<td>/awips/ldad/src/gateway</td>
</tr>
<tr>
<td>LDAD_HOME</td>
<td>/awips/ldad</td>
</tr>
<tr>
<td>LDAD_INCOMING</td>
<td>/data/Incoming</td>
</tr>
<tr>
<td>LDAD_INTERNAL_BIN</td>
<td>/awips/ldad/bin</td>
</tr>
<tr>
<td>LDAD_INTERNAL_CRONTABFILE</td>
<td>/awips/ldad/bin/LDAD/crontab</td>
</tr>
<tr>
<td>LDAD_INTERNAL_CRONTABS</td>
<td>/awips/ldad/bin/crontabs</td>
</tr>
<tr>
<td>LDAD_INTERNAL_DATA</td>
<td>/awips/ldad/data</td>
</tr>
<tr>
<td>LDAD_INTERNAL_HOME</td>
<td>/awips/ldad</td>
</tr>
<tr>
<td>LDAD_INTERNAL_HOST</td>
<td>PX</td>
</tr>
<tr>
<td>LDAD_INTERNAL_LOGDIR</td>
<td>/awips/ldad/logs</td>
</tr>
<tr>
<td>LDAD_INTERNAL_SOURCE_DIR</td>
<td>/awips/ldad/src</td>
</tr>
<tr>
<td>LDAD_INTERNAL_SYSTEM_CRONTABFILE</td>
<td>/usr/spool/cron/crontabs/ldad</td>
</tr>
<tr>
<td>LDAD_LOCAL_TZ</td>
<td>HST10</td>
</tr>
<tr>
<td>LDAD_RAWDATA</td>
<td>/data/fxa/LDAD/Raw</td>
</tr>
<tr>
<td>LDAD_STORAGE_SOURCE_DIR</td>
<td>/awips/ldad/src/decoder/storage</td>
</tr>
<tr>
<td>LOG_DIR</td>
<td>/data/logs/ldad</td>
</tr>
<tr>
<td>LOG_PREF</td>
<td>/awips/ldad/.logPref</td>
</tr>
<tr>
<td>NEWGATEWAY_DB</td>
<td>ldadgateway</td>
</tr>
<tr>
<td>SQLEXEC</td>
<td>sqllrm</td>
</tr>
<tr>
<td>UDUNITS_PATH</td>
<td>/awips/ldad/data</td>
</tr>
</tbody>
</table>

### L.5 AWIPS Firewall Architecture and Configuration

AWIPS has upgraded the Local Data Acquisition and Dissemination (LDAD) network firewalls with redundant SSG320M firewalls to enhance AWIPS network security. Each AWIPS site has two firewalls controlled by central configuration manager servers. The central configuration management servers are located at the Network Control Facility.
(NCF) and the Backup NCF. The Security manager clients are located at the NCF and the Raytheon engineering facility in Silver Spring, MD to support the deployment. As part of the deployment, each regional headquarters will have a Netscreen Security Manager (NSM) client. The regional NSM clients will allow the regions to view the configurations of the firewalls at each of their AWIPS sites. Control over the firewall configurations is tiered, with the central configuration management server having ultimate control; the regional NSM clients can view their sites but they are managed from the central server. Individual AWIPS sites (forecast offices) will not have direct control over their own configurations.

This section describes the architecture and functionality of the new AWIPS LDAD SSG 320M firewalls.

### L.5.1 AWIPS Firewall Architecture

The new AWIPS Site Firewall architecture sets up three zones:

- **Trusted** – inside the AWIPS network
- **Untrusted** – outside AWIPS
- **DMZ** – LDAD server – 1pair- (ls1).

The DMZ is an additional zone where the NOAA ls1 server is placed. The addition of the DMZ zone solves many issues and adds network security to the architecture.

A limitation of the firewall is that for each internal host on the site AWIPS LAN requiring rlogin/rcp/rsh to the ls1, a policy rule must be created on the firewall. Each such rule for a Trust-to-DMZ host pair uses a unique DMZ IP address for source address translation. This need for a 1:1 address translation from internal AWIPS hosts to the ls1 to handle r-commands would have depleted the pool of available IP addresses in the site local LAN had the DMZ not been added. By creating a DMZ using private addresses, the IP address pool is virtually unlimited (nearly 255 available IP addresses) and the site’s local LAN address space is not affected. Additionally, by moving the ls1 server to the DMZ, the ls1 is made potentially more secure from attacks emanating from the site LAN Untrusted zone.

### L.5.2 Firewall Configuration

The new firewalls are configured at AWIPS sites in the following way:

- Ethernet Port 1 on the Firewall connects to the Trust zone (165.92.x.x).
- Ethernet Port 2 connects to the DMZ zone (192.168.1.x).
- Ethernet Port 3 connects to the Untrust zone (site local LAN).
- Ethernet Port 4 connects both firewalls for High Availability (failover).
• ls1-<siteid> IP is a new IP address 192.168.1.10
  3/21/06 B-4

• Source NAT for all **Trust to DMZ** traffic using **DIPs** (dynamic IP address).

• Destination NAT for **DMZ to Trust** (high-port tcp).

• Destination NAT for **Untrust to DMZ** using **MIPs** (mapped IP addresses). [MIPs are similar to proxy arp rules. The Untrust to DMZ interface will proxy arp for the old (original) ls1 IP address, so when ls1 traffic arrives at that interface, the destination address is translated to the new ls1 192.168.1.10 address. Proxy arping is a common technique used on firewalls or routers to make a host (ls1) on one side of the firewall/router appear to be part of the same network as hosts on the other side (site local LAN) of the firewall/router.]

• The MIP is bi-directional so outgoing traffic (DMZ to Untrust) will be handled similarly (192.168.1.10 is translated to the original ls1 IP address).

**L.5.3 LDAD LAN (DMZ) Configuration**

The LDAD LAN Configuration can be altered as follows:

1. Change ls1 IP to 192.168.1.10.
2. Change gateway to 192.168.1.1.
3. Reconfigure EMDS web server IP address.
4. Add new 192.168.1.x IP addresses to /etc/hosts table for each DIP assigned to an internal 165.92 host. (Needed for rsh/rlogin.)
6. For each internal AWIPS host that is permitted to rsh/rlogin to the ls1, add the hostname to root’s and ldad’s rhosts file on ls1.
7. Remove all 165.92 addresses from ls1 hosts table and routing table. **Note:** Aside from ls1, nothing else on the LDAD LAN will be reconfigured during the initial firewall deployment. The LDAD terminal server and modems will function normally.

**L.5.4 AWIPS LAN Reconfiguration**

The AWIPS LAN Reconfiguration can be done by adding new ls1 IP address to dx1 hosts table and then pushed out for NIS. On all internal hosts that access ls1, replace the old route to ls1 with a new route to the DMZ (new ls1 DMZ IP address).
L.5.5 Interconnectivity Between the Firewall Zones and Security Policies (Firewall Rules)

The AWIPS firewall architecture provides for three different zones (Trust, Untrust, and DMZ) and six rules (firewall) as identified below have been established to govern connectivity between two zones:

- **Trust to DMZ**
  - **Two rules** for high-port tcp traffic from DX1 to ls1 (one rule for dx1 and dx2). Uses fixed-port DIP for source address (165.92) translation.
  - **One rule** for ftp/telnet/ssh/http from all internal AWIPS hosts to ls1 (uses egress interface for source address translation).

  **Note:** For each internal AWIPS host that requires rsh/rcp/rlogin to ls1, there is one rsh/rcp/rlogin rule for that host to the ls1 (uses dedicated fixed-port DIP for source address translation). For example, if the site AWIPS LAN has 20 such hosts, then 20 DIPS are needed for 20 policy rules.

- **Trust to Untrust**
  - **One rule:** All internal hosts can perform outgoing ftp/telnet/ssh. Uses egress.

- **DMZ to Untrust**
  - **One rule:** All DMZ (ls1) to Untrust traffic allowed.

- **Untrust to DMZ**
  - **One rule:** MIP rule used for destination translation of old (original) ls1 IP address to the new 192.168.1.10 DMZ address. Proxy arp rule. Standard Firewall Ruleset.

Table L.5.5-1 illustrates the default firewall ruleset.

<table>
<thead>
<tr>
<th>Policy</th>
<th>From Zone</th>
<th>To Zone</th>
<th>Source</th>
<th>Destination</th>
<th>Service</th>
<th>Action</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Trust</td>
<td>Untrust</td>
<td>SAWIPS_NET/SITE</td>
<td>Any</td>
<td>FTP/Telnet/SSH/HTTP</td>
<td>permit</td>
<td>nat src permit log count</td>
</tr>
<tr>
<td>51</td>
<td>Trust</td>
<td>Untrust</td>
<td>SAWIPS_NET/SITE</td>
<td>Any</td>
<td>LDM</td>
<td>permit</td>
<td>nat src permit log count</td>
</tr>
<tr>
<td>52</td>
<td>Trust</td>
<td>Untrust</td>
<td>SAWIPS_NET/SITE</td>
<td>Any</td>
<td>Terminal Server</td>
<td>permit</td>
<td>nat src permit log count</td>
</tr>
<tr>
<td>2</td>
<td>Trust</td>
<td>Untrust</td>
<td>Any</td>
<td>Any</td>
<td>ANY</td>
<td>deny</td>
<td>deny log alert count</td>
</tr>
<tr>
<td>3</td>
<td>Untrust</td>
<td>Trust</td>
<td>Any</td>
<td>Any</td>
<td>ANY</td>
<td>deny</td>
<td>deny log alert count</td>
</tr>
<tr>
<td>4</td>
<td>DMZ</td>
<td>Trust</td>
<td>ls1-dmz</td>
<td>MIP(192.168.1.50)</td>
<td>tcp15008</td>
<td>permit</td>
<td>permit log count</td>
</tr>
<tr>
<td>49</td>
<td>DMZ</td>
<td>Trust</td>
<td>ls1-servers-dmz</td>
<td>MIP(192.168.1.51)</td>
<td>tcp15008</td>
<td>permit</td>
<td>permit log count</td>
</tr>
<tr>
<td>5</td>
<td>DMZ</td>
<td>Trust</td>
<td>Any</td>
<td>Any</td>
<td>ANY</td>
<td>deny</td>
<td>deny log count</td>
</tr>
<tr>
<td>6</td>
<td>Trust</td>
<td>DMZ</td>
<td>dx1-SSITE</td>
<td>ls1-dmz</td>
<td>tcp15007</td>
<td>permit</td>
<td>nat src dip-id 31 permit log count</td>
</tr>
<tr>
<td>Policy</td>
<td>From Zone</td>
<td>To Zone</td>
<td>Source</td>
<td>Destination</td>
<td>Service</td>
<td>Action</td>
<td>Details</td>
</tr>
<tr>
<td>--------</td>
<td>-----------</td>
<td>-------------</td>
<td>---------------</td>
<td>-------------</td>
<td>-----------------</td>
<td>-------------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td>7</td>
<td>Trust</td>
<td>DMZ</td>
<td>dx2-SITE</td>
<td>ls1-dmz</td>
<td>tcp15007</td>
<td>permit</td>
<td>nat src dip-id 32 permit log count</td>
</tr>
<tr>
<td>53</td>
<td>Trust</td>
<td>DMZ</td>
<td>$SITE px Servers</td>
<td>ls1-servers-dmz</td>
<td>tcp15007</td>
<td>permit</td>
<td>nat src dip-id 41 permit log count</td>
</tr>
<tr>
<td>8</td>
<td>Trust</td>
<td>DMZ</td>
<td>SAWIPS_NET/ SITE</td>
<td>ls1-servers-dmz</td>
<td>FTP/Telnet/SSH/ HTTP</td>
<td>permit</td>
<td>nat src permit log count</td>
</tr>
<tr>
<td>9</td>
<td>Trust</td>
<td>DMZ</td>
<td>dx1-SITE</td>
<td>ls1-dmz</td>
<td>NTP</td>
<td>permit</td>
<td>nat src dip-id 40 permit log count</td>
</tr>
<tr>
<td>10</td>
<td>Trust</td>
<td>DMZ</td>
<td>dx2-SITE</td>
<td>ls1-dmz</td>
<td>NTP</td>
<td>permit</td>
<td>nat src dip-id 40 permit log count</td>
</tr>
<tr>
<td>11</td>
<td>Trust</td>
<td>DMZ</td>
<td>dx1-SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 4 permit log count</td>
</tr>
<tr>
<td>12</td>
<td>Trust</td>
<td>DMZ</td>
<td>dx2-SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 5 permit log count</td>
</tr>
<tr>
<td>13</td>
<td>Trust</td>
<td>DMZ</td>
<td>dx1-SITE</td>
<td>ls1-servers-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 6 permit log count</td>
</tr>
<tr>
<td>14</td>
<td>Trust</td>
<td>DMZ</td>
<td>dx2-SITE</td>
<td>ls1-servers-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 7 permit log count</td>
</tr>
<tr>
<td>54</td>
<td>Trust</td>
<td>DMZ</td>
<td>dx3-SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 42 permit log count</td>
</tr>
<tr>
<td>55</td>
<td>Trust</td>
<td>DMZ</td>
<td>dx4-SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 43 permit log count</td>
</tr>
<tr>
<td>15</td>
<td>Trust</td>
<td>DMZ</td>
<td>px1-SITE</td>
<td>ls1-servers-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 8 permit log count</td>
</tr>
<tr>
<td>16</td>
<td>Trust</td>
<td>DMZ</td>
<td>px2-SITE</td>
<td>ls1-servers-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 9 permit log count</td>
</tr>
<tr>
<td>17</td>
<td>Trust</td>
<td>DMZ</td>
<td>ax-SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 10 permit log count</td>
</tr>
<tr>
<td>19</td>
<td>Trust</td>
<td>DMZ</td>
<td>lx1-SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 11 permit log count</td>
</tr>
<tr>
<td>20</td>
<td>Trust</td>
<td>DMZ</td>
<td>lx2-SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 12 permit log count</td>
</tr>
<tr>
<td>21</td>
<td>Trust</td>
<td>DMZ</td>
<td>lx3-SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 13 permit log count</td>
</tr>
<tr>
<td>22</td>
<td>Trust</td>
<td>DMZ</td>
<td>lx4-SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 14 permit log count</td>
</tr>
<tr>
<td>23</td>
<td>Trust</td>
<td>DMZ</td>
<td>lx5-SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 15 permit log count</td>
</tr>
<tr>
<td>24</td>
<td>Trust</td>
<td>DMZ</td>
<td>lx6-SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 16 permit log count</td>
</tr>
<tr>
<td>40</td>
<td>Trust</td>
<td>DMZ</td>
<td>lx7-SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 17 permit log count</td>
</tr>
<tr>
<td>41</td>
<td>Trust</td>
<td>DMZ</td>
<td>lx8-SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 18 permit log count</td>
</tr>
<tr>
<td>42</td>
<td>Trust</td>
<td>DMZ</td>
<td>lx9-SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 19 permit log count</td>
</tr>
<tr>
<td>43</td>
<td>Trust</td>
<td>DMZ</td>
<td>lxa-SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 20 permit log count</td>
</tr>
<tr>
<td>44</td>
<td>Trust</td>
<td>DMZ</td>
<td>lxb-SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 21 permit log count</td>
</tr>
<tr>
<td>Policy</td>
<td>From Zone</td>
<td>To Zone</td>
<td>Source</td>
<td>Destination</td>
<td>Service</td>
<td>Action</td>
<td>Details</td>
</tr>
<tr>
<td>--------</td>
<td>-----------</td>
<td>---------</td>
<td>--------</td>
<td>-------------</td>
<td>---------</td>
<td>--------</td>
<td>---------</td>
</tr>
<tr>
<td>25</td>
<td>DMZ</td>
<td>Untrust</td>
<td>$DMZ_NET/dmz</td>
<td>Any</td>
<td>ANY</td>
<td>permit</td>
<td>permit log count</td>
</tr>
<tr>
<td>26</td>
<td>Untrust</td>
<td>DMZ</td>
<td>Any</td>
<td>MIP($LS1)</td>
<td>ANY</td>
<td>permit</td>
<td>permit log count</td>
</tr>
<tr>
<td>55</td>
<td>DMZ</td>
<td>Untrust</td>
<td>$DMZ_NET/dmz</td>
<td>Any</td>
<td>esportd</td>
<td>permit</td>
<td>permit log count</td>
</tr>
<tr>
<td>27</td>
<td>Untrust</td>
<td>DMZ</td>
<td>Any</td>
<td>Any</td>
<td>ANY</td>
<td>deny</td>
<td>deny log count</td>
</tr>
<tr>
<td>28</td>
<td>DMZ</td>
<td>Untrust</td>
<td>Any</td>
<td>Any</td>
<td>ANY</td>
<td>deny</td>
<td>deny log count</td>
</tr>
<tr>
<td>29</td>
<td>Trust</td>
<td>DMZ</td>
<td>xt1-$SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 22 permit log count</td>
</tr>
<tr>
<td>30</td>
<td>Trust</td>
<td>DMZ</td>
<td>xt2-$SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 23 permit log count</td>
</tr>
<tr>
<td>31</td>
<td>Trust</td>
<td>DMZ</td>
<td>xt3-$SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 24 permit log count</td>
</tr>
<tr>
<td>32</td>
<td>Trust</td>
<td>DMZ</td>
<td>xt4-$SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 25 permit log count</td>
</tr>
<tr>
<td>33</td>
<td>Trust</td>
<td>DMZ</td>
<td>xt5-$SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 26 permit log count</td>
</tr>
<tr>
<td>34</td>
<td>Trust</td>
<td>DMZ</td>
<td>xt6-$SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 27 permit log count</td>
</tr>
<tr>
<td>35</td>
<td>Trust</td>
<td>DMZ</td>
<td>xt7-$SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 28 permit log count</td>
</tr>
<tr>
<td>36</td>
<td>Trust</td>
<td>DMZ</td>
<td>xt8-$SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 29 permit log count</td>
</tr>
<tr>
<td>37</td>
<td>Trust</td>
<td>DMZ</td>
<td>xt9-$SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 30 permit log count</td>
</tr>
<tr>
<td>38</td>
<td>Trust</td>
<td>DMZ</td>
<td>xta-$SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 31 permit log count</td>
</tr>
<tr>
<td>39</td>
<td>Trust</td>
<td>DMZ</td>
<td>xtb-$SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 32 permit log count</td>
</tr>
<tr>
<td>45</td>
<td>Trust</td>
<td>DMZ</td>
<td>px3-$SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 35 permit log count</td>
</tr>
<tr>
<td>46</td>
<td>Trust</td>
<td>DMZ</td>
<td>px4-$SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 36 permit log count</td>
</tr>
<tr>
<td>47</td>
<td>Trust</td>
<td>DMZ</td>
<td>rp1-$SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 37 permit log count</td>
</tr>
<tr>
<td>48</td>
<td>Trust</td>
<td>DMZ</td>
<td>rp2-$SITE</td>
<td>ls1-dmz</td>
<td>RSH/RCP/RLOGIN</td>
<td>permit</td>
<td>nat src dip-id 38 permit log count</td>
</tr>
<tr>
<td>50</td>
<td>Trust</td>
<td>DMZ</td>
<td>Any</td>
<td>Any</td>
<td>ANY</td>
<td>deny</td>
<td>deny log count</td>
</tr>
<tr>
<td>63</td>
<td>DMZ</td>
<td>Trust</td>
<td>ls1-servers-dmz</td>
<td>dx1-dmz</td>
<td>RSHD</td>
<td>permit</td>
<td>nat dst DX1IP permit log count</td>
</tr>
<tr>
<td>64</td>
<td>DMZ</td>
<td>Trust</td>
<td>ls1-servers-dmz</td>
<td>dx2-dmz</td>
<td>RSHD</td>
<td>permit</td>
<td>nat dst DX2IP permit log count</td>
</tr>
<tr>
<td>65</td>
<td>DMZ</td>
<td>Trust</td>
<td>ls1-servers-dmz</td>
<td>px1-dmz</td>
<td>RSHD</td>
<td>permit</td>
<td>nat dst PX1IP permit log count</td>
</tr>
<tr>
<td>66</td>
<td>DMZ</td>
<td>Trust</td>
<td>ls1-servers-dmz</td>
<td>px2-dmz</td>
<td>RSHD</td>
<td>permit</td>
<td>nat dst PX2IP permit log count</td>
</tr>
</tbody>
</table>
L.5.6 LDAD FAX Troubleshooting

If there seems to be a problem with the fax queue filling up with unsent faxes (the transmission relies on the firewall), run the faxstat command on the active LS. It is a good sign if the fax queue status is “Running and idle” or “Initializing server” or something like “Sending job 13...”. However, it is likely a bad sign if the fax queue status is “Waiting for modem to come ready...”.

Sometimes the “Waiting for modem to come ready” message appears even in normal operation; however, if the status line continues to say that for several minutes without change, the modem should be re-seated. From the front of the LDAD modem nest, open the panel and locate modem 14; then press on the top and bottom tabs. Once the modem is ejected, firmly press the modem board back into the slot.

L.5.7 LDAD Firewall Site Configuration Management

All the sites except Raytheon Testbed Silver Spring (ANCF) and Raytheon Testbed Silver Spring (BNCF) are equipped with AWIPS LDAD Firewall Juniper SSG 320M.

- ANCF Raytheon Testbed Silver Spring: AWIPS LDAD Firewall Juniper NSM BASE LIC
- BNCF Raytheon Testbed Silver Spring: AWIPS LDAD Firewall Juniper NSMXPRES HA.

L.5.8 Csportd: Configuration and Setup

In order to make the LDAD server send and receive data through the terminal server, csportd has been used to serve as a utility that gives users connectivity between the LDAD server and terminal server ports. csportd reads data from stdin, a pseudo terminal, and sends the data to the terminal server port. Likewise, data can be read from the terminal server port and sent to the pseudo terminal.

You will find the csportd executable in /usr/bin, but to start up csportd properly, you need to configure it.

Follow these instructions, based on /ldad/bin/ltsConfig.<siteID> using ports 5 and 7 for ASOS and port 30 for fax:

Example of ltsConfig.wncf:

```plaintext
# Filename: ltsConfig.template
#
#   Example configuration file for site WNCF.
#
# fqdn:   ltserv-wncf.wncf.noaa.gov
ipaddress:  10.201.6.220
subnetMask: 255.255.255.0
broadcastAddress:10.201.6.255
```

Appendix L  

LDAD Configuration Samples and Firewall Architecture

gateway: 10.201.6.4

dns: 10.201.6.1

rloginServer: 10.201.6.216

radiusServer: 10.201.6.216

<table>
<thead>
<tr>
<th>Port</th>
<th>IfType</th>
<th>Misc</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7e1out</td>
<td>10.201.6.228</td>
</tr>
<tr>
<td>2</td>
<td>8n1out</td>
<td>10.201.6.229</td>
</tr>
<tr>
<td># 3</td>
<td>csportd</td>
<td>ASOS</td>
</tr>
<tr>
<td># 4</td>
<td>pppInteractive</td>
<td>10.201.6.231</td>
</tr>
<tr>
<td>5</td>
<td>csportd</td>
<td>uArt</td>
</tr>
<tr>
<td>6</td>
<td>pppInteractive</td>
<td>10.201.6.230</td>
</tr>
<tr>
<td>7</td>
<td>csportd</td>
<td>ASOS</td>
</tr>
<tr>
<td># 8</td>
<td>csportd</td>
<td>ASOS</td>
</tr>
<tr>
<td># 9</td>
<td>csportd</td>
<td>ASOS</td>
</tr>
<tr>
<td>#10</td>
<td>csportd</td>
<td>ASOS</td>
</tr>
<tr>
<td>#11</td>
<td>csportd</td>
<td>ASOS</td>
</tr>
<tr>
<td>#12</td>
<td>csportd</td>
<td>ASOS</td>
</tr>
<tr>
<td>#13</td>
<td>csportd</td>
<td>ASOS</td>
</tr>
<tr>
<td>#14</td>
<td>csportd</td>
<td>ASOS</td>
</tr>
<tr>
<td>#15</td>
<td>csportd</td>
<td>dtmf</td>
</tr>
<tr>
<td>#16</td>
<td>csportd</td>
<td>dtmf</td>
</tr>
<tr>
<td>#17</td>
<td>csportd</td>
<td>dtmf</td>
</tr>
<tr>
<td>#18</td>
<td>csportd</td>
<td>dtmf</td>
</tr>
<tr>
<td>#19</td>
<td>csportd</td>
<td>dtmf</td>
</tr>
<tr>
<td>#20</td>
<td>csportd</td>
<td>dtmf</td>
</tr>
<tr>
<td>#21</td>
<td>pppInteractive</td>
<td></td>
</tr>
<tr>
<td>#22</td>
<td>pppInteractive</td>
<td></td>
</tr>
<tr>
<td>#23</td>
<td>pppInteractive</td>
<td></td>
</tr>
<tr>
<td>#24</td>
<td>csportd</td>
<td>ASOS</td>
</tr>
<tr>
<td>#25</td>
<td>csportd</td>
<td>uArt_DTE</td>
</tr>
<tr>
<td>#26</td>
<td>csportd</td>
<td>DirectASOS</td>
</tr>
<tr>
<td>#27</td>
<td>csportd</td>
<td>PCROSA</td>
</tr>
<tr>
<td>28</td>
<td>console</td>
<td>LDAD_Sw_Pnl</td>
</tr>
<tr>
<td>29</td>
<td>console</td>
<td>11sw-wncf</td>
</tr>
<tr>
<td>30</td>
<td>csportd</td>
<td>fax</td>
</tr>
<tr>
<td>31</td>
<td>console</td>
<td>ls2-wncf</td>
</tr>
<tr>
<td>32</td>
<td>console</td>
<td>ls3-wncf</td>
</tr>
</tbody>
</table>

1. Stop csportd on the LS, as user root.
   `/etc/ha.d/resource.d/csportd stop`

   Example:

   ```bash
   csportd -d1 -T/dev/faxmodem -s -i ltserv 30 &
sleep 1
   chown uucp:uucp /dev/faxmodem
   chwrdev /dev/faxmodem
   csportd -d1 -T/dev/suamodem1 -s -i -D ltserv 5 &
sleep 1
   chwrdev /dev/suamodem1
   ```
   Example:
   
   `wncf 5 7`

   **Note:** There should only be one uncommented line in this file, specifying the site ID and LTS ports associated to `csportd` interfaces (ASOS or uART). The `csportd` interface for the fax does not need to be specified here.

   Example:
   
   `faxmodem suamodem1 suamodem2`

   **Note:** The pseudo devices listed in this file need to match up with the LTServ ports in the `/ldad/data/csportd_control` file.

After updating the `csportd` config files, the LDAD processes will need to be restarted, as well:

```
su - ldad
stopLDADexternal.sh
exit
```

as user `root`,

```
/etc/ha.d/resource.d/csportd start
```

```
su - ldad
startLDADexternal.csh
```
Appendix M
Acceptable Formats for Importing Climate Data
Appendix M. Acceptable Formats for Importing Climate Data

Table of Contents

M.1 Data Formats Accepted ........................................................................................................ 1
M.2 General Format ................................................................................................................ 1
M.3 Contents of File ............................................................................................................... 2
  M.3.1 Daily Data ................................................................................................................. 2
  M.3.2 Station Data .............................................................................................................. 2
  M.3.3 Monthly Data ............................................................................................................. 4

List of Exhibits

Exhibit M.2-1. Example 1: Acceptable General Format for Climate Data ......................... 1
Exhibit M.2-2. Example 2: Acceptable General Format for Climate Data ......................... 1
Exhibit M.3.1-1. Example of Bad Format for Daily Climate Data ........................................ 2
Exhibit M.3.1-2. Climate Error Message for Invalid Data ..................................................... 2
Exhibit M.3.2-1. Example 1: Acceptable Format for Multiple Station Names ..................... 3
Exhibit M.3.2-2. Example 2: Acceptable Format for Multiple Station Names ..................... 3
Exhibit M.3.2-3. Example of Acceptable Format for Single Station Name ......................... 3
Exhibit M.3.3-1. Examples of Acceptable Format for Single and Multiple Months of Data .... 4
Exhibit M.3.3-2. Example of Acceptable Format for Multiple Months of Data in MM/DD Format .......................................................... 5
Exhibit M.3.3-3. Acceptable Format for Single Month Data ................................................. 5
Exhibit M.3.3-4. Acceptable Month First Format ................................................................. 5
Exhibit M.3.3-5. Various Acceptable Month First Combinations ...................................... 6
Exhibit M.3.3-6. Various Acceptable Day First Combinations ......................................... 6
M.1 Data Formats Accepted

When importing existing climatology data files into the Daily Normals, Means, and Extremes database, it is important to know the format of the import file(s). Climate will only recognize the limited variety of file formats that are specified below. If the format for the import file is not listed, the Import Climate routine will not execute properly.

M.2 General Format

Headers

The data file can contain any number of text headers or text lines within the body of the data. Headers between sections of the data are acceptable as well.

Columns and Rows

The data portion must be arranged in columnar format with one record per row. A record refers to a row of meteorological variables along with a corresponding date. The date can appear anywhere within the row, and rows can continue as long as needed. Exhibits M.2-1 and M.2-2 display acceptable formats.

<table>
<thead>
<tr>
<th>Date</th>
<th>nh</th>
<th>nl</th>
<th>rh</th>
<th>rl</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>63</td>
<td>45</td>
<td>78</td>
<td>12</td>
</tr>
<tr>
<td>02</td>
<td>63</td>
<td>45</td>
<td>83</td>
<td>14</td>
</tr>
<tr>
<td>03</td>
<td>62</td>
<td>44</td>
<td>80</td>
<td>10</td>
</tr>
<tr>
<td>04</td>
<td>62</td>
<td>44</td>
<td>79</td>
<td>21</td>
</tr>
</tbody>
</table>

Exhibit M.2-1. Example 1: Acceptable General Format for Climate Data

or

<table>
<thead>
<tr>
<th>rh</th>
<th>rhd</th>
<th>rl</th>
<th>rld</th>
<th>pcp</th>
<th>date</th>
</tr>
</thead>
<tbody>
<tr>
<td>78</td>
<td>1979</td>
<td>12</td>
<td>1933</td>
<td>0.14</td>
<td>01</td>
</tr>
<tr>
<td>83</td>
<td>1883</td>
<td>14</td>
<td>1965</td>
<td>0.14</td>
<td>02</td>
</tr>
<tr>
<td>80</td>
<td>1982</td>
<td>10</td>
<td>1967</td>
<td>0.14</td>
<td>03</td>
</tr>
<tr>
<td>79</td>
<td>1945</td>
<td>21</td>
<td>1993</td>
<td>0.14</td>
<td>04</td>
</tr>
</tbody>
</table>

Exhibit M.2-2. Example 2: Acceptable General Format for Climate Data

Delimiters

Delimiters within the data rows can be any symbol except dashes (-), periods (.), or tabs. The program assumes three default delimiters: spaces, asterisks (*), and slashes (/). Any other delimiters must be entered by the user.
M.3 Contents of File

M.3.1 Daily Data

The data file should contain only daily data or only monthly data. In a case where a file contains both, the program is still capable of importing the type of data (daily or monthly) that the user has specified. When it reaches a line that does not match the format, an error is generated. Exhibit M.3.1-1 is an example of a day and month data within the same file.

<table>
<thead>
<tr>
<th>day</th>
<th>rec max</th>
<th>rec min</th>
<th>avg pre</th>
<th>rec pre</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>79</td>
<td>42</td>
<td>0.24</td>
<td>1.36</td>
</tr>
<tr>
<td>31</td>
<td>81</td>
<td>38</td>
<td>0.25</td>
<td>1.15</td>
</tr>
</tbody>
</table>

**Monthly Totals**

<table>
<thead>
<tr>
<th>Avg Max</th>
<th>Avg Min</th>
<th>Avg Precip</th>
</tr>
</thead>
<tbody>
<tr>
<td>70.6</td>
<td>49.8</td>
<td>3.29</td>
</tr>
</tbody>
</table>

**Exhibit M.3.1-1. Example of Bad Format for Daily Climate Data**

Exhibit M.3.1-2 shows the error message window resulting from the file above that contains both daily and monthly data. If the user chooses “daily” for data type, the file properly reads data for Days 30 and 31 and the data are stored in the database. The error message is a result of having both the day and month data in the file.

![Exhibit M.3.1-2. Climate Error Message for Invalid Data](image)

The program recognizes that the particular data in the file is not as specified and warns the forecaster of the discovery. Choosing **Continue** at this point simply allows the reading of this file to continue (if lines remain to be read). **Cancel Import** breaks the read of this file and continues to the next file in the import list, if other files exist.

M.3.2 Station Data

The data file may contain data for any number of stations contained in the station list. All stations in the data file must be in the station database.
Multiple Stations

If there is more than one station in the file, the station names or their four-letter ICAO IDs must either appear in the text headers separating each station’s data or within the data rows themselves. Exhibits M.3.2-1 and M.3.2-2 display acceptable formats regarding the station names.

### PITTSBURGH

<table>
<thead>
<tr>
<th>January</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>date</td>
<td>nh</td>
</tr>
<tr>
<td>30</td>
<td>35</td>
</tr>
<tr>
<td>31</td>
<td>35</td>
</tr>
</tbody>
</table>

**Charleston**

| 30      | 36   | 27  | 72   | -10  |
| 31      | 36   | 37  | 75   | -8   |

Exhibit M.3.2-1. Example 1: Acceptable Format for Multiple Station Names

or

<table>
<thead>
<tr>
<th>January</th>
</tr>
</thead>
<tbody>
<tr>
<td>station</td>
</tr>
<tr>
<td>Pittsburgh</td>
</tr>
<tr>
<td>Pittsburgh</td>
</tr>
<tr>
<td>Charleston</td>
</tr>
<tr>
<td>Charleston</td>
</tr>
</tbody>
</table>

Exhibit M.3.2-2. Example 2: Acceptable Format for Multiple Station Names

If the data file is comparable to Exhibit M.3.2-1, the user should choose Pittsburgh and Charleston in the station list, and the **Station Name(s) in Header(s) box** is automatically selected.

If the data file is comparable to the Exhibit M.3.2-2, the user should select the **Station Name(s) in Data box**.

### Single Station

If a file contains data for one station only, the station name does not necessarily have to appear anywhere within the file.

In Exhibit M.3.2-3, the data will be saved for the station for 29 January and 30 January.

<table>
<thead>
<tr>
<th>JAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>date</td>
</tr>
<tr>
<td>29</td>
</tr>
<tr>
<td>30</td>
</tr>
</tbody>
</table>

Exhibit M.3.2-3. Example of Acceptable Format for Single Station Name
General

If Station Name(s) in Header(s) is selected, either the complete station name (in all capital letters) or the four-letter ICAO station ID must appear in the header(s). Difficulties reading the station name might occur when the station name is abbreviated in the station list, the station ID contains more than one word, or more than one station exists for a city. If these difficulties arise, the forecaster should edit the data file by adding the station name (in all capital letters) or the four-letter ICAO ID to the header. Examples are:

Pittsburgh or K PIT
Charleston or K CRW

If Station Name(s) in Data is selected, either the complete station name (in all capital letters) or the four-letter ICAO station ID must appear in the data. In addition, when arranging the columnar order of the data in the Import Daily or Monthly Climatological Data GUI, station name must be chosen or the data will not be saved into the database.

M.3.3 Monthly Data

The data file can contain data for any number of months.

Multiple Months

If there is more than one month in the file, the months must appear either in the text headers separating the monthly data blocks or within the data rows themselves.

If the data file is comparable in format to Exhibit M.3.3-1, the user should choose January and February from the month list, and the Month located in header(s) button is automatically chosen.

<table>
<thead>
<tr>
<th>JAN</th>
<th>nh</th>
<th>nl</th>
<th>rh</th>
<th>rl</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>39</td>
<td>28</td>
<td>61</td>
<td>-4</td>
</tr>
<tr>
<td>31</td>
<td>39</td>
<td>28</td>
<td>58</td>
<td>-10</td>
</tr>
</tbody>
</table>

Exhibit M.3.3-1. Examples of Acceptable Format for Single and Multiple Months of Data

If the data file is comparable in format to Exhibit M.3.3-2, the user should select the Month located in Date Field button and choose the Month/Day format in the file. See below for a further description on these options.
Appendix M Acceptable Formats for Importing Climate Data

<table>
<thead>
<tr>
<th>Date</th>
<th>nh</th>
<th>nl</th>
<th>rh</th>
<th>rl</th>
</tr>
</thead>
<tbody>
<tr>
<td>01/30</td>
<td>39</td>
<td>28</td>
<td>61</td>
<td>-4</td>
</tr>
<tr>
<td>01/31</td>
<td>39</td>
<td>28</td>
<td>58</td>
<td>-10</td>
</tr>
<tr>
<td>02/27</td>
<td>43</td>
<td>30</td>
<td>72</td>
<td>0</td>
</tr>
<tr>
<td>02/28</td>
<td>43</td>
<td>31</td>
<td>68</td>
<td>-1</td>
</tr>
</tbody>
</table>

*Exhibit M.3.3-2. Example of Acceptable Format for Multiple Months of Data in MM/DD Format*

If a file contains a single month of data, the user should choose the month from the list. If the month appears within the data, the user must select the **Month Located in Data Field** box and choose the appropriate **Month/Day format**.

The forecaster chooses a single month from month list if the following file format in Exhibit M.3.3-3 is comparable to the import file.

<table>
<thead>
<tr>
<th>Date</th>
<th>nh</th>
<th>nl</th>
<th>rh</th>
<th>rl</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>35</td>
<td>26</td>
<td>59</td>
<td>0</td>
</tr>
<tr>
<td>02</td>
<td>35</td>
<td>26</td>
<td>63</td>
<td>-3</td>
</tr>
<tr>
<td>03</td>
<td>35</td>
<td>26</td>
<td>71</td>
<td>-7</td>
</tr>
</tbody>
</table>

*Exhibit M.3.3-3. Acceptable Format for Single Month Data*

The forecaster chooses **Month Located in Date Field** and **Month First** format if the following file format in Exhibit M.3.3-4 is comparable to the import file.

<table>
<thead>
<tr>
<th>Date</th>
<th>nh</th>
<th>nl</th>
<th>rh</th>
<th>rl</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jan/01</td>
<td>35</td>
<td>26</td>
<td>59</td>
<td>0</td>
</tr>
<tr>
<td>Jan/02</td>
<td>35</td>
<td>26</td>
<td>63</td>
<td>-3</td>
</tr>
<tr>
<td>Jan/03</td>
<td>35</td>
<td>26</td>
<td>71</td>
<td>-7</td>
</tr>
</tbody>
</table>

*Exhibit M.3.3-4. Acceptable Month First Format*

If **Month Name(s) in Header(s)** is selected, at least the first three characters of the month (in CAPS) must appear in the text. The following month names are acceptable:

- JAN
- FEB
- MAY
- JANUARY
- FEBRUARY
- MAY

If **Month Located in Date Field** is selected, there are several acceptable date formats. Single-digit months or days are not dependent on a preceding zero. The month and day may be in two separate, successive fields (separated by one of the defined delimiters). When arranging the order of the data for a file with the date in two separate fields, choose date once in the column order list.

- **Month First** option. The month appears before day in date file. Exhibit M.3.3-5 displays the acceptable combinations for month first options.
Appendix M

Acceptable Formats for Importing Climate Data

<table>
<thead>
<tr>
<th>Mar-23</th>
<th>03/23</th>
<th>MAR 23</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAR-23</td>
<td>3/23</td>
<td>mar 23</td>
</tr>
<tr>
<td>mar-23</td>
<td>03 23</td>
<td>MAR/23</td>
</tr>
<tr>
<td>03-23</td>
<td>3 23</td>
<td>mar/23</td>
</tr>
<tr>
<td>3-23</td>
<td>Mar 23</td>
<td>Mar/23</td>
</tr>
</tbody>
</table>

Exhibit M.3.3-5. Various Acceptable Month First Combinations

- **Day First** option. The day appears before month in data file. Exhibit M.3.3-6 displays the acceptable combinations for day first options.

<table>
<thead>
<tr>
<th>23-Mar</th>
<th>23/03</th>
<th>23 MAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>23-MAR</td>
<td>23/3</td>
<td>23 mar</td>
</tr>
<tr>
<td>23-mar</td>
<td>23 03</td>
<td>23/MAR</td>
</tr>
<tr>
<td>23-03</td>
<td>23 3</td>
<td>23/Mar</td>
</tr>
<tr>
<td>23-3</td>
<td>23 Mar</td>
<td>23/mar</td>
</tr>
</tbody>
</table>

Exhibit M.3.3-6. Various Acceptable Day First Combinations

- **Julian Days**. Starting at the beginning of the year, days are numbered numerically (1–365). In this case 29 February either is not counted at all or is equal to 366.

- **Julian Days (Feb29=60)**. Starting at the beginning of the year, days are numbered numerically (1–366). In this case 29 February is assigned day 60.

**NOTE**: In either Julian Days option, 29 February is not stored to the database.
Appendix N
AWIPS National Datasets
This appendix contains the 25 March 2014 listing of the AWIPS National Datasets (NDM files), maintained by the Support Branch of SEC/OST (W/OST33). This listing is updated daily; refer to http://www.nws.noaa.gov/ndm/ for the most current listing and any updated instructions associated with updating the AWIPS national datasets. The datasets are available to AWIPS via sftp to NOAA1 pub/ndm. The URL for the ListServer is http://infolist.nws.noaa.gov/read/login (login with your email address and listserver password).

<table>
<thead>
<tr>
<th>Dataset</th>
<th>File Description/Metadata</th>
</tr>
</thead>
<tbody>
<tr>
<td>afos2awips.080624</td>
<td>EXAMPLE ONLY</td>
</tr>
<tr>
<td></td>
<td>Go to NOAA1 for current version.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Review before using a2a on NOAA1 /awips/fxa/data. It may not be correct for your products.</td>
</tr>
<tr>
<td></td>
<td>Maps AFOS ID to WMO data designator and AWIPS CCCC. Updated by OSO to match NWSTG switching directory. Most accesses are not via a binary search, so this can be time consuming. It is unknown what the afos2awips.realIds and afos2awips.testIds files are used for. wanMsgHnd reads the whole table at initialization; handleOUP.pl validates AWIPS ID using this file every time it is invoked.</td>
</tr>
<tr>
<td>afos_lookup_table.dat</td>
<td>02/07/13</td>
</tr>
<tr>
<td></td>
<td>Maps AWIPS CCCC to AFOS CCC for use by the StdDBDecoder process to build AFOS PILS for text product storage. (Used to decode standard text product). While there are cases where an AWIPS CCCC is associated with multiple AFOS CCCs, the nature of this table means that each CCCC may map to only one CCC. (An AFOS CCC may be associated with many AWIPS CCCs.)</td>
</tr>
<tr>
<td>afosMasterPIL.txt</td>
<td>03/25/14</td>
</tr>
<tr>
<td></td>
<td>Processed by localization (-text option) to create /awips/fxa/data/localizationDataSets/LLL/afosMasterPIL.CCC, which is used to build the text workstation browser contents.</td>
</tr>
<tr>
<td>awips2afos.txt</td>
<td>09/24/13</td>
</tr>
<tr>
<td></td>
<td>Used by the program that handles the RSS ingest into AWIPS. It first uses awips2afos.txt to get the nine letter ID from the KxxxNNNXXX on the incoming product. Later it calls distributeProduct.pl to send out the product to the world - distributeProduct looks up the nine-letter ID in afos2awips.txt to get the WMO header.</td>
</tr>
<tr>
<td>awipsPriorities.txt</td>
<td>06/20/05</td>
</tr>
<tr>
<td></td>
<td>PENDING awipsPriorities.txt description.</td>
</tr>
<tr>
<td>bit_table.dat</td>
<td>01/31/14</td>
</tr>
<tr>
<td></td>
<td>Table entries indicate the NNNs of those IDs that are stored “with the national bit set” (an AFOS term meaning that the user can call up the product without using the CCC because it is always stored under the local CCC). The StdDBDecoder substitutes the local CCC for AAA.</td>
</tr>
<tr>
<td>collective_table.dat</td>
<td>04/26/12</td>
</tr>
<tr>
<td></td>
<td>Maps WMO data designators of collective products with AFOS ID. An entry in this table directs the product to the CollDBDecoder vice the StdDBDecoder. Both decoders read the file. Many entries are of the form AAAXXX; the AAA will be replaced by the local CCC. Other entries use CCCXXX where XXX is determined from the product contents and CCC is obtained from the product contents and the CCC is obtained from the national category table.</td>
</tr>
<tr>
<td>GOESImagerInfo.txt</td>
<td>01/09/07</td>
</tr>
<tr>
<td></td>
<td>See NOAA1 pub/ndm/</td>
</tr>
<tr>
<td></td>
<td>Used by satellite decoder GOESImagerInfo.txt requires updating when a new satellite becomes operational.</td>
</tr>
<tr>
<td>gribModelsECMF_ed1.txt</td>
<td>10/14/03</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> No longer NDM - was returned to baseline Nov 2003</td>
</tr>
<tr>
<td>Dataset</td>
<td>File Description/Metadata</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ispan_table.template 03/25/14</td>
<td>Maps WMO header (data descriptor + origin) for noncollective products with AFOS ID. File is processed by localization (-text option) to produce /awips/ixa/data/localizationDataSets/LLL/ispan_table.dat. The latter is used by the StdDBDecoder as a “last resort” to create the AFOS PIL, particularly in cases where the product origin does not map directly to a CCC (e.g., KWBC). The file is a list of those IDs that cannot be created from the data and their corresponding WMO IDs, and is generally based on the NWSTG switching directory.</td>
</tr>
<tr>
<td>locationIDs (09/24/10)</td>
<td>Contains reference points for aviation advisories and change is required, effective April 15, 2008 (see TIN 07-83)</td>
</tr>
<tr>
<td>maritimeStationInfo.txt 03/14/14</td>
<td>Contains station information for fixed buoy and for CMAN sites. It is read by the MaritimeDecoder, which logs if it does not find a station. This is also used by localization (-station option) to complete BUOY.spi (station plot info), for those stations that do not in BUOY.goodness, allowing them to be plotted.</td>
</tr>
<tr>
<td>metarStationInfo.txt 03/25/14</td>
<td>Contains station information for METAR and SAO sites. It is read by the METAR decoder, which reports in its log if it does not find a station (but stores the decoded report anyway). This data set is also used by localization (-station option) to complete MTR.spi (station plot info), for those stations that do not appear in MTR.goodness, allowing data to be plotted.</td>
</tr>
<tr>
<td>MTR.goodness03/25/14</td>
<td>Used to control plotting. This file lists METAR and SAO station locations and their plotting desirability relative to surrounding stations; used for progressive disclosure. Localization (-station) creates /awips/ixa/data/localizationDataSets/LLL/MTR.spi, which is used to construct the METAR stations map.</td>
</tr>
<tr>
<td>modelBufrStationInfo.txt 03/26/12</td>
<td>Description pending.</td>
</tr>
<tr>
<td>national_category_table.template 03/25/14</td>
<td>Localization (-text) produces /awips/ixa/data/localizationDataSets/LLL/national_category_table.dat, which is used by the CollDBDecoder to map XXX or XXXX station ID to AFOS PIL CCC. There is no longer a limit to the number of entries (had been 3500). This is also used for METAR plot sampling.</td>
</tr>
<tr>
<td>pirepsTable.txt 01/28/14</td>
<td></td>
</tr>
<tr>
<td>profiler.goodness 09/03/02</td>
<td>Description pending.</td>
</tr>
<tr>
<td>profilerStationInfo.txt 09/03/02</td>
<td>Description pending.</td>
</tr>
<tr>
<td>prodList.txt 01/10/11</td>
<td>prodList.txt entries force the radarStorage process to transmit the radar product to the NCF under the WMO heading.</td>
</tr>
</tbody>
</table>
## Appendix N

### AWIPS National Datasets

<table>
<thead>
<tr>
<th>Dataset</th>
<th>File Description/Metadata</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rps-RPGOP-tcp.clear-air 01/10/11 see NOAA1 pub/ndm</td>
<td>rps-RPGOP-tcp.clear-air specifies the WSR-88D radar products which should always be included in the RPS lists when radar is in clear air mode so that they can be centrally collected.</td>
</tr>
<tr>
<td>dual-pol-rps-RGPOP-tcp.clear.air 08/30/11 see NOAA1 pub/ndm</td>
<td></td>
</tr>
<tr>
<td>rps-RPGOP-tcp.storm 01/10/11 see NOAA1 pub/ndm</td>
<td>rps-RPGOP-tcp.storm specifies the WSR-88D radar products which should always be included in the RPS lists when radar is in storm mode so that they can be centrally collected. Includes Mesocyclone MD141 product.</td>
</tr>
<tr>
<td>dual-pol rps-RPGOP-tcp.storm 08/30/11 see NOAA1 pub/ndm/dualpol</td>
<td></td>
</tr>
<tr>
<td>rps-SPGOP-tcp.storm 05/21/08</td>
<td>rps-SPGOP-tcp.storm specifies the TDWR radar products which should always be included in the RPS lists so that they can be centrally collected. This file was introduced in OB9.</td>
</tr>
<tr>
<td>radarInfoMaster.txt is no longer an NDM file</td>
<td>radarInfoMaster.txt is created by localization from fs1-w88d (.shp,.shx,.dbf) during localization</td>
</tr>
<tr>
<td>wmoSiteInfo.txt 09/20/11 See NOAA1 pub/ndm</td>
<td></td>
</tr>
<tr>
<td>Radars – TDWR and FAA Radars-NDM Files</td>
<td></td>
</tr>
<tr>
<td>tdwrFsiScanOrder.txt 8/25/08</td>
<td>tdwrFsiScanOrder.txt contains the modified order of the elevation cuts for every TDWR radar for both Hazard and Monitor modes. This file is used by FSI and was introduced in OB9</td>
</tr>
<tr>
<td>tdwrProdList.txt 08/23/10</td>
<td>tdwrProdList.txt specifies which TDWR radar product should be sent to the WAN for central collection and what WMO header should be used for that product. This file introduced in OB9.</td>
</tr>
<tr>
<td>tdwrElevations.txt 08/07/08</td>
<td>tdwrElevations.txt contains the elevation cuts for every TDWR radar for both Hazard and Monitor modes.</td>
</tr>
<tr>
<td>How to update tdwrRadars.txt 07/30/07</td>
<td>tdwrRadars.txt contains the TDWR radar IDs currently being used in AWIPS.</td>
</tr>
<tr>
<td>wmoSiteInfo.txt 09/20/11 see NOAA1 pub/ndm dx2 (dx1 for failover)</td>
<td>wmoSiteInfo.txt associates the radar RPG/SPGs to each WFO site and the flag of each RPG/SPG indicates which WFO site sends the radar data to the WAN for central collection.</td>
</tr>
<tr>
<td>RAOB Datasets</td>
<td></td>
</tr>
<tr>
<td>raob.goodness 09/17/23</td>
<td>In raob.goodness, the very last (sixth) column is the progressive disclosure hint, as described in staticProgDisc.html or va_driver.doc.html In this file, it is also used to say whether a given station participates in making pop-up skewTs and geographic station selection for the VB. Stations with a progressive disclosure hint greater than zero will participate in pop-up skewTs and geographic station selection for the VB.</td>
</tr>
<tr>
<td>raobDataKeys.txt 09/17/23</td>
<td></td>
</tr>
<tr>
<td>raobDepictKeys.txt 09/17/23</td>
<td></td>
</tr>
<tr>
<td>raobMenus.txt 09/17/23</td>
<td></td>
</tr>
<tr>
<td>raobProductButtons.txt 09/17/23</td>
<td></td>
</tr>
<tr>
<td>raobStationInfo.txt 09/17/23 See NOAA1 pub/ndm</td>
<td></td>
</tr>
<tr>
<td>redbookPurgeInfo.txt 07/22/13</td>
<td>Refer to Redbook description link</td>
</tr>
<tr>
<td>Dataset</td>
<td>File Description/Metadata</td>
</tr>
<tr>
<td>-----------------------------------------</td>
<td>------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>redbookDataKeys.txt 07/22/13</td>
<td>Refer to Redbook description link</td>
</tr>
<tr>
<td>redbookDepictKeys.txt 07/22/13</td>
<td>Refer to Redbook description link</td>
</tr>
<tr>
<td>redbookLegends.txt 11/30/05</td>
<td>Refer to Redbook description link</td>
</tr>
<tr>
<td>redbookProductButtons.txt 07/22/13</td>
<td>Refer to Redbook description link</td>
</tr>
<tr>
<td>redbookCPCMenus.txt 09/15/11</td>
<td>Refer to Redbook description link</td>
</tr>
<tr>
<td>redbookHPCMenus.txt 07/22/13</td>
<td>Refer to Redbook description link</td>
</tr>
<tr>
<td>redbookNCOMenus.text 03/16/09</td>
<td>Refer to Redbook description link</td>
</tr>
<tr>
<td>redbookHazardMenus.txt 03/27/07</td>
<td>Refer to Redbook description link</td>
</tr>
<tr>
<td>redbookMarineMenus.txt 06/21/05</td>
<td>Refer to Redbook description link</td>
</tr>
<tr>
<td>redbookUpperAirMenus.txt 03/13/09</td>
<td>Refer to Redbook description link</td>
</tr>
<tr>
<td>redbookLegends.txt 11/30/05</td>
<td>Refer to Redbook description link</td>
</tr>
<tr>
<td>satSpecificInfo.txt 04/14/10</td>
<td>See NOAA1 pub/ndm This file is used by satellite decoder and the fog monitor, satSpecificInfo.txt requires updating when a new satellite becomes operational.</td>
</tr>
<tr>
<td>How to Update</td>
<td></td>
</tr>
<tr>
<td>selsAnchors.txt 10/25/05</td>
<td>See NOAA1 pub/ndm This file is used to plot SPC watch boxes. The anchor points used in the “public” line (not the aviation line) on SAW products are listed here.</td>
</tr>
<tr>
<td>Station_locations table for climate</td>
<td>Contains the list of METAR, TAF, and SCD stations “known” to the Climate Reports Formatter and AWIPS VER program. Only stations contained in this master list can be set up in these applications. Stations not in this list will be rejected by the application setup user interfaces. The master list is contained in a data table in the hmdb database, is nationally configured, and is not to be modified by users.</td>
</tr>
<tr>
<td>Bob Morris, MDL</td>
<td></td>
</tr>
<tr>
<td>Station_table.dat 05/16/12</td>
<td>Used to create the AFOS ID for upper air (UA) products. Maps a five-digit UA station number to XXXX, which in turn is passed through national_category_table.dat to build an AFOS PIL (in CollDBDecoder).</td>
</tr>
<tr>
<td>synopticStationTable.txt 08/17/10</td>
<td></td>
</tr>
<tr>
<td>tdwrElevations.txt 11/15/05</td>
<td>OB81.1 tdwrElevations.txt</td>
</tr>
<tr>
<td>textCategoryClass.txt 03/19/14</td>
<td>Used to map NNN to one or more classes for menu assignment in the text browser. The list of classes is found in the textBrowser.tcl, procedure classCode. The designated NNN will appear in the browser menus for each listed class.</td>
</tr>
<tr>
<td>textCCChelp.txt 03/25/14</td>
<td>Contains help strings associated with CCC or XXX entries in the text browser.</td>
</tr>
<tr>
<td>textNNNhlp.txt 03/14/14</td>
<td>Contains help strings associated with NNN entries in the text browser.</td>
</tr>
<tr>
<td>textOriginTable.txt 10/20/11</td>
<td>Contains information on which CCCs appear in which Origin list on the text browser. Local sites will need to modify the “R” section, which is set up for Colorado and is not hit by localization.</td>
</tr>
<tr>
<td>UGClookups_changes.txt 12/30/13</td>
<td>Describes what changes have been made in the UGClookups.table file.</td>
</tr>
<tr>
<td>nfs mounted /awips/adapt/NWRWAVES/</td>
<td></td>
</tr>
<tr>
<td>UGClookups.table 12/30/13</td>
<td>Contains counties/zones information and is used by the NWRWAVES scripts.</td>
</tr>
</tbody>
</table>
### Dataset Description/Metadata

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Description/Metadata</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>upair_table.dat 03/27/12</strong></td>
<td>Maps the WMO data designator of the upper air products with the AFOS ID. Similar to collective_table.dat, an entry in this table directs the product to the <code>ColDBDecoder</code> instead of the <code>StdDBDecoder</code>. (Both decoders read the file.) Entries are in the format CCC(nnn)XXX, where XXX is determined from the product contents (via station_table.dat) and CCC is obtained from national_category_table.dat.</td>
</tr>
<tr>
<td><strong>dataInfomanual OB9.11</strong></td>
<td>Description pending</td>
</tr>
<tr>
<td><strong>dataInfo.manual OB81</strong></td>
<td>Description pending</td>
</tr>
<tr>
<td><strong>dataInfo.manual OB82</strong></td>
<td>Description pending</td>
</tr>
<tr>
<td><strong>dataInfo.manual OB83</strong></td>
<td>Description pending</td>
</tr>
<tr>
<td><strong>dataInfo.manual OB9</strong></td>
<td>Description pending</td>
</tr>
<tr>
<td><strong>depictInfo.manual OB9.11</strong></td>
<td>Description pending</td>
</tr>
<tr>
<td><strong>depictInfo.manual OB81</strong></td>
<td>Description pending</td>
</tr>
<tr>
<td><strong>depictInfo.manual OB82</strong></td>
<td>Description pending</td>
</tr>
<tr>
<td><strong>depictInfo.manual OB83</strong></td>
<td>Description pending</td>
</tr>
<tr>
<td><strong>depictInfo.manual OB9</strong></td>
<td>Description pending</td>
</tr>
<tr>
<td><strong>UGClookup_changes.txt 07/18/11</strong></td>
<td>Describes what changes have been made in the UGClookup.table file.</td>
</tr>
<tr>
<td>nfs mounted</td>
<td>nfs mounted</td>
</tr>
<tr>
<td>/awips/adapt/NWRWAVES/</td>
<td>nfs mounted</td>
</tr>
<tr>
<td><strong>UGClookup.table 07/18/11</strong></td>
<td>Contains counties/zones information and is used by the NWRWAVES scripts.</td>
</tr>
<tr>
<td>nfs mounted</td>
<td>nfs mounted</td>
</tr>
<tr>
<td>/awips/adapt/NWRWAVES/bin/</td>
<td>nfs mounted</td>
</tr>
<tr>
<td>How to Update - 6/30/09</td>
<td>nfs mounted</td>
</tr>
<tr>
<td><strong>productButtonInfo.txt OB9.11</strong></td>
<td>Description pending</td>
</tr>
<tr>
<td><strong>productButtonInfo.txt OB81</strong></td>
<td>Description pending</td>
</tr>
<tr>
<td><strong>redbookHPCMenus.txt OB82</strong></td>
<td>Description pending</td>
</tr>
<tr>
<td><strong>productButtonInfo.txt OB83</strong></td>
<td>Description pending</td>
</tr>
<tr>
<td><strong>productButtonInfo.txt OB9</strong></td>
<td>Description pending</td>
</tr>
<tr>
<td><strong>Radar Datasets maintained by ROC</strong></td>
<td>The WSR-88D Hotline and ROC, acting as agents for the respective Regions and NWSH, maintain and distribute AWIPS system-specific &quot;radar files&quot; for WSR-88D and TDWR/SPG access. Routine requests for changes to existing radar access permissions should be coordinated through Regional Focal Points. Emergency requests for changes to existing radar access permissions can be requested to the Hotline directly and will be subsequently coordinated with the respective Regional Focal Point by the Hotline.</td>
</tr>
<tr>
<td>dialRadars.txt</td>
<td></td>
</tr>
<tr>
<td>radarsInUse.txt</td>
<td></td>
</tr>
<tr>
<td>radarsOnMenu.txt</td>
<td></td>
</tr>
<tr>
<td>orpgDedicated.txt</td>
<td></td>
</tr>
<tr>
<td>orpgOTRs.txt</td>
<td></td>
</tr>
<tr>
<td>orpgBackups.txt</td>
<td></td>
</tr>
</tbody>
</table>

### NOTES:

1. The following files are maintained by FSL:
   - radarDataKeys.template
   - radarDataMenus.template
   - radarDepictKeys.template
   - radarImageStyleInfo.template

2. The rcv_handler.tbl file will be delivered with releases when appropriate and has been removed from this list of national datasets.
For most of the files listed in the preceding table, you can download the ndm file (the datasets are available to AWIPS via sftp to NOAA1 pub/ndm) and copy it to /awips2/edex/data/ndm on DX3 or DX4. EDEX will process it and update the appropriate tables.

The site data configuration automation tool will copy those national data management files, already maintained on the AWIPS system, into the endpoint for EDEX, which allows the software to ingest and localize properly based on the file. The files that are supported are identified in the following table.

<table>
<thead>
<tr>
<th>Input File Name</th>
<th>Staging Location</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>afos2awips.txt</td>
<td>/awips/fxa/data</td>
<td>afos_to_awips table in the fxatext database</td>
</tr>
<tr>
<td>modelBufrStationInfo.txt</td>
<td>/awips/fxa/data</td>
<td>modelBufr.spi in /awips2/edex/data/utility/common_static/site/LLL/base maps</td>
</tr>
<tr>
<td>MTR.primary, MTR.goodness</td>
<td>/awips/fxa/data</td>
<td>MTR.spi, MTR.primary, MTR.goodness in /awips2/edex/data/utility/common_static/site/LLL/base maps</td>
</tr>
<tr>
<td>goesBufrStationInfo.txt</td>
<td>/awips/fxa/data</td>
<td>goesBufr.spi in /awips2/edex/data/utility/common_static/site/LLL/base maps</td>
</tr>
<tr>
<td>poesBufrStationInfo.txt</td>
<td>/awips/fxa/data</td>
<td>poesBufr.spi in /awips2/edex/data/utility/common_static/site/LLL/base maps</td>
</tr>
<tr>
<td>maritimeStationInfo.txt</td>
<td>/awips/fxa/data</td>
<td>BUOY.spi in /awips2/edex/data/utility/common_static/site/LLL/base maps</td>
</tr>
<tr>
<td>metarStationInfo.txt</td>
<td>/awips/fxa/data</td>
<td>common_obs.spatial table in metadata database</td>
</tr>
<tr>
<td>raob.goodness, raob.primary</td>
<td>/data/fxa/nationalData</td>
<td>raob.spi in /awips2/edex/data/utility/common_static/site/LLL/base maps</td>
</tr>
<tr>
<td>raobStationInfo.txt</td>
<td>/awips/fxa/data</td>
<td>common_obs.spatial table in metadata database</td>
</tr>
<tr>
<td>ispan_table.dat</td>
<td>/awips/fxa/data/localization/ndataSets/LLL</td>
<td>ispan_table in /awips2/edex/data/utility/edex_static_tree</td>
</tr>
</tbody>
</table>

Please refer to the *AWIPS II Site Migration Guide* for the command line options to copy the above-listed files.

The instructions for updating the radar ndm files follow.

- asrRadars.txt, arsrRadars.txt, tdwrRadars.txt
  - AWIPS 2 does not use these files.

- prodList.txt, tdwrProdList.txt, rps-RPGOP-tcp.storm, rps-SPGOP-tcp.storm, wmoSiteInfo.txt, rps-RPGOP-tcp.clear-air
  1. Copy to /awips2/rcm/data/config/drop-ins/ on dx1 and dx2.
  2. Restart the RadarServer.
• tdwrElevations
  For EDEX, this file may be updated by copying it to /awips2/edex/data/ndm.
  For the RadarServer and CAVE radar applications, the file should be updated as part of a
  software release. A DR will be necessary to enable a practical way to manually update
  the file.

• tdwrFsiScanOrder.txt
  1. Copy these to /awips/fixa/FSIedex/data/ on px1 and px2.
  2. Restart FSIprocessor.
     (/awips/fixa/FSIedex/bin/{stopFSIprocessorEDEX,startFSIprocessorEDEX})
Appendix O
NWRWAVES User’s Manual
& Documentation
The *NWR WAVES User’s Manual & Documentation*, a publication of the National Weather Service, has been reproduced in its entirety in this appendix of the SMM. This version of the manual was released on February 28, 2012, and was the current version as of the publication date of the OB14.1.1 SMM.
# NWRWAVES

(NOAA Weather Radio With All-Hazards VTEC Enhanced Software)

User’s Manual & Documentation

## TABLE OF CONTENTS

1. SOFTWARE SPECIFICS .......................................................................................................................... - 2 -
2. INTRODUCTION ........................................................................................................................................ - 2 -
3. INSTALLATION OF NWRWAVES ....................................................................................................... - 3 -
4. NWRWAVES SETUP UTILITY ............................................................................................................... - 3 -
   Tab 1 – Transmitter Configuration ......................................................................................................... - 6 -
   SPECIAL NOTE ABOUT ROUTINE VS. NON-ROUTE BSA’S .......................................................... - 8 -
   SPECIAL NOTE ABOUT CORE COUNTY/CORE ZONE LOGIC .................................................. - 10 -
   Tab 2 – Product Configuration ........................................................................................................... - 11 -
   SPECIAL NOTES ABOUT VTEC & SHORT-FUSE WATCHES/WARNINGS ................................ - 12 -
   SPECIAL NOTES FOR SITES WITH MARINE RESPONSIBILITIES ........................................ - 12 -
   SPECIAL NOTES ON CONVETIVE WATCH PRODUCT PREPARATION .................................. - 22 -
   SPECIAL NOTES ON NON-WEATHER EMERGENCY MESSAGES (NWEM) .......................... - 22 -
   Tab 3 – Summary Message/Miscellaneous Settings Configuration .................................................. - 23 -
   Tab 4 – Marine/Tropical Product Configuration ................................................................................ - 26 -
      Marine VTEC Processing ................................................................................................................... - 28 -
      Hurricane Local Statement .............................................................................................................. - 28 -
      Tropical Cyclone Product (TCP) .................................................................................................... - 28 -
      Great Lakes Forecasts .................................................................................................................... - 29 -
5. SETTING AWIPS TEXT DATABASE TRIGGERS ................................................................................... - 30 -
   SPECIAL NOTES REGARDING TEXT DATABASE TRIGGERS ...................................................... - 31 -
   NWEM messages: ................................................................................................................................. - 31 -
   Convective watches: .............................................................................................................................. - 32 -
   Selectively specify transmission status based on issuing WFO in OB9: ........................................ - 34 -
6. NWRWAVES AND YOUR CRS DATABASE ......................................................................................... - 35 -
   TRANSMITTER SPECIFIC MESSAGE TYPES ................................................................................ - 35 -
   GENERIC MESSAGE TYPES .............................................................................................................. - 36 -
   SETTING UP MESSAGE TYPES IN YOUR CRS DATABASE ........................................................... - 37 -
   SETTING UP TRANSMITTER SPECIFIC MESSAGE TYPES ........................................................ - 38 -
   SETTING UP GENERIC MESSAGE TYPES ....................................................................................... - 39 -
   IMPORTANT NOTES ABOUT CRS MESSAGE TYPES AND YOUR CRS DATABASE .................. - 42 -
7. NWRWAVES OPERATIONS ...................................................................................................................... - 42 -
   NWRWAVES BROWSER ..................................................................................................................... - 43 -
8. NWRWAVES PROGRAM EXECUTION (LOGGING, TROUBLESHOOTING) ..................................... - 47 -
9. WORD/PHRASE REPLACEMENT CAPABILITY ................................................................................. - 50 -
10. SWAPS Capability ................................................................................................................................. - 51 -
11. OPTIONS TO PARSE CALL-TO-ACTION IN NWS ALERT MESSAGES ........................................ - 51 -
12. LESSONS LEARNED AND RESOURCES AVAILABLE ...................................................................... - 54 -
   APPENDIX A – COMPLETE NWRWAVES FILE LISTING .............................................................. - 55 -
   APPENDIX B – EXAMPLE OF EAX siteTrigger.template FILE FOR NWRWAVES ................ - 57 -
   APPENDIX C – NWRWAVES Cron Jobs ........................................................................................ - 59 -
   APPENDIX D – Instructions for implementing MWW Watch/Warnings/Advisories ..................... - 60 -
1. SOFTWARE SPECIFICS

Version: AWIPS II
Programming Language: Tcl/Tk
Design Team:
- Evan Bookbinder – WFO Pleasant Hill, MO
- Brian Walawender – Central Region Headquarters Kansas City, MO
- Michael Hudson – Central Region Headquarters Kansas City, MO
- Peter Browning – Central Region Headquarters Kansas City, MO

Maintenance Lead:
- Warrick Moran – NWR Headquarter Silver Spring, MD

Maintenance Support:
- Nancy Helderman – NWR Headquarter Silver Spring, MD

Maintenance Programmers:
- NWRWAVES: Sung Vo and Peter Wu (OPS23) 301 713-0191

2. INTRODUCTION

NWRWAVES is a comprehensive formatter for NOAA Weather Radio (NWR) products. Its purpose is threefold. First, it is designed to replace all existing formatter capabilities found in the AWIPS WWA program and also the capabilities found in CAFÉ. Second, NWRWAVES utilizes VTEC coding found in an increasing suite of NWS products to better identify, produce and manage outbound CRS Weather Messages. The use of the Message Reference Descriptor (MRD) number will allow sites to better automate their CRS Broadcast Cycle management. Third, NWRWAVES is designed for better sustainability over the numerous scripts used in the CAFÉ approach.

As a new site preparing to use NWRWAVES, there are three basic steps necessary to configure the software for your local area after installation (Section 2 – INSTALLATION OF NWRWAVES).

A) The first is to configure your local settings in NWRWAVES. These include your transmitters or broadcast programs, the products that can be processed through NWRWAVES, summary message settings and marine/tropical product settings. NWRWAVES has powerful local configuration tools for product-by-product settings, adding local Listening Area Codes (LACs) and local products, and a customizable word/phrase replacement capability (similar to CAFÉ). Instructions on how to configure these settings can be found in Section 3 – NWRWAVES SETUP UTILITY.

B) The second step is to establish triggers from the AWIPS database, which will send products through NWRWAVES. NWRWAVES does feature a test mode, allowing the formatter program to run in conjunction with existing CAFÉ formatters in order to facilitate side-by-side comparisons of output prior to formal activation (by product) of NWRWAVES. Instructions on how to establish AWIPS text database triggers are located in Section 4 – SETTING AWIPS TEXT DATABASE TRIGGERS.

C) The third step (and a necessary one before you transition NWRWAVES to operational mode) is to ensure that your CRS database is structured to accept inbound Weather Messages from NWRWAVES. NWRWAVES uses an approach where each transmitter/broadcast program receives a unique Message Type naming scheme. An explanation of how NWRWAVES generates Weather Messages, along with suggestions on an approach to the structure of the local CRS database, is located in Section 5 – NWRWAVES AND YOUR CRS DATABASE.
3. INSTALLATION OF NWRWAVES

If your site has never installed and configured NWRWAVES prior to the installation of OB9, you must first grant permission for users in your office to access the NWRWAVES Setup Utility. This will allow you to configure the software in your office. In the /awips/adapt/NWRWAVES/browser directory, there is a file called “admin.list”. You will need to edit this file and add any AWIPS user accounts that you want to have access to the setup utility. All other users will be blocked.

To edit this file from any AWIPS workstation:

- Open a shell with a user account that has fxalpha group privileges. In this shell, change directories into the NWRWAVES Browser directory:

  cd /awips/adapt/NWRWAVES/browser

- Edit the “admin.list” file:

  nedit admin.list

- Add your local AWIPS user accounts as needed, and delete any existing entries from the default list. Make sure there is only one user account on each line.

- When you are done, save the file and then exit from the editor. Exit from the shell.

You are now ready to run the NWRWAVES setup utility. This utility will allow you to configure NWRWAVES for your local site.

4. NWRWAVES SETUP UTILITY

The next two steps require that you are an AWIPS user whose AWIPS user account is listed in the “admin.list” file as described above.

A) Open the NWRBrowser on any AWIPS workstation. You will be greeted by the NWRWAVES Browser (Figure 1).
The NWRWAVES Browser functionality mirrors that of the NWR Browser, and there are also added capabilities. Details on NWRWAVES Browser operation can be found in Section 6 – NWRWAVES OPERATION.

B) From the “NWRWAVES” pull-down menu, select the option “NWRWAVES Setup”. You will see the NWRWAVES Setup GUI pop-up on your AWIPS screen (Figure 2). The initial display will start on the first of four tabs, which is the transmitter/broadcast program configuration tab.

NOTE: If you get an error message that a version is already open somewhere else, but you are confident that there is not a version open anywhere on your AWIPS, perform the following steps:

- Open a shell on any AWIPS workstation as a user with ‘fxalpha’ group privileges.
- Type 
  
  cd /awips/adapt/NWRWAVES
- Type 
  
  rm –f setup.lock
- Exit from the shell
Figure 2 – NWRWAVES Setup GUI

There are four tabs on the NWRWAVES Setup GUI. Each one governs a unique section of NWRWAVES configuration. You can cycle to each one by simply clicking on the tab label located at the top of the interface. There is a pull-down menu, “File”, which will allow you to ‘Save and Exit’ or simply ‘EXIT WITHOUT SAVING’ the program. The “Help” pull-down menu contains information on the product version.
On each tab, there are buttons labeled with question marks. Each of these buttons contains information on what you need to do for that particular configuration item. Be sure to make liberal use of these question boxes for hints and help.

**NOTE:** When you make adjustments to NWRWAVES configuration data, the current version of your configuration files are saved off into the `/awips/adapt/NWRWAVES/BACKUP` directory. Details on these file names are in the different ‘Tab’ sections below. When they are saved off into the `/BACKUP` directory, the naming scheme used is the file name, followed by a date/time stamp. This will allow you to quickly restore from any mistakes made in configuring different components of NWRWAVES.

To configure NWRWAVES, you will work from left to right on these tabs to establish your office configuration. The first tab is our first tab – establishing your transmitter information.

**Tab 1 – Transmitter Configuration**

To configure your transmitter (broadcast program) information, complete the following steps:

1. At the top of the interface there is a box for you to enter the number of transmitters (number of broadcast programs) you have in your office. Left click in this box, enter in the number of transmitters in your CWA and press the “Enter” key. NWRWAVES will create a list of transmitters (over several rows) for your use. By default, the names of these transmitters will be a default ID of \texttt{NW} x (where x is 1 to the number of transmitters). For transmitters over 10, the naming scheme is \texttt{Nxx}.

2. Select a transmitter by clicking on the radio button by its label \texttt{NW} x with the left mouse button. For the first transmitter, this will be the one labeled “NW1”.

3. Click the “\textbf{Change Transmitter Name}” button to set a 3-letter ID for that transmitter. \textbf{NOTE:} the 3-letter ID will be used to create the PIL when generating CRS messages (a transmitter called MCI would produce products like STLNOWMCI, STLSVAMCI, etc). The first three letters of the CRS message is your legacy state AFOS identifier. \textbf{NOTE:} Be sure to make note of the three letter mnemonic used for each transmitter. This will be important as the naming scheme must match how your CRS database is configured (see \textbf{Section 5} for information on CRS Database configuration).

4. If you desire to assign a preamble phrase for your transmitter, enter in a preamble for that transmitter (if desired) in the appropriate box. \textbf{NOTE:} Clicking on the “?” button located past the “Preamble For XXX” data entry box provides a good example of preamble text.

5. Repeat steps 2 through 4 for each transmitter. Each transmitter must have a unique three letter mnemonic, and this three letter mnemonic will be used for the creation of unique Weather Messages for each transmitter. The preamble phrase is strictly optional.

The next step is to create the list of counties (parishes)/independent cities, and public zones/marine zones/fire weather zones which represent the service area for each transmitter.

6. Click on the radio button representing the first transmitter. Left-click in the box to the right of the label “Selected WFO/Center Site ID”. Your office ID should default as the entry for this box. If it is missing for any reason, enter in your office’s three letter identifier into this box, then hit the ‘Enter’ key.

7. You will see a list of county/parish/independent cities appear in the left hand column. There are two radio buttons next to the box you entered your site ID into in (6) above. The default radio button set is “Select Counties/Independent Cities for XXX”, where “XXX” is your site ID. You will assign county (FIPS)
codes and zone (Z) codes separately for each transmitter. This will allow maximum flexibility for unique circumstances in managing your service area.

8) The list of county (parish)/independent cities and zones is kept in a default distribution file in the /awips/adapt/NWRWAVES/bin directory and is called “UGClookup.table”. This file was derived from AWIPS shape files and is as complete as possible. Sites will not modify the UGClookup.table file! If you note errors in the names of your counties (parishes) or zones, please post them to the NWRWAVES listserver, so they can be fixed. This file is now (from OB9.2) part of the National Datasets Maintenance (NDM) files on NOAA1 baseline file. See the following URL for details (close to the bottom): [http://www.weather.gov/ndm/](http://www.weather.gov/ndm/). Sites with marine zones may want to pay close attention, as some of the marine zone names are somewhat verbose in the default file.

Your transmitter configuration tab will initialize with your WFO identifier’s list of counties (parishes)/zones.

There may be instances where a county/zone is missing, or where your office uses a local “dummy” county or zone code on your transmitters. You can add any missing or custom listening area Codes (LAC’s) into NWRWAVES. Likewise, you can also edit or delete any existing local LAC’s. To do this:

a) Click on the pink button labeled, “Edit Local LAC”.
b) A pop-up GUI will appear, and you will be prompted to enter data into it (Figure 2a).

c) From left to right, enter the associated WFO (likely yours), the six-character UGC code (i.e., MOC998, KSZ999), the name of your new LAC (i.e., Generic County), a two-letter state ID (i.e., KS) and the time zone of this new LAC. Note that in the Help Message of this LAC Editor, the word “MARINE” is suggested to be used to specify a marine zone. This feature is now available in version OB9.

For the time zone, the primary option is a capital letter for the appropriate time zone (E-Eastern, C-Central, M-Mountain, P-Pacific, A-Alaskan, H-Hawaiian, g-Guam, a-Virgin Islands/Atlantic Time). If an LAC doesn’t observe Daylight Savings Time, use a lower case letter from the ones listed above in lieu of the capital letter. For example, an entry of “c” represents that this LAC observes Central Time and does not observe Daylight Time.

If an LAC is large enough to encompass two time zones, the use of a two-letter code is appropriate. For example, “eC” would be used for an LAC that covers part of the Eastern Time zone that doesn’t observe DST, but also one that covers part of an area in the Central Time zone that does observe DST.)
If you enter the same data as an existing local LAC, you will see the buttons for ‘Save Edits’ and ‘Delete’ become usable. If the data is new, you will only be given the option to ‘Add LAC’.

d) Click the appropriate button for the action needed when you have finished entering in your local LAC.

Repeat step 8 as necessary to add all local FIPS and zone codes for this transmitter. Any local LAC’s will be added into the /awips/adapt/NWRWAVES/bin directory in the file “localUGClookup.table”.

9) To select counties (zones) for inclusion in a transmitter’s service area, left-click on any applicable county names in the left blue list box. This list will include all baseline counties (zones) as well as any local entries you have entered. You can select multiple counties all at once simply by clicking on the county names (there is no need to hold the “SHIFT” or “CTRL” buttons to multi-select counties). Once you have them selected, you can assign them to either the Routine Broadcast Service Area (BSA) or the Non-Routine Broadcast Service Area (NBSA) by clicking the appropriate button. Likewise, you can remove counties back to the master list by highlighting them in the right blue list boxes and clicking the button.  

**SPECIAL NOTE ABOUT ROUTINE VS. NON-ROUTINE BSA’S**

Each transmitter has a separate list for Routine Counties/Zones and Non-Routine Counties/Zones. In the context of NWRWAVES, this is **not** necessarily a one-to-one relationship with the routine service area of your CRS transmitters. Establish your lists by using any county/zone for which you routinely broadcast information (tone alerts for warnings, etc.) in your Routine BSA. Users would then include county/zone entries that reside outside a normal reception area in the Non-Routine BSA. This is useful as you can include counties in your Non-Routine BSA for which you do unique tone-alert settings by product, counties that are out of your transmitter’s coverage area but for which you carry WRSAME bursts for an area radio station, or to set a county from a distant office to carry a product such as an HLS (where you are an inland office but wish to broadcast information about a land falling tropical system from a nearby coast).

Once you have selected all the counties within your CWA, you can select county/counties from other CWA’s if your transmitter overlaps another office’s CWA. To do this, go back to step (6), and change the office ID from yours to the other office, then hit “ENTER”. **NOTE:** to access counties (parishes)/zones from another office, type that office’s 3-letter ID into the box next to the label “Select WFO/Center Site ID” to populate that office’s list. To enable a list of off-shore marine zones, type “OPC” into this box to access the off-shore zones that the Ocean Prediction Center uses for the OFF products.

10) Repeat steps 7, 8 and 9 for counties from that office. Repeat step (10) for all offices that affect the service area of this transmitter.

11) **CORE COUNTY SETTINGS (NEW):** Once you have established your set of routine/non-routine counties (parishes/independent cities), you may assign one of these to the transmitter’s core county. To do this, right click on the label of one of your counties/parishes/independent cities (in either the Routine or Non-Routine BSA lists), and you will see its appropriate LAC code listing appear in the box above labeled “Core County”. This is your core county for this transmitter.
12) Once you have assigned the county (parish)/independent cities for this transmitter, click on the radio button labeled “Select Zones for XXX”. This will allow you to assign the “Z” or zone codes for this transmitter.

13) Repeat steps (6) through (10) to select all zones applicable to this transmitter. The zone listings are inclusive to all forecast, marine and fire weather zones.

14) **CORE ZONE SETTINGS (NEW):** Once you have established your set of routine/non-routine zones, you may assign one of these as the transmitter’s core zone. To do this, **right click on the label** of one of your selected zones (Routine or Non-Routine), and you will see the appropriate LAC code listing appear in the box labeled “Core Zone”. This is your core zone for this transmitter.

15) Click the green “Save Transmitter Edits” button to save off the configuration data for this transmitter.

16) Left-click on the radio button for the next transmitter in your configuration list.

17) Repeat steps (6) through (15) for each transmitter.

Once you complete these steps, you will have selected county (parish)/independent cities and zones listings for all of your transmitters. Each transmitter will have a unique three-letter mnemonic, and (if applicable), you will have assigned each transmitter with an optional preamble phrase and an optional core county (parish)/independent city and core zone.

The transmitter configuration settings are stored in the /awips/ adapt/NWRWAVES directory in the file “transmitter_cfg.XXX”, where ‘XXX’ is your AWIPS site identifier. **You should not hand-edit this file!** Any changes required must be made through the NWRWAVES Setup GUI.

Once you have configured the transmitters, your “Transmitter Configuration” tab should look similar in format to Figure (3). The next step is to configure products for processing through NWRWAVES.
SPECIAL NOTE ABOUT CORE COUNTY/CORE ZONE LOGIC
Sites have the ability to assign a core county and/or a core zone to each of their individual transmitters. The logic behind a “core county/core zone” is that for certain products, only the segment which contains this core LAC will be processed and sent to that transmitter. Furthermore, only the core LAC will be used in message assembly; therefore, Identical Message Type/Identical LAC replace will occur dependably. Most sites will use core county/core zone for legacy products such as the ZFP, NOW, HWO, etc. Use of the core county/core zone is not appropriate for watch/warning products.

The core county and core zone are global by transmitter. In other words, you only have one of each for every unique transmitter. You do not have the ability to use one ‘core zone’ for the ZFP, and a second for the NOW. If you followed instructions in v2.4 to assign counties/zones to the Routine vs. Non-Routine BSA’s to replicate a core county/zone functionality, you should reassign these lists to take advantage of process/no process and unique tone alert capabilities.
Tab 2 – Product Configuration

After you are done configuring your transmitters, click the **Product Configuration** tab (Figure 4)

![Product Configuration GUI](image)

**Figure 4 – Product configuration GUI**

This tab is used to set all the configuration items for each NWRWAVES product type. Those used to WWA or CAFÉ formatters should note some similarities, but many new features will also allow for extensive customization of CRS messages.

In the top left corner you will see a red pull-down menu which contains a list of all of the baseline product types that can be formatted to CRS using this software. The list is arranged alphabetically by a leading three letter identifier. Some of these products have a period in them (i.e., SV.W) which indicates a VTEC product code. Others do not (i.e., LEW) which indicate a non-VTEC product.

VTEC products listed do not at times correspond directly to their AWIPS product identifier (i.e., HS.W is issued in AWIPS as a WSW). *When you think about products and configuring them for NWRWAVES, think of them in terms of their VTEC code and not their AWIPS product identifier.* There are legacy AWIPS product identifiers in...
the product configuration list (i.e., SV.W is there for a Severe Thunderstorm Warning, and SVR is also listed as a Severe Thunderstorm Warning). NWRWAVES will use the VTEC phenomena and significance codes as its primary lookup, so the entry for SVR is redundant and only used if for any reason the VTEC line was somehow corrupt or missing from a Severe Thunderstorm Warning.

Both product types (VTEC and non-VTEC) share many common configuration settings. VTEC products have additional unique configuration settings for alert tone settings, based on the VTEC action code. VTEC products are also tracked internally by NWRWAVES, and the advanced feature of MRD Replace is used to facilitate better automation of information replacement in the CRS Broadcast Cycle. In the NWRWAVES distribution file, all these configuration settings are set to “No” and the product dissemination is set to the “Pending” side of the NWRWAVES Browser (as they are shown in Figure 4 above). Sites will need to adjust these settings *by product* to meet their local needs.

**SPECIAL NOTES ABOUT VTEC & SHORT-FUSE WATCHES/WARNINGS**
In the VTEC sense, a tornado watch is defined as a TO.A, a severe thunderstorm watch a SV.A, a tornado warning as a TO.W, and a severe thunderstorm warning as an SV.W. When you begin to configure products in NWRWAVES, you must think of these products as their VTEC hazard, and not their AWIPS product. For example, you would configure all options for tornado warnings under the “TO.W” option in the product list in NWRWAVES. If you wish to have the follow-up SVS statement’s headlines read, you will need to turn the “Include Headlines” option to “On” under the “TO.W”. Why? The AWIPS product does come out as an SVS, but it has the “TO.W” VTEC code in it, which is what NWRWAVES will use to determine what to do with the inbound tornado warning follow-up.

There are legacy AWIPS product identifiers in the NWRWAVES product list (e.g. TOR, SVR, NPW, etc…), just in case a product is inadvertently sent without VTEC coding. You should also configure these just in case.

VTEC coded short fuse watch and warning products are as follows:

- **SV.A** – Severe Thunderstorm Watch
- **TO.A** – Tornado Watch
- **TO.W** – Tornado Warning
- **SV.W** – Severe Thunderstorm Warning

**SPECIAL NOTES FOR SITES WITH MARINE RESPONSIBILITIES**
Marine products such as the Near Shore Forecast (NSF) and Coastal Waters Forecast (CWF) (and soon by directive to include the Offshore Forecast (OFF) and Great Lakes Forecast (GLF)) have numerous VTEC possibilities. You have the option in NWRWAVES to enable/disable VTEC use for these products (settings are established in Section 3 – Tab 4 below), as VTEC can lead to some complicated processing of segments and messages on the air.

If you decide to have NWRWAVES track VTEC, you will need to configure NWRWAVES for each of these VTEC event and significance codes (each is considered a unique product). You will be able to tone alert (by VTEC phenomenon). You will not be able to set a default under the NWRWAVES product types of NSH and CWF (as appropriate) and see the products process. These VTEC event codes are as follows:

- **CF.A** – Coastal Flood Watch
- **CF.S** – Coastal Flood Statement
CF.W – Coastal Flood Warning
CF.Y – Coastal Flood Advisory
GL.W – Gale Warning
HF.W – Hurricane Force Wind Warning
LO.Y – Low Water Advisory
LS.A – Lakeshore Flood Watch
LS.W – Lakeshore Flood Warning
LS.S – Lakeshore Flood Statement
LS.Y – Lakeshore Flood Advisory
MA.F – Routine Marine Forecast
MA.S – Special Marine Statement
MA.W – Special Marine Warning
RB.Y – Small Craft Advisory for Rough Bar
SC.Y – Small Craft Advisory
SE.Y – Hazardous Seas Advisory
SI.Y – Small Craft Advisory for Wind
SR.W – Storm Warning
SU.W – High Surf Warning
SU.Y – High Surf Advisory
SW.Y – Small Craft Advisory for Hazardous Seas
TS.A – Tsunami Watch
TS.W – Tsunami Warning

If you disable VTEC tracking for these marine products, NWRWAVES will treat them as non-VTEC products, and the settings under the NWRWAVES product types of “NSH” and “CWF” (also “OFF” and “GLF”) will be used universally regardless of the VTEC hazards contained in each segment. You will not be able to assign tone alerts by VTEC phenomenon. You would only need to configure these four NWRWAVES product entries (as applicable to your office):

CWF – Coastal Waters Forecast
NSH – Near Shore Forecast
OFF – Offshore Waters Forecast
GLF – Great Lakes Forecast

Starting August 2007, many Weather Forecast Offices (WFO)s have started issuing experimental Marine Weather Warning (MWW) bulletins. The MWW is intended to better inform mariners of adverse marine weather hazards and serve as a dedicated long duration marine Watch, Warning and Advisory product. The MWW bulletin will provide the marine community with more specificity and vital marine hazard information, patterned after the winter weather watch/warning/advisory (WSW) bulletin and the non-precipitation watch/warning/advisory (NPW) bulletin. A complete VTECs support listing of MWW and sample instructions for implementing Small Craft Advisory (SC.Y) is included in Appendix D.

To begin modifying product parameters:
1) Select a product from the drop down menu. (NOTE: These items can be arranged alphabetically by VTEC phenomena or by product type by clicking the appropriate radio button).

2) The settings for this particular product are displayed.

3) Each configuration item on the GUI has a question mark box to its right. Click on that question mark box to see more information on what that particular item governs. Referencing Figure 4 above, here is a breakdown of the interface:

   a) Starting to the immediate right of the product drop-down list, there are two boxes labeled “Issuance Header” and “Followup Header”. Assign the CRS product identifier (middle three letters) into these boxes. **These entries can differ from the VTEC phenomena/significance codes or the AWIPS product identifier!** For example, a Severe Thunderstorm Warning has a VTEC phenomena/significance code “SV.W”, an EAS product issuance code of “SVR” and a follow-up code of “SVS”. You want to ensure that the “Issuance Header” and “Followup Header” are consistent with EAS codes (and resultant CRS Weather Messages). You will see this with much of your winter weather, non-precipitation and severe weather product suite.

   **NOTE:** Where appropriate, the default NWRWAVES distribution file will contain EAS codes for various watch and warning as mandated by directive. Sites should not change the issuance or follow-up headers for warning or watch products without first checking the applicable directives.

   Down the left-hand side of the product configuration interface, you will see yes/no selectable options for the following items labeled “Message Properties”:

   b) **Process as Generic Message Type:** This option, if selected “Yes”, instructs NWRWAVES to assemble a single generic message for each VTEC line/product segment versus the default transmitter specific messages. CRS would then sort the generic weather message on the LAC list assigned to that message. Specific details regarding generic message versus transmitter specific message assembly is discussed in Section 5 – NWRWAVES And Your CRS Database.

   c) **Use MRD Replace on CRS:** This option is grayed out (not selectable) and set to “Yes” by default for all VTEC coded products. The option is also grayed out (non-selectable) for all non-VTEC products with the exception of those listed for exclusion from this rule in the file /awips/adapt/NWRWAVES/NONVTECMRD.txt. If a three letter non-VTEC product is listed in this file, then you can choose the option of using MRD Replace for those non-VTEC products.

   The content of the NONVTECMRD.txt file follows the following example:

   NOW,N
   SPS,Y
   HWO,N

   The NOW, SPS and HWO products can be configured by the local site to use non-VTEC product MRD Replace. Only the SPS will use the MRD Replace in the above configuration. The NOW,
SPS and HWO are the only products listed in the baseline distribution of NWRWAVES, due to somewhat unpredictable behavior that can occur in the absence of VTEC.

More information on what MRD replace is, and how it functions within NWRWAVES, can be found in Section 6 – NWRWAVES OPERATION.

d) **Process Only for Core County/Zone**: If “Yes” is selected, NWRWAVES will process only those product segments whose UGC listings contain the core county (parish/independent city) or core zone as established for each transmitter. This option would be effective for products such as the ZFP where an office may only want to broadcast a specific zone segment for each transmitter on NOAA Weather Radio. Core county/zone settings are global per transmitter, so the same core zone would be used on transmitter ‘ABC’ for the NOW as well as the ZFP.

e) **Process for Non-Routine Broadcast Service Area(s)**: If set to “No”, then an inbound product/segment will not be processed if the UGC line contains only non-routine BSA counties/zones for a given transmitter. If set to “Yes”, then the inbound product will be processed if it contains UGC codes found only in the non-routine BSA settings for that transmitter. Any alarm/alert settings used will be governed by the non-routine BSA alert settings (covered later).

f) **Intro**: An optional text box where you can type in an intro phrase for this product. For example, “NOW HERE IS YOUR SHORT TERM FORECAST” could be used for the “NOW” product. The intro phrase can be used in combination with the transmitter preamble phrase (set in Section 3 above) and also with a county (parish), zone and independent city listing to create lead-ins for your CRS products. **NOTE**: you must have an Intro phrase to include the county/zone list in (f) below. Otherwise, you would not get radio-ready text from NWRWAVES for these two sections.

g) **Include preamble**: If set to “Yes”, the preamble you defined per transmitter in Section 3 will be used as part of the lead-in for this product. You will want to use this option with the Intro or County/Zone list, as the wording will not turn out very cleanly. An example preamble might read, “In the K I D 77 listening area”.

h) **Include County/Zone list**: If set to “Yes”, this will return a listing of counties, parishes, independent cities or zones affected on this transmitter by this product. An example phrase might be, “for the following counties, in Missouri, Bates, Jackson and Johnson. This also includes the following Virginia independent cities, Alexandria and Richmond.” **NOTE**: This feature will only work if you define in Intro in (d) above.

i) **Include Issue Time**: If set to yes, the time the product was run through NWRWAVES will be included. An example phrase would be, “Issued at 3:15 pm CDT”.

If you have options (f) through (i) set to yes in these examples, an intro phrase for the Short Term Forecast (example given in f) might read as follows:

“HERE IS YOUR SHORT TERM FORECAST, for the following counties In the K I D 77 listening area, in Missouri, Bates, Jackson and Johnson”.
j) **Include Headlines:** If set to “Yes”, headlines contained in the inbound product will be formatted correctly and included in the CRS product, otherwise they will be omitted. You will likely want to set this feature to “Yes” for most products that could contain headlines. **NOTE:** You may wish to disable headlines for products produced from GHG, as they duplicate the headline in the attribution statement. This will give you two headlines on the air.

k) **Include Overview:** If set to “Yes”, NWRWAVES will generate a separate overview product, containing the overview and any overview headlines contained in the product. This outbound message will be generated under the CRS ID -- CCCOVRXXX where CCC is the issuing office, OVR is literal, and XXX is the AWIPS Product ID. For example, if product STLWSWEAX contained an overview and this option is set to YES for any of the VTEC phenomena it contains, NWRWAVES would generate a separate overview product under the CRS ID EAXOVRWSW. Since the OVR is a separate message, offices can configure this product as well by modifying the settings in this section under the “OVR - Overview” product from the pull down list.

l) **Include Supplemental Text:** Offices can use the “&&” delimiter to separate the main body of text from a supplemental text section in products. If this is set to “Yes”, then the CRS product will include any text below the “&&” and before the “$$” that signifies the end of the segment (product). Otherwise, anything below the “&&” will be omitted from the CRS product.

m) **Repeat Headline:** If set to “Yes”, then the event headline will be repeated at the end of the text captured from the product. If set to “No”, then the hazard headline will not be repeated. This is useful in the case of a Tornado Warning. The resultant text added would read like this example, “Repeating, a tornado warning has been issued for Jackson County, Missouri, effective until 4:45 PM CDT.” **NOTE:** This functionality exists only for VTEC encoded products.

n) **In Summary Message:** If set to “Yes”, then this particular product (hazard) will be tracked as a summarized hazard, depending on your Summary Message settings that you establish in Tab 3 later. For now, if you wish this hazard to be summarized on your transmitters, set it to “Yes”, otherwise leave it set as “No”.

The right hand side of the GUI interface contains the “Transmission Properties” section of settings (Figure 4a and 4b) for both VTEC and non-VTEC products.
o) For all products, you can toggle between the settings for your routine BSA and your non-routine BSA listening areas. Click the radio button between the two to switch between them. When you are configuring your non-routine BSA settings, the background of this section will switch from grey to an aqua-blue (Figure 4c on the following page).

p) For both VTEC and non-VTEC products, you can toggle ‘Interrupt’ status to “On” or “Off”. If set to “On”, all instances of this product that are sent to CRS will interrupt the routine broadcast.

For the ‘Alert Tones’ setting for both product types, you have three options: “On”, “NWR Only” and “Off”. If set to “On”, then the 1050Hz and WRSAME tones will be sent with this product.

**IMPORTANT NOTE:** For all short fuse watches and warnings, including Tornado Watches/Warnings, Severe Thunderstorm Watches/Warnings, and Flash Flood Warnings,
leave the default settings of “On” for the “Alert Tones” option, and for the VTEC options of “EXA” and “EXB” as shown on the left-hand side of Figure 4a. By default, for all short-fuse warning products listed in Directive 10-1710, alert tones come in NWRWAVES set to “On” for initial issuance, as well as for extensions in area (EXA), and extensions in both area and time (EXB).

The “Off” setting will generate no tones at all.

The “NWR Only” setting will allow you to use your CRS Message Type settings to govern whether to include just the SAME tones and/or the 1050 Hz tone (if defined under unique properties).

IMPORTANT NOTE: Take special care when using the “NWR Only” option for alert tones. Unless you’ve specifically assigned the 1050 Hz tone *by transmitter* under the ‘Transmitter Specific’ blue button on the CRS Message Type GUI, the “NWR Only” option will simply check for any SAME settings in your Message Type to use. If you haven’t done anything special to assign the 1050 Hz tone by product, you can think of the “NWR Only” option as a “SAME-only (if configured)” option.

For the VTEC coded product, this setting only governs the “NEW” action code within the VTEC line. You can then set alert tones settings for additional VTEC action codes shown on the GUI interface. These can be quite powerful in configuring alert tones for your warning suite. For example, a “TO.W” Tornado Warning product would have the “Alert Tones” set to “On” for initial issuance. As follow-up SVS products are issued, you could set the “CON” action code to “NWR Only” (and your SVS Message Type set to WRSAME only in your CRS database) to send only SAME tones for follow-up SVS statements that continue the Tornado Warning. Your setting for “EXP” and “CAN” could be set to “No”, so that no tones of any kind are sent for an expired or cancelled warning.

For non-VTEC products, the “Alert Tones” setting governs any issuance of the product (governed by AWIPS product identifier). An example of this is the “FFW” (non-VTEC coded product). While the follow-up product for an FFW is an FFS, “FFS” has its own product identifier listed in the NWRWAVES product database, therefore the alert tone settings of the FFW will not be used for the FFS. When this product becomes VTEC coded, the product identifier “FF.W” in the NWRWAVES product list will then be used, and the alert settings for follow-up statements will be those you assign.

q) **Silence period**: If this box is checked, you can override the “Alert Tones” settings for specific periods during the day. You set these by clicking on the boxes labeled “Begin Time” and “End Time”. Figure 4d shows the GUI used to set these times.
If this **Silence period** box is selected and the silence period is specified, NWRWAVES will send a Red Banner (See Figure 4d.1) when the silence period is ended to alert the forecasters to use the CRS system to resend the product with Alert/Interrupt if required.

![Figure 4d – Silence period time setting](image)

**Figure 4d.1 – Red Banner when Blackout period has ended**
The last section of the “Transmission Settings” GUI sets the disposition, periodicity and disposition of CRS messages (Figure 4e).

Figure 4e – final transmission settings

r) **Storage**: From the pull-down menu, you can select from ‘Active’, ‘Inactive’, ‘Synthetic Speech Override’, or ‘VIP (Default)’. By default, every product in the baseline NWRWAVES distribution is set to ‘VIP (Default)’.

s) **Transmission Status**: If set to ‘CRS’, this product will be sent directly to CRS for airplay. If set to ‘Pending’, the outbound product(s) will be placed in the Pending side of the NWRWAVES Browser. Every product in the NWRWAVES distribution is set to ‘Pending’.

**NOTE**: Sites should leave the disposition of CRS products to ‘Pending’ for every product until they are ready to begin active use of NWRWAVES. At that time, products can be converted from ‘Pending’ to ‘CRS’ one-by-one to ensure a smooth transition.

t) **Periodicity**: You can type in a number (in minutes) or select a number from the drop-down menu, to assign to this product. If you wish to remove a periodicity setting, you can left-click in the text box next to the triangle, backspace over the number, then left-click in the gray area to reset it to a blank.

u) **CRS Effective Time**: This setting allows you to establish the minute (00-59) at which you’d like an outbound message to begin broadcast. From an automated standpoint, this feature is best used with the periodicity setting above to establish set times for message broadcasting. For example, setting a product’s effective time to 00 and the periodicity to 15 minutes would cause that message to play with predictability “on the 15’s” (provided the message is not in a Message Group in your Broadcast Suite on CRS).

v) **Default duration**: You can type in a number (in minutes) or select one from the drop-down menu to set a default duration for any product that is missing a UGC line. If you have a UGC coded product but do not wish to play it for an extended time, you can click the “Use in lieu of UGC” radio button to force the default duration time no matter what the UGC expiration time is in the product.
w) When all these settings have been configured for this product, click the green “Save Edits” button at the top of the interface.

4) Repeat steps 1, 2, and 3a through 3w for every product in the drop down list that your office processes for CRS.

5) You may find that a local product issued by your office is missing from the product configuration file, or that a regional/national product is also missing. You can also add products locally to your product configuration file. In the case of the regional/national product in scope, you will want to add it but also post it as a future addition for the baseline to the NWRWAVES list server.

To add a local product to the Product Configuration list:

a) Click on the pink “Add Product” button found towards the top right of the Product Configuration tab GUI.

b) A pop-up GUI interface will display (Figure 4f). In the text boxes provided, fill in the basic information of the new product, similar to what is displayed below.

c) If the product is VTEC encoded, be sure to include a period between the second and third letter of the VTEC Phenomenon or Product PIL entry. Otherwise, just enter the three letter identifier of the CRS Product PIL.

d) Click the ‘OK’ button once these entries are complete.

e) You will be returned to the Product Configuration Tab GUI. Complete step 3 in its entirety to finish configuration of this new product. When done, click the green “Save Edits” button to save your new product. NWRWAVES will double-check to ask if you really want to save this new product to your product configuration list.

Product configuration data is stored in /awips/adapt/NWRWAVES in the file product.cfg. Sites will not hand-edit this file! It is comma delineated, and there is a significant risk of messing up critical NWRWAVES configuration data by hand-editing the file. Sites will use the NWRWAVES Setup GUI to make adjustments to product configuration. Your local product.cfg file will be restored every time you upgrade to a newer version of NWRWAVES.
NWRWAVES functions similarly to CAFÉ with respect to convective watch message generation. CAFÉ would process either the WOU or WCN bulletins, and through the use of VTEC, CAFÉ would then generate an output message customized for NWR internally, using VTEC phenomenon/significance codes and VTEC action codes to govern tone alert settings (CAFÉ WBC4CRS formatter).

For CONUS sites, NWRWAVES begins with the WOU(x) issued by the SPC for all convective watches. The WOU(x) includes all affected land FIPS codes and marine zones (regardless of state and CWA), VTEC, with the VTEC phenomenon and significance codes, the VTEC action code (NEW, CON, CAN, etc.), and an event tracking number. The WOU(x) product makes a perfect vehicle from which to prepare all convective watch bulletins. NWRWAVES will build a watch message from the decoded elements of the WOU (FIPS/marine zone codes, VTEC) through internal code and phrasing. No part of the actual WOU(x) bulletin is included in the outbound CRS Weather Message. WOU update and cancellation products are ignored. **NOTE:** For OCONUS sites where the SPC does not issue convective watches, NWRWAVES will use the initial WCN for tone alerting purposes. This setting is governed by your OCONUS global variable sourced from the AWIPS operating environment.

For both CONUS and OCONUS sites, NWRWAVES will then use any *followup* WCNxxxx messages to track the status of a convective watch. NWRWAVES will internally create a message/call-to-action for all affected areas, with CRS Weather Message attributes (SVA, TOA) based on the hazard. No part of the actual WCN bulletin is included in the outbound CRS Weather Message.

Alert tones will be sent/not-sent based on the VTEC action code customized by each office under Tab 2 above. By default settings (per directive 10-1710), the only VTEC action codes that would trigger a new tone alerted SV.A/TO.A product from a WCN would be the codes “EXA” and “EXB”, which add new counties to a watch through the WCN.

**NOTE:** You will need text database triggers for all WOU statements (WOU0 through WOU9), your local office WCN, and any other surrounding office’s WCN’s that apply to areas covered by your transmitters. Text database triggers are covered in Section 4.

### SPECIAL NOTES ON NON-WEATHER EMERGENCY MESSAGES (NWEM)

NWRWAVES will be fully compatible with the new HazCollect system slated for operational status in 2006. NWRWAVES will replace the NWEM CAFÉ formatter formally in OB9.2. In anticipation of this switch in 2007, all the NWEM CRS Weather Messages are included in the default product configuration, with EAS PILS (issuance header and followup header) that are consistent with 10-1710. **OB9.1 or older**, HazCollect is still calling the legacy NWEM CAFÉ formatter. **NOTE:** In the pre and post-OB9.2 environment, sites will **not** establish text database triggers for any NWEM messages! HazCollect calls its NWR formatter internally, so there is no need to duplicate the call in a separate trigger. No action is required prior to OB9.1 to configure NWRWAVES for NWEM messages.

After installation of OB9.2 NWRWAVES will become the default NWR formatter for NWEM products. This migration will provide logic to place “test products” into the NWRWAVES pending directory and software defect support by the OPS23 CRS Staff. Another enhanced capability of NWRWAVES is that a site can change the
EAS PILS (issuance header and followup header) if state or local broadcaster policy is to not use the new EAS PILS. For example, the Missouri Broadcasters Association has asked the NWS to use the EAS PIL “CEM” for all these new hazards until further notice, as many member stations have not upgraded their EAS equipment. All these NWEM hazards can be set to convert their AWIPS product identifier to a “CEM” CRS Weather Message.

Once all products have been configured for your site as needed, you are ready to move on to the Summary Message Configuration tab.

**Tab 3 – Summary Message/Miscellaneous Settings Configuration**

After you have set properties for all your products, the next step is to configure summary message settings. The summary message can be a powerful addition to your CRS product suite. Generating a summary of active hazards was a function of the WWA NWR formatters.

The basic function of the NWRWAVES summary message function is as follows: sites can configure their product configuration settings to select which (if any) product types are tracked by NWRWAVES for summarization purposes. Once a local threshold is exceeded, NWRWAVES will generate a message summarizing any applicable hazards. The summary message can be used within a regular Broadcast Suite to bring a periodic short summary of these hazards, or the summary message can itself be used to trigger a higher suite on your CRS, thus shortening a lengthy broadcast if many hazards are in effect.

In the NWRWAVES Setup GUI, click on the third tab, which is labeled **Summary Message/Misc. Settings**.
Figure 5 – Summary Message Configuration Tab

Under this tab you can select the following options:

1) **Summary Message PIL:** Your AWIPS legacy state node will make up the first three letters of the summary PIL, and the last three letters (labeled ‘NWX’ on the GUI) will be the unique three-letter mnemonic that you gave each of your transmitters. You can select the middle three letters of the summary message. For example, for transmitter MCI for the Kansas City office (state node is STL), the summary message type (from the GUI above) would be ‘STLSUMMCI’.
2) **# of Bulletins to Trigger Summary:** This is the minimum number of active hazards (with the summary message option set to “YES” in the product configuration) that are required before NWRWAVES begins to generate a summary message.

3) **Include intro (yes/no):** If “Yes” is selected, NWRWAVES will include the text, “The following watches (warnings) (and advisories) is/are in effect” as an intro phrase for its summary of active hazards.

4) **Include Preamble (yes/no):** The same as the Product Configuration setting, if this is set to “Yes”, then NWRWAVES will include the preamble in its intro phrase for the summary message.

5) **Include Issue Time? (yes/no):** Identical to the Product Configuration setting, if this is set to “Yes”, then NWRWAVES will include the time the summary message was prepared in the intro phrase of the summary message.

6) **Transmission:** You can set whether the summary message is sent directly to CRS for airplay, or whether it gets placed in the ‘Pending’ side of the NWRWAVES Browser. The default setting is set to ‘Pending’.

The top right hand side of the GUI contains settings that affect the wording of the summary message, as well as the wording of the optional lead-in and county/zone listing phrases by transmitter (that you established under Tab 1 earlier):

7) **County (FIPS) Codes:** If this radio button is selected, NWRWAVES will include the words ‘county’, ‘parish’, ‘counties’, ‘parishes’, or ‘independent city(ies)’ after its summary phrase “for the following”. Proper wording will be used for whether the FIPS code is for a county, a parish, or an independent city.

8) **Zone codes:** You can govern (by selecting the appropriate option) whether for UGZ codes NWRWAVES follows its summary phrase “for the following” with the word ‘county (parish), counties (parishes)’, ‘area(s)’, ‘location(s)’, or simply nothing at all.

9) **Marine zones codes:** Functions like #8 but for marine zones. Options are to follow the phrase “for the following” with nothing, ‘coastal location(s)’, or ‘marine location(s)’.

10) **Marine Zones Sort By:** Options are Alphabetical or Numeric.

11) **Land Zones Sort By:** Options are Alphabetical or Numeric.

The bottom left hand side of the GUI contains settings that affect the NWRWAVES Browser:

12) **Generate Red Banner Alert for Stale Products in NWRWAVES Browser Pending Folder:** the NWRWAVES Browser provides an optional D2D red banner alert when products have become stale in the Pending side of the browser. This option is designed to prompt the forecaster/HMT to take an action on these messages -- transmit or delete them. To turn this feature on, simply check the “Yes” option box and then set the time duration at which products in the Pending side of the browser will become “stale” and generate the red banner alert.

13) The NWRWAVES Browser also provides a means of allowing a user to filter the Sent list by one or more products. Previously the default list of available products could be modified by editing a text file. This has been integrated into the GUI. To Add/Remove product types from the NWRWAVES Browser filter list, simply use the GUI provided. To Add a product, enter its ID into the text box and click the right arrow. The Remove a product, select it with the left mouse from the list and click the left arrow button.
The bottom right hand side of the GUI contains settings that affect NWRWAVES purging capabilities.

14) On the bottom right hand side of the GUI, you can change the number of days that NWRWAVES will store output and debug related files in the following directories: DEBUG, LOGS, OUTPUT, INPUT and TEST. It is recommended that you keep this number at least 10 days for troubleshooting purposes, but not more than 15 days. A cron task will automatically purge files older than the number of days established in this section.

When you are satisfied with the summary settings, click the green "STORE CONFIGURATION SETTINGS" to save them.

You have now configured the first three tabs of the NWRWAVES Setup GUI. If your site is not a coastal site, and you have no marine responsibilities, you have completed the software configuration, and you may exit the NWRWAVES Setup GUI (File pull-down menu, click “Save and Exit”). You may proceed directly to Section 4 – Setting AWIPS Text Database Triggers, to configure AWIPS to send products through NWRWAVES.

If you do have marine responsibilities or are a coastal office, you must complete Tab 4 – Marine/Tropical Product Configuration before you are done.

**Tab 4 – Marine/Tropical Product Configuration**

Most marine products will play through NWRWAVES with little customization needed for the individual products. Extensive work in the word replacement dictionary (Section 6) or VIP will be needed to replace acronyms and abbreviations with full word phrases, to render a quality NWR product.

There are four main sections under the Marine/Tropical Product Configuration Tab (Figure 6), Marine VTEC Processing, Hurricane Local Statement, Tropical Storm Product and Great Lakes Forecast.
Figure 6 – Marine/Tropical Product Configuration GUI

- Hard copies uncontrolled. Verify effective date prior to use. -
**Marine VTEC Processing**

Next to the label “Process VTEC for Routine Marine Products (CWF, NSH, OFF, GLF)?” is a ‘yes/no’ option. Check ‘yes’ to enable VTEC usage in your marine products. Check ‘no’, and NWRWAVES will process all marine products but ignore any VTEC lines. **NOTE:** see pages 11 and 12 of NWRWAVES documentation for details on whether you choose to use or not use VTEC with your marine product suite.

**Hurricane Local Statement**

The Hurricane Local Statement (HLS) Section is used to configure how the HLS is formatted for airplay. NWRWAVES can be configured to replace the section headlines form the HLS with more radio friendly text. An example of this can be seen in figure 6. You can also omit a section headline individually by leaving the **REPLACED CRS PHRASE** section blank.

To configure this section:

1) If you do not want any of the sub-headlines broadcast from your office’s (or an adjacent office’s) HLS, simply check the box next to the label “Do Not Broadcast HLS Sub-headlines”.

2) If you do wish to broadcast these sub-headlines, left-click in the box labeled, “Number of sub-headlines”, enter the number of sub-headlines your office (or an adjacent office) uses as their standard HLS format, and then left-click the blue ‘Set’ button. Sub-headlines are those phrases within the HLS that look like this: “…WATCHES/WARNINGS…”

3) A number of boxes will appear in the HLS section, which will match the number you just entered.

4) For each sub-headline, enter its original text from the HLS product, and then list its CRS replacement phrase to the right of the original phrase. For example, you might want to replace the phrase “…WATCHES/WARNINGS…” with “watches and warnings in effect” to enhance airplay.

In OB9, a couple of new features related to the processing of the HLS products have been introduced. Please refer to the Appendix- F for details.

**Tropical Cyclone Product (TCP)**

From the 2010 hurricane season on, only the “DISCUSSION AND 48-HOUR OUTLOOK” text section of the TCP products (see Figure-6 highlighted in red) will be processed by the NWRWAVES. The others sections of the TCP text will be discarded in order to reduce the broadcast cycles in the NWR.

There are two options you can configure in this section that affect processing of the TCP product from TPC and HPC.

5) Process NHC Public Cyclone Advisories? (yes/no): if this is set to ‘No’, then NWRWAVES will **not** process the TCPAT(x) bulletins issued by the Tropical Prediction Center, even if you have the text database trigger set to send the TCPAT(x) bulletin to NWRWAVES. **If the answer is set to “Yes”,** you also have additional option to delete the metric values found in the DISCUSSION AND 48-HOUR OUTLOOK section. All the metric speed (in KM/HR) or distance (in KM) values can be filtered out. **NOTE:** The administrator must select this option from the GUI first since the default is not to delete the metric values.

6) Process HPC Public Cyclone Advisories? (yes/no): this flag tells the software whether to ignore or process the TCPAT(x) issued by the HPC once a tropical system has made landfall. If the answer is
set to “Yes”, you also have additional flags to either read the entire advisory as written, or to filter out any tabular data (such as inland rainfall totals) that HPC may include in the product.

SPECIAL NOTES ON TROPICAL CYCLONE PRODUCT PREPARATION

NWRWAVES functions much differently than CAFÉ with respect to its tropical cyclone message generation. CAFÉ would process the TPC TCPAT(x) bulletins, and through the use of phrase searches and key wording, CAFÉ would then generate an output message customized by a latitude/longitude box, and where the storm was with respect to this box. If it met certain phrase and location criteria, the inbound CRS Weather Message would be assigned the proper EAS code (TRA, TRW, HUA, HUW) for alert on NWR.

NWRWAVES treats the TCP just like the HLS, as a regular product that contains information on the storm. 

NOTE: A strong recommendation for sites is that they should invoke Generic Message Type assembly for the TCP product in NWRWAVES, so as to minimize the delay of producing transmitter specific output. The product suite needed to process hurricane watch/warning and tropical storm watch/warning bulletins is the new HLS instead of the TCV or TCVAT(x) suite issued by the TPC since the HLS is carrying the same VTEC. If a site has both HLS and TCV, there will be failures of the HLS.

NOTE: You will need text database triggers for HLS to properly prepare and disseminate tropical watches and warnings. Text database triggers are covered in Section 4.

Great Lakes Forecasts

The Great Lakes Forecast (GLF) section is used to configure how the GLF is formatted for airplay. These capabilities replace the customized settings found in the CAFÉ GLF formatter. Effectively October 2007, all GLF would change from the non-segmented format to a new segmented format that would allow WFOs issuing the GLF products to be more area-specific. For example, the geographic divisions used in the GLF such as “NORTH HALF” will be replaced by smaller marine zones grouped in segments by similar weather conditions. The old non-segmented format will be completely removed in OB9 release.

To configure (if needed) the Great Lakes Forecast options, complete these steps:

1. Choose a Great Lake from the Select a GLF Product Below dropdown list.
2. Enter the AWIPS product identifier for the GLF (example CHIGLFLM).
3. Enter total number of possible breakpoint sections in the GLF (usually, this is denoted within the text of the product such as “NORTH HALF”, “SOUTH HALF”, etc.) This should include all possible breakpoints.
4. Click the Set button
5. Edit the Optional Intro Statement if desired. Any text entered will be used as an intro phrase for the GLF output.
6. Click the Broadcast Synopsis (If Applicable) button if desired. If this is checked, the “SYNOPSIS” section of the GLF will be included in the CRS output, otherwise it will be omitted.
7. Fill in the SECTION HEADERS entry boxes to reflect the break points used in the GLF. Again, an example would be phrases such as “NORTH HALF” and “SOUTH HALF”.
8. For each SECTION HEADER, click each transmitter checkbox that you wish to have formatted for airplay. (This feature allows WFO’s to assign certain GLF sections to certain transmitters).
In Figure 6 above, the configuration settings for the GLF are as follows:

- Process the AWIPS product (if triggered) CLEGLFLE for Lake Erie
- Do not include any intro statement
- Broadcast the “.SYNOPSIS” section within the output of the CRS GLF forecast

Section headers (as defined) would only play for those transmitters checked for each header.

When you have completed all applicable sections of this tab, click the green STORE MARINE CONFIGURATION SETTINGS button to save your settings. You can also establish periodicity and/or effective time settings for the GLF under the product configuration tab. NOTE: if you checked the option “No” for using VTEC for marine products, the “GLF” entry is the option you’ll need to configure. Otherwise, you will need to configure for each individual VTEC header to accomplish this task.

You have now completed customization of NWRWAVES for your office. Proceed to Section 4 – Setting AWIPS Text Database Triggers, to configure AWIPS to send products through NWRWAVES.

5. SETTING AWIPS TEXT DATABASE TRIGGERS

NWRWAVES software is executed as an automated response to an AWIPS text database trigger (Postgres). Offices that have used CAFÉ will notice some degree of familiarity in the establishment of database triggers.

You can, and should, run triggers for both your CAFÉ and NWRWAVES formatters during the transition period between CAFÉ and NWRWAVES. You can run NWRWAVES in a test mode, which is what you will need to do in this transition period to ensure a smooth switch between the two formatters. Once you decide to switch a product/products over to NWRWAVES in an operational mode, you would need to remove the legacy CAFÉ triggers. To establish a suite of triggers needed for NWRWAVES operation, perform the following steps:

1) Log into an AWIPS workstation with a user account that has fxalpha group privileges and open up a terminal window.

2) Change directories to the trigger directory.

   cd /data/fxa/siteConfig/textApps

3) Make a backup copy of your site trigger information, in case you need to revert back to it.

   cp siteTrigger.template siteTrigger.template.mmddyyyy

   where ‘mmddyyyy’ is the numeric current date

4) Edit your site trigger configuration.

   gedit siteTrigger.template &

When you view this file, you will see a listing of AWIPS product identifiers, along with an action that is to be completed upon receipt of that product identifier into the text database. If you are a CAFÉ site, there will be
several entries that call CAFÉ scripts to format data. An example of a CAFÉ entry in your site trigger configuration will look something like the following:

**STLSVREAX** /home/CRS/SVR/nwrsvr.csh STLSVREAX

**Do not change or remove these CAFÉ text database triggers!** Your initial configuration will be to set NWRWAVES into test mode, so you can monitor its performance and ensure that it will work successfully with your CRS database prior to making the switch over to operational mode. The WWA NWR formatters do not use text database triggers.

5) For each text product that you wish to be formatted by NWRWAVES, you will need to add a line to the siteTrigger.template file using the following syntax (CCCNNNXXX is the 8/9 letter product PIL):

```
CCCNNNXXX /awips/adapt/NWRWAVES/nwrwavestest.csh
```

The script ‘nwrwavestest.csh’ is used by NWRWAVES to execute the formatter in TEST mode.

You will need an entry for each product that you wish to send through NWRWAVES. This entry (the CCCNNNXX field) is the exact AWIPS product identifier for each product you wish to process! **An example of text database triggers set at WFO Pleasant Hill for NWRWAVES execution can be found in Appendix B.**

**SPECIAL NOTES REGARDING TEXT DATABASE TRIGGERS**

**NWEM messages:**
Sites do not need to add any triggers for their NWEM messages. In OB9.1 or older NWEM messages are formatted out of HazCollect, and HazCollect will call its own Café NWEM formatter internally.

**Starting with AWIPS OB9.2** NWRWAVES will automatically become the default formatter for processing all NWEM messages. The old standalone Café NWEM formatter will cease to be used for HazCollect NWEM products. The purpose of the switch is to provide HazCollect with Spanish support and to resolve issues that existed in the legacy Café formatter. It should be noted that installation of OB9.2 is the preferred manner to migrate from the Café NWEM formatter to NWRWAVES. If sites are unable to install AWIPS OB9.2 due to emergency circumstances and would like to MANUALLY migrate Café NWEM to NWRWAVES for HazCollect, the following instructions should be executed:

1. log in as fxa username
2. cd /awips/fxa/bin
3. cp /awips/fxa/bin/transmitHazardWarnings.pl /awips/fxa/bin/transmitHazardWarnings.plORG
4. make the following changes

In the send to CRS section of the transmitHazardWarnings.pl script change

```
my $formatter = "'/home/CRS/NWEM/nwrnwem.csh $product_id $crs_section $trans_file";
```

- Hard copies uncontrolled. Verify effective date prior to use. -
my $formatter = "/awips/adapt/NWRWAVES/nwrwaves.csh $product_id";

5. Save of the transmitHazardWarnings.pl script
6. ls -lart transmitHazardWarnings.pl

Make sure the permission, owner and group should look like the below example

-r-xr-xr-x 1 fxa fxalpha 29964 Aug 10 2009 transmitHazardWarnings.pl

**Convective watches:**
CONUS sites will need to include all ten WOU(x) products issued by the Storm Prediction Center, along with all WCN products that apply to your CRS broadcast service areas. OCONUS sites only need the WCN products. For the WCN products, these will include your WCN, and any WCN’s issued by surrounding offices. For the WOU(x) products, a section of your siteTrigger.template will need to include these entries:

```
CCCWOU0    /awips/adapt/NWRWAVES/nwrwavestest.csh
CCCWOU1    /awips/adapt/NWRWAVES/nwrwavestest.csh
CCCWOU2    /awips/adapt/NWRWAVES/nwrwavestest.csh
CCCWOU3    /awips/adapt/NWRWAVES/nwrwavestest.csh
CCCWOU4    /awips/adapt/NWRWAVES/nwrwavestest.csh
CCCWOU5    /awips/adapt/NWRWAVES/nwrwavestest.csh
CCCWOU6    /awips/adapt/NWRWAVES/nwrwavestest.csh
CCCWOU7    /awips/adapt/NWRWAVES/nwrwavestest.csh
CCCWOU8    /awips/adapt/NWRWAVES/nwrwavestest.csh
CCCWOU9    /awips/adapt/NWRWAVES/nwrwavestest.csh
CCCWCNXXX  /awips/adapt/NWRWAVES/nwrwavestest.csh
```

CCC should be set to your legacy state node for all WOU products and for your WCNXXX product. CCC should be set to the legacy state node for all other office WCNXXX products that you need to cover your broadcast service areas.

For tropical cyclone products needed to generate Tropical Storm Watches/Warnings, and Hurricane Watches/Warnings, you will need to include HLS products issued by the WFOs. If your site is a coastal office, for the HLS product, a section of your siteTrigger.template file will need to include these entries:

```
CCCHLSXXX  /awips/adapt/NWRWAVES/nwrwaves.csh
```

**NOTE:** The current OB9 version does not support the TEST mode for HLS product.

6) After you have finished adding in the suite of triggers, we will need to save the file and re-localize for these new text database triggers.
7) For site on AWIPS II software, log onto the DX3 or DX4.

```
su – fxa ( or as root)
cd /data/fxa/sdc
./config_awips2.sh triggers XXX (where XXX is the 3 characters site ID)
```

Be sure to run “/awips2/fxa/bin/subscription –o read -t ldad –r ldad” command on any X-term window to verify the text database triggers have been set up accordingly.

The end result of completing steps (1) through (7) is that NWRWAVES will begin to process products in TEST mode for your site. In TEST mode, generated CRS output will be placed in the /awips/adapt/NWRWAVES/TEST directory instead of being sent to CRS or the ‘Pending’ side of the NWRWAVES Browser. You can use your favorite editor, viewer or browser to review the output over time, to ensure that NWRWAVES is processing inbound data and producing correctly formatted outbound CRS products.

The end result of completing steps (1) through (7) is that NWRWAVES will begin to process products in TEST mode for your site. In TEST mode, generated CRS output will be placed in the /awips/adapt/NWRWAVES/TEST directory instead of being sent to CRS or the ‘Pending’ side of the NWRWAVES Browser. You can use your favorite editor, viewer or browser to review the output over time, to ensure that NWRWAVES is processing inbound data and producing correctly formatted outbound CRS products.

Having completed Sections 1 through 4 through this point, you have successfully configured NWRWAVES for your local site. You have established transmitters (broadcast programs) with the appropriate listing of county (parish), independent cities and zone codes. You have customized your product list for processing, and you have established the AWIPS products from your text database that are to be processed through NWRWAVES.

NWRWAVES is now running on your AWIPS in TEST mode.

The end result of your work to date will be visible in the /awips/adapt/NWRWAVES/TEST directory, where NWRWAVES will place its output for your review. You will need to run in TEST mode until you have the opportunity to properly configure your CRS database in Section 5, and you also have the chance to review NWRWAVES output and validate that it is working properly. Do not plan to transition to an operational mode until both of these have been accomplished and the results are acceptable to you.

Perform steps 9 through 15 below only when you are ready to begin operational use of NWRWAVES.

TRANSITIONING PRODUCTS TO OPERATIONAL MODE

When you are ready to begin transitioning NWRWAVES to operational status, you can do this on a product-by-product basis. **Do not plan to move from TEST to OPERATIONAL mode in NWRWAVES** until you ensure your CRS is correctly configured to accept the format of Weather Messages from NWRWAVES.

Section (5) – NWRWAVES AND YOUR CRS DATABASE gives you the information you need to ensure your CRS is ready to accept NWRWAVES output.
The steps necessary to transition a product or products to operational mode are as follows:

9) Repeat step (1) through step (7) above in Section 4. For all instances of a product in step (5) that you wish to transition to operational status, replace the call:

```
/awips/adapt/NWRWAVES/nwrwavestest.csh
```
with
```
/awips/adapt/NWRWAVES/nwrwaves.csh
```

10) Remove the associated CAFÉ trigger for this product.

**NOTE:** There may be multiple AWIPS product identifiers for this product if you format other WFO AWIPS products for this CRS product type. Be sure to remove all applicable CAFÉ product triggers for this product or you will see duplicate CRS Weather Messages!

**NOTE:** If you are a site that uses the WWA NWR formatters, you will need to go into the WWA Admin GUI and disable NWR formatting for this product type. Alternatively, you can use GHG in lieu of WWA, as the WWA NWR formatter only works for products actually prepared through WWA.

11) Repeat steps (7) and (8) to re-localize your new triggers to establish this set of new products to operational mode.

Once you have set the products to process operationally, the resultant CRS output will (by default) be placed in the ‘Pending’ side of the NWRWAVES Browser. This is because your Product Configuration was set to place the output there in your original configuration. To facilitate automated transmission to CRS:

12) Go back to Section 3 – Tab 2.
13) Repeat steps (1), (2) and (3r) under the ‘Product Configuration’ tab. You will be switching the “Transmission Status” section from ‘Pending’ to ‘CRS’ (Figure 7).
14) Click the green “SAVE EDITS” button to save the product configuration
15) Repeat steps (13) and (14) above for all products you are transitioning to operational mode and automated CRS broadcast.

![Figure 7 – example of Transmission Status set to automated relay to CRS](image)

**Selectively specify transmission status based on issuing WFO in OB9:**
In OB9, the local WFO will have the capability to place specified products particularly for other offices’ products to go to the Pending window for further review/editing if necessary. A new input argument (e.g. – EXTERNAL with 2 dashes) can be passed to the nwrwaves.csh to redirect the output to the Pending directory as follow:
The example CCCZFPXXX product will appear in the ‘Pending’ side of the NWRWAVES Browser. Manual intervention through the browser will be required to send the output to CRS.

6. NWRWAVES AND YOUR CRS DATABASE

NWRWAVES was initially designed to produce transmitter (broadcast program) specific output, to best focus the content of Weather Messages on the listening area of that transmitter. In this paradigm, unique CRS Weather Messages would be produced for each product, or each segment within a segmented product, using a standardized naming scheme.

After NWRWAVES deployment, issues were noted regarding the volume of messages produced by NWRWAVES, and the subsequent affect that multiple transmitter-specific messages were having on the CRS architecture (specifically the VIP processor). As a result of these issues, **NWRWAVES allows an office to opt for transmitter specific –OR- “generic” message assembly.** If a site opts for generic Message Type assembly, only one Weather Message will be produced for CRS per each valid product segment; its ultimate destination on your CRS Broadcast Cycles will be governed by the LAC’s assigned to the message.

**NOTE:** Generic message type assembly is **STRONGLY** recommended for all short-fuse warning products (SV.W, TO.W, FFW, FFS, FF.W) so as to minimize delays in VIP processing and subsequent alert tone broadcast. Generic message type assembly should also be invoked for products that yield identical output on each transmitter (i.e., RWS/SWS, PNS type products) and anything else that an office feels comfortable in switching.

The assembly of both types of messages is as follows:

**TRANSMITTER SPECIFIC MESSAGE TYPES**
The standard nomenclature for a CRS Message Type is **CCCNNNXXX**. This format is very similar to AWIPS product identifiers. NWRWAVES prepares its transmitter specific Weather Messages for the three components (CCC, NNN, XXX) using the following logic:

- **CCC** – derived from your AWIPS legacy state node environmental variable. This is sourced from the file /awips/fxa/data/localizationDataSets/XXX/AFOS_CCC_Name.txt. ‘XXX’ is your AWIPS site identifier.

- **NNN** – the middle three letters will come from your settings for each product in the Product Configuration settings. Remember, the term ‘product’ is not just inclusive as a one-to-one relationship to an AWIPS product identifier. ‘Product’ refers to the combination of VTEC phenomenon and significance code when appropriate (several AWIPS product identifiers can have multiple VTEC phenomenon/significance code combinations such as the AWIPS WSW product). The middle three letters are either the “Initial Header” or “Followup Header” entries (based upon VTEC and follow-up statements) as you assigned back in Section 3 – Tab 2 (Figure 8).
Figure 8 – an example of NWRWAVES product SV.W and it’s unique NNN codes assigned by this product type

XXX – the last three letters used in Weather Message assembly are the three-character mnemonics you assigned unique to each transmitter back in Section 3 – Tab 2.

In the case of the Kansas City/Pleasant Hill office, our AFOS state node identifier is STL. We have eight transmitters (MCI, STJ, IRK, MIA, CRL, CLN, MYV, TTN). Therefore, for the NWRWAVES product SV.W (Severe Thunderstorm Warning) as shown above in Figure 8, an inbound AWIPS product of STLSVREAX for all of our CWA would yield the following Weather Messages (depending on the VTEC action code):

Initial Issuance (/O.NEW.KEAX.SV.W)          Follow-up Issuance (/O.CON.KEAX.SV.W)
STLSVRMCI      STLSVSMSCI
STLSVRSTJ      STLSVSSTJ
STLSVIRK       STLSVSIRK
STLSVRMIA      STLSVSMA
STLSVRCL       STLSVSCL
STLSVRCRL      STLSVSCRL
STLSVRCRL      STLSVSCL
STLSVRMYV      STLSVSMYV
STLSVRTTN      STLSVSTTN

For non-VTEC coded products, the only header that will be used by NWRWAVES will be the “Issuance Header” entry, as there is no way to accurately determine if the product is an initial issuance or a follow-up. Thus, you will only need to ensure the presence of the list under the “Issuance Header” in your CRS database.

Therefore, in your CRS database, you will need to ensure that for every product you produce through NWRWAVES, you have a unique Message Type defined that matches this naming scheme, for both VTEC and non-VTEC products. In the case of the Pleasant Hill office where we have eight (8) transmitters, every VTEC encoded product will yield a maximum of 16 unique Message Types, and a maximum of eight unique Message Types for every non-VTEC product (or VTEC product where the two headers are the same).

**GENERIC MESSAGE TYPES**

In generic message type assembly, NWRWAVES makes extensive use of the inbound AWIPS product identifier (PID) for assembling its equivalent Message Type). For all inbound AWIPS PID’s that are nine characters in length (i.e., “STLRWSEAX”), the resulting outbound CRS Message Type used will match the inbound product’s identifier exactly (STLRWSEAX). If the inbound AWIPS PID is an eight character identifier (i.e., “STLRWSMO”), NWRWAVES will yield a nine character Message Type using the logic below (in this case, STLRWSSMO).

NWRWAVES prepares its transmitter specific Weather Messages for the three components (CCC, NNN, XXX) using the following logic:

**CCC** – derived from the first three letters of the AWIPS Product Identifier (PID) from the inbound product.

**NNN** – the middle three letters will come from your settings for each product in the Product Configuration settings. Remember, the term ‘product’ is not just inclusive as a one-to-one relationship to an AWIPS
product identifier. ‘Product’ refers to the combination of VTEC phenomenon and significance code when appropriate (several AWIPS product identifiers can have multiple VTEC phenomenon/significance code combinations such as the AWIPS WSW product). The middle three letters are either the “Initial Header” or “Followup Header” entries (based upon VTEC and follow-up statements) as you assigned in Section 3 – Tab 2 above (Figure 8).

XXX – the last three letters used in Weather Message assembly are the last three characters of the AWIPS PID.

In the case of the Kansas City/Pleasant Hill office, our AFOS state node identifier is STL. We have eight transmitters (MCI, STJ, IRK, MIA, CRL, CLN, MYV, TTN). Therefore, for the NWRWAVES product SV.W (Severe Thunderstorm Warning) issued from Pleasant Hill, an inbound AWIPS product of STLSVREAX for all of our CWA would yield the following Weather Messages (depending on the VTEC action code):

**Initial Issuance (/O.NEW.KEAX.SV.W)**

STLSVREAX

**Follow-up Issuance (/O.CON.KEAX.SV.W)**

STLSVSEAX

For the NWRWAVES product SV.W (Severe Thunderstorm Warning) issued from Des Moines, an inbound AWIPS product of DSMSVRDMX would yield the following Weather Messages (depending on the VTEC action code):

**Initial Issuance (/O.NEW.KDMX.SV.W)**

DSMSVRDMX

**Follow-up Issuance (/O.CON.KDMX.SV.W)**

DSMSVSDMX

For the NWRWAVES product RWS (Regional Weather Synopsis) issued from Wichita, an inbound AWIPS product of TOPRWSKS would yield the following Weather Message:

**Any Issuance (non-VTEC coded product)**

TOPRWSKS

**NOTE:** since the inbound AWIPS PID was only eight characters, NWRWAVES assembles its last three characters of the Generic Message Type from the last three characters (SKS) of the AWIPS PID.

For non-VTEC coded products, the only header that will be used by NWRWAVES will be the “Issuance Header” entry, as there is no way to accurately determine if the product is an initial issuance or a follow-up. Thus, you will only need to ensure the presence of the list under the “Issuance Header” in your CRS database.

Therefore, in your CRS database, you will need to ensure that for every product you produce through NWRWAVES using Generic Message Type, you have a unique Message Type defined that matches this naming scheme, for both VTEC and non-VTEC products. In the case of the Pleasant Hill office where we cover seven (7) different offices and ourselves, we would have **8 unique Message Types per NWRWAVES product (16 for each product if the Issuance/Follow-Up headers are different like the TOR/SVS)**; one for each office’s AWIPS PID of their issuing product.

**SETTING UP MESSAGE TYPES IN YOUR CRS DATABASE**
The preferred method to adjust and configure your CRS database is through the GUI system on your CRS MP workstation. Sites are discouraged from hand-editing a copy of their ASCII database, unless the site focal point has a thorough knowledge of its structure. Sites that have run the WWA NWR formatters in the past will already have many of these unique Message Types defined in their database, as WWA produced transmitter-specific output. CAFÉ users will likely find a mix of unique and global Message Types depending on the AWIPS product identifier type.

The following is a recommended methodology to use in ensuring your CRS database is ready to accept output from NWRWAVES. It will work well in preparing CRS for NWRWAVES on a product-by-product basis.

NOTE: Before you begin making any changes to your CRS database, use the 0MP Database backup utility to make a good backup copy of your existing database before you begin! This way, if you need to revert back to the original database, you can do so.

1) Choose a unique product that you wish to prepare and/or ensure your CRS database to accept.
2) Make note of the “Issuance Header” and “Followup Header” of that product in the NWRWAVES Product Configuration GUI (Section 3 – Tab 2). **NOTE:** If this is a non-VTEC product such as the HWO, NOW, etc., you only need to note the “Issuance Header”.
3) In the CRS Message Type GUI (on your MP), you can find the “Message Types” GUI under the ‘Messages’ pull-down menu. You have two options on how to proceed, preparing place holders for either transmitter specific or generic message types. **NOTE:** If you already have transmitter-specific message types, you are advised to leave them be and only add those generic message types you need on a case basis.

**SETTING UP TRANSMITTER SPECIFIC MESSAGE TYPES**

For those that will involve tone alerts (WRSAME, or WRSAME and 1050Hz), you want to be sure that the **only transmitter** that has a red ‘check mark’ next to its label in the “SAME Transmitters” column is the one that matches the last three letters of the Message Type. If all the boxes are checked, then the inbound NWRWAVES product will tone alert multiple times, on multiple transmitters, in instances where an LAC overlaps multiple transmitters. Ensure that only one “SAME Transmitters” option is selected for each Message Type to prevent this from occurring.

Also, you will likely also want to *not* include these unique Message Types (if they are applicable watch/warning/advisory type products) in your Emergency Override product list. The list in Emergency Override can get quite lengthy, and the process of initiating multiple EO sessions can lead to too much work and a delay in getting a broadcast completed for multiple transmitters.

a) If you already have unique CRS Message Types defined per transmitter, and these Message Types begin with your state node CCC and end in the three-character combinations you have set in your NWRWAVES transmitter configuration, you have the correct naming scheme for transmitter-specific products, and you can complete this step (3a). Otherwise, proceed to step (3b) below.

Ensure that **only one** ‘SAME Transmitters’ transmitter is checked as described above, if this Message Type is one that involved alert tones of any type. Make sure the ‘Emergency Override’ checkbox is not selected for this unique Message Type. Your Message Groups, Broadcast Suites (and triggers) will already contain these Message Types, and no additional
work will be needed to facilitate NWRWAVES output from being scheduled on your CRS. Save the Message Type and proceed to step (4).

b) If you already have unique CRS Message Types defined per transmitter, and these Message Types begin with your state node CCC but *do not* end in the three-character combinations you have set in your NWRWAVES transmitter configuration (say you have a scheme such as “NW1, NW2”, etc., and you wish to change these last three letters), complete this step (3b). Otherwise, proceed to step (3c) below.

Perform the following steps:

1. Call each Message Type up through the GUI.
2. Edit the Message Type XXX for each individual product (in the CCCNNNXXX format), and change the last three characters to match that of your transmitter.
3. Ensure that only one ‘SAME Transmitters’ transmitter is checked as described above, if this Message Type is one that involved alert tones of any type.
4. Make sure the ‘Emergency Override’ checkbox is not selected for this unique Message Type.
5. Save the Message Type.
6. Repeat (1.), (2.), and (3.) for all other existing Message Types of this particular product category.

The CRS software will take your changes to these existing Message Types and will propagate the adjusted Message Type identifiers through all existing Message Groups and Broadcast Suites (including triggers). Proceed to step (4).

c) If you do not have unique CRS Message Types defined for each transmitter (say you have a global Message Type such as CCCSVRXXX such as in CAFÉ), you will need to add them to your CRS database. To do this:

1. Through the CRS Message Type GUI, create a new Message Type for a transmitter.
2. Establish its baseline attributes.
3. Ensure that only one ‘SAME Transmitters’ transmitter is checked as described above, if this Message Type is one that involved alert tones of any type.
4. Save the Message Type.
5. Repeat (1.), (2.), (3.) and (4.) to create Message Types of this particular product category for all transmitters.
6. You will then need to add these Message Types to any applicable Message Groups and/or Broadcast Suites for each transmitter.
7. Once the new Message Types have been assigned to a Broadcast Suite in some fashion, be sure to re-download the applicable Broadcast Programs to each transmitter. Proceed to step (4) below.

**SETTING UP GENERIC MESSAGE TYPES**
For those that will involve tone alerts (WRSAME, or WRSAME and 1050Hz), you want to be sure that all applicable transmitters have a red ‘check mark’ next to their labels in the “SAME Transmitters” column. Since you’re opting to use Generic Message Type, LAC’s assigned to the product will govern which transmitter(s) are affected. Unlike the transmitter-specific product assembly, there is no danger of the inbound NWRWAVES product tone alerting multiple times, on multiple transmitters, in instances where an LAC overlaps multiple transmitters.

You will likely *want* to include only one of these Generic Message Types (if they are applicable watch/warning/advisory type products) in your Emergency Override product list. The one you’ll likely want to include will be the one for the product issued from your office (as it will be included in every transmitter’s Broadcast Suites).

d) If you already have generic CRS Message Types defined in your database, and these Message Types mirror the issuing office’s AWIPS PID for those particular products (i.e., the old CAFÉ configuration for much of your short-fuse warning products), you have the correct naming scheme for generic products, so you can complete this step (3d). Otherwise, proceed to step (3e) below.

Ensure that all applicable ‘SAME Transmitters’ transmitter buttons are checked as described above, if this Message Type is one that involved alert tones of any type. Make sure the ‘Emergency Override’ checkbox is not selected for this unique Message Type. Your Message Groups, Broadcast Suites (and triggers) will already contain these Message Types, and no additional work will be needed to facilitate NWRWAVES output from being scheduled on your CRS. Save the Message Type and proceed to step (4).

e) If you already have generic CRS Message Types defined in your database, but these Message Types do not match the naming scheme invoked by NWRWAVES, complete this step (3e). Otherwise, proceed to step (3f) below.

Perform the following steps:

1. Call each Message Type up through the GUI.
2. Edit the Message Type XXX for each individual product (in the CCCNNNNXXX format), and change the first and/or last three characters to match the naming scheme used by NWRWAVES.
3. Ensure that all applicable ‘SAME Transmitters’ transmitters are checked as described above, if this Message Type is one that involved alert tones of any type.
4. Make sure the ‘Emergency Override’ checkbox is *not* selected for this unique Message Type.
5. Save the Message Type.
6. Repeat (1.), (2.), and (3.) for all other existing Message Types of this particular product category.

The CRS software will take your changes to these existing Message Types and will propagate the adjusted Message Type identifiers through all existing Message Groups and Broadcast Suites (including triggers). Proceed to step (4).
f) If you do not have global CRS Message Types defined, you will need to add them to your CRS database. To do this:

1. Through the CRS Message Type GUI, create a new Message Type for a transmitter. Remember that the naming scheme will mirror the AWIPS PID of the inbound office’s product if it is nine characters in length; if it is eight characters, the last three characters of the CRS Message Type will be the right three characters of the AWIPS PID.
2. Establish its baseline attributes.
3. Ensure that all applicable ‘SAME Transmitters’ transmitters are checked as described above, if this Message Type is one that involved alert tones of any type.
4. Save the Message Type.
5. Repeat (1.), (2.), (3.) and (4.) to create Message Types of this particular product category for all transmitters.
6. You will then need to add these Message Types to any applicable Message Groups and/or Broadcast Suites for each transmitter.

Once the new Message Types have been assigned to a Broadcast Suite in some fashion, be sure to re-download the applicable Broadcast Programs to each transmitter. Proceed to step (4) below.

4) Once you have established any needed Message Types for this particular hazard, it is recommended that you also create (if you don’t have one already as in the generic message type assembly) a generic Message Type for this NWRWAVES product. The generic Message Type would have the format CCCNNNNXXX, where the ‘XXX’ in this case would be your office AWIPS identifier. At Pleasant Hill, we used the format STLNNNEAX for all our generic Message Types. **NOTE:** if you have defined generic message types in steps 3(d) through 3(f), you can simply use the one for your office’s product for your manual recording needs.

For the generic message type that matches your offices issuing AWIPS PID, ensure that all applicable ‘SAME Transmitters’ transmitters are checked as described above, if this Message Type is one that involved alert tones of any type. Also, use this one (if needed) as the ONLY one set to appear in your Emergency Override GUI (this is a check box on the Message Type GUI).

In addition to CRS/VIP loading issues through NWRWAVES, the generic Message Type is quite useful for manual recordings, should NWRWAVES fail for any reason. Since the Message Type is assigned to all transmitters and can be scheduled for any transmitters, if an office has to send a warning via EO, the operator only has to make the one ‘live’ broadcast and then have the message go into all applicable transmitter programs. If a site uses the specific Message Types for their EO entries, the NWR Operator will have to send the ‘live’ warning multiple times, once for each specific Message Type.

5) Repeat steps (1) through (4) for the “Followup Header” settings if the AWIPS product for this hazard is a VTEC product, and this entry is different from the “Issuance Header”. If both are the same, you can skip step (5).
6) Repeat steps (1) through (5) for NWRWAVES products as you are ready, to ensure they are ready for transition into an operational NWRWAVES environment.

**IMPORTANT NOTES ABOUT CRS MESSAGE TYPES AND YOUR CRS DATABASE**

Some helpful hints as you transition from existing NWR formatters (WWA NWR, CAFÉ) to NWRWAVES:

a) If you opt to use the approach of having transmitter-specific Message Types in your database for NWRWAVES, also have a generic Message Type for that same NWRWAVES product to create an optimal CRS operating environment. For the NWRWAVES product WS.W (Winter Storm Warning) in the Pleasant Hill CRS database (we have eight transmitters), we have nine different Message Types. We include the generic STLWSWEAX Message Type in our Emergency Override GUI for manual recording purposes. Any warning or update for any transmitter can be broadcast through EO; then it can be scheduled on any applicable transmitter with nominal work.

b) Utilize the generic Message Type for all short-fuse warning products (SV.W, TO.W, FFW, FFS, FF.W) so as to minimize the amount of time it takes to process a warning through VIP and ultimately to the air on CRS. Sites should also look to maximize the use of generic Message Type wherever possible for messages that will feature the same content regardless of approach (RWS, PNS, RER).

c) **Do not delete, change or remove any existing Message Types that are used by your operational NWR formatter until you have fully switched that product to operational status in NWRWAVES!** Once you make the full transition to NWRWAVES for this particular product, you can safely remove any legacy Message Types through the CRS GUI interface.

d) Make sure that for any transmitter-specific Message Type that is used for tone-alert products, that the only ‘SAME Transmitter’ checked in the list is the one whose three-character mnemonic matches that of the last three letters of the Message Type. This will prevent multiple alerts being sent for products that involved LAC(s) shared amongst more than one transmitter. Make sure any generic Message Type used for alert purposes has all applicable ‘SAME Transmitters’ checked on the Message Type GUI.

e) A suggested transition is to make the switch on a product-by-product basis. This will avoid wholesale changes to your CRS database at one time. Also, after each NWRWAVES product is configured with the Message Types needed in your CRS database (or a small subset of products), make a backup copy of that CRS database (through the 0MP database backup utility) in case you need to revert back to a previous version.

f) Remember, if you add new Message Types to your CRS database, you also need to add them to any applicable Broadcast Suites, and adjust any triggers as necessary to ensure proper broadcast of information.

7. **NWRWAVES OPERATIONS**

This section of documentation will describe the NWRWAVES Browser, NWRWAVES program execution, and detail the logging performed by NWRWAVES to assist in troubleshooting procedures.
**NWRWAVES BROWSER**

The main interface that users will notice as they use NWRWAVES is the NWRWAVES Browser. This new browser will be linked during the install such that it can be launched using the standard “**NWRBrowser**” entry in the AWIPSII workstation left mouse click on AWIPS start-up menu. The overall look and feel of the NWRWAVES Browser is similar to the original NWRBrowser, however there are a number of new features and enhancements that are noted below.

There are a series of pull-down menus across the top of the NWRWAVES interface:

1) **File** Menu (Keyboard Shortcut Alt-F)
   a. Print – Print highlighted items from Pending/Sent lists (see Figure 9)
   b. Exit – Used to close the browser

![NWRWAVES Browser](image)

*Figure 9 – NWRWAVES Browser (front-end application to NWRWAVES)*
2) **Sort** Menu (Keyboard Shortcut Alt-S) – Because the number of items in the browser list boxes can become quite lengthy, a number of sorting capabilities were added to allow users to quickly locate outbound or transmitted CRS messages.
   a. **None** – No sorting is done and the browser will function similar to the old NWR Browser.
   b. **Alphabetical (Default)** – Files in the Pending and Sent list boxes will be sorted in alphabetical order.
   c. **Expiration Time** – Files in the Pending and Sent list boxes will be sorted in the order in which CRS messages are scheduled to expire.
   d. Applying the “**Decreasing Sort**” checkbox will sort the list boxes in either reverse alphabetical order (alphabetical sort applied) or in the reserve order in which CRS messages are scheduled to expire (Expiration Time sort).

3) **Filter** Menu (Keyboard Shortcut Alt-i) – The ability to filter items in the NWR Browser was added as another means to quickly located products in the Pending and Sent list boxes
   a. **Filter by Transmitter** – If you have configured your transmitters as in Section 1, you should see your list of transmitters present. Clicking any of the transmitter mnemonic radio buttons will cause the NWRWAVES browser to display only those outbound/transmitted products which are applicable to the selected transmitter.
   b. **Filter by Product Type/WFO** – The NWRWAVES Browser filter function for products contains a vast list of the most widely used products. This list can be modified under the Summary Message/Miscellaneous Configuration Tab back in Section 3, Item 11. Clicking on any product type radio button will filter the display to only show those product types.

   **NOTE:** The “output” item is used to filter the AWIPS climate program CRS messages
   **NOTE:** You can apply a “combined” filter by selecting an item under both the Filter by Transmitter and Filter by Product Type menus.

4) **Options** Menu (Keyboard Shortcut Alt-O)
   a. **Clear Old Products** (and its equivalent Purge button on the main browser display) will allow a user to delete files whose CRS message expiration time has expired. The user will be prompted to remove products from both the Pending and Sent directories. This feature can be quite helpful in eliminating dead products from an extensive file listing, but be wary that there may be instances when a user may want to maintain an expired product for later retransmission.
   b. **Update Expiration Time** – This option will become selectable once a user highlights a product in the Sent list box with the left mouse button. This feature allows a user to modify the expiration time of a product and retransmit it to CRS. This method is a great way to allow a forecaster/HMT to remove a product from the CRS broadcast cycle without having to physically remove it at the CRS console.
   c. **Auto Refresh (Default On)** – This selection tells the NWRWAVES browser to auto update the Pending and Sent list boxes.
   d. **Set Refresh Interval**… - The refresh interval can be changed from the default of 30 seconds by clicking on this menu item and entering a new number (in seconds) into the box.
   e. **Highlight Warnings (Default On)** – This selection will cause the NWRWAVES browser list boxes to briefly change from a blue to a purple/pink background color when a short-fuse warning has been issued.
NOTE: Previous versions of NWRWAVES allowed sites to switch between three other LINUX editors available for use. After complications of sites using other editors and having corrupt CRS output, the baseline NWR Browser editor is now the only editor called by NWRWAVES.

5) NWRWAVES Menu (Keyboard Shortcut Alt-N)
   a. NWRWAVES Setup – Refer to Section 3 for launching the NWRWAVES Setup Utility. This option is administratively restricted by AWIPS user accounts. Most NWRWAVES users will not have access to the Setup Utility.
   
   NOTE: You will only be allowed to open one version of the Setup GUI on your AWIPS at one time. This is controlled by a lock file which is created when the Setup GUI is launched, and is removed when you exit the Setup GUI.
   
   If you get an error message that a version is already open somewhere else, but you are confident that there is not a version open anywhere on your AWIPS, perform the following steps:

   - Open a shell on any AWIPS workstation as a user with ‘fxalpha’ group privileges.
   - Type `cd /awips/adapt/NWRWAVES`
   - Type `rm -f setup.lock`
   - Exit from the shell

   b. View/Edit VTEC_Summary File – For offices using the summary message capabilities described in Section 3 – Tab 3 (page 17), users can manually edit/delete specific product summary messages should the automated VTEC replacement method fail. The pop-up GUI which appears when there are active hazards being track for summarization (Figure 10) is a simple interface. You select the hazard you which to modify from a list of buttons, and then you can either clear all LAC(s) at once or selected LAC(s) through their individual radio buttons. Sites will rarely (if ever) need to use this interface, since active hazards are tracked through VTEC action codes in follow-up statements. This will be a useful interface though, if you track a non-VTEC hazard such as an FFW in the interim period before full VTEC implementation.

   ![Figure 10 – NWRWAVES summary message editing GUI interfaces](image)

   c. Regenerate a CRS message – Allows a user to call an AWIPS PIL from the text database and reprocess the product through the NWRWAVES formatter. When this option is selected, the user is prompted to enter an AWIPS product identifier then click ‘OK’. The latest version of the product will appear in the ‘Regenerate Product’ GUI (Figure 11). Sites can toggle between
database versions of the product type by using the “Previous Version” or “Latest Version” buttons. Once you have the product you wish to re-transmit in the viewer, click the “Send to NWRWAVES” button to re-process this product.

**NOTE:** If you use the “Include Issue Time” lead-in option for a product, the time that will be used is the time that the AWIPS product is processed by NWRWAVES, and not the time it was issued in its Mass Media header (MND date/time line). If you re-transmit a product, the time mentioned as the issuance time on the air will be the time you requested its retransmission.

![NWRWAVES Retransmit product GUI interface](image)

Figure 11 – NWRWAVES Retransmit product GUI interface

There are also a series of buttons down the middle of the interface. Some have been mentioned above, and there are a few additional buttons as well:

d. **View button:** If a user left-clicks on a product (in either the ‘Pending’ or ‘Sent’ side of the NWRWAVES browser) and clicks this button, it will bring up the product in a viewer. This function is identical to the legacy NWRBrowser. Starting in version OB8.3, the user can also print the product in the viewer via the pull-down menus across the top left (under File->Print option).

e. **Refresh button:** If clicked, it will force a manual auto-refresh of the NWRWAVES Browser.

f. **Delete button:** If a user left-clicks on a product or products on either side of the browser and then clicks this button, these NWRWAVES products will be deleted from the browser. The user is prompted to confirm the delete (yes/no/yes to all). An improvement over the NWRBrowser is that a user can select multiple products at the same time.
g. **Send button:** If a user left-clicks on a product or products on either side of the browser and clicks the ‘Send’ button, these product(s) will be sent to CRS. An improvement over the NWRBrowser is that a user can select multiple products at the same time.

h. **Exit button:** This button exits the NWRWAVES Browser.

8. **NWRWAVES PROGRAM EXECUTION (LOGGING, TROUBLESHOOTING)**

NWRWAVES is called through the use of AWIPS text database triggers. You can call for NWRWAVES to run manually two different ways: you can ‘Regenerate a CRS Message’ or edit the summary message. Both of these methods are detailed above.

Here is a breakdown of the execution of NWRWAVES for the receipt of a product:

1) An inbound product is received into the AWIPS database.
2) A text database trigger is executed to initiate NWRWAVES for the inbound AWIPS product. Depending on your settings in your siteTrigger.template file, you will either call the c-shell script that runs NWRWAVES in TEST mode (/awips/adapt/NWRWAVES/nwrwavestest.csh) or the c-shell script that runs NWRWAVES in operational mode (/awips/adapt/NWRWAVES/nwrwaves.csh).
3) These c-shell scripts perform the following functions:
   a) The script takes the inbound product and ‘cleans’ it up for use.
   b) The inbound product is moved from /data/fxa/trigger to the /awips/adapt/NWRWAVES/QUEUE directory for processing.
   c) A check is performed to ensure that only one version of NWRWAVES is running. If one is currently active, this iteration will sleep for 30 seconds before checking again. If the same version is still running after 30 seconds, these scripts will assume a hung process, kill the original one so that this one can begin. Thorough testing has indicated that a time period much less than 30 seconds is all that NWRWAVES requires for proper execution.
   d) These scripts perform a ‘checksum’ on the main program (nwrwaves.tcl) and write this checksum to the product’s debug file.

**NOTE:** The ‘checksum’ value is a fixed value of the nwrwaves.tcl code as prepared and released into AWIPS baseline software. If your site has trouble with NWRWAVES, support staff will ask for this value. If it is anything other than the baseline value, it is an indication that someone at your site has made edits to the baseline software. **SITES WILL NOT MAKE ANY LOCAL CHANGES TO THE NWRWAVES SCRIPTS AS DELIVERED!!** Making local changes to the baseline software may introduce unintended instability to the program, and it will be impossible to troubleshoot your problem if you have made local changes. Off-site troubleshooting will likely be denied until you restore the baseline version of ‘nwrwaves.tcl’, and the debug log file reflects the same ‘checksum’ value as the baseline value.
NOTE: An extensive debug and operational logging scheme is used in NWRWAVES, to assist in local and off-site troubleshooting. This scheme is detailed in (5) below.

e) These scripts call the main formatter program ‘nwrwaves.tcl’. The ‘TEST’ argument is the only difference between the two c-shell scripts. The AWIPS PID (file name) of the product being processed is also passed as an argument.

4) The main file ‘nwrwaves.tcl’ is executed next. All processing related to product preparation, assembling and formatting is performed by this script. This script does utilize three additional scripts at different points during message preparation and assembly. These are located in the /awips/adapt/NWRWAVES/bin directory and are as follows:

a) **UGC_VTEC_Decoder.tcl**: This module is called by nwrwaves.tcl to decode the UGC and VTEC lines (as appropriate) from an inbound AWIPS product. If there is no UGC (UGZ) line in the inbound product (such as an RER), then this module will assign the product a ‘dummy list’ which is one county from each transmitter. NWRWAVES will use the first county (alphabetically) from each transmitter. There is no way to control a non-UGC (UGZ) coded product by transmitter. Every transmitter will receive a copy of the product, with an assigned default expiration time of 360 minutes.

b) **UGCLookup.table**: This is the master baseline lookup table for all counties/parishes/independent cities and zones/fire weather zones/marine zones. Sites will not modify this file! If an area is missing, you do have the ability to add it manually through the Transmitter Configuration GUI (Section 3 – Tab 1: Page 7). Any local LAC’s added will be placed in the file localUGCLookup.table in this same /bin directory. NWRWAVES will source both lookup tables (if applicable) when it searches for UGC/UGZ codes.

**NOTE**: If you do find missing counties/zones that should be in the baseline file, please post them to the NWRWAVES list server (Section 7) to ensure they do get added in a future build release of NWRWAVES.

c) **WordFile.txt**: This file is a legacy from CAFÉ, and you can add words, phrases or wild card entries. NWRWAVES will make substitutions from the AWIPS product to the outbound CRS products as needed. The WordFile.txt file is detailed below.

d) **WordFile_SPN.txt**: This optional file operations exactly in the same manner as the WordFile.txt above, but allows offices that broadcast separate Spanish products to develop a separate word/phrase replace file for Spanish phrases.

5) Debugging and logging information: NWRWAVES makes extensive use of logging to assist in troubleshooting. Unless mentioned specifically, the files below are placed in the directory listed which can be found under the root NWRWAVES directory (/awips/adapt/NWRWAVES).

Here is a breakdown of available resources to you:
a) INPUT directory: NWRWAVES will archive every AWIPS product that is sent through NWRWAVES. The product will be named with the 8-9 letter AWIPS Product Identifier, followed by the date/time stamp of the product when it was processed. This date/time stamp will match the stamp assigned to its associated ‘debug’ and ‘output’ files, so that you have the input and output that can be easily matched, along with additional debug information.

b) OUTPUT directory: Similar to (a) above, but this directory contains all the output (CRS Weather Messages) prepared by NWRWAVES from the files stored in the INPUT directory.

c) LOGS directory: NWRWAVES keeps a daily log of all its activity. These log files are named using this scheme: nwrwaves.mmddyy.log. These daily logs will track the major processing points of NWRWAVES for every product processed on that day. The ‘checksum’ value is also written here.

d) DEBUG directory: NWRWAVES will create a unique ‘debug’ file for each product processed in this directory. The debug file naming scheme is: CCCNNNXXX_yymmddhhmmssdebug.txt where the CCCNNNXXX is the inbound AWIPS product identifier. You will be able to match this file with its corresponding entries in the INPUT and OUTPUT directories. The debug file contains a log of the major and minor processing points of NWRWAVES execution for this particular product.

e) SUMMARY directory: Any summary messages generated by NWRWAVES are kept in this directory.

f) ERROR directory: If NWRWAVES crashes for any reason, the core dump of the tcl/tk interpreter, along with a copy of the inbound AWIPS product are placed in this directory. There is also a file kept in the /awips/adapt/NWRWAVES directory called ‘errorout.txt’. This file will exist, and its size will be zero, if NWRWAVES has run successfully for any product.

NOTE: The contents of these directories are scoured by an automated cron at the intervals established under Section 3, item 12 of this document under the Summary Message/Miscellaneous Configuration settings. These files contain activity logs of everything sent into, and that comes out of, NWRWAVES.

NOTE: NWRWAVES has robust error trapping capabilities built into the software. Should NWRWAVES fail, the operational staff will be alerted by a red-banner message (fxaAnnounce) that the program failed, and they will be directed as to why it failed. Examples of some common failure conditions include: expired UGC time in a product, corrupt product configuration file (hand editing), and corrupt VTEC or UGC lines (hand editing). If a product fails to transmit, sites will at least be able to recognize what didn’t process, then take appropriate action to ensure that the data is broadcast.

6) When ‘nwrwaves.tcl’ has finished, it will take appropriate action with its output file(s) for CRS. If you called this file with the ‘nwrwavestest.csh’ file, all output files will be copied to the /awips/adapt/NWRWAVES/TEST directory. If you used the ‘nwrwaves.csh’ file, then the output will be copied to one of two locations:
a) If your transmission status is set to “CRS” for this NWRWAVES product type, the output will be copied to the ‘/data/fxa/workFiles/nwr/ready’ directory. Baseline AWIPS processes that monitor this directory will then take the output files and send them to CRS.

b) If your transmission status is set to “Pending” for this NWRWAVES product type, the output will be copied to the ‘/data/fxa/workFiles/nwr/pending’ directory. They will appear in the ‘Pending’ side of the NWRWAVES Browser. Manual intervention through the browser will be required to send the output to CRS.

A complete listing of the NWRWAVES file structure is included in Appendix A.

9. WORD/PHRASE REPLACEMENT CAPABILITY

The use of the Voice Improvement Processor (VIP) pre-processor and dictionary capabilities is encouraged to handle most word replacements, but there are some instances where you may want to make corrections before a product is transmitted to CRS. A number of such features are already built into NWRWAVES, including the correction of punctuation, removal of ellipses, etc. However a WordFile_SPN.txt file is provided with NWRWAVES. Instructions are provided at the top of this file, along with a number of examples within. There are two word replacement files: WordFile.txt (used for all but inbound AWIPS products that end in “SPN”), and WordFile_SPN.txt (used for Spanish inbound products that have an AWIPS PID that ends in “SPN”).

To modify the word replacement file:
1) Open a shell on any AWIPS workstation, with a user account that has ‘fxalpha’ group privileges.
2) Type cd /awips/adapt/NWRWAVES/bin.
3) Type gedit /awips/adapt/NWRWAVES/bin/WordFile.txt.
4) Add any words or phrases that you want in the prescribed format.
5) Save and exit from the editor when you are done.
6) Close the active shell.

Some hints and guidelines as to the structure of this file (NOTE: It is similar to the CAFÉ WordFile.txt file. You should be able to easily port any local changes you have made there into NWRWAVES):

a) Use a double pipe (||) to separate the original string from the replacement string.
b) All white space will be ignored.
c) Do not use punctuation in the original phrase. NWRWAVES will automatically replace the original phrase even if it is followed by a comma, period, dash, new line or space.
d) Phone numbers do not require any special entries for replacement, if they are included in your AWIPS text products in a traditional phone number format #-###-#### or ###-###-####. These formats will be handled correctly in NWRWAVES. If a site formats phone numbers differently in their actual text products, it may not be necessary to do so anymore, as they might get unexpected results.
e) Related to (d), sites that typically use ranges with a dash (NORTH WINDS 7-11 MPH, HIGHHS 77-82) will not get correct wording in an outbound CRS product. These ranges will need to be coded up differently in AWIPS text products.
f) For advanced users, the use of regular expressions is allowed and encouraged. Do not surround any regular expressions with quotes or braces { }. 

- 50 -
An excellent reference to how regular expressions can be crafted can be found in the black Tcl/Tk Welch Book, "Practical Programming in Tcl and Tk". Pages 121-126 are most insightful. Examples of some common regular expressions are:

\[0-9][0-9][0-9][0-9]Z ||

Replaces UGC time references in products such as ‘1200Z’

\[HPMCEA][DS]T ||

For time zone replacement, you can use this regular expression to take out time zone references in a product. In the baseline WordFile.txt, this line is commented out. Uncomment the line (take out the leading # symbol) if you want to filter out all time zones. If your office has multiple time zones, you may want to modify this to take out the most common one (i.e., take out C in the first bracketed area to mention Central Daylight/Standard Time but mention Mountain Time.

SEE LAKE ONTARIO OPEN LAKES FORECAST FOR [^/n]+ ||

This is an example of how to code up a replacement for a common phrase, where wording may vary at the end such as the end of the week. The following phrase might be commonly found in a Near Shore Forecast: SEE LAKE ONTARIO OPEN LAKES FORECAST FOR FRIDAY THROUGH THURSDAY.

g) Do not duplicate entries here with those that already exist in your VIP dictionary. This file is ideal for overriding CRS generalities such as:
   - FT to be read as FEET
   - Stripping the time zones off of time stamps in products (see (d) above)
   - Pausing long phrases

10. SWAPS Capability
SWAPS capability is another level of word replacement within the lead-in’s for different NWRWAVES products. This feature allows the forecasters to substitute/replace the wordings for 2 or more or a group of counties by a simple phrase. See example in the localSWAPS.txt of how to use the SWAPS option.

11. OPTIONS TO PARSE CALL-TO-ACTION IN NWS ALERT MESSAGES
In OB9 release, AWIPS warning generation tools such as WarnGen, GFE/GHG, and RiverPro will add a pair of unique markers between the Call-To-Action (CTA) Statements in many of the text products (see Appendix-E). It is part of the capabilities specified in OSIP #07-024 project - Formatting NWS Alert Messages in CAP. The current proposed solution for the CTA markers is:

(blank line)
PRECAUTIONARY/PREPAREDNESS ACTIONS...
(blank line)
(single- or multi-line text content of call-to-action, instructions, etc. here)
(blank line)
&
(blank line)
In OB9 release, AWIPS warning generation tools such as WarnGen, GFE/GHG, and RiverPro will add a pair of unique markers between the Call-To-Action (CTA) Statements in many of the text products (see Appendix-E). It is part of the capabilities specified in OSIP #07-024 project - Formatting NWS Alert Messages in CAP. The current proposed solution for the CTA markers is:

For the NWR purpose, NWRWAVES will remove the markers. Also, based on the particular hazard, NWRWAVES will provide the WFOs the option to either broadcast the corresponding CTAs or not (i.e., turn on/off the broadcast of the CTAs). In AWIPS OB8.3 release, an experimental option was added in the NWRWAVES GUI to allow the WFOs to turn on/off the CTAs for all the Marine Weather Warnings (MWW) products. The option applied globally for all the MWW products but not the individual VTEC phenomena/significance. In OB9 release, this MWW CTA GUI selection has been removed and a site-level (non-GUI base) configuration file would be used to control the CTA statements.

In OB9, all CTAs will NOT be removed by default unless the WFO NWRWAVES focal point would enter an entry or entries of configuration data in a newly created ASCII configuration file /awips/adapt/NWRWAVES/bin/NO_CTA.ini. The only way to filter out the CTA is to enter both the Product Type AND the corresponding VTEC entry or a VTEC entry with action codes separate by a comma.

The following is an example of the CTA configuration file:

```
################################################################
#### NWRWAVES Call-To-Action related config FILE: NO_CTA.ini ####
####                                                         ####
#### By default, all the CTA statements will be part of      ####
#### the warning products. The only way to remove the CTA    ####
#### statements is by creating the following entries:        ####
####                                                         ####
#### (1) Product type e.g. NPW AND (2) either a VTEC event    ####
#### code e.g. TO.W (apply for all action codes)             ####
#### or a VTEC phenomena/significance plus action codes      ####
#### separate by a comma.                                   ####
####                                                         ####
#### The following are two examples of removing the CTA      ####
#### statements associate with wind advisory and Lake wind   ####
#### in the NPW product type:                               ####
####                                                         ####
#### NPW                                                     ####
#### WI.Y                                                    ####
#### LW.Y,NEW,CON                                            ####
####                                                         ####
#### These three entries combination will instruct the       ####
#### NWRWAVES to remove the CTA statements from the NPW      ####
#### product output.                                         ####
####                                                         ####
#### In the case of mutiple VTECs within a given segment, if ####
#### one or more VTEC(s) is not specified in this file, then  ####
#### all the CTA statements will be output automatically for ####
```

Appendix O

NWRWAVES User's Manual & Documentation

AWP.MAN.SMM.A2

- Hard copies uncontrolled. Verify effective date prior to use. -
### that particular segment.

There are 2 exceptions that user can turn off the OVR overview products without CTA by entering the following entries:

- FLW_OVR
- FL_S_OVR

The FLW or FLS (hydro) products are the only two with the CTA in the overview section being removed from the OVR product.

`#` Base on product type and VTEC phenomena/significance e.g. NPW and WI.

NPW
WI.Y

`#` Base on product type and VTEC phenomena/significances plus selected actions

MWW
SC.Y,CON,EXA

**EndOfFile**

To completely remove all the CTAs for the VTEC phenomena/significance such as "SV.W" for example, the user needs to enter two product types (e.g. SVR and SVS) in the configuration file:

SVR
SVS
SV.W,NEW,CON,EXP,CAN

or

SVR
SV.W,NEW
SVS
SV.W,CON,EXP,CAN

Say if the user removes the "SVR" line from the configuration file:

SVS
SV.W,NEW,CON,EXP,CAN

Then, the CTA associates with the action code of "NEW" will never be removed from the output since only the "SVR" products can have the "NEW" action. Or specifically, NWRWAVES skips the code to process CTA since the product type "SVR" is not specified in the configuration file.

When there are multiple VTEC events described within the same product or product segment, there may be more than one CTA. If one or more of the VTEC events are not specified in the CTA configuration file in a given product or product segment, then the entire CTA section between the CTA markers shall be broadcast. In other words, more than one CTA may be broadcast. If only the start CTA marker exists in a product without the end CTA marker (&&), NWRWAVES would broadcast the text that follows the start CTA marker regardless of any WFO-level CTA option configuration.
12. LESSONS LEARNED AND RESOURCES AVAILABLE

An NWRWAVES resource web site has been created for support purposes. This site is located under the CRS web site maintained by NWS Headquarters (http://www.weather.gov/ops2/crs/nwrwaves.htm). This web site contains an FAQ list, various documentation for NWRWAVES, and a link to the NWRWAVES listserver.

NWRWAVES support is administered through the NCF. Sites experiencing difficulties with NWRWAVES should call the NCF and open a trouble ticket. The NCF will provide Tier 1 support for offices using NWRWAVES. If the NCF believes that Tier 2 support is necessary, they will contact the appropriate Tier 2 support group. Tier 2 support is now the CRS Operations Group at NWSHQ.

General support questions can and should be referred to the NWRWAVES Listserver. If you’re in need of general support or just have a question regarding NWRWAVES (no imminent problem or NCF support needed for a failure), the best mechanism for support is as follows:

a. If the problem you’re experiencing is impacting operational use, contact the NCF and open a trouble ticket.

b. If it is not an imminent issue, check your question against the FAQ list on the NWRWAVES web site. If your question has been encountered before, you will find the solution posted here.

c. If the question is not readily visible on the FAQ list, post it to the list server. This way, everyone who has been working with the software can see your situation, and you can get help from other NWRWAVES users. Operational questions, configuration issues and questions should be submitted to the NWRWAVES list server to ensure the quickest response possible.

To sign up for the NWRWAVES list server, go to the following URL: http://infolist.nws.noaa.gov/read/login/ Input your e-mail address and e-mail password, then select the All Forums Tab. Scroll down to NWRWAVES and click on subscribe. You will receive information back from the listserver concerning use of the forum.
APPENDIX A – COMPLETE NWRWAVES FILE LISTING

MAIN INSTALL DIRECTORY LISTING

drwxrwxrwx  2 fxalpha fxalpha  28672 Apr 27 16:47 BACKUP
-drwxrwxrwx  2 fxalpha fxalpha  4096 Apr 23 11:29 bin
-drwxrwxrwx  2 fxalpha fxalpha  4096 Apr 24 13:35 browser
-drwxrwxrwx  2 fxalpha fxalpha 139264 Apr 28 21:14 DEBUG
-drwxrwxrwx  2 fxalpha fxalpha  4096 Apr 15 22:10 ERROR
-rw-rw-r--  1 fxalpha fxalpha  0 Apr 28 21:14 errorout.txt
-rw-rw-r--  1 fxalpha fxalpha  37 Apr 28 21:14 FILENAME.LAST
-drwxrwxrwx  2 fxalpha fxalpha  73728 Apr 28 21:14 INPUT
-drwxrwxrwx  2 fxalpha fxalpha  4096 Apr 28 01:40 LOGS
-rw-rw-r--  1 fxalpha fxalpha  18 Apr  5 19:49 NONVTCEMRD.txt
-rwxrwxr-x  1 fxalpha fxalpha  4265 Apr  9 17:47 nwrwaves.csh
-rwxrwxrwx  1 fxalpha fxalpha 1026 Feb 23 18:48 NWRWAVESpurge.sh
-rw-rw-r--  1 fxalpha fxalpha  4 Apr 27 17:01 purge.cfg
-rwxrwxr-x  1 fxalpha fxalpha 212320 Mar 24 23:45 nwrwaves_setup.tcl
-rwxrwxr-x  1 fxalpha fxalpha 160858 Apr  9 18:10 nwrwaves.tcl
-rwxrwxr-x  1 fxalpha fxalpha  4270 Apr  9 18:21 nwrwavestest.csh
-drwxrwxrwx  2 fxalpha fxalpha  49152 Apr 28 21:14 OUTPUT
-rwxrwxr-x  1 fxalpha fxalpha  788 Feb 18 15:17 pendingDirCheck
-rw-rw-r--  1 fxalpha fxalpha 14738 Apr 26 19:37 product.cfg
-rw-rw-r--  1 fxalpha fxalpha 14375 Apr  2 16:22 product.cfg.V2.4
-rwxrwxrwx  1 fxalpha fxalpha  15 Apr 27 17:01 QUEUE
-rwxrwxrwx  1 fxalpha fxalpha 1467 Feb 23 18:48 removeExpiredNWR.sh
-drwxrwxrwx  2 fxalpha fxalpha  4096 Apr 16 01:21 SUMMARY
-drwxrwxrwx  2 fxalpha fxalpha  4096 Mar 24 16:32 TEST
-rw-rw-r--  1 fxalpha fxalpha  4704 Mar 25 00:03 UGClookup_changes.txt
-rwxrwxr-x  1 fxalpha fxalpha  2941 Feb 18 15:17 updateSUMMARY.csh

NWRWAVES bin DIRECTORY LISTING
-rwxrwxr-x  1 fxalpha fxalpha  9039 Feb 18 15:17 clocktime.tcl
-rwxrwxr-x  1 fxalpha fxalpha 15098 Feb 18 15:17 color_msgbox.tcl
-rwxrwxr-x  1 fxalpha fxalpha  64430 Feb 18 15:17 comboBox.tcl
-rwxrwxr-x  1 fxalpha fxalpha  2211 Feb 18 15:17 dialog.tcl
-rw-rw-r--  1 fxalpha fxalpha  44 Dec  2 08:52 localUGClookup.table
-rwxrwxr-x  1 fxalpha fxalpha  5104 Feb 18 15:17 notebook.tcl
-rwxrwxr-x  1 fxalpha fxalpha  923 Feb 18 15:17 pkgIndex.tcl
-rwxrwxrwx  1 fxalpha fxalpha  1545 Feb 18 15:17 relinkNWRWAVESBrowser.sh
-rw-rw-r--  1 fxalpha fxalpha 230042 Mar 25 00:06 UGClookup.table
-rwxrwxr-x  1 fxalpha fxalpha  27118 Mar 14 16:51 UGC_VTEC_Decoder.tcl
-rw-rw-r--  1 fxalpha fxalpha  751 Feb 18 15:17 VTECrank.ini
-rwxrwxr-x  1 fxalpha fxalpha  717 Nov 13 13:01 WordFile_SPN.ini
-rwxrwxrwx  1 fxalpha fxalpha  3315 Jan 24 19:47 WordFile.txt
-rw-rw-r--  1 fxalpha fxalpha  717 DEC 13 13:01 localSWAPS.txt
-rwxrwxr-x  1 fxalpha fxalpha  717 Dec 13 13:01 NO_CTA.ini
-rwxrwxr-x  1 fxalpha fxalpha  717 Mar 21 15:17 NWRWAVESBrowser.sh

- Hard copies uncontrolled. Verify effective date prior to use. -
NWRWAVES BROWSER DIRECTORY LISTING

-rw-r--r--     1 fxa      fxalpha        57 Apr 24 13:33 admin.list
-rw-rw-rw-     1 fxa      fxalpha      1118 Mar 24 22:36 browser.cfg
-rw-rw-rw-     1 fxa      fxalpha       267 Mar 24 22:36 browser.ini
-rwxrwrxr-x     1 fxa      fxalpha       66988 Mar 24 23:42 browser.tcl
-rw-rw-rw-     1 fxa      fxalpha         5 Apr 27 17:01 pending.cfg
-rw-rw-rw-     1 fxa      fxalpha        95 Apr 27 17:01 pils.list
-rwxrwrxr-x     1 fxa      fxalpha      12788 Mar 24 22:36 update.tcl
APPENDIX B – EXAMPLE OF EAX siteTrigger.template
FILE FOR NWRWAVES

STLSVREAX /awips/adapt/NWRWAVES/nwrwaves.csh
STLSVRLSX /awips/adapt/NWRWAVES/nwrwaves.csh
STLSVRSFGF /awips/adapt/NWRWAVES/nwrwaves.csh
TOPSVRTOP /awips/adapt/NWRWAVES/nwrwaves.csh
DMSSVRDNX /awips/adapt/NWRWAVES/nwrwaves.csh
CHISVRSVDN /awips/adapt/NWRWAVES/nwrwaves.csh
OMASVROAX /awips/adapt/NWRWAVES/nwrwaves.csh
STLTOREAX /awips/adapt/NWRWAVES/nwrwaves.csh
STLTORLSX /awips/adapt/NWRWAVES/nwrwaves.csh
STLTORSFGF /awips/adapt/NWRWAVES/nwrwaves.csh
TOPTORTOP /awips/adapt/NWRWAVES/nwrwaves.csh
DSMTORDMX /awips/adapt/NWRWAVES/nwrwaves.csh
CHITORDVN /awips/adapt/NWRWAVES/nwrwaves.csh
OMATOROAX /awips/adapt/NWRWAVES/nwrwaves.csh
STLFFWEAX /awips/adapt/NWRWAVES/nwrwaves.csh
STLFFWLSX /awips/adapt/NWRWAVES/nwrwaves.csh
STLFFWSFGF /awips/adapt/NWRWAVES/nwrwaves.csh
TOPFFWTOP /awips/adapt/NWRWAVES/nwrwaves.csh
DSMFFWDMX /awips/adapt/NWRWAVES/nwrwaves.csh
CHIFFWDVN /awips/adapt/NWRWAVES/nwrwaves.csh
OMAFFWOAX /awips/adapt/NWRWAVES/nwrwaves.csh
STLFFAEAX /awips/adapt/NWRWAVES/nwrwaves.csh
STLFFALSX /awips/adapt/NWRWAVES/nwrwaves.csh
STLFFASFGF /awips/adapt/NWRWAVES/nwrwaves.csh
TOPFFATOP /awips/adapt/NWRWAVES/nwrwaves.csh
DSMFFADMX /awips/adapt/NWRWAVES/nwrwaves.csh
CHIFFADVN /awips/adapt/NWRWAVES/nwrwaves.csh
OMAFFAOAX /awips/adapt/NWRWAVES/nwrwaves.csh
STLFFSAX /awips/adapt/NWRWAVES/nwrwaves.csh
STLFFSLSX /awips/adapt/NWRWAVES/nwrwaves.csh
STLFFSSGF /awips/adapt/NWRWAVES/nwrwaves.csh
CHIFFSDVN /awips/adapt/NWRWAVES/nwrwaves.csh
DSMFFSDMX /awips/adapt/NWRWAVES/nwrwaves.csh
TOPFFSTOP /awips/adapt/NWRWAVES/nwrwaves.csh
OMAFFSOAX /awips/adapt/NWRWAVES/nwrwaves.csh
STLFLWEAX /awips/adapt/NWRWAVES/nwrwaves.csh
STLFLWLSX /awips/adapt/NWRWAVES/nwrwaves.csh
STLFLWSGF /awips/adapt/NWRWAVES/nwrwaves.csh
CHIFLWDVN /awips/adapt/NWRWAVES/nwrwaves.csh
TOPFLWTOP /awips/adapt/NWRWAVES/nwrwaves.csh
OMAFLWOAX /awips/adapt/NWRWAVES/nwrwaves.csh
STLWSWEAX /awips/adapt/NWRWAVES/nwrwaves.csh
STLWSWLSX /awips/adapt/NWRWAVES/nwrwaves.csh
TOPWSWTOP /awips/adapt/NWRWAVES/nwrwaves.csh
OMAWSWOAX /awips/adapt/NWRWAVES/nwrwaves.csh
Appendix O

NWRWAVES User's Manual & Documentation

CHIWSDVN /awips/adapt/NWRWAVES/nwrwaves.csh
DSMWSWDMX /awips/adapt/NWRWAVES/nwrwaves.csh
STLWSWSGF /awips/adapt/NWRWAVES/nwrwaves.csh
STLNOWEAX /awips/adapt/NWRWAVES/nwrwaves.csh
STLNOWLSX /awips/adapt/NWRWAVES/nwrwaves.csh
STLNOWSGF /awips/adapt/NWRWAVES/nwrwaves.csh
CHINOWDVN /awips/adapt/NWRWAVES/nwrwaves.csh
DSMNOWDMX /awips/adapt/NWRWAVES/nwrwaves.csh
TOPNOWTOP /awips/adapt/NWRWAVES/nwrwaves.csh
OMANOWOAX /awips/adapt/NWRWAVES/nwrwaves.csh
STLNPWEAX /awips/adapt/NWRWAVES/nwrwaves.csh
STLPWLSX /awips/adapt/NWRWAVES/nwrwaves.csh
CHINPWDVN /awips/adapt/NWRWAVES/nwrwaves.csh
DSMNPWDMX /awips/adapt/NWRWAVES/nwrwaves.csh
OMANPWOAX /awips/adapt/NWRWAVES/nwrwaves.csh
TOPNPWTOP /awips/adapt/NWRWAVES/nwrwaves.csh
STLNPWSGF /awips/adapt/NWRWAVES/nwrwaves.csh
STLNPSEAX /awips/adapt/NWRWAVES/nwrwaves.csh
STLHWOEAX /awips/adapt/NWRWAVES/nwrwaves.csh
STLSPESEAX /awips/adapt/NWRWAVES/nwrwaves.csh
STLRWSEAX /awips/adapt/NWRWAVES/nwrwaves.csh

- Hard copies uncontrolled. Verify effective date prior to use. -
APPENDIX C – NWRWAVES Cron Jobs

NWRWAVES has four background cron jobs. All the cron jobs run on dx2 as user fxa, except during a failover when they will run on dx1. The cron jobs are listed below:

# Crontab file for starting dx2apps NWRWAVES purge processes.
# Any entry that needs to use ${FXA_HOME} or ${FXA_DATA} should use "csh -c"
# to run the command. The command and any output redirection to a file must
# all be include in single quotes after the "-c". The output redirection
# will then be done by the csh so it must use csh syntax.

# PURGER/SCOUR...
# Run NWRWAVESpurge.sh daily to clean DEBUG LOGS OUTPUT INPUT
# and TEST Directories
45 0 * * * csh -c '/awips/adapt/NWRWAVES/NWRWAVESpurge.sh >& /dev/null'

# Run removeExpiredNWR.sh twice an hour to remove any expired or
# corrupt files from pending and sent side of NWRWAVESBrowser
20,40 * * * * csh -c '/awips/adapt/NWRWAVES/removeExpiredNWR.sh >& /dev/null'

# OTHER NWRWAVES Cron jobs
# Summary Update Script, updateSUMMARY.csh, four times an hour to
# update the summary message
00,15,30,45 * * * * csh -c '/awips/adapt/NWRWAVES/updateSUMMARY.csh >& /dev/null'

# Run the Pending Directory Check Script, pendingDirCheck, every ten minutes
# to check for old messages in the pending directory
00,10,20,30,40,50 * * * * csh -c '/awips/adapt/NWRWAVES/pendingDirCheck >& /dev/null'

**NWRWAVESpurge.sh** – Runs once a day 00:45Z. It purges files in the DEBUG, LOGS, INPUT, OUTPUT and TEST directories. Purge settings are contained in the file /awips/adapt/NWRWAVES/purge.cfg and are modified by using the NWRWAVES Setup GUI.

**removeExpiredNWR.sh** – Runs twice an hour at 20 and 40 minutes past. It removed expired and corrupt messages from the /data/fxa/workFiles/nwr/pending and /data/fxa/workFiles/nwr/sent directories.

**updateSUMMARY.csh** – Runs four times an hour at 00, 15, 30, and 45 minutes past. It forces an update of any active NWRWAVES Summary messages.

**pendingDirCheck** – Runs six times an hour at 00, 10, 20, 30, 40, and 50 minutes past. It checks for stale messages in the /data/fxa/workFiles/nwr/pending directory. If a stale message is detected, then an fxaAnnounce Red Banner message is sent. The time a messages needs to sit in the pending directory before it is considered stale is configured in the NWRWAVES Setup GUI. This script can also be disabled in the NWRWAVES Setup GUI.
APPENDIX D – Instructions for implementing MWW Watch/Warnings/Advisories

The following is a list of the applicable Watches, Warnings and Advisories that can be issued with the Marine Weather product (MWW) along with the associated V-TEC code, Issuance and Followup headers.

### Marine Watches

<table>
<thead>
<tr>
<th>Products</th>
<th>VTEC</th>
<th>Issuance Header</th>
<th>Followup Header</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hurricane Force Wind Watch</td>
<td>HF.A</td>
<td>MWW</td>
<td>MWW</td>
</tr>
<tr>
<td>Storm Watch</td>
<td>SR.A</td>
<td>MWW</td>
<td>MWW</td>
</tr>
<tr>
<td>Gale Watch</td>
<td>GL.A</td>
<td>MWW</td>
<td>MWW</td>
</tr>
<tr>
<td>Hazardous Seas Watch</td>
<td>SE.A</td>
<td>MWW</td>
<td>MWW</td>
</tr>
<tr>
<td>Heavy Freezing Spray Watch</td>
<td>UP.A</td>
<td>MWW</td>
<td>MWW</td>
</tr>
</tbody>
</table>

### Marine Warnings

<table>
<thead>
<tr>
<th>Products</th>
<th>VTEC</th>
<th>Issuance Header</th>
<th>Followup Header</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hurricane Force Wind Warning</td>
<td>HF.W</td>
<td>HUW</td>
<td>HUW</td>
</tr>
<tr>
<td>Storm Warning</td>
<td>SR.W</td>
<td>MWW</td>
<td>MWW</td>
</tr>
<tr>
<td>Gale Warning</td>
<td>GL.W</td>
<td>MWW</td>
<td>MWW</td>
</tr>
<tr>
<td>Hazardous Seas Warning</td>
<td>SE.W</td>
<td>MWW</td>
<td>MWW</td>
</tr>
<tr>
<td>Heavy Freezing Spray Warning</td>
<td>UP.W</td>
<td>MWW</td>
<td>MWW</td>
</tr>
</tbody>
</table>

### Marine Advisories

<table>
<thead>
<tr>
<th>Products</th>
<th>VTEC</th>
<th>Issuance Header</th>
<th>Followup Header</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ashfall Advisory</td>
<td>AF.Y</td>
<td>NPW</td>
<td>NPW</td>
</tr>
<tr>
<td>Brisk Wind Advisory</td>
<td>BW.Y</td>
<td>MWW</td>
<td>MWW</td>
</tr>
<tr>
<td>Dense Fog Advisory</td>
<td>FG.Y</td>
<td>NPW</td>
<td>NPW</td>
</tr>
<tr>
<td>Dense Smoke Advisory</td>
<td>SM.Y</td>
<td>NPW</td>
<td>NPW</td>
</tr>
<tr>
<td>Freezing Spray Advisory</td>
<td>UP.Y</td>
<td>MWW</td>
<td>MWW</td>
</tr>
<tr>
<td>Low Water Advisory</td>
<td>LO.Y</td>
<td>MWW</td>
<td>MWW</td>
</tr>
<tr>
<td>Small Craft Advisory</td>
<td>SC.Y</td>
<td>MWW</td>
<td>MWW</td>
</tr>
<tr>
<td>Small Craft Advisory for Hazardous Seas</td>
<td>SW.Y</td>
<td>MWW</td>
<td>MWW</td>
</tr>
<tr>
<td>Small Craft Advisory for Rough Bar</td>
<td>RB.Y</td>
<td>MWW</td>
<td>MWW</td>
</tr>
<tr>
<td>Small Craft Advisory for Winds</td>
<td>SL.Y</td>
<td>MWW</td>
<td>MWW</td>
</tr>
</tbody>
</table>

Each of the marine sites will have to configure the MWW entries according to their needs. The following is an example of how to configure an MWW for Small Craft Advisory (SC.Y).
Instructions for implementing Small Craft Advisory (SC.Y)  
NWRWAVES (OB8.2 and older)

There are two basic steps needed to implement SC.Y products into NWRWAVES. The first is to create the product and configure it within NWRWAVES, and the second is to create the textdb trigger for the new MWW AWIPS product.

1. Create the SC.Y product within NWRWAVES
   a) From the NWRBrowser on AWIPS, launch the NWRWAVES Setup GUI.
   b) Click on the second tab, which is labeled “NWRWAVES Product Configuration” (below).
   c) Click the pink 'Add Product' button.
d) In the new GUI that pops up, enter in the text boxes exactly what you see in the example below.

![NWRWAVES New Product Addition](image1)

```
VTEC Phenomenon or Product PIL  Product Name  Issuance Header  Followup Header
SC.Y  Small Craft Advisory  MWW  MWW
OK  Cancel
```

e) Click OK, then click the green ‘Save Edits’ button in the NWRWAVES Setup GUI.

f) Confirm your settings: from the NWRWAVES product dropdown menu, select the product “SC.Y – Small Craft Advisory”. Your GUI should match the one below; if it doesn’t make adjustments to the appropriate fields and then save those changes. Don’t forget to ensure your alert settings for the Non-Routine Broadcast Area!

![NWRWAVES Setup Utility](image2)

You are done configuring the SC.Y product in NWRWAVES. Since the SC.Y uses the CWF code for
EAS activation (and for simplicity sake we’ll use the MWW follow-up header), your CRS database is already configured to accept the new SC.Y products.

2. Configuring your textdb triggers to execute NWRWAVES for the new MWW product(s). For a complete explanation of setting up text database triggers for NWRWAVES, please refer to the NWRWAVES User Documentation, pages 28-29:

a. Log into an AWIPS workstation with a user account that has fxlalpha group privileges and open up a terminal window.
b. Change directories to the trigger directory.

```
cd /data/fxa/siteConfig/textApps
```
c. Make a backup copy of your site trigger information, in case you need to revert back to it.

```
cp siteTrigger.template siteTrigger.template.mmddyyyy
where ‘mmddyyyy’ is the numeric current date
```
d. Edit your site trigger configuration.

```
gedit siteTrigger.template &
```

When you view this file, you will see a listing of AWIPS product identifiers, along with an action that is to be completed upon receipt of that product identifier into the text database. Enter the following line into your trigger file:

```
CCCMWWXXX /awips/adapt/NWRWAVES/nwrwaves.csh
```

Be sure to include every local office MWW product that you need for your NWR coverage areas. “CCC” refers to the legacy state node, while “XXX” is the 3-character WFO identifier.

e. Save and exit your siteTrigger.template file

f. Log onto dx3/dx4

```
ssh dx3/dx4
```

g. Change user to fxa and type the following commands for AWIPS II:

```
su – fxa (may require to be root)
cd /data/fxa/sdc
./config_awips2.sh triggers XXX (where XXX is the 3 letters site ID)
```
APPENDIX E – A List of Product Identifier in OB9 with the new Instruction (CTA) Markers

AVA, AVW, CAE, CDW, CEM, CWF, EQW, EVI, FFA, FRW, HLS, HMW, LAE, LEW, MWS, MWW, NPW, NUW, RFW, RHW, SPW, TOE, VOW, WSW, FFA, FLW, EWW, FFS, FFW, SMW, SVR, SVS, TOR

NOTE: Some products, namely the stand-alone MWS (MWS that does not follow-up a SMW) and Non-Weather Emergency Message (NWEM) products that originate in HazCollect, initially will not contain CTA Markers.

APPENDIX F – New features to support HLS products and operations

Since June 2008, the Hurricane Local Statement (HLS) products issued by the sites are segmented products with VTEC containing the Tropical Storm/Hurricane Watch/Warnings. During the 2008 tropical season, NWRWAVES created HLS radio products that were very lengthy and essentially unreadable on NWR. The resulting radio products made cycles upwards of 30 minutes long which is unacceptable for a severe weather event. A small enhancement was made in OB9 release to allow sites to choose overview and to reduce the HLS sub-headlines sent to the NWR. Instead of formatting the incoming HLS messages without user’s attention, the NWRWAVES is now requires some user’s interaction by sending a series (total of 3) of HLS related Guardian banners to alert users to run the HLS Setup GUI to make sub-headline selections when a new HLS message arrives. See Figure 1 for samples. This will require even more interaction for sites with multiple HLS products. From the HLS operation perspective, the NWRWAVES software now requires the support of two separate roles - administrators to set up the HLS configuration and regular users to make sub-headline selections when a new HLS message arrives.

This is the 1\textsuperscript{st} call message (priority is set to 3):

This 2\textsuperscript{nd} message sends with an audio alarm (priority is set to 2):

This is the 3\textsuperscript{rd} or last call or 2 minutes warning (priority is set to 1):
To set up the HSL banners in AWIPS II, the NWRWAVES administrators may need to logon to the graphic workstation(s) where you want the HLS banners show up and perform the following steps:

1) Right click on the AlertViz icon, and select “Configuration…”
2) At the top left box under Category, clicked on New button and added TROPICAL (as shown in Figure 1-a below).
3) Under Sources, select “NWRWAVES” as shown.
4) Under PRIORITIES, set the check boxes for priorities 1 to 3 accordingly.

Select “Save” to save and exit.

Users should consult the AWIPS II Alert Visualization Configuration related documentation for details.
**Part I - Administrator’s Role: (must be performed in-advance)**

The NWRWAVES administrators will need to set up the AWIPS text database triggers in order to process the HLS products automatically (see section 5 of this User’s Guide).
The following are the procedures to launch the HLS Setup GUI for the administrator:
From the AWIPS graphical workstation, select NWRBrowser via the AWIPS start-up menu (left mouse click). Go to the “NWRWAVES” pull-down menu, select the option “NWRWAVES Setup” (Figure 2). You will see the NWRWAVES Setup GUI pop-up on your AWIPS screen.

![Figure 2- Launching the HLS Setup GUI for administrators via the NWRBrowser](image)

Select the 4th “Marine/Tropical Product Configuration” tab (see Figure 3). Under the “Hurricane Local Statement” section, you should see the label “Subheadlines Selection and HLS Transmitter Configuration” and click on the new “Change Settings” button (CYAN color) to activate the HLS Setup GUI.
Figure 3- A new button is added to run HLS Setup from the NWRWAVES Setup GUI
Upon the selection of “Change Settings” button, the new HLS Setup Utility GUI should pop up as shown in Figure 4. For sites that only have a local HLS product set up, you will not see the File->“Open … Site ID” (highlighted in red).

ATAN 991 was delivered to address a deficiency of not being able to support multiple HLSs found in the OB9 version. Many sites want to have the capability to make subheadline selections customizable for different sites. For example, Sterling, Va WFO Office (LWX) also process the neighbors’ HLS messages (e.g. WBCHLSAKQ and PHLHLSPHI) in addition to the local HLS (e.g. WBCHLSLWX). In order to support the subheadline selections for multiple HLSs, the administrator can create (via your favorite editor) an ASCII file “hls_sites.list” (all lower case) in the NWRWAVES directory (/awips/adapt/NWRWAVES on dx1 or dx2 machines) with a 3 characters neighbor site ID per line. For example (LWX Office), the file contains the following 2 lines:

```
akq
phi
```

Once the “hls_sites.list” file is created, the HLS Setup GUI should be able to create and read a site dependent configuration file (e.g. “hls_cfg.XYZ, where XYZ is the site ID) for the HLS users.
Figure 4 – A new HLS Setup GUI for Sub-headlines configuration for multiple HLSs
The administrators (and also users as well) will be able to set up and control the First Wait time and Final Wait times, and an automatic TWO minutes warning will be issued if no user’s respond after the 2 waits. If no further response, the NWRWAVES will go on to parse the input HLS message using the previous HLS configuration. The First and Second Wait times are recommended and set to be 10 and 8 minutes by default respectively. If no user edits (touches) the HLS configuration file after the 3 Guardian banners, the HLS input will be processed after 20 minutes (First wait Time plus Final wait Time and the TWO minutes warning) elapsed.

One of the major duties of the administrator is to create or to delete any additional local sub-headlines for the users to select. This will occur well in advance of the event or after the software installation is completed. Note that the following 9 standard sub-headlines will be created in alphabetical order and cannot be deleted:

- NEW INFORMATION
- PRECAUTIONARY/PREPAREDNESS ACTIONS
- PROBABILITY OF TROPICAL STORM/HURRICANE CONDITIONS
- WINDS
- WINDS AND SEAS
- STORM SURGE AND STORM TIDE
- INLAND FLOODING
- TORNADOES
- TORNADOES AND WATERSPOUTS

To populate additional sub-headlines, the administrator would input the new sub-headline one at a time via the “Enter new Sub-headlines (no ellipses):” entry. After completing a new sub-headline entry, click on the “Add to Subheadline list” button, and the new sub-headline entry will show up as a new sub-headline list on the left scroll window. Likewise, to delete the entry, just highlight the entry on the left scroll window and click on the “Delete” button to remove the selected sub-headline(s). Again, you cannot remove those eleven standard sub-headlines. Be sure to save the configuration before existing.

Another major task required by an administrator is to assign counties/zones mapping to the transmitters for only the HLS products. This is an experimental feature introduced particularly for the HLS product in OB9. By selecting the 1st tab of “Transmitter Configuration for HLS” in the HLS Setup GUI (see Figure 5), the administrator can select the transmitters’ counties/zones relationship for only the HLS product without impacting all other products. This GUI is identical and operated the same way as the old familiar NWRWAVES Setup GUI does with the exception of the new “Excluding marine zones” radio buttons (see the RED oval). Also consult section 4 – NWWAVES Setup Utility (Tab 1- Transmitter Configuration) of this User’s Guide for details operations.

For the first time when the administrator brings up this HLS Setup GUI, an exact duplicate of the regular NWRWAVES transmitters’ configuration data file will be copied over for this newly created HLS transmitters’ configuration. It is important to know that whatever the changes made here will only impact the HLS product and not any other products.
Figure 5- A new GUI to configure transmitters for HLS product.

The following is for those sites with multiple HLSs configuration set up:
Unlike the multiple HLS sub-headline configuration files for neighbor sites, there is only one (single) copy of this HLS transmitters file for the local WFO purpose. If the current Active WFO is one of your neighbors (e.g. via the Open …site ID), you will be warned by some red colored frames (see Figure 6) that there is really only one HLS transmitters file designated for your local site. The proper way to assign zones to transmitters mapping is to select your local WFO as the current active WFO.
Figure 6- A warning example of transmitter configuration when the current active site is not your local site.

**Part II - User’s Role:**

Once the administrator completes the required HLS configuration in Part I, the users should be able to run the HLS GUI. Normally, the HLS users will not run the HLS GUI unless a new HLS message has just arrived and triggers the Guardian banners to alert them to run the HLS GUI.
When a new HLS message arrives, a Guardian banner will alert the HLS users. For multiple HLS sites, the user will need to watch out the 3 characters (see below e.g. LWX in this case) site ID that is issuing the HLS:

![Alert Visualization Popup Message Dialog](image)

The following are the procedures to launch the HLS Setup GUI for the general users:

Upon alerting by the 1st Guardian banner from the AWIPS graphic workstations, the HLS user should login to any of the AWIPS graphic workstation.

From the AWIPS graphical workstation, select NWRBrowser via the appLauncher.

Go to the NWRBrowser pull-down menu, select 4th option entry “HLS Subheadlines Selection” (Figure 7). If you don’t see this option, this means that the administrator has not activated the HLS Setup GUI yet. Upon the selection, user will see the HLS Setup GUI pops up on his/her AWIPS screen (Figure 8) along with the incoming HLS message displays in the File viewer window.
Figure 7- Launching the HLS Setup GUI for HLS users via the NWRBrowser
<table>
<thead>
<tr>
<th>Hurricane Local Statement Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Wait Time in 10 Minutes (default 10 min.) before the 2nd Guardian reminder message is issued.</td>
</tr>
<tr>
<td>Final Wait Time in 8 Minutes (default 10 min.) before the 2nd Guardian reminder message is issued.</td>
</tr>
</tbody>
</table>

**Figure 8 – HLS Setup GUI (user mode) with a File viewer option turn on**

**WTUS62 KMHX312213 HLSMNX**

Urgent - Immediate Broadcast Requested
Hurricane Earl Local Statement
National Weather Service Newport/Morehead City NC
6:13 PM EDT Tue Aug 31 2010

...Hurricane Watch Issued as Hurricane Earl Approaches the Coast of North Carolina...

**Areas Affected...**
This local statement provides important information and recommended actions for people and marine interests in select locations and coastal water legs of eastern North Carolina and adjacent coastal waters.

**Watches/Warning...**
A hurricane watch continues for the following locations...
Washington...Tyrrell...Mainland Dare...Beaufort...Mainland Hyde...Croatan...Pamlico...Carteret...Onslow...Outer Banks Dare and Outer Banks Hyde.

For marine interests...A hurricane watch continues for all of Atlantic coastal waters including Albemarle and Pamlico sounds.

Please check the latest public and marine forecasts for detailed information about additional hazards.

**Storm Information...**
At 5 PM EDT...the center of Hurricane Earl was located near latitude 22.0N...longitude 68.8W. This was about 1000 miles south-southeast of Buxton NC. Storm motion was NW or 305 degrees at 14 MPH. Storm intensity was 135 MPH.

**Situation Overview...**
When making decisions...do not focus on the exact forecast track.

Hurricane Earl will pass east of North Carolina starting Thursday afternoon and will continue to affect the area through Friday afternoon. Increased winds and rain are likely along the outer banks and coastal counties...and the coastal waters from Cape Lookout to Duck...including the Albemarle and Pamlico sounds...starting Thursday morning and continuing into Friday afternoon. Strong rip currents and large breaking waves are also likely.

It is too early to provide exact wind and surge forecast values for specific locations. A general concern should be for the possibility of at least damaging winds somewhere within eastern North Carolina.

**Precautionary/Preparedness Actions...**
Precautionary/Preparedness Actions...
Briefly, the main functional steps for the HLS users are summarized below:

- Open the active WFO as indicated by the Guardian message (for multiple HLSs sites only)
- View the incoming HLS message from the File viewer window for subheadlines if desirable
- Choose the option (Y/N) for “Include Overview”
- Make selections of which subheadlines to process using “→” or “←” buttons
- Hit File->“Save Config” when done
- Select File->“Exit” to quit the HLS GUI.

ATAN 989 fix was delivered to fix a time delay problem. The NWRWAVES parsing script should now run within 10 seconds after users select a “Save Config” option.

User must select and save the sub-headlines to be used when a new HLS product arrives each time. By highlighting the sub-headline(s) on the left scroll window (see Figure 9), the user then clicks on the “→” button to select the desired sub-headline(s). This selected sub-headline(s) should show up on the right scroll window and eventually will be used by the NWRWAVES formatting script. Note that the NWRWAVES script does not process the sub-headlines in alphabetical order as shown on the right scroll window as indicated. The HLS output ordering should match the pre-fix order of the input HLS product. Likewise, you can unselect the sub-headlines by highlight the items on the right scroll window for broadcast and click on the “←” button to unselect the sub-headlines. Notice that users (only administrators can) do not have the role to add any new sub-headlines (all the associated buttons or entry are disabled). The last important step is to save the new configuration by selecting the “File” menu-> “Save Config” option to save the changes before the “Exit”.

- Hard copies uncontrolled. Verify effective date prior to use. -
Figure 9 – A new HLS Setup GUI for Subheadlines configuration.

If users do not see the separate File viewer window pops up along with the HLS GUI, they can request the administrator to create a blank (an empty) flag file “hls_show_input” in the NWRWAVES’s bin directory (/awips/adapt/NWRWAVES/bin). If the users don’t want this option, the administrator can rename or remove the file.
New changes for the 2011 hurricane season (AWIPS OB9.3 delivery)

1) By default, the overview product (OVR) will be generated for each transmitter for the HLS (see Figure 10 which is set to YES) - set for each hazard it contains (HU.A/W, TR.A/W, TY.A/W)

2) The OVR toggle option would be turn on as default but the site can elect to turn it off.

3) The “Include Overview?” radio button from HLS Subheadline selection gui has been removed.

4) The tropical hazards of HI.A, HI.W, TI.A and TI.W have been removed from the product configuration file.

5) Made TR.A, HU.A and TY.A as default to not trigger 1050 Hz or SAME code (no change to tropical warnings)

6) The new Subheadline selections in the GUI have the following selectable choices:
   - NEW INFORMATION
   - PRECAUTIONARY/PREPAREDNESS ACTIONS
   - PROBABILITY OF TROPICAL STORM/HURRICANE CONDITIONS
   - WINDS
   - WINDS AND SEAS
   - STORM SURGE AND STORM TIDE
   - INLAND FLOODING
   - TORNADOES
   - TORNADOES AND WATERSPOUTS
### NWRWAVES Setup Utility

#### Message Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Yes</th>
<th>No</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process as Generic Message Type?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Use MRD Replace on CRS?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Process ONLY for Core County/Zone?</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Process for Non-Routine Broadcast Service Area(s)?</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Intro</td>
<td>Tropical Storm Watch</td>
<td></td>
</tr>
<tr>
<td>Include Preamble?</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Include County/Zone List?</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Include Issue Time?</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Include Headlines?</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Generate Overview Product?</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Include Supplemental Text?</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Repeat Headline?</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>In Summary Message?</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

#### Transmission Properties

<table>
<thead>
<tr>
<th>Transmission Status:</th>
<th>CRS</th>
<th>Pending</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alert Tones:</td>
<td>On</td>
<td>NWR Only</td>
</tr>
<tr>
<td>COR:</td>
<td>On</td>
<td>NWR Only</td>
</tr>
<tr>
<td>CAN:</td>
<td>On</td>
<td>NWR Only</td>
</tr>
<tr>
<td>CON:</td>
<td>On</td>
<td>NWR Only</td>
</tr>
<tr>
<td>EXA:</td>
<td>On</td>
<td>NWR Only</td>
</tr>
<tr>
<td>EXT:</td>
<td>On</td>
<td>NWR Only</td>
</tr>
<tr>
<td>EXB:</td>
<td>On</td>
<td>NWR Only</td>
</tr>
<tr>
<td>EXP:</td>
<td>On</td>
<td>NWR Only</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Silence Period</th>
<th>On</th>
<th>Begin Time</th>
<th>End Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Periodicity:</td>
<td></td>
<td>Minutes</td>
<td></td>
</tr>
<tr>
<td>CRS Effective Time:</td>
<td></td>
<td>minutes after the hour</td>
<td></td>
</tr>
<tr>
<td>Default Duration:</td>
<td></td>
<td>Minutes</td>
<td></td>
</tr>
</tbody>
</table>

- Hard copies uncontrolled. Verify effective date prior to use.
Figure 10 – The “Generate Overview Product?” button is set to “Yes” by default and Alert Tones set to “Off” for tropical watches (not warnings).

**How to test the HLS products at the field sites?**
The best way to test the HLS products using NWRWAVES (no user interaction/ Guardian banners) is to set up the text trigger via the “nwrwavestest.csh” script (see section 5 of User’s Guide). The output will go to the NWRWAVES’s /TEST subdirectory instead. Be sure to put the “nwrwaves.csh” back in the text triggers for operation when you are done with the testing.

**Routine Cleaning of some of the NWRWAVES directories**
This is just a reminder that there are several NWRWAVES subdirectories (e.g. /BACKUP and /ERROR) that require manual clean up after a long period of time. Remember that every time when the administrator brings up the HLS GUI, a back up copy of the HLS configuration file will be saved in the /BACKUP subdirectory.
APPENDIX G – NWRWAVES NWEM Processing

Non-Weather Emergency Product Information

<table>
<thead>
<tr>
<th>Event Code</th>
<th>AWIPS Priority</th>
<th>Event (Product) Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVW</td>
<td>Warning/Exclusive</td>
<td>Avalanche Warning</td>
</tr>
<tr>
<td>CDW</td>
<td>Warning/Exclusive</td>
<td>Civil Danger Warning</td>
</tr>
<tr>
<td>CEM</td>
<td>Warning/Exclusive</td>
<td>Civil Emergency Message</td>
</tr>
<tr>
<td>EQW</td>
<td>Warning/Exclusive</td>
<td>Earthquake Warning</td>
</tr>
<tr>
<td>EVI</td>
<td>Warning/Exclusive</td>
<td>Immediate Evacuation Warning</td>
</tr>
<tr>
<td>FRW</td>
<td>Warning/Exclusive</td>
<td>Fire Warning</td>
</tr>
<tr>
<td>HMW</td>
<td>Warning/Exclusive</td>
<td>Hazardous Materials Warning</td>
</tr>
<tr>
<td>LEW</td>
<td>Warning/Exclusive</td>
<td>Law Enforcement Warning</td>
</tr>
<tr>
<td>NUW</td>
<td>Warning/Exclusive</td>
<td>Nuclear Power Plant Warning</td>
</tr>
<tr>
<td>RHW</td>
<td>Warning/Exclusive</td>
<td>Radiological Hazard Warning</td>
</tr>
<tr>
<td>SPW</td>
<td>Warning/Exclusive</td>
<td>Shelter In Place Warning</td>
</tr>
<tr>
<td>VOW</td>
<td>Warning/Exclusive</td>
<td>Volcano Warning</td>
</tr>
<tr>
<td>AVA</td>
<td>Watch/High</td>
<td>Avalanche Watch</td>
</tr>
<tr>
<td>CAE</td>
<td>Watch/High</td>
<td>Child Abduction Emergency</td>
</tr>
<tr>
<td>LAE</td>
<td>Watch/High</td>
<td>Local Area Emergency</td>
</tr>
<tr>
<td>TOE</td>
<td>Watch/High</td>
<td>911 Telephone Outage Emergency</td>
</tr>
<tr>
<td>ADR</td>
<td>Other/General</td>
<td>Administrative Message/Follow up Statement</td>
</tr>
<tr>
<td>DMO</td>
<td>Configurable</td>
<td>Practice/Demo Warning</td>
</tr>
<tr>
<td>NIC*</td>
<td>Other/General</td>
<td>National Information Center</td>
</tr>
<tr>
<td>NPT*</td>
<td>Configurable</td>
<td>National Periodic Test</td>
</tr>
<tr>
<td>RMT**</td>
<td>Configurable</td>
<td>Routine Monthly Test</td>
</tr>
<tr>
<td>RWT**</td>
<td>Configurable</td>
<td>Routine Weekly Test</td>
</tr>
</tbody>
</table>

* not yet defined and not implemented in HazCollect at this time.
** not implemented in HazCollect at this time.

Ensure county code “000” in the localUGClookup.table and product.cfg file

All National and state-wide NWRWAVES will use “000” in the UGC rather than individual county Listening Area Codes (LACs). For example, a statewide message for Kentucky would be encoded as “KYC000-” rather than “KYC007-035-039-” for the three counties in the state. The “KYC000” code needs to map to every transmitter which will broadcast to any part of Kentucky.

The following is the procedure for adding county “000” in NWRWAVES:

1. Create an entry of county code “000” for localUGClookup.table
   a. Select NWRWAVES setup utility window from NWRWAVE browser
   b. The first tab of “Transmitter Configuration” should be displayed
   c. The first transmitter (BLF in Paducah case) will be selected by default
   d. Click on the “Edit Local LACs” button
e. Enter each entry accordingly (see Figure 1) for creating “KYC000”
f. Select “Save Edits”
g. Repeat steps d. to f. if the site involves multiple states (e.g. “ILC000” and “MOC000” for Paducah).
h. When completed, the localUGClookuptable file in /awips/adapt/NWRWAVES/bin directory should contain:

```
#LOCAL UGC LOOKUP TABLE CREATED BY NWRWAVES
ALL Kentucky (AMBER ONLY),KY,PAH,C,KYC000
ALL Missouri (AMBER ONLY),MO,PAH,C,MOC000
ALL Illinois (AMBER ONLY),IL,PAH,C,ILC000
```

2. Assigning the county code for a transmitter (a minimal of one will do)
i. Locate the lower left scroll window for “Unused Counties/Indep. Cities for SiteID (e.g. PAH)”
j. Left click on the “000” county code (e.g. KYC000)
k. Hit the “⇒” button associates with the “Routine Broadcast Selected Counties/Zones” on top right scroll window.
l. Repeat steps j. to k. if the site involves multiple states. (see Figure 2)
m. Select “Save Transmitter Edits” (Green button)
n. Select File menu ->Save & Exit option.

Once the county code of “000” is defined and configured, the NWRWAVES should be able to locate the newly defined UGC code (e.g. KYC000, ILC000 and MOC000).
Basic checklist for testing HazCollect with NWRWaves

1. Check /awips/adapt/NWRWAVES/product.cfg via NWRWAVES setup utility GUI. From the NWRWAVES setup utility GUI select Product Configuration (second tab) and for each NWEM product, select the following options and save it when it is done.

   · Process as Generic Message Type: *Yes*
   · Process Only for Core County/Zone: *No*
   · Process for Non-Routine Broadcast Service Area(s): *Yes*
   · Include Country/Zone List: *Yes*
   · Include Issue Time: *No*
   · Include Headlines:* Yes*
   · Generate Overview Product: *No*
   · Include Supplemental text: *No*
   · Storage: *VIP (Default)*
   · Transmission Status: *CRS *
   · Use if UGC expiration missing: *Yes*

2. Run “/awips2/fxa/bin/subscription –o read -t ldad –r ldad” to make sure NWEM products have the text database triggers ready, for example,

   SF DADRPAH /awips/fxa/bin/startTransmitHazWarnings.csh GEN
   ...
   ...

Appendix O

- Hard copies uncontrolled. Verify effective date prior to use. -
AWIPS Software Modification

- Changes to /awips/txa/bin/transmitHazardWarnings.pl

In the sendToCRS() section of the script make the following changes.

Modify the following line from:

\[ \text{my $formatter = "/home/CRS/NWEM/nwrnwem.csh $product_id $crs_section$trans_file";} \]

to

\[ \text{my $formatter = "/awips/adapt/NWRWAVES/nwrwaves.csh $product_id";} \]
Appendix P
Diagnosing System Health
Appendix P. Diagnosing System Health

Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>P.1 AWIPS II Communications Review</td>
<td>1</td>
</tr>
<tr>
<td>P.1.1 SBN Ingest</td>
<td>1</td>
</tr>
<tr>
<td>P.1.2 ORPG Ingest</td>
<td>3</td>
</tr>
<tr>
<td>P.1.3 LDAD Ingest</td>
<td>4</td>
</tr>
<tr>
<td>P.1.4 Product Retrieval</td>
<td>6</td>
</tr>
<tr>
<td>P.1.5 CAVE Communication</td>
<td>7</td>
</tr>
<tr>
<td>P.2 Diagnostic Tools</td>
<td>8</td>
</tr>
<tr>
<td>P.3 AWIPS II System Monitor</td>
<td>13</td>
</tr>
<tr>
<td>P.3.1 System Monitor Main Configuration File</td>
<td>17</td>
</tr>
</tbody>
</table>

List of Exhibits

<table>
<thead>
<tr>
<th>Exhibit</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exhibit P.1-1. High-Level Communications Flow in AWIPS II</td>
<td>2</td>
</tr>
<tr>
<td>Exhibit P.1.2-1. RPG Ingest Data Flow</td>
<td>3</td>
</tr>
<tr>
<td>Exhibit P.1.3-1. High-Level LDAD Ingest Data Flow</td>
<td>5</td>
</tr>
<tr>
<td>Exhibit P.3-1. AWIPS II System Monitor</td>
<td>13</td>
</tr>
<tr>
<td>Exhibit P.3-2. Radar.xml Configuration File</td>
<td>15</td>
</tr>
<tr>
<td>Exhibit P.3-3. Radar Product</td>
<td>15</td>
</tr>
<tr>
<td>Exhibit P.3.1-1. params.xml Configuration File</td>
<td>17</td>
</tr>
</tbody>
</table>

List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table P.1.1-1. LDM Processes, Users, Log Files</td>
<td>2</td>
</tr>
<tr>
<td>Table P.1.3-1. LDAD Router Processes in AWIPS II</td>
<td>6</td>
</tr>
<tr>
<td>Table P.2-1. Tools Pertinent to AWIPS II</td>
<td>8</td>
</tr>
<tr>
<td>Table P.2-2. Log Locations for AWIPS II Components</td>
<td>10</td>
</tr>
<tr>
<td>Table P.2-3. Specific Information for Log Files</td>
<td>10</td>
</tr>
<tr>
<td>Table P.3-1. AWIPS II System Monitor Cron Entries</td>
<td>14</td>
</tr>
<tr>
<td>Table P.3-2. AWIPS II System Monitor Configuration Files</td>
<td>14</td>
</tr>
<tr>
<td>Table P.3-3. AWIPS II System Monitor Configuration Files Tags</td>
<td>16</td>
</tr>
<tr>
<td>Table P.3.1-1. AWIPS II System Monitor params.xml XML Definition</td>
<td>18</td>
</tr>
</tbody>
</table>
This appendix provides an overview of tools that can be used to diagnose system health. The focus is on those components that are changed or completely new in the AWIPS II architecture. A brief overview is provided to help define the flow of some of the diagnostic tools.

AWIPS II introduces a number of new processes, and with those processes come a new set of log files to monitor. Also, the move to a load-balanced clustered suite of software (Environmental Data Exchange (EDEX)) introduces a new processing paradigm in which the server that is not tied up with other tasks performs the next one. Because of this, it is worth noting that there is no guarantee that one server’s software will process a specific piece of data; administrators need to become comfortable with checking two sets of logs.

The flow of data within the AWIPS II system (as described in Chapter 4) differs from the way AWIPS I stores and retrieves data. This will affect how certain scenarios will be diagnosed in comparing similar situations with AWIPS I.

Finally, the change in data storage paradigm that heavily relies on a combination of metadata and Hierarchical Data Format – 5 (HDF5) flat file storage was the catalyst for redesigning the AWIPS System Monitor. The new tool will provide information similar to that provided by the AWIPS I version in diagnosing system health.

P.1 AWIPS II Communications Review

AWIPS II relies heavily on communication between processes in storing and retrieving products. A high-level diagram of process communication within AWIPS II is provided in Exhibit P.1-1. A brief review of specific communications within AWIPS II follows the exhibit.

P.1.1 SBN Ingest

Data is received via the Satellite Broadcast Network (SBN) Digital Video Broadcaster (DVB)s and multicast over a private Local Area Network (LAN) to the Communications Processor – Satellite Broadcast Network (CPSBN) processors where it is captured by the noaaportIngest process. noaaportIngest replaces the dvb_multicast and readnoaaport process. noaaportIngest reads data off the SBN feed and inserts it into LDM. LDM is configured with pqact.conf (running on cpsbn1); it writes the product to physical storage on a filesystem in /data_store and sends a message into the Queue Processor Interface Daemon (Apache AMQP Message Broker) (QPID) Java Messaging Service (JMS) external.dropbox queue indicating that a product is available for decoding. The message contains the first 100 characters of the product, which includes the World Meteorological Organization (UN) (WMO) header for product identification, as well as the file path and file name under which the product was stored under /data_store.

The next available EDEX process will read from the JMS queue and will act on the next available message, and then initiate the proper plugins to work on the data based on the information contained in the JMS message. The EDEX process identifies the products using the WMO header contained in the message and compares it to a pattern in the distribution eXtensible Markup Language (XML).
Data storage by EDEX is done by sending messages to the PostgreSQL engine and to Python Process Isolated Enhanced Storage (PyPIES) running on dx2f. PostgreSQL stores necessary data into the proper tables and PyPIES is charged with HDF5 file storage.

**Note:** The “old” way will still be in use at VRH, PBP, GUM, and HFO because these sites have remote CPs, and /data_store cannot be mounted on the CP. In the future, they will get additional hardware so that they will have a local CP and will be able to baseline their configuration, as the other sites do.

**Potential Things to Check:** 

- Are all the ingest processes running on the LDM host? Table P.1.1-1 shows a list of processes and which user(s) run(s) each process for checking via the `ps` command.

**Table P.1.1-1. LDM Processes, Users, Log Files**

<table>
<thead>
<tr>
<th>Process</th>
<th>User</th>
<th>Log</th>
</tr>
</thead>
<tbody>
<tr>
<td>noaaportIngester -m 224.0.1.1 -n -u 3 -t mhs -r 1 -s NMC</td>
<td>ldm</td>
<td>~ldm/logs/nwstg.log</td>
</tr>
<tr>
<td>noaaportIngester -m 224.0.1.2 -n -u 4 -t mhs -r 1 -s GOES</td>
<td>ldm</td>
<td>~ldm/logs/goes.log</td>
</tr>
<tr>
<td>noaaportIngester -m 224.0.1.3 -n -u 5 -t mhs -r 1 -s NMC2</td>
<td>ldm</td>
<td>~ldm/logs/nwstg2.log</td>
</tr>
<tr>
<td>noaaportIngester -m 224.0.1.4 -n -u 6 -t mhs -r 1 -s NOAAPORT_OPT</td>
<td>ldm</td>
<td>~ldm/logs/oconus.log</td>
</tr>
<tr>
<td>noaaportIngester -m 224.0.1.5 -n -u 7 -t mhs -r 1 -s NMC3</td>
<td>ldm</td>
<td>~ldm/logs/nwstg3.log</td>
</tr>
</tbody>
</table>
- Check to ensure the LDM process is configured properly to act on a product based on its WMO header as configured in the /usr/local/ldm/etc/pqact.conf file.
- Check to ensure that the /data_store directory is mounted on the NAS2.
  **Note:** /data_store volume is moved off the NAS1 onto NAS2.
- Check to ensure that QPID is running and it is not having problems accepting connections. Look in the LDM ldmd.log file for the string “Cannot connect to remote EDEX instance” and check to make sure qpidd is running on the cp1f host.
- Check to ensure the EDEX processes are running on DX3 and DX4, and are not processing slowly. Check the latency and processing times in the EDEX logs.
- Check to ensure the /data_store directory is also mounted off NAS2 on the DX3 and DX4 so EDEX can access the files it is supposed to decode. You may see a lot of “file not found” messages inside of large java stack trace error messages in the EDEX logs.

**P.1.2 ORPG Ingest**

Data is received via the Transport Control Protocol (TCP) connection between the AWIPS II Radar Configuration Manager (RCM) (also known as the AWIPS II Radar Server) and the Open Radar Product Generator (ORPG). As products are received, the RCM process will write to physical storage and place a message in the QPID JMS queue. From this point on, the path is the same as SBN Ingest. Exhibit P.1.2-1 provides a high-level overview of the ingest flow from the radar product generator (RPG).

![Exhibit P.1.2-1. RPG Ingest Data Flow](image)

---

**Appendix P Diagnosing System Health**

---

**AWP.MAN.SMM.A2**

-- Hard copies uncontrolled. Verify effective date prior to use. --
Potential Issues / Things To Check: Note: The following is not an all-inclusive list; rather, it is a list of common things to check when troubleshooting ORPG ingest issues.

- A network path to the ORPG must be in place. Ping is a good utility to check this path.
- RCM must be configured with the correct Internet Protocol (IP) of the ORPG, as well as the port and password. Check the config.xml in /awips2/edex/data/config/persist.
- The RCM process must be running. Use the ps command and tail the log in /awips2/rcm/data/logs to check for the status of the RCM process.
- The /data_store directory is mounted on the NAS2. [Note: All hosts mount volume off NAS2.]
- Check to ensure the /data_store directory is also mounted off NAS2 on the DX3 and DX4 so EDEX can access the files it is supposed to decode. You may see a lot of “file not found” messages inside of large java stack trace error messages in the EDEX logs.
- QPID must be running and must not be having problems accepting connections.
- The EDEX processes must be running on DX3 and DX4, and must not be processing slowly. Check the latency and processing times in the EDEX logs.

P.1.3 LDAD Ingest

The ingest of Local Data Acquisition and Dissemination (LDAD) into the LDAD Servers (LS) and across the firewall has not changed. In fact, up to the point of preprocessing, ingest has not changed in AWIPS II. The following steps still occur in AWIPS II:

1. Data is placed into /data/Incoming on LS1.
2. The listener process on px2f is notified of new products by the newLDADdataNotification and CO_Serv processes on the LS1.
3. The listener process SCPs the data file over to px2f into /data/pxa/LDAD/tmp on px2f and the newLDADdataNotification moves the product into /data/ldad/Processed on LS1.
4. The listener process checks the LDADinfo.txt file to see if preprocessing is needed. If so, it runs the appropriate preprocessor where the data is reformatted and placed into /data/pxa/LDAD/Raw. If no preprocessing is needed, it puts the data directly into /data/pxa/LDAD/Raw.

At this point the routing paradigm is different. Rehosted LDAD processes write to raw data storage and place a message in the QPID JMS external.dropbox queue. From this point on, the path is the same as the SBN Ingest. Exhibit P.1.3-1 is a high-level depiction of the changed portion of the process.
There is potential for data to come in which is not handled by a router process in AWIPS I or in AWIPS II. For instance, raw Gridded Binary (GRIB) or GRIB2 data, which is ingested through LDAD, needs to be put somewhere for processing. That data can be placed into /awips2/edex/data/manual, which is an ingest endpoint that EDEX monitors for new products. When found, it attempts to decode the data based on the same rules it uses if it is notified of a raw product storage via the QPID JMS queue, with one extra step. If no WMO header is found in the product, then it will attempt to use the file name when matching regular expressions within distribution XML files.

**Potential Issues / Things To Check:** Note: The following is not an all-inclusive list; rather, it is a list of common things to check when troubleshooting LDAD ingest issues.

- All current processes that run on LS1 should be running and configured as they are today. Normal troubleshooting techniques apply.
- The file LdadPatterns.txt must be rehosted to the AWIPS II version of the file, which allows execution of the new router*EDEX processes. Table P.1.3-1 lists the router processes that should be running in AWIPS II and whether or not they are modified for use in AWIPS II or the exact same process as in AWIPS I. Use a `ps` command to check to see if they are all running. Note that only the modified, rehosted applications are ones which put messages on the QPID queue for EDEX decoding.
### Table P.1.3-1. LDAD Router Processes in AWIPS II

<table>
<thead>
<tr>
<th>Process Name</th>
<th>Modified or Legacy</th>
</tr>
</thead>
<tbody>
<tr>
<td>routerLdadDecoder</td>
<td>Legacy</td>
</tr>
<tr>
<td>routerStoreTextEDEX</td>
<td>Modified</td>
</tr>
<tr>
<td>routerStoreNetcdf</td>
<td>Legacy</td>
</tr>
<tr>
<td>routerStoreEDEX</td>
<td>Modified</td>
</tr>
<tr>
<td>routerShefEncoderEDEX</td>
<td>Modified</td>
</tr>
<tr>
<td>routerStoreAkBlpEDEX</td>
<td>Modified</td>
</tr>
</tbody>
</table>

- The proper libraries need to be installed for the rehosted router*EDEX processes to put messages on the QPID queue. Use `rpm -q awips2-notification` to check for this rpm.
- The file `/awips/fxa/data/edex-info.txt` must exist and must have the proper host running QPID in the first line, which should look as shown below:

  ```
  edex.msg amqp:tcp:cp1f:5672
  ```

- The `/data_store` directory is mounted on the NAS2.
- The `/awips2/edex/data/manual` directory needs to be mounted from the Network Attached Storage (NAS1) to place files that are not handled by one of the rehosted router*EDEX processes listed in Table P.1.3-1.
- The EDEX processes need to be running on DX3 and DX4, and you need to ensure they are not processing slowly. Check the latency and processing times in the EDEX logs.
- Not all the data that comes in via the LDAD system has WMO headers; therefore, the EDEX distribution XML files need to be properly configured to notify the proper decoding plugin based on the file name stored into the `/data_store` directory.

### P.1.4 Product Retrieval

CAVE is the main interface between a user and the data in AWIPS II. It first connects to EDEX through Internet Protocol Virtual Server (IPVS) when it starts. The product retrieval itself is performed through communications with the EDEX request server.

When a menu is first displayed, CAVE is requesting the most recent product time from the metadata database for each of those configured menu items. This request is sent through the connection with EDEX request process. While the request is being sent, and before a reply is received, the menu will show `???.????` for the product time. Once a reply is received, the question marks will change to the product time of the most recent product for that menu item; or, if the product has no recent time in the metadata database, it will display as `--.----`.

If the question marks stay (i.e., they do not change to product times or to the dashed notation) because no product stored matched that menu item, there could be a problem with the communication to the EDEX request process, or from the EDEX request process to the metadata database.
Once a product is loaded to a perspective’s display in CAVE, the product itself is updated by a broadcast to a specific topic to which CAVE is listening housed within QPID. When the successful product store is broadcast, the data should update, as well as the time, for the most recent product on the menus that have been previously displayed.

All product time retrievals go through an EDEX request process. However, in accessing the information within HDF5, CAVE communicates directly to PyPIES. For example, if the client has requested data which resides in hdf5 store, then it communicates directly with PyPIES on dx2f to retrieve the necessary information to visualize the data.

**Potential Issues / Things to Check.** The following is not an all-inclusive list; rather, it is a list of common things to check when troubleshooting product retrieval issues.

- Ensure IPVS is running, and that it detects at least one EDEX request server running on either DX3 or DX4.
- Ensure that there is a network path to the floating IP managed by IPVS from each Linux Workstation (LX) or Linux Text Workstation (XT). Using `ping ec` is a good start. If a user gets a banner when starting Alert Visualization (AlertViz) (or when it automatically starts when the user logs in to the desktop environment) asking for localization preferences, and the box for the localization server is red, then the path to the server is broken.
- Ensure that the user is configured to use `ec` as the localization server, and not DX3, DX4 or any other server.
- Ensure that at least one EDEX request process is running on either DX3 or DX4. Look in the `/awips2/edex/logs` if there are problems starting the EDEX request process.
- Ensure that the PostgreSQL database is running on dx1f.
- Ensure that PyPIES is running on dx2f.
- Ensure that the `/awips2/edex/data/hdf5` mount from the Direct Attached Storage (DAS) is mounted on dx2f.

**P.1.5 CAVE Communication**

Different applications integrated within the CAVE have need to communicate among themselves – for example, the WarnGen application within the D2D perspective and the TextWS application. When WarnGen is running, and a polygon and associated warning attributes have been established to the forecaster’s liking, the Create Text button is pressed. At this point, the Text Workstation application running on the configured host will display the text message for editing and transmission. This inter-application communication is done through a QPID JMS queue. There is no direct traffic to the XT from the LX in terms of communication between the AWIPS II programs.

CAVE also needs to communicate in order to disseminate products. This process is completed by sending a message to the EDEX request server to which it was connected via IPVS. The EDEX server, once it receives a transmission request, uses an application developed to transmit the product using the AWIPS I Message Handling System (MHS) code already installed on the system.
Potential Issues / Things To Check: Note: This is not an all-inclusive list; rather, it is a list of common things to check when troubleshooting inter-CAVE communication issues.

- Ensure that the Text Workstation host setting under the Text Workstation preferences is set to the accompanying XT to the LX which is being used. Access the preferences by opening CAVE → Preferences, then selecting ‘Text Workstation’ from the selection list to the left.
- Ensure that there is a network path to the floating host of cp1f, where QPID runs. Using ping cp1f is a good test of this path.
- Ensure that QPID is running along with the a2cp1apps package, and that there are no errors in the qpid logs.

P.2 Diagnostic Tools

Most of the tools that are useful in AWIPS I at a system level can be used in AWIPS II to help diagnose system issues. These include, but are not limited to:

- top (CPU monitoring)
- vmstat, free (memory monitoring)
- iostat (disk Input/Output (IO) monitoring)
- df (disk usage monitoring)
- tail, grep, less (log viewers and parsers)

However, a few new tools, or lesser known tools, can be of use as well. Table P.2-1 organizes and describes some of the new tools pertinent to AWIPS II.

<table>
<thead>
<tr>
<th>Tool</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LDM</td>
<td></td>
</tr>
<tr>
<td>ldmadmin watch</td>
<td>Running as system user ldm on the host running the LDM, one can use this command to watch the most recent data products to arrive in the queue (successful inserts). Can be used with optional -f FEEDTYPE arguments where FEEDTYPE is a valid LDM feedtype. This utility can also be used to monitor the latency of the data coming from the LDM host. Note that this does not indicate a successful action taken by pqact process.</td>
</tr>
<tr>
<td>ldmadmin watch -f FEEDTYPE</td>
<td></td>
</tr>
<tr>
<td>ldmadmin pqactcheck</td>
<td>Utility run as system user ldm that checks the syntax of the pqact.conf file.</td>
</tr>
<tr>
<td>ldmadmin pqactHUP</td>
<td>Utility run as system user ldm which will force the LDM system to re-read the active pqact.conf file(s) and begin using the new rules, assuming the file is correctly formatted.</td>
</tr>
<tr>
<td>ldmadmin tail</td>
<td>This utility, run as system user ldm, tails the LDM log file for easy monitoring without being in the logs directory. Useful for seeing which products have matched an entry in the pqact.conf file.</td>
</tr>
<tr>
<td>Tool</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Ldmping</td>
<td>Run as system user <strong>ldm</strong>. Specifically designed to check the availability of an LDM server.</td>
</tr>
<tr>
<td>notifyme</td>
<td>Run as system user <strong>ldm</strong>. This utility connects to an LDM instance and issues a specific REQUEST to see if a product can be requested. Can be used with the optional –p argument which allows a regular expression pattern to be sent.</td>
</tr>
<tr>
<td>notifyme -v -h cp1f -p ‘REGEX’</td>
<td>Use of for testing regular expression that can be used in the pqact.conf file.</td>
</tr>
<tr>
<td>pqmon</td>
<td>Utility that monitors the LDM queue file on the disk. The following are useful columns in the output of the utility:</td>
</tr>
<tr>
<td>pqmon -i X</td>
<td>- <em>nprods</em> – Number of products in the queue</td>
</tr>
<tr>
<td></td>
<td>- <em>maxprods</em> – Maximum number of products in the queue so far</td>
</tr>
<tr>
<td></td>
<td>- <em>age</em> – The age, in seconds, of the oldest product in the product queue</td>
</tr>
<tr>
<td></td>
<td>This can be used with an optional -i argument to iterate a new line of information every X number of seconds.</td>
</tr>
<tr>
<td>Database</td>
<td><strong>pgadmin3</strong></td>
</tr>
<tr>
<td></td>
<td>Graphical tool which allows a user to explore the database structure and table schema in a visual way.</td>
</tr>
<tr>
<td></td>
<td>Database queries can be made to extract data.</td>
</tr>
<tr>
<td></td>
<td><strong>Exercise caution</strong> when using this tool, as it has the ability to change rows in the database if care is not taken.</td>
</tr>
<tr>
<td></td>
<td>Utility is available in AWIPS I; however, the database user name has changed. It should now be <strong>awips</strong> and not <strong>pguser</strong>.</td>
</tr>
<tr>
<td>psql</td>
<td>Command line tool which allows a user to test a connection to, and peruse the data within, a PostgreSQL database engine.</td>
</tr>
<tr>
<td></td>
<td>Utility is available in AWIPS I; however, the database user name has changed. It should now be <strong>awips</strong> and not <strong>pguser</strong>.</td>
</tr>
<tr>
<td>QPID</td>
<td><strong>qpid-stat -q</strong></td>
</tr>
<tr>
<td></td>
<td>Command line tool run on the host which is running QPID that allows a user to see the status of individual JMS queues and topics which are housed by QPID.</td>
</tr>
<tr>
<td></td>
<td>Optional command line arguments allow for sorting by column name (-S colname) and limiting the number of queues returned (-L NUM).</td>
</tr>
<tr>
<td>HDF5</td>
<td><strong>hdfview</strong></td>
</tr>
<tr>
<td></td>
<td>Graphical user interface for looking in HDF5 files.</td>
</tr>
<tr>
<td>h5stat</td>
<td>Command line utility which can provide information about an HDF5 file. This can provide the number of datasets in a file, and other information about the HDF5 file itself.</td>
</tr>
<tr>
<td>h5ls</td>
<td>Command line tool which can list the information in each of the datasets. The –r option is a powerful recursive option that will list all data in all groups of the HDF5 file. Or a user can view group-by-group.</td>
</tr>
<tr>
<td>h5dump</td>
<td>Command line utility which can be used to view the contents of an HDF5 file.</td>
</tr>
<tr>
<td>IPVS/Pulse</td>
<td><strong>ipvsadm</strong></td>
</tr>
<tr>
<td></td>
<td>Command line tool that can be used to query how many downstream EDEX Request servers IPVS/Pulse has registered, and to see how many connections to each EDEX Request server currently are made.</td>
</tr>
</tbody>
</table>
The available logs are shown in Table P.2-2.

**Table P.2-2. Log Locations for AWIPS II Components**

<table>
<thead>
<tr>
<th>Server</th>
<th>Process</th>
<th>Log Directory</th>
</tr>
</thead>
<tbody>
<tr>
<td>dx1f</td>
<td>Radar Server</td>
<td>/awips2/rcm/data/logs/radarserver.log</td>
</tr>
<tr>
<td></td>
<td>PostgreSQL DBMS</td>
<td>/awips2/data/pg_log/postgresql-YYYY-MM-DD_HHMMSS.log</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/awips2/data/pg_log/postgresql-{Monday-Sunday}.log</td>
</tr>
<tr>
<td>dx2f</td>
<td>PyPIES</td>
<td>/awips2/pypies/logs/pypies.log.YYYY-MM-DD</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/awips2/pypies/logs/pypiesHourlyStats.log.YYYY-MM-DD</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/awips2/pypies/logs/pypiesMinuteStats.log.YYYY-MM-DD</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/awips2/httpd_pypies/var/log/httpd/access_log</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/awips2/httpd_pypies/var/log/httpd/error_log</td>
</tr>
<tr>
<td>DX3/DX4</td>
<td>EDEX</td>
<td>/awips2/edex/logs</td>
</tr>
<tr>
<td>GFE</td>
<td>GFE-ServiceBackup</td>
<td>/awips2/GFESuite/logs/&lt;site&gt;/&lt;yyyymmdd&gt;/ingestAT.log</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/awips2/GFESuite/logs/&lt;site&gt;/&lt;yyyymmdd&gt;/irtServer.log</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/awips2/GFESuite/logs/&lt;site&gt;/&lt;yyyymmdd&gt;/iscDataRec.log</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/awips2/GFESuite/logs/&lt;site&gt;/&lt;yyyymmdd&gt;/requestAT.log</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/awips2/GFESuite/logs/&lt;site&gt;/&lt;yyyymmdd&gt;/sendAT.log</td>
</tr>
<tr>
<td></td>
<td></td>
<td>awips2/GFESuite/ServiceBackup/logs/&lt;yyyymmdd&gt;/svcbu_export_grids_1553</td>
</tr>
<tr>
<td>CPSBN1/</td>
<td>QPID</td>
<td>/awips2/qpid/var/log</td>
</tr>
<tr>
<td>CPSBN2</td>
<td></td>
<td>/var/log</td>
</tr>
<tr>
<td></td>
<td>scour</td>
<td>/usr/local/ldm/logs</td>
</tr>
<tr>
<td></td>
<td>LDM</td>
<td>/usr/local/ldm/logs</td>
</tr>
<tr>
<td>LX/XT</td>
<td>AlertViz</td>
<td>~/caveData/logs</td>
</tr>
<tr>
<td></td>
<td>CAVE</td>
<td>~/caveData/etc/user/${USER}/logs</td>
</tr>
</tbody>
</table>

The logs identified in Table P.2-3 provide specific information.

**Table P.2-3. Specific Information for Log Files**

<table>
<thead>
<tr>
<th>Server</th>
<th>Log</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DX3/DX4</td>
<td>edex-ingest-YYYYMMDD.log</td>
<td>General Ingest and Operations Log</td>
</tr>
<tr>
<td>DX3/DX4</td>
<td>edex-ingestDat-YYYYMMDD.log</td>
<td>DATIngest and processing log</td>
</tr>
<tr>
<td></td>
<td>edex-ingest-gen_areal_ffg-YYYYMMDD.log</td>
<td>FFG Ingest and processing log</td>
</tr>
<tr>
<td></td>
<td>edex-ingest-gen_areal_qpe-YYYYMMDD.log</td>
<td>QPE Ingest and processing log</td>
</tr>
<tr>
<td></td>
<td>edex-ingestGrib-YYYYMMDD.log</td>
<td>Grib data ingest and processing log</td>
</tr>
<tr>
<td></td>
<td>edex-ingest-purge-YYYYMMDD.log</td>
<td>Data purge log</td>
</tr>
<tr>
<td></td>
<td>edex-ingest-radar-YYYYMMDD.log</td>
<td>Radar data ingest and processing log</td>
</tr>
<tr>
<td></td>
<td>edex-ingest-satellite-YYYYMMDD.log</td>
<td>Satellite data ingest and processing log</td>
</tr>
<tr>
<td></td>
<td>edex-ingest-shef-YYYYMMDD.log</td>
<td>Shef data ingest and processing log</td>
</tr>
<tr>
<td></td>
<td>edex-ingest-shef-performance-YYYYMMDD.log</td>
<td>Shef performance log</td>
</tr>
<tr>
<td></td>
<td>edex-ingest-smartInit-YYYYMMDD.log</td>
<td>GFE SmartInit processing log</td>
</tr>
</tbody>
</table>
### Server Logs

<table>
<thead>
<tr>
<th>Server</th>
<th>Log</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>edex-ingest-unrecognized-files-YYYYMMDD.log</td>
<td>All data which does not match a distribution pattern XML shows up in this log</td>
</tr>
<tr>
<td></td>
<td>edex-request-YYYYMMDD.log</td>
<td>EDEX Request server log</td>
</tr>
<tr>
<td></td>
<td>edex-request-thriftSrv-YYYYMMDD.log</td>
<td>Most of the requests from CAVE and a few requests from python command line utilities (e.g., some GFE items)</td>
</tr>
<tr>
<td></td>
<td>start-edex-ingest-YYYYMMDD.log</td>
<td>Log created from running service edex_camel start</td>
</tr>
<tr>
<td></td>
<td>start-edex-ingestGrib-YYYYMMDD.log</td>
<td>Log created from running service edex_camel start</td>
</tr>
<tr>
<td></td>
<td>start-edex-ingestDat-YYYYMMDD.log</td>
<td>Log created from running service edex_camel start</td>
</tr>
<tr>
<td></td>
<td>start-edex-request-YYYYMMDD.log</td>
<td>Log created from running service edex_camel start</td>
</tr>
<tr>
<td></td>
<td>start-edex-trigger-YYYYMMDD.log</td>
<td>Trigger log</td>
</tr>
<tr>
<td></td>
<td>edex-ingest-archive-YYYYMMDD.log</td>
<td>Ingest Archive log</td>
</tr>
<tr>
<td></td>
<td>edex-ingest-GFEPerformance-YYYYMMDD.log</td>
<td>GFE Performance Ingest log</td>
</tr>
<tr>
<td></td>
<td>edex-ingest-activeTableChange-YYYYMMDD.log</td>
<td>activeTableChange logs</td>
</tr>
<tr>
<td></td>
<td>edex-ingestDat-activeTableChange-YYYYMMDD.log</td>
<td>ingestDat activeTableChange logs</td>
</tr>
<tr>
<td></td>
<td>edex-ingestDat-performance-YYYYMMDD.log</td>
<td>IngestDat Performance logs</td>
</tr>
<tr>
<td></td>
<td>edex-ingestGrib-activeTableChange-YYYYMMDD.log</td>
<td>Ingest Grib activeTableChange log</td>
</tr>
<tr>
<td></td>
<td>edex-request-activeTableChange-YYYYMMDD.log</td>
<td>EDEX request activeTableChange log</td>
</tr>
<tr>
<td></td>
<td>edex-ingest-performance-YYYYMMDD.log</td>
<td>EDEX ingest performance log</td>
</tr>
<tr>
<td></td>
<td>edex-request-performance-YYYYMMDD.log</td>
<td>EDEX request performance log</td>
</tr>
<tr>
<td></td>
<td>edex-ingest-gen_areal_qpe-YYYYMMDD.log</td>
<td>EDEX ingest QPE log</td>
</tr>
<tr>
<td></td>
<td>edex-request-GFEPerformance-YYYYMMDD.log</td>
<td>GFE Performance Request Log</td>
</tr>
<tr>
<td></td>
<td>edex-request-hibernate-YYYYMMDD.log</td>
<td>EDEX request Hibernate Log (See note below.)</td>
</tr>
<tr>
<td></td>
<td>edex-ingest-hibernate-YYYYMMDD.log</td>
<td>EDEX ingest Hibernate Log (See note below.)</td>
</tr>
<tr>
<td></td>
<td>edex-ingestGrib-hibernate-YYYYMMDD.log</td>
<td>EDEX ingestGrib Hibernate log (See note below.)</td>
</tr>
<tr>
<td>dx1f</td>
<td>radarserver.log</td>
<td>RCM process log</td>
</tr>
<tr>
<td></td>
<td>postgresql-YYYY-MM-DD-HHMMSS.log</td>
<td>PostgreSQL log (new log started on new day and on server restart).</td>
</tr>
<tr>
<td></td>
<td>access_log</td>
<td>httpd-pypies access log</td>
</tr>
<tr>
<td></td>
<td>error_log</td>
<td>httpd-pypies error log</td>
</tr>
<tr>
<td>dx2f</td>
<td>pypies.log</td>
<td>pypies process log</td>
</tr>
</tbody>
</table>
Appendix P

Diagnosing System Health

<table>
<thead>
<tr>
<th>Server</th>
<th>Log</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPSBN1/2</td>
<td>ldm.log</td>
<td>LDM log</td>
</tr>
<tr>
<td></td>
<td>nwstg.log</td>
<td>readnoaaport for NWSTG feed log</td>
</tr>
<tr>
<td></td>
<td>nwstg2.log</td>
<td>readnoaaport for NWSTG2 feed log</td>
</tr>
<tr>
<td></td>
<td>goes.log</td>
<td>readnoaaport for GOES feed log</td>
</tr>
<tr>
<td></td>
<td>oconus.log</td>
<td>readnoaaport for Outside Conterminous United States (OCONUS) feed log</td>
</tr>
<tr>
<td></td>
<td>qpidd.log</td>
<td>QPID</td>
</tr>
<tr>
<td>LX/XT</td>
<td>$(hostname).alertviz.log</td>
<td>AlertViz Start Log</td>
</tr>
<tr>
<td></td>
<td>$(hostname)_caveConsole.log</td>
<td>CAVE stdout messages that would normally be seen in the console window if you ran cave.sh</td>
</tr>
</tbody>
</table>

**Note:** The hibernate log file will now be produced for each EDEX instance. Any errors produced by org.hibernate will be logged to this file. This log file should primarily be referenced whenever there are persistence errors in the EDEX instance logs.

**Known Error Messages.** Not all error messages can be listed ahead of the time that they will be seen, nor can it be foretold that one will show up when you are expecting it. However, a list of previously identified messages and what they mean follows.

**PostgreSQL Errors**

**FATAL:** no pg_hba.conf entry for host "165.92.24.3", user "awips", database "dc_ob7mhx"

This error indicates that some process is attempting to make a database connection from host 165.92.24.3 to database dc_ob7mhx as database user awips. The rules for defining this access are in the /awips2/data/pg_hba.conf. If this file does not have an entry granting host access to this IP address, the error will be seen.

**FATAL:** the database system is starting up

This error indicates that some process is attempting to make a connection to the database engine while it is starting and before it is ready. This may happen if the engine was not stopped cleanly, and is in the process of replaying transactions logs to recover lost data.

**LDM**

**ERROR:** Could not connect to the remote EDEX instance.

This error shows that the LDM processes could not connect to the QPID JMS server through the edexBridge executable.

**EDEX**

**[EDEXMain] GFESiteActivation: GFE: GFESiteActivation - OAX Error activating site OAX due to configuration**

This error shows that GFE activation failed on the server side, meaning that no GFE clients will be able to connect for the failed site. This particular error shows the problem to be with the configuration, which would be the siteConfig.py or localConfig.py file.
Appendix P

Diagnosing System Health

RPGEnvironmentalDataManager: RPGEnvData - Error connecting to RadarServer

The EDEX processes are those that will send the necessary environmental data to the ORPG. This operation is performed through the RCM process running on dx1f. If EDEX cannot connect to the RCM process, this error will show in the EDEX logs.

Note: Do not add declarations to localConfig.py that override system variables. Now that GFE processing has been incorporated into EDEX, these declarations could negatively affect the behavior of multiple systems. For example, do not set TZ inside localConfig.py as it will have an adverse effect of the behavior of WarnGen.

P.3 AWIPS II System Monitor

The AWIPS II system monitor has been changed as a result of the changes made in how data is stored and retrieved with AWIPS I. In order to keep the two separate, the AWIPS II system monitor will run from the apache default /etc/httpd directory on the server hosting the a2px1apps package. Note that the location of the http server will not change. Also staying the same will be the URL to the system monitor, http://px1f. This means that the AWIPS I bookmark will still be used. The look, however, has been changed to a more readable design as shown in Exhibit P.3-1.

Exhibit P.3-1. AWIPS II System Monitor
The AWIPS II system monitor leverages the notion that each data piece has a metadata entry within the database. It works on a timed refresh, which means it is no longer dependent on individual collection processes. This benefits the system because individual processes could fail or crash, causing data not to update. It is also a good indication of how things are behaving within the system. For example, CAVE also uses the metadata to populate the most recent product time for each of the menu entries. So, if the AWIPS II System Monitor is showing stale or missing data, it should be showing stale or missing information in the CAVE menus as well. One difference between how the system monitor and CAVE get these times is that the AWIPS II System Monitor page pulls directly from the database, while CAVE uses requests through EDEX. This is a good troubleshooting tool to use in order to isolate any problems with CAVE’s connection to EDEX, or problems with the underlying backend.

While data monitoring is no longer dependent on individual processes, disk and CPU monitoring are still done via script, which kicks off via the root crontab associated with the a2px1apps package. The crons are set up in the a2px1cron file which will be active in the /etc/cron.d directory when the package is started. Table P.3-1 lists the cron entries.

### Table P.3-1. AWIPS II System Monitor Cron Entries

<table>
<thead>
<tr>
<th>Crontab to Update Disk Monitor Portion of the AWIPS II System Monitor</th>
</tr>
</thead>
<tbody>
<tr>
<td>*/20 * * * * root /data/fxa/A2_SysMonitor/scripts/diskMon.sh &gt;&gt; /dev/null</td>
</tr>
<tr>
<td>*/20 * * * * root /data/fxa/A2_SysMonitor/scripts/ldad_diskMon.sh &gt;&gt; /dev/null</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Crontab to Update CPU Monitor Portion of the AWIPS II System Monitor</th>
</tr>
</thead>
<tbody>
<tr>
<td>*/20 * * * * root /data/fxa/A2_SysMonitor/scripts/cpuMon.sh &gt;&gt; /dev/null</td>
</tr>
<tr>
<td>*/20 * * * * root /data/fxa/A2_SysMonitor/scripts/ldad_cpuMon.sh &gt;&gt; /dev/null</td>
</tr>
</tbody>
</table>

The scripts in Table P.3-1 create an actual php file that is viewable through a link on the system monitor page. This shows the CPU utilization on the various devices. Notice that the stdout of the scripts is sent to /dev/null, which means if the disk or CPU Utilization is not working it might be useful to run one of these scripts via the command line to see if there is extra information provided to stdout which might help isolate any problems.

Configuration files that help customize the data monitoring and refresh are xml files stored in /data/fxa/A2_SysMonitor/config. Each piece of the system monitor can be configured by manipulating the delivered xml files. These files are listed in Table P.3-2.

### Table P.3-2. AWIPS II System Monitor Configuration Files

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Configuration File Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Redbook Graphics Monitoring</td>
<td>graphic.xml</td>
</tr>
<tr>
<td>Grid Data Monitoring</td>
<td>grid.xml</td>
</tr>
<tr>
<td>LDAD Acquisition Data</td>
<td>ldad_acq.xml</td>
</tr>
<tr>
<td>LDAD Dissemination</td>
<td>ldad_dissem.xml</td>
</tr>
<tr>
<td>LDAD Processes</td>
<td>ldad.xml</td>
</tr>
<tr>
<td>General System Monitor Parameters</td>
<td>params.xml</td>
</tr>
<tr>
<td>Point Data Monitoring</td>
<td>point.xml</td>
</tr>
<tr>
<td>Radar Data Monitoring</td>
<td>radar.xml</td>
</tr>
<tr>
<td>Satellite Data Monitoring</td>
<td>satellite.xml</td>
</tr>
</tbody>
</table>
Exhibit P.3-2 illustrates one of the configuration files, radar.xml. It is noted that there is site-specific data that should be filled in during the installation of the AWIPS II software.

```xml
<radar>
  <data>
    <productname>8-Bit Base Reflectivity (Z) elev0.5</productname>
    <sqltxt>SELECT reftime FROM radar WHERE productcode = '94' AND location_rda_id = 'RADAR_ID' AND primaryelevationangle = '0.5' ORDER BY reftime DESC LIMIT 1</sqltxt>
    <warnthresh>720</warnthresh>
    <errorthresh>1440</errorthresh>
  </data>
</radar>
```

**Exhibit P.3-2. Radar.xml Configuration File**

In the case of the radar file in Exhibit P.3-1, the text in the `<productname>` tags are what is shown in the Radar Product portion as seen in Exhibit P.3-3.

![Radar Product](image)

**Exhibit P.3-3. Radar Product**

The `<sqltxt>` tags enclose the SQL (Structured Query Language) statement used to pull the most recent product from the appropriate metadata table. In this case, it is selecting from the radar table.

The `<warnthresh>` is the time in seconds before the yellow ⚠️ is shown in the Status column. In this example, 12 minutes old.

The `<errorthresh>` is the time in seconds before the red ❌ is shown in the Status column. In this example, 24 minutes old.

To add a new entry, a full new `<data>` section between `<radar>` and `</radar>` tags will need to be added. The way the xml is configured, however, provides added flexibility into which data products are monitored and what each threshold will be for the alarms.

Similarly, data can be configured and added in the other configuration files listed in Table P.3-2. Table P.3-3 describes the tags used in each.
### Table P.3-3. AWIPS II System Monitor Configuration Files Tags

<table>
<thead>
<tr>
<th>XML Tag</th>
<th>Tag Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ffmp_grids.xml</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;rfc&gt;</code></td>
<td>A valid 3-letter River Forecast Center (RFC) AWIPS Identifier</td>
</tr>
<tr>
<td><code>&lt;errorthresh&gt;</code></td>
<td>The number of seconds before the data is shown as missing</td>
</tr>
<tr>
<td><code>ffmp_radar.xml</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;rda&gt;</code></td>
<td>A valid 4-letter radar identifier</td>
</tr>
<tr>
<td><code>&lt;errorthresh&gt;</code></td>
<td>The number of seconds before the data is shown as missing</td>
</tr>
<tr>
<td><code>graphic.xml</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;productname&gt;</code></td>
<td>Text to describe the product being monitored</td>
</tr>
<tr>
<td><code>&lt;sqltxt&gt;</code></td>
<td>A valid SQL query which returns the most recent inserttime of the product being monitored</td>
</tr>
<tr>
<td><code>&lt;warnthresh&gt;</code></td>
<td>The number of seconds before the data is shown as late</td>
</tr>
<tr>
<td><code>&lt;errorthresh&gt;</code></td>
<td>The number of seconds before the data is shown as missing</td>
</tr>
<tr>
<td><code>grid.xml</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;productname&gt;</code></td>
<td>Text to describe the product being monitored</td>
</tr>
<tr>
<td><code>&lt;sqltxt&gt;</code></td>
<td>A valid SQL query which returns the most recent inserttime of the product being monitored</td>
</tr>
<tr>
<td><code>&lt;warnthresh&gt;</code></td>
<td>The number of seconds before the data is shown as late</td>
</tr>
<tr>
<td><code>&lt;errorthresh&gt;</code></td>
<td>The number of seconds before the data is shown as missing</td>
</tr>
<tr>
<td><code>ldad_acq.xml</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;productname&gt;</code></td>
<td>Text to describe the product being monitored</td>
</tr>
<tr>
<td><code>&lt;filename&gt;</code></td>
<td>The full path, including file name, of the file to be monitored for punctuality on the LDAD server</td>
</tr>
<tr>
<td><code>&lt;warnthresh&gt;</code></td>
<td>The number of seconds before the data is shown as late</td>
</tr>
<tr>
<td><code>&lt;errorthresh&gt;</code></td>
<td>The number of seconds before the data is shown as missing</td>
</tr>
<tr>
<td><code>ldad_dissem.xml</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;productname&gt;</code></td>
<td>Text to describe the product being monitored</td>
</tr>
<tr>
<td><code>&lt;filename&gt;</code></td>
<td>The full path, including file name, of the file to be monitored for proper dissemination on the LDAD server</td>
</tr>
<tr>
<td><code>&lt;warnthresh&gt;</code></td>
<td>The number of seconds before the data is shown as late in being disseminated</td>
</tr>
<tr>
<td><code>&lt;errorthresh&gt;</code></td>
<td>The number of seconds before the data is shown as missing</td>
</tr>
<tr>
<td><code>ldad.xml</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;processname&gt;</code></td>
<td>The name of a process to monitor</td>
</tr>
<tr>
<td><code>&lt;host&gt;</code></td>
<td>Where the process normally runs</td>
</tr>
<tr>
<td><code>point.xml</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;productname&gt;</code></td>
<td>Text to describe the product being monitored</td>
</tr>
<tr>
<td><code>&lt;sqltxt&gt;</code></td>
<td>A valid SQL query which returns the most recent reftime of the product being monitored</td>
</tr>
<tr>
<td><code>&lt;warnthresh&gt;</code></td>
<td>The number of seconds before the data is shown as late</td>
</tr>
<tr>
<td><code>&lt;errorthresh&gt;</code></td>
<td>The number of seconds before the data is shown as missing</td>
</tr>
<tr>
<td><code>radar.xml</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;productname&gt;</code></td>
<td>Text to describe the product being monitored</td>
</tr>
<tr>
<td><code>&lt;sqltxt&gt;</code></td>
<td>A valid SQL query which returns the most recent inserttime of the product being monitored</td>
</tr>
<tr>
<td><code>&lt;warnthresh&gt;</code></td>
<td>The number of seconds before the data is shown as late</td>
</tr>
</tbody>
</table>
### P.3.1 System Monitor Main Configuration File

The AWIPS II System Monitor has one main configuration file which also resides in the `/data/fxa/A2_SYSMonitor/config` directory structure. The name of the file is `params.xml` and is shown in Exhibit P.3.1-1.

```xml
<params>
    <config>
        <dbserver>dx1f</dbserver>
        <edexserver>dx3</edexserver>
        <dbuser>awips</dbuser>
        <dbpass>awips</dbpass>
        <refresh>120</refresh>
        <ldad_refresh>600</ldad_refresh>
        <scan_refresh>120</scan_refresh>
        <ffmp_refresh>120</ffmp_refresh>
        <edex_home>/awips2/edex</edex_home>
        <common_base>/data/utility/common_static/base</common_base>
        <common_site>/data/utility/common_static/site</common_site>
        <monitor_plugin>/monitoring/MonitorPluginState.xml</monitor_plugin>
        <has_ldad>yes</has_ldad>
        <root_dir>/data/fxa/A2_SYSMonitor</root_dir>
    </config>
</params>
```

**Exhibit P.3.1-1. params.xml Configuration File**

The values shown in the exhibit are the **baselined** configuration values. They should not change very often, if at all. It is not recommended that this file be changed by the sites; however, a description of the XML tags is provided in Table P.3.1-1.
Table P.3.1-1. AWIPS II System Monitor params.xml XML Definition

<table>
<thead>
<tr>
<th>XML Tag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;dbserver&gt;</td>
<td>The server upon which the PostgreSQL database engine runs</td>
</tr>
<tr>
<td>&lt;primary_edex&gt;</td>
<td>The hostname of a server running EDEX</td>
</tr>
<tr>
<td>&lt;client_edex&gt;</td>
<td>OBE parameter</td>
</tr>
<tr>
<td>&lt;dbuser&gt;</td>
<td>The database user as whom to connect to the PostgreSQL database</td>
</tr>
<tr>
<td>&lt;dbpass&gt;</td>
<td>The password for the configured &lt;dbuser&gt;</td>
</tr>
<tr>
<td>&lt;refresh&gt;</td>
<td>How often, in seconds, the main page auto-refreshes</td>
</tr>
<tr>
<td>&lt;ldad_refresh&gt;</td>
<td>How often, in seconds, the LDAD page auto-refreshes</td>
</tr>
<tr>
<td>&lt;scan_refresh&gt;</td>
<td>How often, in seconds, the SCAN page auto-refreshes</td>
</tr>
<tr>
<td>&lt;ffmp_refresh&gt;</td>
<td>How often, in seconds, the FFMP page auto-refreshes</td>
</tr>
<tr>
<td>&lt;edex_home&gt;</td>
<td>The directory into which the EDEX software is installed</td>
</tr>
<tr>
<td>&lt;common_base&gt;</td>
<td>The base directory into which the common configuration localization tree is stored</td>
</tr>
<tr>
<td>&lt;monitor_plugin&gt;</td>
<td>The path under common_base to find the plugin state of EDEX</td>
</tr>
<tr>
<td>&lt;has_ldad&gt;</td>
<td>yes if the site has an LDAD, no if it does not</td>
</tr>
<tr>
<td>&lt;root_dir&gt;</td>
<td>The root directory of the AWIPS II System Monitor</td>
</tr>
</tbody>
</table>
Appendix Q
AWIPS Applications and Version Numbers
This appendix identifies the AWIPS applications and their correct version numbers. Please refer to Appendix U for the component categorization.

### AWIPS Applications and Version Numbers

<table>
<thead>
<tr>
<th>AWIPS Application</th>
<th>Version for FOT&amp;E Release</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aviation Forecast Preparation System (AvnFPS)</td>
<td>FOT&amp;E – 14.1.1</td>
<td>AvnFPS is designed to ease monitoring, improve production, and facilitate Quality Control (QC) of Terminal Aerodrome Forecasts (TAF) and Transcribed Weather Enroute Broadcasts (TWEB). The AvnFPS monitoring capability gives forecasters quick and continuous feedback on TAF's as well as associated observations. This monitoring capability consists of a color-coded scheme and is site configurable. The TAFs, surface observations, and guidance are easily accessible in text or graphical form and can be enhanced to alert forecasters to weather significant to aviation operations.</td>
</tr>
<tr>
<td>AWIPS Decision Assistance and Production Preparation Tools (ADAPPT)</td>
<td>FOT&amp;E – 14.1.1</td>
<td>ADAPPT Foundation software is composed of the foundation routines that support the IFPS, LAMP, Pre-LAMP, and SCAN.</td>
</tr>
<tr>
<td>CLIMATE</td>
<td>FOT&amp;E – 14.1.1</td>
<td>The CLIMATE software provides the capability to automatically initiate (three times daily) daily, monthly, seasonal, and annual climate reports. It also formats these reports for dissemination (similar to HWR). CLIMATE also provides review and edit capability for products prior to dissemination.</td>
</tr>
<tr>
<td>Dam Catalog Reviewer and Estimating Tool (DamCREST)</td>
<td>OB9</td>
<td>A Graphical User Interface application available on both Linux and Windows operating platforms.</td>
</tr>
<tr>
<td>Distributed Hydrologic Modelling (DHM)</td>
<td>OB9</td>
<td>DHM is the set of features integrated into NWSRFS for distributed hydrologic modeling.</td>
</tr>
<tr>
<td>Flash Flood Monitoring Program: Advanced (FFMPA)</td>
<td>FOT&amp;E – 14.1.1</td>
<td>An integrated suite of multi-sensor applications that detects, analyzes, and monitors precipitation and generates short-term warning guidance for flash flooding automatically within AWIPS.</td>
</tr>
<tr>
<td>AWIPS Application</td>
<td>Version for FOT&amp;E Release</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>---------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Fog Monitor (FM).</td>
<td>FOT&amp;E – 14.1.1</td>
<td>An AWIPS application that applies various algorithms to visible and infrared satellite images in order to identify potential areas of fog. At night, the application primarily uses the well-known infrared “Fog Product” (the difference of the 10.7 micron and 3.9 micron brightness temps) to highlight potential fog areas. During the daytime, several algorithms attempt to discern fog areas by brightness, shape, and other characteristics. Also applied are filters that help to distinguish fog from possible false signal features such as snow cover and mid-level clouds.</td>
</tr>
<tr>
<td>Four-dimensional Stormcell Investigator (FSI)</td>
<td>0.4</td>
<td>An innovative base radar data 4-panel display application that is based on the National Severe Storms Laboratory Warning Decision Support System - Integrated Information GUI. FSI allows users to create and manipulate dynamic cross-sections (both vertical and at constant altitude), so that they can &quot;slice and dice&quot; storms and view these cross-section data in three-dimensions and across time.</td>
</tr>
<tr>
<td>Gridded Forecast Environment Suite (GFE Suite)</td>
<td>FOT&amp;E – 14.1.1</td>
<td>A series of programs that provide an interactive gridded forecast preparation capability. Components derive surface sensible weather elements from model data, manage the forecast data and metadata in a database, and generate forecast products in a variety of formats.</td>
</tr>
<tr>
<td>Hourly Weather Roundup (HWR)</td>
<td>Version 05/18/05</td>
<td>HWR automatically receives and processes METAR surface observations, Satellite Cloud Products (SCP), Supplementary Climatological Data (SCD), and marine observations. It also formats a text portion of the data for dissemination over NWR, a text portion for the Console Replacement System (CRS), and a tabular product for dissemination over NWWS. Furthermore, the HR CSCI provides review and edit capability for products prior to dissemination.</td>
</tr>
<tr>
<td>AWIPS Application</td>
<td>Version for FOT&amp;E Release</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>---------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Hydro Database Manager (HydroBase)</td>
<td>FOT&amp;E – 14.1.1</td>
<td>HydroBase is the component of the AWIPS system that permits the management of the reference portion of the database. Adding, deleting, and editing of hydrometeorological data collection and river forecast data locations are handled within this application. It consists of a series of pull-down menus that allow for interacting with the information within the reference database.</td>
</tr>
<tr>
<td>HydroGen</td>
<td>OB9</td>
<td>HydroGen is a suite of software programs that collect data from the Integrated Hydrologic Forecast System (IHFS) database and prepare eXtensible Markup Language (XML) files and hydrograph graphics (or simply “hydrographs”) for the web.</td>
</tr>
<tr>
<td>HydroView and MPE Editor</td>
<td>HydroView: FOT&amp;E – 14.1.1</td>
<td>HydroView and MPE Editor have been reengineered into a CAVE and other CAVE plug-ins.</td>
</tr>
<tr>
<td>Interactive Forecast Preparation System (IFPS)</td>
<td>FOT&amp;E – 14.1.1</td>
<td>IFPS enables the forecaster to generate and manipulate a digital database of observed and forecasted meteorological variables, which then can be used to automatically generate various products with different formats.</td>
</tr>
<tr>
<td>Localized Aviation MOS Program (LAMP)</td>
<td>OB8.2</td>
<td>A statistical system that provides forecast guidance for sensible weather elements. The GFS-LAMP guidance currently runs operationally at NCEP. Guidance is available from the following cycles: 0000, 0300, 0600, 0900, 1200, 1500, 1800, and 2100 UTC</td>
</tr>
<tr>
<td>Local Analysis and Prediction System (LAPS)</td>
<td>OB9</td>
<td>Reads METAR, mesonet, satellite, radar, and profiler data stored at the WFO and performs an analysis and generates forecasts.</td>
</tr>
<tr>
<td>Local Data Acquisition and Dissemination (LDAD)</td>
<td>OB9</td>
<td>The LDAD software system acquires and integrates data available from automated systems and human observers in the local WFO area. It also disseminates critical and non-critical AWIPS weather information to local users, particularly state and local government emergency management agencies.</td>
</tr>
<tr>
<td>AWIPS Application</td>
<td>Version for FOT&amp;E Release</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------</td>
<td>---------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Local Storm Report (LSR)</td>
<td>FOT&amp;E – 14.1.1</td>
<td>A stand-alone AWIPS application designed to provide forecasters with an easy and quick way to create, manage, and send the LSR public text product. This text product contains noteworthy weather events for which the forecaster has either received or sought real-time observations.</td>
</tr>
<tr>
<td>NOAA Weather Radio With All-Hazards VTEC Enhanced Software (NWRWAVES)</td>
<td>Browser: Version 9.7 (12/15/08)</td>
<td>Developed to replace all the existing formatter capabilities in WWA and CAFE. NWRWAVES utilizes VTEC coding found in an increasing suite of NWS products to better identify, produce and manage outbound CRS Weather Messages.</td>
</tr>
<tr>
<td>NWS River Forecast System (NWSRFS)</td>
<td>OB9</td>
<td>A comprehensive set of hydrologic techniques used by the RFCs to perform their forecast functions.</td>
</tr>
<tr>
<td></td>
<td>OB8.3(01/02/08) OB8.3(01/02/08) OB8.3(01/02/08) OB8.3(01/02/08) OB8.3(01/02/08)</td>
<td></td>
</tr>
<tr>
<td>RFC Archive Server (RAX)</td>
<td>OB6</td>
<td>The RAX CSCI is a member of the Data Server (DS) family but is a stand-alone Linux PC at the RFCs. The RAX CSCI is dedicated to providing additional disk storage capability and processing power to the RFCs for hydrological data.</td>
</tr>
<tr>
<td></td>
<td>OB8.3 (02/01/08) OB8.3 (02/01/08) OB8.3 (02/01/08) OB8.3 (02/01/08) OB8.3 (02/01/08)</td>
<td></td>
</tr>
<tr>
<td>River Product Formatter (RiverPro)</td>
<td>OB9.2.9 (07/01/2010)</td>
<td>RiverPro application, which generates text products for the National Weather Service Hydrology program.</td>
</tr>
<tr>
<td>RiverMonitor</td>
<td>OB8.3 (03/18/07)</td>
<td>Used to provide automatically updated tabular information summarizing river conditions. Delivered in part for RiverPro VTEC monitoring; provides general monitoring of river data.</td>
</tr>
<tr>
<td>System for Convection Analysis and Nowcasting (SCAN)</td>
<td>FOT&amp;E – 14.1.1</td>
<td>An integrated suite of multi-sensor applications that detects, analyzes, and monitors convection and generates short-term probabilistic forecast and warning guidance for severe weather automatically within AWIPS.</td>
</tr>
<tr>
<td>AWIPS Application</td>
<td>Version for FOT&amp;E Release</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------</td>
<td>---------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>System for Nowcasting of Winter Weather (SNOW)</td>
<td>FOT&amp;E – 14.1.1</td>
<td>An AWIPS application suite that continuously monitors surface observations for winter weather hazards. SNOW automatically alerts the forecasters whenever such conditions are detected, and provides capabilities to display observed winter weather threats in ways that help forecasters focus on what they consider most important.</td>
</tr>
<tr>
<td>System on AWIPS for Forecasting and Evaluation of Seas and Lake (SAFESEAS)</td>
<td>FOT&amp;E– 14.1.1</td>
<td>A set of AWIPS applications that continuously monitor marine and adjacent overland conditions for specific marine weather hazards. SAFESEAS automatically alerts the forecasters whenever such conditions are detected, and provides capabilities to display observed marine threats in ways that help forecasters focus on what they consider most important.</td>
</tr>
<tr>
<td>Warning Generation (WarnGen)</td>
<td>FOT&amp;E – 14.1.1</td>
<td>WarnGen enables you to issue flash flood, severe thunderstorm, tornado, and other short-term warnings for a single storm or a line of storms. In addition, you can issue text products (follow-up statements) that update the progress of the storm, cancel the warning if conditions change, re-issue another warning on the same storm, or note the expiration of the warning. WarnGen also enables you to provide warning backup support to neighboring sites.</td>
</tr>
<tr>
<td>AWIPS Archive Server</td>
<td>Version: AWIPS II</td>
<td>It has Level IV archiving functionality and supports the Weather Event Simulator at the WFOs.</td>
</tr>
<tr>
<td>AWIPS Archive Setup</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AWIPS Archive Compressor</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CD Burner</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NMAP</td>
<td>OB9</td>
<td>NMAP displays and animates different types of meteorological data on a geographic background. The current version supports overlay of satellite, radar, model, METAR, MOS, and upper-air data. A Motif graphical user interface controls the program functions.</td>
</tr>
<tr>
<td>Site Specific Hydrologic Prediction System (SSHPS)</td>
<td>OB8.2 (9/21/07)</td>
<td>A local hydrologic model provided to allow the WFO forecaster to supplement RFC river forecast guidance by generating forecast river stages for fast-response headwater and river basins.</td>
</tr>
<tr>
<td>AWIPS Application</td>
<td>Version for FOT&amp;E Release</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------------------------</td>
<td>----------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Hydro Time Series</td>
<td>FOT&amp;E – 14.1.1</td>
<td>A subset of data within an Arc Hydro geodatabase that includes measurement values and the time they were collected. Also includes a description of the properties for a particular type of time series. Tools for visualizing time series have been developed. The tools allow the user to query the Arc Hydro database for time series records related to a particular feature, and plot one or more types of time series related to one or more spatial feature.</td>
</tr>
<tr>
<td>Precipitation Processing (PP)</td>
<td>OB9</td>
<td>This is collection of OHD applications which comprise a sub-system. RFC applications and WHFS (WFO Hydrologic Forecast System) applications require precipitation totals for station locations and grids. It is one of the prime considerations used in forecasting river stage. It is also a key indicator of flash flood potential. The processing applications ingest and process station precipitation reports and gridded precipitation data from radar and satellite sensors. The data are stored in the Integrated Hydrologic Forecast System (IHFS) database. They include 10+ applications that are non-interactive. The primary functional areas are the Gage Precipitation Pre-Processor (GPP), Multi-sensor Precipitation Estimator (MPE), and the Hi-Resolution Precipitation Estimator (HPE).</td>
</tr>
<tr>
<td>WFO Hydrologic Forecast System (WHFS) and Integrated Hydrologic Forecast System (IHFS) Database</td>
<td>FOT&amp;E – 14.1.1</td>
<td>This is a collection of OHD applications which comprise a sub-system. The WHFS/IHFS applications incorporate a comprehensive set of data ingest, data quality, and data processing operations. Associated with each observed and forecast value in the database is a quality control code which indicates the quality of the value based on external and internal tests of the value. Depending on the value of the quality code, the data are handled in different ways within the IHFS data flow and applications may or may not use the data. They contain 10+ applications that are non-interactive, including the SHEF decoder, the database purger, the rate-of-change checker, alert/alarm report generator, observed-forecast monitor, HydroGen data extractor, and SiteSpecific data handling operations.</td>
</tr>
<tr>
<td>AWIPS Application</td>
<td>Version for FOT&amp;E Release</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>High-resolution Precipitation Estimator (HPE)</td>
<td>FOT&amp;E – 14.1.1</td>
<td>HPE provides frequent (by default every 5 minutes) high resolution (~1km x 1km) rain rate and 1-hour rain accumulations covering a Weather Forecast Office’s (WFO) (or River Forecast Center’s (RFC)) area of responsibility. These gridded datasets can then be used by the Flash Flood Monitoring and Prediction Advanced (FFMP-A) system or can be displayed on D2D. The mosaicking uses data from multiple radars minimizing the distance from each radar. As a result, HPE reduces the need to run multiple instances of FFMP when convection covers more than one radar. HPE is an entirely separate program from the Multi-sensor Precipitation Estimator (MPE) and will not replace it. HPE uses different data, produces different output grids more frequently and at higher resolution, and runs with little user interaction.</td>
</tr>
</tbody>
</table>
Appendix R
System Architecture Diagrams
## Appendix R. System Architecture Diagrams

### List of Exhibits

<table>
<thead>
<tr>
<th>Exhibit R-n</th>
<th>Architecture Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>R-1</td>
<td>CONUS WFO Architecture</td>
<td>2</td>
</tr>
<tr>
<td>R-2</td>
<td>OCONUS WFO Architecture</td>
<td>3</td>
</tr>
<tr>
<td>R-3</td>
<td>RFC Architecture</td>
<td>4</td>
</tr>
<tr>
<td>R-4</td>
<td>NCEP Architecture</td>
<td>5</td>
</tr>
</tbody>
</table>
This appendix graphically represents the AWIPS software components associated with the various AWIPS hardware components.

- Exhibit R-1 depicts the CONUS WFO architecture.
- Exhibit R-2 depicts the OCONUS WFO architecture.
- Exhibit R-3 depicts the RFC architecture.
- Exhibit R-4 depicts the NCEP architecture.
CONUS WFO Architecture

Legend:
- AWIPS II Component

AWIPS II Data
GFESuite2

BOIVerify
User_storage

LX

DX1

DX2

DX3

DX4

NAS

CP1

CP2

AX

DX1f
Radar Server
PostgreSQL Database Engine

DAS

• NIS
• MHS processes
dhcpd (printers)
• NTP
• Trap Daemon
sendmail
• NWWS Product
notificationServer
• nwrTrans.pl
• DNS
dhcpd (printers)

DX2f

• PostgreSQL Data
• HDF5

PyPIES

• LAPS
• MSAS
• HWR
• Climate
• Web Server
• asyncScheduler
• NWWS Scheduler
• SBNebNoMonitor
• VIR Test
• Xyplex hosting
• Print spooler
• purge_files
• floodseq
• report_alarm
• roc_checker
• hmMonitorServer

LAP S
• M S A S
• HWR
Climate
• Web Server
asyncScheduler
NWWS Scheduler
SBNebNoMonitor
VIR Test
Xyplex hosting
Print spooler
purge_files
floodseq
report_alarm
roc_checker
hmMonitorServer

CP1f

LDM Writer
DVB
EDEX Bridge
QPID
IPVS

LX

DX3

DX4

NAS

CP1

CP2

AX

C A V E

F I R E W A L L

Firewall

Serial Mux

• N I S
• N T P
• Trap Daemon
sendmail
• NWWS Product
notificationServer
• nwrTrans.pl
• DNS
dhcpd (printers)

LDAD

LDAD routers
dhcpd (printers)

LS1

LS2

LS3

Legend:
- AWIPS II Component

Exhibit R-1. CONUS WFO Architecture
OCONUS WFO Architecture

Exhibit R-2. OCONUS WFO Architecture
RFC Architecture

Exhibit R-3. RFC Architecture
Appendix S
CAVE Localization Perspective
AWIPS II supports a base/site/user localization pattern similar to AWIPS I. In AWIPS II, the localization data is preserved in a Localization Store that is physically mounted on the NAS and is accessible via the EDEX Request process running on the DX3/4 servers. The Localization Store includes configuration for both CAVE and EDEX.

The CAVE Localization Perspective provides a graphical interface to access the CAVE localization data. This perspective provides a familiar file browser interface for locating configuration files. The Localization Perspective also shields the user somewhat from the physical structure of the Localization Store, providing a logical grouping of files by application/perspective (D2D, GFE, etc.), and it enforces access restrictions for editing certain files.

The Localization Perspective contains three views: the File Browser view; the File Editor view; and the File Outline view.

1. The File Browser view provides a familiar tree approach to locating a specific configuration file. The configuration files are organized by application (CAVE, D2D, GFE, etc.) and then by function and file name. In AWIPS II, there may be several versions of each file, including BASE, CONFIGURED (GFE only), SITE,
WORKSTATION, and USER. Each of these is listed as a separate entry under the actual file name.

2. The File Editor view provides an editor for the selected file. When a configuration file is selected, it opens in an appropriate editor; for example, a Python file is opened in the Python editor, and an XML file is opened in an XML editor.

3. The File Outline view provides a hierarchical view of certain files. When available, you can click on an element in the File Outline to select that element in the file.

CAVE is installed on the LX and XT workstations, and is normally started via the desktop application launcher mechanism. On the LX workstations, CAVE is launched in full visualization mode, which allows access to the Localization Perspective; on the XT workstations, CAVE is launched as the Text Workstation, which does not allow access to the Localization Perspective.

This exercise provides basic procedures for using the Localization Perspective.

Note: AlertViz must be running on the LX workstation before you can start CAVE; normally AlertViz starts when the user logs into the LX workstation.

1. Open the CAVE Localization Perspective.

   CAVE is normally started from the popup Start Menu on the LX workstation. This procedure walks you through the initial startup of CAVE and opening the Localization Perspective.

   Note: This procedure provides the basic steps for accessing the localization perspective. These steps should be used at the start of each of the remaining procedures.

   a. Log into the LX workstation using your student login.

   b. Start CAVE. First, left click on the LX desktop to display the Start Menu.

   c. Select ‘Start CAVE’.

   d. If CAVE starts correctly, a splash screen will be displayed as CAVE starts.
After a short pause, the main CAVE window will display:

e. Open the Localization Perspective. First, click the ‘perspective’ button (to the left of the D2D tab) to display the list of available perspectives.
f. Select ‘Localization’.

**Note:** If *Localization* is not listed, select ‘Other...’ to open the *Open Perspective* dialog.
Once the Open Perspective dialog is displayed, select ‘Localization’ and click ‘OK’. The Localization Perspective will open.

g. After selecting your GFE role by clicking ‘OK’, the Localization Perspective opens.

2. Edit configuration files in the Localization Perspective.
   Once the Localization Perspective has been opened, specific configuration files are located using the File Browser located at the left of the window.

   In this procedure, you will locate and open two files using the Localization Perspective: an XML file and a Python code file.

   a. If CAVE is not running on your workstation, follow the steps in Procedure 1 to start CAVE; once CAVE is running, open the Localization Perspective.

   b. The File Browser on the left is used to locate various configuration files. Expand CAVE→Bundles→scales→scalesInfo.xml to locate the file that controls the scale selector drop-down in the D2D perspective of CAVE.
Note that there is a SITE level and BASE level version of this file.

c. Double click on ‘SITE’ to open the site-level version in the XML editor.
Note: The XML Editor has two display options; Design and Source (as noted at the bottom of the scalesInfo.xml-SITE window). By default, the file is opened in Design mode. Design mode provides a tree-like view of the XML file. In design mode, existing values may be modified without directly editing the XML.

d. To view the actual XML, click the ‘Source’ tab at the bottom of the editor.

e. Expand D2D→Derived Parameters→python→DerivParamImporter.py to locate the Derived Parameters Python Importer script.
Note that there is only a single version of this file, the BASE version.

f. Double click on ‘BASE’ to open the BASE version of the file in the editor.

Note: For the Python file, a Python language enabled file editor is provided.
g. Close any open editors.

3. Use the perspective to make a change to a file that can be seen in CAVE.

This procedure steps through the creation of a local (USER) version of a site-level file. Once created, the file may be modified and saved. Finally, the file system on DX3 is checked to verify the file modification. The file you will copy is the *scalesInfo.xml* menu dropdown file that you opened earlier in this exercise.

   a. If CAVE is not running on your workstation, follow the steps in Procedure 1 to start CAVE; once CAVE is running, open the *Localization Perspective*. Once the *Localization Perspective* is open, locate *scalesInfo.xml* in the *File Browser*.

   b. Right click on ‘SITE’ and select ‘Copy to→User (“xxxxx”)’ from the popup menu. This will make a copy of the file into the selected site localization.
Replace xxxxx with your user name on the system.

c. Double click ‘USER (xxxxx)’ to open scalesInfo.xml for editing.
d. Use the arrow next to the last mapScale in the editor to expand down the field. Notice that it is the entry for the WFO scale.

![Image of CAVE editor with mapScale settings]

See what happens when this scale is deleted! Right click on the “mapScale” and select ‘Remove’.

![Image of Remove dialog box]

```
Remove
Add DTD Information...
Edit Namespaces...
Add Attribute
Add Child
Add Before
Add After
```

e. Click on the ‘Source’ tab at the bottom of the middle window and view the XML syntax. Notice that there is no `<mapScale>` for WFO now. Click the ‘Save’ button in the upper left corner of CAVE (under the CAVE menu).
f. Use the perspective-switching tabs at the top to go back to the D2D perspective. Use the menu dropdown selector, and try to switch to WFO scale. It should not be available!

g. Switch back to the localization perspective by clicking on the tab. Click the ‘Source’ tab at the bottom of the window if scalesInfo is open in Design mode. Then hold down the right mouse button over the XML itself and navigate to ‘Compare With → Local History’.
You will notice that a History tab appears on the right side of the perspective.
h. Double click on the previous version and note that a comparison tab appears in the middle portion of the window. Note that the righthand side of the file shows the WFO entry present and highlighted.

Close the ‘Compare scalesInfo.xml Current and Local Revision’ tab by using the “x” in the title bar

i. Use the right mouse button to select the older version in the History browser, taking note to hold down the button until a menu appears. Select ‘Open’.
Note that a new tab has opened, which has the XML of the old version of the file. It does not have the string “USER” in the tab’s title and the WFO entry is present.

Close the tab by using the “x” in the tab title bar.

j. Hold down the right mouse button in the XML of the scalesInfo.xml – USER tab and select ‘Replace With → Previous From Local History’.

Note that the file now shows the WFO entry present.

k. Use the perspective-switching tabs at the top to go back to the D2D perspective. Use the menu dropdown selector, and try to switch to WFO scale. It is now available again!

l. Switch back to the localization perspective and right click on the USER (xxxxx) version of scalesInfo.xml in the leftmost column (File Browser). Select ‘Delete’.

You will receive a confirmation dialog, to which you should answer ‘OK’.

4. Look at a file that is not in XML format.

There are other types of files accessible via the Localization Perspective, and this procedure will show a few of them.

a. If CAVE is not running on your workstation, follow the steps in Procedure 1 to start CAVE; once CAVE is running, open the Localization Perspective. Once the Localization Perspective is open, do the following to find 8 bit Refl.cmap
- Expand CAVE in the File Browser by clicking the ►
- Expand Color Maps in the File Browser by clicking the ►
- Expand Radar folder by clicking the ►
- Expand 8 bit Refl.cmap by clicking the ►

**Note:** cmap files are color map files.

b. Right click on ‘BASE’ and select ‘Copy to → User(xxxxx)’ from the popup menu. This will make a copy of the file into the selected site localization.
Replace xxxxx with your user name.

c. Double click ‘User (xxxxx)’ to open the color map for editing.
Note that the color wheel launches within the Localization Perspective to allow for easy editing of the color map.

d. Close the color wheel editor by clicking the ‘x’ in the tab titled “8 bit Refl.cmap – USER”.

e. Open a terminal window on your workstation.

f. Log into DX3 using your student login.
Enter: `ssh dx3`

g. Change directory to the EDEX Localization Store.
Enter: `cd /awips2/edex/data/utility/cave_static/user/xxxxx/colormaps/Radar`

h. Perform a file listing to show the file has been copied to the EDEX Localization Store.
Enter: `ls -l`

i. Log off DX3.
Enter: `exit`

j. Expected results: Your terminal session should be similar to that shown below.

```
8 bit Refl.cmap 8 bit Refl.cmap.md5
```

k. Use the already-open CAVE Localization Perspective to remove the user-level color map by holding down the right mouse button while selecting the USER (xxxxx) file in the File Browser and selecting ‘Delete’.

5. Edit a location plot information file to add a new location.

Location plot information (.lpi) can also be edited via the localization perspective to add or remove stations.

a. If CAVE is not running on your workstation, follow the steps in Procedure 1 to start CAVE; once CAVE is running, open the Localization Perspective. Once the Localization Perspective is open, do the following to find 8 bit Refl.cmap

- Expand CAVE in the File Browser by clicking the ►
- Expand Basemaps in the File Browser by clicking the ►
- Expand ldad15.spi folder by clicking the ►

b. Right click on the SITE (XXX) file and select Copy To → User (xxxxx).

c. Double click on the file and note that it opens in plain text format in the middle pane. Add a line at the top of the file that looks like the other entries. Note: Use spaces and not TABs in this file. Use the following to decide what to add:
If on LX1:
0  PRESIDENT  38.9000  -77.0000  9999  30000.000  PRES-METAR

If on LX2:
0  RUSHMORE  43.72  -102.78  9999  30000.000  RUSH-METAR

If on LX3:
0  LIBERTY  40.68  -74.03  9999  30000.000  LIBERTY-METAR

If on LX4:
0  GRANDCANYON  35.25  -112.76  9999  30000.000  CANYON-METAR

If on LX5:
0  SPACENEDDLE  47.43  -112.67  9999  30000.000  NEEDLE-METAR

Save the file using the button at the top left of the CAVE window.

d. Switch over to the D2D perspective and load “Maps → LDAD Stations” and look for your new station!

e. Switch back to the Localization perspective and delete the USER (xxxxx) file by holding down the right menu button while selecting the file in the File Browser and selecting ‘Delete’.

f. Confirm the delete by clicking ‘OK’.
Appendix T
Service Backup
Appendix T. Service Backup

Table of Contents

Page
T.1 Scope ............................................................................................................................. 1
T.2 AWIPS I Service Backup ............................................................................................ 1
T.3 AWIPS II Service Backup .......................................................................................... 2
   T.3.1 Service Backup Design and Configuration ........................................................... 3
      T.3.1.1 Service Backup GUI ..................................................................................... 5
         T.3.1.1.1 Emergency GFE .................................................................................... 6
         T.3.1.1.2 Import Configuration .............................................................................. 6
         T.3.1.1.3 Export Digital Data to Failed Site ........................................................... 9
         T.3.1.1.4 Export Configuration .............................................................................. 10
         T.3.1.1.5 Export Digital Data ................................................................................. 11
         T.3.1.1.6 Export Failed Site’s Digital Data to Central Server .................................. 13
   T.3.2 Initiating Service Backup .................................................................................... 14
T.4 Intrasite Coordination in AWIPS II ......................................................................... 17
T.5 EDEX Service Backup Service ................................................................................ 18
T.6 IFPS Service Backup in AWIPS II .......................................................................... 18
T.7 Modified Functionality ............................................................................................. 20

List of Exhibits

Exhibit T.2-1. AWIPS I Service Backup GUI ................................................................. 1
Exhibit T.3-1. AWIPS II Service Backup Interface ....................................................... 2
Exhibit T.3.1-1. AWIPS II Service Backup Directory Structure .................................. 3
Exhibit T.3.1.1-1. Service Backup GUI ......................................................................... 6
Exhibit T.3.1.1.2-1. Import Configuration GUI ............................................................. 7
Exhibit T.3.1.1.2-2. Import Configuration Execution .................................................... 7
Exhibit T.3.1.1.2-3. Digital Data Import Option ............................................................. 8
Exhibit T.3.1.1.2-4. Import Digital Data Execution ....................................................... 8
Exhibit T.3.1.1.3-1. Export Grids to Failed Site ............................................................. 9
Exhibit T.3.1.1.3-2. Export Data to Failed Site - Sender ............................................... 10
Exhibit T.3.1.1.3-3. Export Data to Failed Site - Receiver ............................................. 10
Exhibit T.3.1.1.4-1. Configuration Exported Table Item ............................................... 11
Exhibit T.3.1.1.5-1. Digital Data Exported Table Item .................................................. 12
Exhibit T.3.1.1.5-2. Export Digital Data Execution ....................................................... 13
Exhibit T.3.1.1.6-1. Grids Exported to Central Server .................................................. 13
Exhibit T.3.1.1.6-2. Export Digital Data to Central Server Execution ........................................ 14
Exhibit T.6-1. AWIPS II App Launcher ..................................................................................... 19
Exhibit T.6-2. AWIPS II Service Backup Application ................................................................... 19

List of Tables
Table T.4-1. ISC Configuration Files .......................................................................................... 18
The AWIPS service backup capability is designed to allow one WFO to assume the functions of another site that has failed in order to retain high availability of services and products. Therefore, an effective method of transferring configuration and forecast data was incorporated into the AWIPS system. AWIPS II seeks to incorporate this capability into the new architecture.

### T.1 Scope

This document describes the service backup capabilities present in the AWIPS II system. The major functions present in the AWIPS I service backup component have been incorporated into the AWIPS II architecture. There are several important differences between the existing AWIPS I service backup component and the service backup component present in AWIPS II. These differences are described in this appendix, and the new procedure for initiating and using service backup in AWIPS II is explained in detail.

### T.2 AWIPS I Service Backup

The AWIPS I service backup (see Exhibit T.2-1) component exposes 10 operations to the user via a graphical user interface for executing service backup.

![Service Backup GUI](image)

**Exhibit T.2-1. AWIPS I Service Backup GUI**
These operations are:

1. Emergency GFE
2. Import Configuration Data
3. Import Failed Site's Digital Data
4. Export Grids to Failed Site
5. Export CCC’s Configuration to the Central Server
6. Export CCC’s Digital Forecast to the Central Server
7. Export a Failed Site's Digital Data to the Central Server
8. Start GFE
9. Exit Service Backup
10. Clean Up Service Backup

T.3 AWIPS II Service Backup

AWIPS I uses a stand-alone graphical user interface for executing service backup operations. AWIPS II has replaced the stand-alone GUI with an interface that is a component of CAVE for controlling service backup operations. (See Exhibit T.3-1.)

Exhibit T.3-1. AWIPS II Service Backup Interface
A fundamental difference between the AWIPS I and AWIPS II architectures is that AWIPS II can run as two or more WFOs simultaneously. The primary site is your site. Any additional sites that are currently running on your EDEX server are listed under the active sites list.

### T.3.1 Service Backup Design and Configuration

AWIPS II reuses much of the functionality of the scripts from AWIPS I, but these have been significantly cleaned up to only perform the needed functions. In AWIPS I the scripts contained a lot of code that is not used.

These scripts are included in the AWIPS II installer and are placed in the appropriate directories upon successful installation of AWIPS EDEX.

The service backup scripts are installed to {INSTALL_DIR}/GFESuite. The directory structure of the GFESuite directory is shown in Exhibit T.3.1-1.

![Exhibit T.3.1-1. AWIPS II Service Backup Directory Structure](image)
The main environment file (svcbu.env) is installed to ServiceBackup/configuration. The new configuration file svcbu.env, sources in the settings from the svcbu.properties file found in the same directory and sets the python environment. Note that the overall number of configurations has been significantly reduced from is found in AWIPS I. The system attempts to set up the AWIPS_HOME variable automatically if one is not set. Typically, this variable would be set to /awips2.

Two example local environment files have been included and reside in the ServiceBackup/configuration folder. These files contain the local environment settings for a particular site. Several modifications have been made to these files so as to conform to the AWIPS II directory hierarchy. These are:

- SVCBU_HOST variable needs to be changed to the name of the main EDEX server.
- CDSPORT variable needs to be changed to the port of the main EDEX server, i.e., 9581.
- PYTHONPATH should not be set as it is already configured by EDEX.
- PRIMARY_SITES should only be used by multi-domain sites. Otherwise, the setting must remain blank. As of 13.5.1, the software automatically uses the AW_SITE_IDENTIFIER setting. The setting is a comma-separated list of sites for the export grids cron to run for instead of the site set as AW_SITE_IDENTIFIER.
- NATIONAL_CENTER setting has been deprecated as of 13.5.1 and should no longer be part of configuration.

The file svcbu_export_elements.ccc lists the weather elements that are to be packaged when transmitting digital data back to the central server. If you choose to use it, place it in the ServiceBackup/data folder. Additional files may be placed in this directory for different sites.

The scripts used for executing the service backup operations are included in the folder ServiceBackup/scripts. They are a subset of the scripts used in AWIPS I for service backup and contain several modifications for use with AWIPS II. The changes include:

- **Removal of ProgressBar.py and showProgress.py.** These have been replaced by equivalent Java dialogs.
- **Removal of the tcl scripts.** Tcl is not used in AWIPS II service backup for GUIs. The Java interface is used instead.
- **Removal of cron script.** The cron script that executes the export digital data operation has been removed. In AWIPS II, this task is executed from a quartz timer defined in the gfe-request.xml file in the EDEX GFE plug-in. A quartz timer has also been added to the same file to regularly remove accumulated log file directories. The /awips2/edex/conf/spring/cron.properties file defines time settings for the above quartz jobs.
• **Modifications to export and import configuration scripts.** Several modifications were made to the export and import configuration scripts. AWIPS II now bundles the information in \{EDEX_HOME\}/data/utility/common_static/site/\{site\} \{EDEX_HOME\}/data/utility/edex_static/site/\{site\}, and \{EDEX_HOME\}/data/utility/cave_static/site/\{site\} into the tarball to be exported. The import configuration scripts also expect these directories to be present. Upon receipt, the configuration tarball is unpacked and the site configuration is placed into the appropriate localization directories.

The scripts are executed from a class called ServiceBackupJobManager.

Upon startup, EDEX adds the bin directories contained in the ServiceBackup directory structure to the PATH variable (see start.sh located in edex/bin directory). The directory structure containing the executables is different (simplified) from those found in AWIPS I. It is contained in the GFESuite/ServiceBackup directory. Since AWIPS II supports the idea of defining a dynamic install location, these files are not placed in the root of the system.

### T.3.1.1 Service Backup GUI

The Service Backup interface handles the same operations as the A1 service backup GUI operations critical to service backup. These are:

1. Emergency GFE
2. Import Configuration Data
3. Import Failed Site's Digital Data
4. Export Grids to Failed Site
5. Export CCC's Configuration to the Central Server
6. Export CCC's Digital Forecast to the Central Server
7. Export a Failed Site's Digital Data to the Central Server
8. Start GFE
9. Exit Service Backup
10. Clean Up Service Backup

The Export configuration and Export digital data operations are for use by your site. The remaining options operate on data used for backing up the failed site. Upon completion of the selected operation, a pop-up message will appear notifying the user of the success or failure of the operation. The following sections describe the available operations.

The service backup GUI is shown in Exhibit T.3.1.1-1.
T.3.1.1.1 Emergency GFE

Emergency GFE serves the purpose of a ‘quick’ backup. When the user chooses this option, a failed site’s configuration will be downloaded from the central server and GFE will open. No grids are downloaded using this option.

T.3.1.1.2 Import Configuration

The import configuration operation imports the configuration information for a site so that service backup can be initiated. This operation is executed by clicking the Import Configuration Data radio button followed by the OK button at the bottom of the GUI as shown in Exhibit T.3.1.1-1. The user will be presented with a prompt to enter the failed site id (see Exhibit T.3.1.1.2-1).
This operation communicates with the Message Handling System (MHS) to retrieve the configuration for the desired site. Once the configuration data arrives, the MHS kicks off a script called receive_configuration. This script unpacks the retrieved data and places the configuration information in the appropriate localization directories for the requested site. Exhibit T.3.1.1.2-2 shows the flow of execution for importing a configuration.

The digital data operation imports grid data for a site so service backup can be initiated. This operation is executed either by selecting it at the time the import of configuration is initiated, or by clicking the Import Digital Forecast radio button, followed by the OK button at the bottom of the main Service Backup GUI (see Exhibit T.3.1.1.2-3). Note: The Import Digital Forecast button in Exhibit T.3.1.1.2-3 will be in disable mode (grayed out) while you are in the process of importing a site’s configuration or digital data.
Exhibit T.3.1.2.3. Digital Data Import Option

This operation sends a request message to the MHS via the msg_send application. The MHS requests the data and places it into MHS X400 directory. The MHS then kicks off a script that integrates the data into the data stores via iscMosaic. After this operation is completed, service backup is ready to use for backing up a failed site. Exhibit T.3.1.1.2-4 shows the flow of execution for importing digital data.

**Note:** The Start GFE button in the above Exhibit will only be enabled after you have successfully imported the failed site’s configuration. Also, it will be disabled while you are importing a failed site’s configuration or digital data or when you are not in Service backup mode.

Exhibit T.3.1.2.4. Import Digital Data Execution
T.3.1.1.3 Export Digital Data to Failed Site

The “export digital data to failed site” operation (when you are in the Service Backup mode) exports the digital data from your site back to the failed site. This operation is executed by clicking the Export Grids to Failed Site radio button followed by the OK button at the bottom of the GUI. (See Exhibit T.3.1.1.3-1.)

![Export Grids to Failed Site](service_backup.png)

**Exhibit T.3.1.1.3-1. Export Grids to Failed Site**

Executing this operation kicks off a sequence of execution on both the sender side and the side receiving the data. The sending side packages the grids in netCDF format and sends a message to the MHS via msg_send, which sends the grids back to the failed site. The MHS at the receiving site kicks off a script to unpack and place the received grids in the RESTORE database using iscMosaic.

Exhibit T.3.1.1.3-2 shows the flow of execution on the sending side when the Export to Failed Site operation is initiated.
Exhibit T.3.1.1.3-2. Export Data to Failed Site - Sender

Exhibit T.3.1.1.3-3 shows the flow of execution on the receiving site when the export to failed site operation is initiated.

Exhibit T.3.1.1.3-3. Export Data to Failed Site - Receiver

### T.3.1.1.4 Export Configuration

The export configuration operation packages your site’s configuration information and transmits it to the central server. This operation is executed by clicking the Export CCC’s Configuration to the Central Server radio button followed by the OK button at the bottom of the GUI. (See Exhibit T.3.1.1.4-1.)
Exhibit T.3.1.1.4-1. Configuration Exported Table Item

Exporting a configuration consists of:

1. Executing the export_configuration script to package up the site configuration for your site from the common_static, edex_static, and cave_static directories and
2. Transmitting them via msg_send to the central server.

**T.3.1.1.5 Export Digital Data**

The export digital data operation exports your site’s digital data to the central server. This operation is executed by clicking the Export CCC’s Digital Forecast to the Central Server radio button followed by the OK button at the bottom of the GUI. (See Exhibit T.3.1.1.5-1.)
Exhibit T.3.1.1.5-1. Digital Data Exported Table Item

Exporting digital data consists of packaging up the site grids using ifpnetCDF and placing them in the export grids directory located at \
\{AWIPS_HOME\}/GFESuite/exportgrids/primary for grids exported for the primary site(s), and \{AWIPS_HOME\}/GFESuite/exportgrids/backup for grids exported for the failed site. The grids placed in the exportgrids subdirectories get uploaded to the central server via the rsync process.

Exhibit T.3.1.1.5-2 shows the flow of execution for exporting digital data.
### T.3.1.1.6 Export Failed Site’s Digital Data to Central Server

This operation exports the failed site’s digital data from your site to the central server. The operation is executed by clicking the Export [a Failed Site’s Data] to the Central Server radio button followed by the OK button. (See Exhibit T.3.1.1.6-1.)

![Exhibit T.3.1.1.6-1. Grids Exported to Central Server](grid_export.png)
This operation packages the digital data for the failed site and sends it to the central server via the MHS.

In Exhibit T.3.1.1.6-1, the Exit Service Backup button will be enabled only while you are in Service Backup Mode. It will allow you to clean up the Failed Site’s Configuration, and it will close the Service Backup GUI when finished.

The Clean Up Service Backup button in Exhibit T.3.1.1.6-1 will allow you to clean up a Failed Site’s Configuration if you are in Service Backup Mode. If you are not currently in the Service Backup Mode, you will be prompted for the Site ID that you wish to clean up. Also, this button will be disabled while you are importing a failed site’s digital data.

Exhibit T.3.1.1.6-2 shows the flow of execution for exporting digital data to the central server.

![Exhibit T.3.1.1.6-2. Export Digital Data to Central Server Execution](image)

**T.3.2 Initiating Service Backup**

1. Service backup for a failed site is initiated through the command line, or by using appLauncher (described shortly). Start the Service backup interface either through appLauncher, or by entering at the command line

   **TYPE:**

   `/awips2/cave/cave.sh -component ServiceBackup`
2. Import the configuration data for the failed site. If the configuration files are already present on your system, this step is not necessary. If the configuration data for that site is not present on your system, it needs to be retrieved. Click the Import Configuration Data radio button followed by the OK button at the bottom of the GUI. Prompts for the failed site id and additional operations to execute will display in succession. Enter the failed site id and click OK to proceed. If the configuration is successfully retrieved, a pop-up message will confirm it. At the time of requesting configuration, you will also have the option of importing Digital Forecast Data and starting GFE:
3. Assuming the configuration was imported successfully and the site was activated successfully, the digital data may now be imported.
4. Service backup scripts will contact the MHS and retrieve the digital data for that site. If you had not selected to start GFE during the first step, you would be presented with the option to do.

![Service Backup window]

Service backup is now ready for use for the selected site. CAVE can now be started localized to the failed site.

**T.4 Intrasite Coordination in AWIPS II**

ISC has been incorporated into the EDEX Request Processes that run on DX3 and DX4. The EDEX Request process performs ISC routing tasks based on configuration in two configuration files:

1. `localConfig.py`. Defines ISC transmission information.
2. `siteConfig.py`. Defines ISC server information.

Both of these files are converted from the existing AWIPS I files at software installation; In AWIPS II, these files are located in the AWIPS II Localization Store on DX3/4. The file locations are shown in Table T.4-1.
Table T.4-1. ISC Configuration Files

<table>
<thead>
<tr>
<th>File</th>
<th>Location (Relative to /awips2/edex/data/utility)</th>
</tr>
</thead>
<tbody>
<tr>
<td>localConfig.py</td>
<td>edex_static/site/XXX/config/gfe</td>
</tr>
<tr>
<td>siteConfig.py</td>
<td>edex_static/site/XXX/config/gfe</td>
</tr>
</tbody>
</table>

In Table T.4-1, XXX represents the site identifier, e.g., OAX.

**Note:** Do not add declarations to localConfig.py that override system variables. Now that GFE processing has been incorporated into EDEX, these declarations could negatively affect the behavior of multiple systems. For example, do not set TZ inside localConfig.py as it will have an adverse effect on the behavior of WarnGen.

It is important to recognize that ISC operation remains essentially unchanged. As grids are modified and/or published in the CAVE GFE perspective, they are forwarded to the ISC server. Depending on the configuration values in localConfig.py, this may be automatic or it may be by forecaster action.

### T.5 EDEX Service Backup Service

The server side applications (ifpServer, etc.) supporting Service Backup have been reengineered as EDEX Services. Operationally, there is little change in AWIPS II. The configuration location for Service Backup has changed. The primary configuration file, located in /awips2/GFESuite/ServiceBackup/configuration, is svcbu.properties.

The EDEX Service Backup Service is part of the EDEX request process. It is activated by a Quartz trigger that fires at 15 minutes after each hour. Service Backup events are logged in two places: 1) in the EDEX request log in /awips2/edex/logs; and 2) in the Service Backup log in /awips2/GFESuite/ServiceBackup/logs/<yyyymmdd>, where yyyymmdd is the date stamp for the logs.

The rsync process is still used to send exported grids to the Central Server. Those grids are rsync’ed from the /awips2/GFESuite/exportgrids directory.

### T.6 IFPS Service Backup in AWIPS II

For AWIPS II, GFE has been reengineered into the GFE perspective in CAVE; as previously discussed, the GFE server applications have been reengineered as EDEX plugins. The AWIPS I Interactive Forecast Preparation System (IFPS) Service Backup Graphical User Interface (GUI) application has been reengineered as a Java application for AWIPS II.

On the LX workstation, Service Backup is accessed from the AWIPS II App Launcher, shown in Exhibit T.6-1 as Start IFPS Service Backup GUI. Alternately, Service Backup can be started on LX stations by entering /awips2/cave/cave.sh –component ServiceBackup at the command prompt.
When Service Backup starts, the display is similar to Exhibit T.6-2.

Exhibit T.6-2. AWIPS II Service Backup Application
T.7 Modified Functionality

In AWIPS I, while in GFE service backup, many sites manipulate their backup site’s GFE grids using their own procedures and tools. This allows sites to manage these grids efficiently, using techniques (familiar) to them. In AWIPS I, a common method used to accomplish this was to create and pre-stage a backup GFE user account (i.e., backup_pub, backup_gjt...etc). This account would contain all the local GFE tools and procedures that the sites wish to use during service backup. Sites would copy and stage these tools and procedures into their appropriate directories in /awips/GFESuite/primary/data/databases/<backupuser>/. When GFE service backup is invoked, sites would select the appropriate user account (i.e., backup_pub) to start GFE.

In AWIPS II this functionality can be replicated with the following:

In /awips2/GFESuite/ServiceBackup/configuration/svcbu.properties, modify SVCBU_USER=1 from the original setting of 0, and SVCBU_USER_ID to any generic tester id available on the system.

If the generic user has been created, and the Start GFE option has been selected, GFE will run as the generic user.

Additionally, dual domain sites should set the PRIMARY_SITES variable to any sites being used as primary, separated by commas in the svcbu.properties to get Service Backup to acknowledge that additional site IDs are primary, and thus allow the export of configuration to the central server correctly. For normal sites, this is not a permitted operation, and the PRIMARY_SITES variable should be left blank. When PRIMARY_SITES is blank, the value is defaulted to the AW_SITE_IDENTIFIER value.

For releases prior to OB13.3.1, invoking Service backup as the National Center requires this command: /awips2/cave/cave.sh -component ServiceBackup -site XXX -nc TRUE. Thereafter Service Backup can be invoked normally without any extra flags.

Please Note: As of OB13.5.2, the output directories for export_grid operations changed to /awips2/GFESuite/exportgrids/primary and /awips2/GFESuite/exportgrids/backup versus the old /awips2/GFESuite/exportgrids and /awips2/GFESuite/exportgrids2, which were previously used by the dual domain sites. Changes were made to the central server script to allow it to differentiate between single and multi-domain sites, and process accordingly.
Appendix U
AWIPS II Component Categorization
# Appendix U. AWIPS II Component Categorization

## Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>U.1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>U.2 Component Categorization</td>
<td>1</td>
</tr>
</tbody>
</table>
**U.1 Introduction**

AWIPS II provides for the migration of a portion of AWIPS functionality into a new component-based structure. Some AWIPS software applications have been reengineered into CAVE, EDEX, and the Radar Server for AWIPS II, while others were either rehosted or wrapped. Still others required no modification.

**U.2 Component Categorization**

This appendix identifies AWIPS II software applications and categorizes them according to their migration paths. Applying the following definitions, most can be categorized as reengineered, rehosted, wrapped, or untouched:

- **Reengineered.** A *reengineered* application is an application or functionality that has been rewritten from top to bottom. The replacement of an existing functionality with a FOSS (free and open source software) solution is also considered reengineering. Reengineered applications have generally been incorporated into CAVE, EDEX, and the Radar Server. Some data ingest functionality has been replaced by the Local Data Manager (LDM).

- **Rehosted.** A *rehosted* application is one that has been packaged for delivery with the AWIPS II software; it may be executed from CAVE or EDEX, or from an operating system launching mechanism. As a general rule, a rehosted application runs as an independent process; the application being executed is obtained from the latest Operational Build. Rehosted applications generally utilize their own configuration and log files although the configuration files are usually contained in the AWIPS II localization store.

- **Wrapped.** A *wrapped* application or functionality is one that has been incorporated into AWIPS II by means of a bridge (for example, the JEPP Java-Python bridge), between AWIPS II code and pre-AWIPS II code.

- **Untouched.** An *untouched* application (also known as an *unmodified* application) is one that has not been touched. In a few cases (primarily, interactive hydro applications), the application has also been relocated into the EDEX data directory structure; in these cases, the configuration has been moved into the AWIPS II localization store.

### AWIPS II Migration and Component Categorization

<table>
<thead>
<tr>
<th>AWIPS Application</th>
<th>Acronym</th>
<th>Migration Path to AWIPS II</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hydro</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Metar2shef</td>
<td>HF/MS</td>
<td>Reengineered into a endpoint transformer to the Shef Plug-In</td>
</tr>
<tr>
<td>ShefDecoder</td>
<td>HF/SHEF</td>
<td>Reengineered into Shef Plug-In</td>
</tr>
<tr>
<td>DPA Processing</td>
<td>HF/DPA</td>
<td>Wrapped into a SOA service end point</td>
</tr>
<tr>
<td>GagePP</td>
<td>HF/GPP</td>
<td>Reengineered into Shef Plug-In</td>
</tr>
</tbody>
</table>
## AWIPS II Migration and Component Categorization

<table>
<thead>
<tr>
<th>AWIPS Application</th>
<th>Acronym</th>
<th>Migration Path to AWIPS II</th>
</tr>
</thead>
<tbody>
<tr>
<td>FloodEvent Archiver</td>
<td>HF/FEA</td>
<td>Rehosted</td>
</tr>
<tr>
<td>ReportAlarm</td>
<td>HF/RA</td>
<td>Wrapped into a SOA service and end point activated by Quartz</td>
</tr>
<tr>
<td>Data Alert/Alarm Checker</td>
<td>HF/AAC</td>
<td>Reengineered into Shef Plug-In</td>
</tr>
<tr>
<td>MPE FieldGenerator</td>
<td>HF/FG</td>
<td>Wrapped into an SOA component activated by Quartz or MPE Editor</td>
</tr>
<tr>
<td>File Purge</td>
<td>HF/PF</td>
<td>Wrapped and activated by Quartz</td>
</tr>
<tr>
<td>Database Purge</td>
<td>HF/PDB</td>
<td>Wrapped and activated by Quartz</td>
</tr>
<tr>
<td>Hydro Point Data Control</td>
<td>HF/PDC</td>
<td>Reengineered into CAVE HydroView Perspective</td>
</tr>
<tr>
<td>WHFS Utilities</td>
<td>HF/WU</td>
<td>Wrapped into a library</td>
</tr>
<tr>
<td>HydroGen</td>
<td>HF/HG</td>
<td>Rehosted</td>
</tr>
<tr>
<td>HydroBase</td>
<td>HF/HB</td>
<td>Reengineered into CAVE HydroView Perspective</td>
</tr>
<tr>
<td>HydroView</td>
<td>HF/HV</td>
<td>Reengineered into CAVE HydroView Perspective</td>
</tr>
<tr>
<td>TimeSeries</td>
<td>HF/TS</td>
<td>Reengineered into CAVE HydroView Perspective</td>
</tr>
<tr>
<td>RiverPro</td>
<td>HF/RPF</td>
<td>Rehosted</td>
</tr>
<tr>
<td>Site Specific Hydrologic Predictor</td>
<td>HF/SSHP</td>
<td>Untouched except to activate from CAVE</td>
</tr>
<tr>
<td>DamCrest</td>
<td>HF/DC</td>
<td>Untouched except to activate from CAVE</td>
</tr>
<tr>
<td>River Monitor/Precip Monitor</td>
<td>HF/RM</td>
<td>Untouched except to activate from CAVE</td>
</tr>
<tr>
<td>MPE Editor</td>
<td>HF/MED</td>
<td>Reengineered into MPE CAVE Perspective</td>
</tr>
<tr>
<td>Database schema</td>
<td>HY/DBS</td>
<td>Untouched</td>
</tr>
<tr>
<td>Database access</td>
<td>HY/DBA</td>
<td>AddedHibernate Object to relational mapping and access</td>
</tr>
<tr>
<td>WHFS support functions</td>
<td>HY/UF</td>
<td>Wrapped into Library</td>
</tr>
<tr>
<td>Calb</td>
<td>RF/CALB</td>
<td>Rehosted</td>
</tr>
<tr>
<td>Distributed Hydrologic Modeling</td>
<td>RF/DHM</td>
<td>Rehosted</td>
</tr>
<tr>
<td>Ensemble ENS</td>
<td>RF/ENS</td>
<td>Rehosted</td>
</tr>
<tr>
<td>Flash Flood Guidance (FFG)</td>
<td>RF/FFG</td>
<td>Reengineered into CAVE-EDEX plug-ins.</td>
</tr>
<tr>
<td>Gribit application</td>
<td>RF/GRIB</td>
<td>Rehosted</td>
</tr>
<tr>
<td>Historical Data Browser</td>
<td>RF/HDB</td>
<td>Rehosted</td>
</tr>
<tr>
<td>Interactive Calibration Program</td>
<td>RF/ICP</td>
<td>Rehosted</td>
</tr>
<tr>
<td>Interactive Double Mass Analysis</td>
<td>RF/IDMA</td>
<td>Rehosted</td>
</tr>
<tr>
<td>Interactive Forecast Program</td>
<td>RF/IPF</td>
<td>Rehosted</td>
</tr>
<tr>
<td>Operational Forecast System</td>
<td>RF/OFS</td>
<td>Rehosted</td>
</tr>
<tr>
<td>OSFDE</td>
<td>RF/OFSDE</td>
<td>Rehosted</td>
</tr>
<tr>
<td>Send_RTC</td>
<td>RF/Send_RFC</td>
<td>Rehosted</td>
</tr>
<tr>
<td>Util</td>
<td>RF/Util</td>
<td>Rehosted</td>
</tr>
<tr>
<td>Verification</td>
<td>RF/VERIF</td>
<td>Rehosted</td>
</tr>
<tr>
<td>XDAT</td>
<td>RF/XDAT</td>
<td>Reengineered into CAVE HydroView Perspective</td>
</tr>
<tr>
<td>XNAV</td>
<td>RF/XNAV</td>
<td>Rehosted (RFC only application)</td>
</tr>
</tbody>
</table>
## AWIPS II Migration and Component Categorization

<table>
<thead>
<tr>
<th>AWIPS Application</th>
<th>Acronym</th>
<th>Migration Path to AWIPS II</th>
</tr>
</thead>
<tbody>
<tr>
<td>XSETS</td>
<td>RF/XSETS</td>
<td>Rehosted</td>
</tr>
<tr>
<td>SHEF Decoder for RAX</td>
<td>RAX/SHEF</td>
<td>Rehosted</td>
</tr>
<tr>
<td>RAXUM</td>
<td>RAX/RAXUM</td>
<td>Rehosted</td>
</tr>
<tr>
<td><strong>GFE</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ifpServer</td>
<td>GF</td>
<td>Reengineered into EDEX Plug-In and EDEX service</td>
</tr>
<tr>
<td>IFPS Data Server</td>
<td>GF</td>
<td>Reengineered into EDEX Plug-In and EDEX service</td>
</tr>
<tr>
<td>GFE Graphical Forecast Editor</td>
<td>GF</td>
<td>Reengineered into CAVE Perspective</td>
</tr>
<tr>
<td>Common Libraries</td>
<td>GF</td>
<td>Reengineered</td>
</tr>
<tr>
<td>Smart Tools</td>
<td>GF</td>
<td>Ported Baseline Tools to run in CAVE</td>
</tr>
<tr>
<td>Text Formatters</td>
<td>GF</td>
<td>Ported Baseline Formatters to run in CAVE including auto testing</td>
</tr>
<tr>
<td>Other GFESuite Program</td>
<td>GF</td>
<td>Reengineered (iscNetCDF, ifpImage, ...)</td>
</tr>
<tr>
<td>Data</td>
<td>GF</td>
<td>Stored in HDF5</td>
</tr>
<tr>
<td><strong>WFO-A</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GEO Datasets</td>
<td>WF</td>
<td>Reengineered into postgisSQL tables</td>
</tr>
<tr>
<td>InteractiveDepict</td>
<td>WF</td>
<td>Reengineered into CAVE resources</td>
</tr>
<tr>
<td>Monitoring/Control</td>
<td>WF</td>
<td>Replaced with JMX</td>
</tr>
<tr>
<td>CommsRouter</td>
<td>CO</td>
<td>Replaced with EDEX ESB</td>
</tr>
<tr>
<td>Data Controller</td>
<td>CO</td>
<td>Replaced with EDEX ESB</td>
</tr>
<tr>
<td>Data Mgmt Message</td>
<td>CO</td>
<td>Replaced with EDEX ESB</td>
</tr>
<tr>
<td>Monitor</td>
<td>CO</td>
<td>Replaced with EDEX ESB</td>
</tr>
<tr>
<td>Notification Server</td>
<td>CO</td>
<td>Replaced with EDEX ESB</td>
</tr>
<tr>
<td>Radar Ingest</td>
<td>RD</td>
<td>Replaced with Java Radar Server</td>
</tr>
<tr>
<td>Radar Applications</td>
<td>RD</td>
<td>Reengineered into CAVE D2D Perspective</td>
</tr>
<tr>
<td>Radar Display</td>
<td>RD</td>
<td>Reengineered into CAVE D2D Perspective</td>
</tr>
<tr>
<td>WarnGen</td>
<td>WG</td>
<td>Reengineered into CAVE D2D Perspective with velocity templates</td>
</tr>
<tr>
<td>Graphic Workstation</td>
<td>GW</td>
<td>Reengineered into CAVE D2D Perspective</td>
</tr>
<tr>
<td>WFO Archive Server</td>
<td>WAX</td>
<td>Replaced with NAS raw archive</td>
</tr>
<tr>
<td>Local Storm Reporting</td>
<td>LR</td>
<td>Reengineered into CAVE</td>
</tr>
<tr>
<td>BUFR Model Output Statistics MOS</td>
<td>BM</td>
<td>Reengineered into EDEX Plug-In</td>
</tr>
<tr>
<td>NWR Editor</td>
<td>NE</td>
<td>Rehosted</td>
</tr>
<tr>
<td>Web Dissemination</td>
<td>WD</td>
<td>No longer part of AWIPS</td>
</tr>
<tr>
<td>IFPS</td>
<td>IF</td>
<td>Reengineered into EDEX Plug-In</td>
</tr>
<tr>
<td>ADAPPT Foundation Software</td>
<td>MN</td>
<td>Reengineered into EDEX</td>
</tr>
<tr>
<td>Climate Software</td>
<td>CL</td>
<td>Rehosted</td>
</tr>
<tr>
<td>Hourly Weather Roundup (HWR)</td>
<td>HR</td>
<td>Rehosted</td>
</tr>
<tr>
<td>NWWS</td>
<td></td>
<td>Rehosted</td>
</tr>
<tr>
<td>LDAD Software</td>
<td>LD</td>
<td>Reused with added interfaces to AWIPS-II</td>
</tr>
<tr>
<td>NCF Enterprise Management</td>
<td>EI</td>
<td>Untouched</td>
</tr>
<tr>
<td>AWIPS Foundation Software</td>
<td>AF</td>
<td>Overcome by EDEX ESB</td>
</tr>
</tbody>
</table>
## AWIPS II Migration and Component Categorization

<table>
<thead>
<tr>
<th>AWIPS Application</th>
<th>Acronym</th>
<th>Migration Path to AWIPS II</th>
</tr>
</thead>
<tbody>
<tr>
<td>Freeware Software</td>
<td>PS</td>
<td>New AWIPS-M set</td>
</tr>
<tr>
<td>Commercial Off-the Shelf (COTS)</td>
<td>CS</td>
<td>Untouched</td>
</tr>
<tr>
<td>Asynchronous Scheduler</td>
<td>ASY</td>
<td>Untouched except to reconfigure and interface to AWIPS-M</td>
</tr>
<tr>
<td>Text Database System</td>
<td>DB</td>
<td>Reengineered into EDEX Plug-In</td>
</tr>
<tr>
<td>Decoding</td>
<td>DE</td>
<td>Reengineered into EDEX Plug-In</td>
</tr>
<tr>
<td>Hydro Point Data Control</td>
<td>HPD</td>
<td>Reengineered into CAVE HydroView Perspective</td>
</tr>
<tr>
<td>Localization</td>
<td>LOC</td>
<td>Reengineered to be dynamic rather than defined at install time</td>
</tr>
<tr>
<td>Message Handling System (MHS)</td>
<td>MH</td>
<td>Untouched with new Java interface API</td>
</tr>
<tr>
<td>Satellite Decoding and Display</td>
<td>ST</td>
<td>Reengineered into EDEX Plug-In and CAVE Plug-In</td>
</tr>
<tr>
<td>Text Workstation</td>
<td>TW</td>
<td>Reengineered into EDEX Plug-In and CAVE Plug-In</td>
</tr>
<tr>
<td>WAN Msg Handling</td>
<td>WMH</td>
<td>Untouched with new Java interface API</td>
</tr>
<tr>
<td>NWS Site Specific Forecasts</td>
<td>NWS SPOT</td>
<td>Untouched</td>
</tr>
<tr>
<td>Legal Archiver</td>
<td>LGL</td>
<td>Untouched</td>
</tr>
<tr>
<td>FXA file system</td>
<td></td>
<td>Controlled by install but internal structure unchanged</td>
</tr>
<tr>
<td>IFPS Service Backup Central Server</td>
<td>ISB</td>
<td>Untouched</td>
</tr>
<tr>
<td>Chatserver - 12Planet</td>
<td></td>
<td>Untouched</td>
</tr>
<tr>
<td>Test Mode Control Program (tmcp)</td>
<td></td>
<td>Reengineered</td>
</tr>
<tr>
<td>AvnFPS</td>
<td>AV</td>
<td>Reengineered into EDEX Plug-In and CAVE Plug-In</td>
</tr>
<tr>
<td>Flash Flood Monitoring FFMP</td>
<td>FF</td>
<td>Reengineered into EDEX Plug-In and CAVE Plug-In</td>
</tr>
<tr>
<td>WWA to NWR</td>
<td>WW</td>
<td>Removed from AWIPS</td>
</tr>
<tr>
<td>Local MOS (LAMP)</td>
<td>LM</td>
<td>Removed from AWIPS</td>
</tr>
<tr>
<td>Pre-LAMP Subsystem</td>
<td>PL</td>
<td>Removed from AWIPS</td>
</tr>
<tr>
<td>SCAN</td>
<td>SC</td>
<td>Reengineered into EDEX Plug-In and CAVE Plug-In</td>
</tr>
<tr>
<td>SNOW</td>
<td>SN</td>
<td>Reengineered into EDEX Plug-In and CAVE Plug-In</td>
</tr>
<tr>
<td>SAFESEAS</td>
<td>SS</td>
<td>Reengineered into EDEX Plug-In and CAVE Plug-In</td>
</tr>
<tr>
<td>LAPS</td>
<td>LS</td>
<td>Rehosted by GSD to run with AWIPS-M interfaces</td>
</tr>
<tr>
<td>MSAS</td>
<td></td>
<td>Rehosted by GSD to run with AWIPS-M interfaces</td>
</tr>
<tr>
<td>FOG (Fog Monitor)</td>
<td>FOG</td>
<td>Reengineered into EDEX Plug-In and CAVE Plug-In</td>
</tr>
<tr>
<td>Rapid SCAN</td>
<td>RSC</td>
<td>Reengineered into EDEX Plug-In and CAVE Plug-In</td>
</tr>
<tr>
<td>NWRWAVES</td>
<td>NW</td>
<td>Rehosted</td>
</tr>
<tr>
<td>4D Stormcell (FSI)</td>
<td>FSI</td>
<td>Untouched</td>
</tr>
<tr>
<td>HazCollect</td>
<td></td>
<td>Rehosted and send notifications to AlertViz</td>
</tr>
<tr>
<td>Guardian</td>
<td>GDN</td>
<td>Reengineered into AlertViz</td>
</tr>
<tr>
<td>Digital Mesocyclone Display</td>
<td>DMD</td>
<td>Reengineered into CAVE and RadarServer</td>
</tr>
<tr>
<td>Marine Weather Warning (MWW)</td>
<td>MWW</td>
<td>Port to AWIPS-M</td>
</tr>
<tr>
<td>Degrib</td>
<td></td>
<td>Reengineered into EDEX Plug-In with NCEP decoder</td>
</tr>
</tbody>
</table>
## AWIPS II Migration and Component Categorization

<table>
<thead>
<tr>
<th>AWIPS Application</th>
<th>Acronym</th>
<th>Migration Path to AWIPS II</th>
</tr>
</thead>
<tbody>
<tr>
<td>NDFD</td>
<td></td>
<td>Untouched</td>
</tr>
<tr>
<td>D2D Station OBS Viewer, Point Data Control</td>
<td></td>
<td>Reengineered into CAVE hydroview perspective</td>
</tr>
<tr>
<td>Time of Arrival</td>
<td>TOA</td>
<td>Reengineered into CAVE D2D perspective tool</td>
</tr>
<tr>
<td>GFE ISC</td>
<td>ISC</td>
<td>Reengineered into EDEX GFE plug-in and CAVE plug-in</td>
</tr>
</tbody>
</table>
Appendix V
Using D2D Gridded Datasets in GFE
Appendix V. Using D2D Gridded Datasets in GFE

Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>V.1</td>
<td>Introduction</td>
<td>1</td>
</tr>
<tr>
<td>V.2</td>
<td>Create a parameterInfo XML File</td>
<td>1</td>
</tr>
<tr>
<td>V.3</td>
<td>Add Any New Parameters Names to gfeParamName.xml</td>
<td>3</td>
</tr>
<tr>
<td>V.4</td>
<td>Add Any New Levels to gfeLevelMappingFile.xml</td>
<td>4</td>
</tr>
<tr>
<td>V.5</td>
<td>Add Your New parameterInfo File to gfeParamInfo.xml</td>
<td>5</td>
</tr>
<tr>
<td>V.6</td>
<td>Add Any Accumulative Parameters to D2DAccumulativeElements</td>
<td>5</td>
</tr>
<tr>
<td>V.7</td>
<td>Add the New Model to D2DMODELS</td>
<td>6</td>
</tr>
</tbody>
</table>

List of Tables

Table V.1-1. Conventions and Variables Used in This Appendix ............................................. 1
Table V.2-1. Convert CDL to XML Information .............................................................................. 2
Table V.2-2. parameterInfo Format ............................................................................................. 3
Appendix V

Using D2D Gridded Datasets in GFE

V.1  Introduction

This appendix provides an overview of the use of D2D Gridded Datasets in GFE. Once the user’s gridded data is ingesting and visible via the Product Browser in D2D, the user can set up access to this data from GFE by using the information presented here.

This appendix covers the creation of a parameterInfo XML file, the addition of new parameter names to the gfeParamName.xml, the addition of new levels to the gfeLevelMappingFile.xml, the addition of a new parameterInfo file to gfeParamInfo.xml, the addition of accumulative parameters to D2DAccumulativeElements, and the addition of the new model to D2DMODELS.

Note: Ingest of non-baseline GriB datasets is outside the scope of this appendix.

Certain conventions and variables are used throughout this appendix. Table V.1-1 details these conventions and variables, and defines any globally applicable information.

Table V.1-1. Conventions and Variables Used in This Appendix

<table>
<thead>
<tr>
<th>Convention</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Commands to be Executed</td>
<td>Commands are in Courier New font face and are assumed to be executed on a single line (although formatting conventions and command length might cause commands to wrap to a new line in this document).</td>
</tr>
<tr>
<td>CCC</td>
<td>This identifies the AW_SITE_IDENTIFIER under which EDEX is running as defined in /awips2/edex/bin/setup.env, i.e., the “localization” regardless of the GFE_DOMAINNAME. For dual-domain sites, the configurations are to be placed in the AW_SITE_IDENTIFIER site-level localization tree and not in the GFE_DOMAINNAME.</td>
</tr>
<tr>
<td>Site Level Overrides</td>
<td>All files are complete overrides; that is, copy the file(s) from its base location to site and edit the base file making your additions. Do not create empty files under the site as this will affect base definitions. There are DRs to create/append overrides. Also, all JVMs on all EDEX Processing Servers are required to be bounced after changes are made. Testing can be done by bouncing a single EDEX Processing Server’s JVMs and running a smartInit with the –h dx3 flag (or other EDEX Processing Server taken as the flag’s argument) to not pass through the edexCluster and ensure utilization of the new caches created by the edits. However, all EDEX Processing Servers are required to be updated for new model data ingest and automatic smartInit running.</td>
</tr>
</tbody>
</table>

V.2  Create a parameterInfo XML File

/awips2/edex/data/utility/common_static/base/grid/parameterInfo
/awips2/edex/data/utility/common_static/site/CCC/grid/parameterInfo
This file is usually given a name that is similar to the datasetID of the D2D dataset with which it is associated; however, it is actually associated with the datasetID by the gfeParamInfo.xml file (see section V.4).
The parameterInfo files contain information that was in the .cdl/.cdf files in AWIPS I.

- If you have a .cdl file from AWIPS I, you can use the **convertCDL2XML** utility to create your parameterInfo file. To convert a pre-existing AWIPS I CDL file into an AWIPS II XML, follow the steps in Table V.2-1.

### Table V.2-1. Convert CDL to XML Information

<table>
<thead>
<tr>
<th>Step</th>
<th>Action(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Locate the CDL to convert</strong></td>
<td>Most non-baseline (site-specific) CDL files are in /data/fxa/customFiles but they may also be in /awips/fxa/data. A simple find command similar to <code>find /awips/fxa -name &quot;*.cdl&quot;</code> can be performed to locate any files. Note that /awips/fxa is a local mount and most likely any AWIPS I CDL files were stored on DX3/DX4 where a grids localization would have been executed.</td>
</tr>
<tr>
<td><strong>Copy files to a staging directory</strong></td>
<td>The easiest approach is to stage the CDL files; however, you can also simply provide the location in AWIPS I as an argument to the conversion utility, as well as the output directory for the XML, when running the script in the next step. Typically, a good location is /localapps to ensure retention of files.</td>
</tr>
</tbody>
</table>
| **Run the conversion utility** | The convertCDL2XML exists anywhere that the `awips2-cli-${RELEASE}` RPM is installed and the script resides in /awips2/fxa/bin. The ncgen utility is required for the script to run, which is provided with the netcdf base RedHat RPM on most systems. If any requirement for the script to run is not present, the script will exit and report the error.  
To run the script (assuming that the CDL files to be converted were copied to /localapps/CDL2XML and you have changed directories into /localapps/CDL2XML):

```
/awips2/fxa/bin/convertCDL2XML -f /localapps/CDL2XML
```
Simply running the convertCDL2XML without any arguments or with the `-h` flag will report back syntax and arguments. The default output directory will be `pwd/convertedCDL` (where `pwd` is the directory in which the shell is currently running when the script is executed), passing the `-o` flag followed by an output directory to which the converted XML file will be stored; for example, `-o /data/local` will put the AWIPSII XML converted by the script into /data/local and not `pwd/convertedCDL`. |
| **Verify and copy the XML** | It is good practice to verify that the XML was created properly. In most instances it will be correct. However, if the CDL is sufficiently unique (various levels for certain parameters, for example) then some XML entries may be handled improperly by ncgen. If you notice instances where CDL files are not handled properly, open a ticket with the NCF. Verify the XML against the parameterInfo Format shown in Table V.2-2.  
**Note:** Copy the XML file from the convertedCDL output directory to the proper location, **common_static/site/grid/parameterInfo**. |

- If you do not have a CDL file, just copy one of the baseline parameterInfo files and use it as a template. These files are located in **edex_static/base/grid/parameterInfo**.
Regardless of whether you are using the `convertCDL2XML` utility or copying a baseline XML `parameterInfo` file as a template, the file requires some important tags. These tags are shown in Table V.2-2.

### Table V.2-2. parameterInfo Format

<table>
<thead>
<tr>
<th>XML Tag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;fcst&gt;</code></td>
<td>These tags define the forecast times for the model in seconds relative to the model reference time.</td>
</tr>
<tr>
<td></td>
<td>There should be one tag for every forecast time you wish to use in GFE.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Upstream of OB13.3.1-21, the <code>&lt;fcst&gt;</code> tag is now mandatory in the <code>parameterInfo</code> for all datasets and for all forecast hours that may be required in GFE. This is true regardless of whether or not they exist in the GriB data. (See section V.4 for more information.)</td>
</tr>
<tr>
<td><code>&lt;short_name&gt;</code></td>
<td>This tag defines the parameter name as it will be seen in GFE (e.g., in WxElementBrowser using IFP as source). It is associated the D2D parameter name via the gfeParamName.xml file. (See section V.3 for more information.)</td>
</tr>
<tr>
<td><code>&lt;units&gt;</code></td>
<td>This tag defines the units of the ingested data and should normally match the value from postgres. This unit string must be parsable by the <code>javax.measure.unit</code>.</td>
</tr>
<tr>
<td></td>
<td>You can use the following SQL query to determine the appropriate unit string for a parameter while connected to dx1f:</td>
</tr>
<tr>
<td></td>
<td><code>psql -U awips -d metadata -c &quot;SELECT (abbreviation,unit) FROM parameter ORDER BY abbreviation&quot;</code></td>
</tr>
<tr>
<td></td>
<td>You can also use various other SQL statements to determine units for individual parameters; for example, to determine units for potential vorticity abbreviations, use:</td>
</tr>
<tr>
<td></td>
<td><code>psql -U awips -d metadata -c &quot;SELECT (abbreviation,unit) FROM parameter WHERE abbreviation ilike '%pv%' ORDER BY abbreviation&quot;</code></td>
</tr>
<tr>
<td><code>&lt;valid_range&gt;</code></td>
<td>This pair of tags defines the minimum and maximum valid values for the parameter.</td>
</tr>
<tr>
<td></td>
<td>Values outside this range will be clamped to the min/max value.</td>
</tr>
<tr>
<td><code>&lt;fillValue&gt;</code></td>
<td>This tag defines the value in the grid that will be used as the “no data” value.</td>
</tr>
<tr>
<td></td>
<td>This value would normally lie outside the range of valid values.</td>
</tr>
<tr>
<td><code>&lt;level&gt;</code></td>
<td>This tag defines the level names of the data that will appear in GFE. They are associated with the actual D2D level values by the gfeLevelMappingFile.xml (see section V.4).</td>
</tr>
</tbody>
</table>

### V.3 Add Any New Parameters Names to gfeParamName.xml

```
/awips2/edex/data/utility/common_static/base/parameter/alias
/awips2/edex/data/utility/common_static/site/CCC/parameter/alias
```

Entries in this file are of the form:
Appendix V

Using D2D Gridded Datasets in GFE

V.4 Add Any New Levels to gfeLevelMappingFile.xml

/awips2/edex/data/utility/edex_static/base/grid
/awips2/edex/data/utility/edex_static/site/CCC/grid

Entries in this file are of the form:

```
<Level key="gfelevel">
  <DatabaseLevel levelName="name" [ levelOneValue="xx" [ levelTwoValue="yy" ] ] />
</Level>
```

You can determine the appropriate D2D level values by running the following SQL query against the postgres database (on dx1f) after the model has been ingested.

Replace the value for datasetID below with the modelname being mapped, ETA218 used as an example:

```
psql -U awips -d metadata -c "SELECT parameter_abbreviation, masterlevel_name, levelonevalue, leveltwovalue FROM grid_info, level WHERE level_id=level.id AND datasetid='ETA218' ORDER BY parameter_abbreviation"
```

- In the decoded D2D data, levels are represented by a level name, and one or two optional numeric values.
- In GFE, levels are represented by a string name.
- The mapping of the GFE string names to D2D levels is contained in gfeLevelMappingFile.xml.
- The levelOneValue and levelTwoValue attributes are optional.
- The GFE level name string can be whatever you desire but is typically of the form:

  `levelName[ levelOneValue [ levelTwoValue ] ]`

Note: These mappings are not tied to a particular dataset so they affect all parameters in all datasets.

- If you are adding a new model you can skip this step and the GFE parameter name will be the same as the name of the D2D parameter.
- If you want to use a different name to match existing smartInits/Tools/Procedures you can map your desired GFE parameter name to the D2D parameter name by adding an entry to the gfeParamName.xml file if one does not already exist.
V.5 Add Your New parameterInfo File to gfeParamInfo.xml

This file associates the parameterInfo XML file with the D2D datasetID.

Entries in this file are of the form:

```
<alias
    base="D2DdatasetID">parameterInfoFileName</alias>
```

- Copy the file from base to site and make changes to the site-level file. The base file is located at `common_static/base/grid/dataset/alias/gfeParamInfo.xml`.
- If a <fcst> tag does not exist for a given forecast time in the gridded dataset’s parameterInfo file, missing time steps may be noticed in the grid manager for that particular model.

If all time steps are not displaying in the grid manager for a particular parameter (when compared to the time steps that are available in D2D for that same parameter), check the following to determine if there are missing <fcst> tags in the parameterInfo file.

On an EDEX Processing Server (e.g., DX3, DX4, DX5, DX6):

```
grep 'No time range found for' /awips2/edex/logs/edex-ingest-smartInit-`date +%Y%m%d`.log | awk '{print $13}' | cut -d_ -f5 | uniq
```

Any dataset returned references the GFE model name mapped in the D2DMODELS array in the localConfig.py for the active GFE domain.

Assuming that the above grep command returned `{MODELNAME}` you can then determine the <fcst> hours that are missing via the following:

```
grep 'No time range found for' /awips2/edex/logs/edex-ingest-smartInit-`date +%Y%m%d`.log | grep ${MODELNAME} | awk '{print $17}' | sort -n | uniq
```

Then add these values inside the <fcst> </fcst> XML tag in the appropriate parameterInfo file.

V.6 Add Any Accumulative Parameters to D2DAccumulativeElements

These would be defined in your localConfig.py file.

- This setting indicates to GFE that the associated D2D parameter is an accumulative element and will change how GFE determines the time range associated with the forecast hours.
- To add parameters for a new model the entry should have the following form:
serverConfig.D2DAccumulativeElements['GFEmodelname']=['parm1','parm2',...]

- To add more parameters for an existing model, the entry should have the following form:

  serverConfig.D2DAccumulativeElements['GFEmodelname']=['parm1','parm2',...]

\[V.7\] Add the New Model to D2DMODELS

These would be defined in your localConfig.py file.

This setting associates the D2D datasetID with the desired GFE model name.

serverConfig.D2DMODELS.append(('D2DdatasetID', 'GFEmodelname'))

Note: For reference, the GFE configuration file locations mentioned in this document are as follows:

edex_static/site/CCC/grid/parameterInfo/xxxxx.xml
edex_static/site/CCC/grid/gfeLevelMappingFile.xml
common_static/site/CCC/parameter/alias/gfeParamName.xml
common_static/site/CCC/grid/dataset/alias/gfeParamInfo.xml
edex_static/site/CCC/config/gfe/localConfig.py

All files should be copied from their BASE location to the SITE location before editing.
Appendix W
WarnGen Urban Boundaries
Appendix W. WarnGen Urban Boundaries

Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>W.1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>W.2 OB13.5.2 WarnGen Template Activation</td>
<td>2</td>
</tr>
<tr>
<td>W.3 Shape File Installation</td>
<td>3</td>
</tr>
<tr>
<td>W.3.1 Site-Level Shape File Installation</td>
<td>4</td>
</tr>
<tr>
<td>W.3.2 Base-Level Shape File Installation</td>
<td>5</td>
</tr>
<tr>
<td>W.3.3 Verify the WarnGenLoc Installation</td>
<td>6</td>
</tr>
<tr>
<td>W.4 Urban Boundaries Specific Template Information</td>
<td>7</td>
</tr>
<tr>
<td>W.4.1 General Background</td>
<td>7</td>
</tr>
<tr>
<td>W.4.2 WarnGenLoc Shape File Fields</td>
<td>8</td>
</tr>
<tr>
<td>W.4.3 Template Configuration Fields</td>
<td>8</td>
</tr>
<tr>
<td>W.4.4 Filters and Constraints</td>
<td>9</td>
</tr>
<tr>
<td>W.4.5 Third Bullet Example</td>
<td>10</td>
</tr>
<tr>
<td>W.4.6 Fourth Bullet Example</td>
<td>12</td>
</tr>
<tr>
<td>W.5 Template Customization, Configuration and Testing</td>
<td>14</td>
</tr>
</tbody>
</table>

List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table W.4.2-1. WarnGenLoc Shape File Fields</td>
<td>8</td>
</tr>
<tr>
<td>Table W.4.3-1. Template Configuration Fields</td>
<td>9</td>
</tr>
<tr>
<td>Table W.4.3-2. Template Configuration Fields</td>
<td>9</td>
</tr>
<tr>
<td>Table W.4.4-1. Constraint Values Use to Control Filters</td>
<td>10</td>
</tr>
<tr>
<td>Table W.4.4-2. Landwater Field Values Used to Control Locations in Land and Marine Products</td>
<td>10</td>
</tr>
</tbody>
</table>


W.1 Introduction

**Note:** Release OB14.1.1 delivers no WarnGen template changes. If the WarnGen templates delivered with OB13.5.2 were implemented, no action is needed to configure WarnGen when OB14.1.1 is installed.

**WARNING:** The WarnGen templates delivered with OB13.5.2 must be activated in order to insert Emergency headlines and text into Tornado and Flash Flood Warnings. When the OB13.5.2 templates are activated, two new template configuration files must be used – config.vm and geospatialConfig*xml. However, for WarnGen to work after the install, a full set of the previous WarnGen templates must exist at the site or user level. See Section W.2 for details.

The Urban Boundaries functionality was added to WarnGen in OB13.4.1. This function allows more accurate and detailed city information in the WarnGen third and fourth bullets. Release OB13.4.1 delivered a large number WarnGen template changes to implement the Urban Boundaries functionality and correct various other errors.

Each site needs the following three pieces of software to implement the WarnGen Urban Boundaries functionality:

1. WarnGen source code changes (delivered in OB13.4.1 and OB13.5.2).
2. Installation of the WarnGenLoc shape file (the shape file is not delivered with the AWIPS software release; sites can complete this step at any time).
3. Activation of the new WarnGen templates (after the WarnGenLoc shape file is installed).

Generally, these three pieces are all independent of one another. The only exception is that activating the 13.4.1 (and 13.5.2) WarnGen templates requires that the WarnGenLoc shape file be installed in Postgres. The 13.4.1 (and 13.5.2) WarnGen source code changes are backward compatible, and the 13.3 templates will continue to work after the 13.5.2 install. Therefore, sites may implement the 13.5.2 WarnGen templates at their convenience. Installing the WarnGenLoc shape file does not affect anything in AWIPS II. WarnGen is the only application that uses the WarnGenLoc shape file. The WarnGenLoc shapefile is not used by the OB13.3 WarnGen templates.

**WARNING:**
The OB13.4 and later WarnGen templates are not compatible with previous templates and require that the WarnGenLoc shape file be installed in order to function.

**Note:** This appendix describes the configuration in the AWIPS II clustered (operational) environment. Some modifications will be needed on the ADAM system. An updated version of this document is maintained on the WarnGen Template wiki at the following location: https://collaborate.nws.noaa.gov/trac/siteconfig/wiki/WarnGenUrbanBoundsInfo
W.2  **OB13.5.2 WarnGen Template Activation**

Release OB13.5.2 delivers the following major template changes:

1. Auto insertion of Emergency headline, third bullet text and CTA to Tornado and Flash Flood Warning products (DR 15669).

2. A new configuration file (config.vm), which easily sets various template global configuration fields.

3. Three new configuration files, which easily set template global geographic configuration fields. The three files are geospatialConfig_COUNTY.xml, geospatialConfig_ZONE.xml, and geospatialConfig_MARINE.xml.

The OB13.5.2 templates are not compatible with previous WarnGen templates, and the WarnGenLoc shape file must be installed.

When OB13.5.2 is installed, all of the previous WarnGen base-level templates will be overwritten. If there are any local WarnGen template changes that need to be preserved, the changes must exist at the site or user level.

| **Note:** | Never modify WarnGen baseline templates. To add customizations to the new OB13.5.2 templates, follow the steps in Section W.5. |

If there is a need to preserve custom template changes after the OB13.5.2 install, a full set of the current templates must exist at the site or user level. Before the OB13.5.2 install, verify that a full set of WarnGen template files exists at the site or user level. Generally this includes all files with the *.vm and *.xml file extensions. If there is a site or user version of the config.xml file (main WarnGen configuration file), do not copy the baseline version of this file to the site or user level. The file VM_global_library.vm contains macros used by all templates and must exist at the site or user level. All the ancillary files that are used (such as forecasterName.vm, mileMarkers.vm and mileMarkers.xml) must exist at the site or user level.

After OB13.5.2 is installed, the previous site-level templates will still be active. The new OB13.5.2 templates can be activated when convenient. To activate the OB13.5.2 templates, complete the following steps:

1. Make a backup of the full set of legacy site-level templates (plus ancillary template files).

2. Copy the full set of OB13.5.2 base-level templates (and ancillary files except config.xml) to the user level.

3. Add any site-specific template changes to the OB13.5.2 user-level templates. The OB13.5.2 templates will use the new config.vm configuration file and geospatialConfig*xml files. Make any needed changes to these configuration files.

4. Test the user level OB13.5.2 templates using practice mode.
5. After verifying that the user-level templates work correctly, copy the full set of user-level OB13.5.2 templates and ancillary files to the site level. Do not copy the file config.xml.

W.3 Shape File Installation

The new WarnGenLoc shape file defines the location information for WarnGen. The shape file set is available on the NOAA1 server in the pub/maps/currentBaseline directory with file names wgddmmyy.*. At the time of this writing (October 25, 2013), the latest version of the shape file set has the following file names:

- wg03se13.dbf
- wg03se13.shp
- wg03se13.shx

The 03se13 version of the shape file has only been quality controlled for the eight AWIPS II beta sites. The data for the other WFOs is derived from AWIPS I and will be good enough for training, testing, and demo purposes. As each WFO implements AWIPS II, it will update the local geography and provide updates for the national version of the shape file. New versions of the shape file will be released periodically as more WFOs come on line with AWIPS II.

**Note:** The above files are very large and may take a long time to download.

When installing the clipped (or local) version of WarnGenLoc to do Quality Control work on AWIPS (or ADAM), the shape file should be installed at the site level. After the local cities have been corrected and incorporated into the national version of WarnGenLoc, the shape file should be installed at the AWIPS base level.

The WarnGenLoc base-level map bundle is provided in OB13.5.2 (DR 16535). Therefore, the CAVE base-level WarnGenLoc configuration will not be possible until OB13.5.2 is installed. Before that time, it will be only be possible to configure CAVE to display WarnGenLoc at the site level. If CAVE is configured to display WarnGenLoc at the site level, extra steps are required and it will be necessary later to remove the site configuration in order to use the base-level configuration.

The latest version of the instructions is available at the WarnGen template wiki at the following location:

https://collaborate.nws.noaa.gov/trac/siteconfig/wiki/WarnGenUrbanBoundsInfo

If installing WarnGenLoc at the site level, follow the instructions in Section W.3.1. If installing WarnGenLoc at the base level, follow the instructions in Section W.3.2. Upon completing an installation at either the site level or the base level, follow the instructions in Section W.3.3 to verify successful completion of the installation.
W.3.1 Site-Level Shape File Installation

**STEP 1: Create the WarnGenLoc Map Bundle and Staging Directory**

Create the site level map bundle directory on dx3. (The owner:group should be awips:fxalpha, permissions 775.)

```
dx3:/awips2/edex/data/utility/cave_static/site/LLL/bundles/maps
```

Download the map bundle tar file from the WarnGen template wiki at the following location:


Use the following command to extract the bundle file:

```
tar xvf warnngenloc_mapbundle.tar
```

Place the warnngenloc.xml bundle file in the maps directory created above.

Create the following site level staging directory on dx3 (owner:group should be awips:fxalpha, permissions 775):

```
dx3:/awips2/edex/data/utility/edex_static/site/LLL/shapefiles/WarnGenLoc
```

Make sure that the proper WarnGenLoc mixed case is used.

**STEP 2: Copy the WarnGenLoc Shape file**

- Copy all three shape files (e.g., wgddmmyy13.*) into the staging directory created in Step 1 (edex_static/site/LLL/shapefiles/WarnGenLoc/)
- On ADAM (as root), scp the shape files from dx1 to adam1:/data/fxa/nationalData, because dx1 is not mounted on ADAM.

| Note: The downloaded shape file name (e.g. wgddmmyy.*) MUST be maintained. |

**STEP 3: Install the WarnGenLoc Shape Files**

SSH to the dx1f server as the root user then enter the following commands:

```
cd /data/fxa/sdc
./config_awips2.sh shp LLL
```

where LLL is the site identifier for your AWIPS system.

On ADAM, the script is normally located in /var/tmp/sdc

The config_awips2.sh script may prompt for each new shape file found. If shape files other than WarnGenLoc are found, enter "N" until prompted for WarnGenLoc. The script will take a few minutes and return with "Complete" notification when done.
The final step is to ensure that the installation was completed successfully. To do this, follow the instructions in Section W.3.3, Verify the WarnGenLoc Installation.

### W.3.2 Base-Level Shape File Installation

Use the following instructions only for Release OB13.5.2 and later:

**STEP 1: Verify the base-level Configuration**

The OB13.5.2 installation created the base level stating directory. Verify that the following base level staging directory exists:

```
dx3:/awips2/edex/data/utility/edex_static/base/shapefiles/WarnGenLoc
```

If the directory does not exist, it will be impossible to install the shape file at the base level.

If WarnGenLoc was previously installed at the site level, the site-level items will override the base-level configuration. The site-level map bundle and staging directory must be removed so they do not override the base-level items. No Postgres change is needed because the same table name and schema are used for both the site level and the base level.

Remove the following site-level map bundle file, if it exists:

```
dx3:/awips2/edex/data/utility/cave_static/site/LLL/bundles/maps/warngenloc.xml
```

Remove the following site-level staging directory and all of its contents if it exists:

```
dx3:/awips2/edex/data/utility/edex_static/site/LLL/shapefiles/WarnGenLoc
```

**STEP 2: Copy the WarnGenLoc Shape Files**

Copy all three of the wgddmmyy.* files into directory `/data/fxa/nationalData`

*Note:* The downloaded shape file name (e.g., wgddmmyy.*) MUST be maintained.

On ADAM (as root), scp the shape files from dx1 to `adam1:/data/fxa/nationalData` because this is not mounted on ADAM.

**STEP 3: Install the WarnGenLoc Shape Files**

SSH to the dx1f server as the root user. Then enter the following commands:

```
cd /data/fxa/sdc
./config_awips2.sh shp LLL
```

where LLL is the site identifier for your AWIPS system.

On ADAM, the script is normally located in `/var/tmp/sdc`
The config_awips2.sh script may prompt for each new shape file found. If shape files other than WarnGenLoc are found, enter “N” until prompted for WarnGenLoc. The script will take a few minutes; it will return with “Complete” notification when done.

The final step is to ensure that the installation was completed successfully. To do this, follow the instructions in Section W.3.3, Verify the WarnGenLoc Installation.

W.3.3 Verify the WarnGenLoc Installation

[Note: The following process applies to both site-level and base-level installations.]

Check Postgres to verify that the shape file was stored by performing the following steps. Note: This cannot be performed as a root:

1. ssh to dx1f as “ncfuser”
2. psql –d maps (enter the password)
3. select * from mapdata.map_version where table_name like 'warngen%';
4. \q (exit psql).

The following fields should display from the mapdata.map_version table:
• table_name: warnngenloc
• filename: WarnGenLoc.shp
• import_time: date and time when config_awips2.sh was run

Some other useful psql commands for viewing the WarnGenLoc fields are the following.

To view the WarnGenLoc field names:

TYPE: \d mapdata.warngenloc

To list some important WarnGenLoc data values (in this example, for WFO LWX):

TYPE: select cwa, name, state, warnngenlev, population, usedirs, supdirs, landwater from mapdata.warngenloc where cwa='LWX' order by state, name asc;

Verify that the shape file can be displayed in CAVE by executing the following steps:

1. Launch CAVE.
2. In the D2D perspective, select MAPS -> WarnGenLoc (or the menu item created above).
3. Zoom to your area of interest.
4. Right click on the map; select "Show Map Legends" to display the map legends.
5. Right click on the WarnGenLoc map label.
6. Select "shade" and choose the field to shade (e.g., name). The polygons should be shaded.

7. Right click on the WarnGenLoc map label.

8. Select label and choose the field to label (e.g. name). The location names should display.

9. The labels can be made easier to read by increasing the magnification and changing the color of the label.

If the WFO needs to make corrections to the urban boundaries shape file, they should follow the instructions from the WarnGen wiki at the following location:

https://collaborate.nws.noaa.gov/trac/siteconfig/wiki/warnGenShapefileQC

W.4 Urban Boundaries Specific Template Information

This section describes the urban boundaries-specific template information needed for the sites.

W.4.1 General Background

This section describes the WarnGen template logic that uses the new WarnGen locations shape file (WarnGenLoc). Many of the items are site configurable. Configurations described here are the baseline template configurations.

The WarnGenLoc items are used in the third and fourth bullets of WarnGen products with a storm track. These are the third bullet current storm location and the fourth bullet pathcast or list of cities. In general, the WarnGen hydrologic products will only use the WarnGenLoc items in the fourth bullet.

The urban boundaries functionality replaces the legacy AWIPS local cities and marine sites functionality. Highway mile markers are treated as simple point locations and are coded in a separate section after the fourth bullet and above the CTA section. Mile marker configuration is a separate topic and is covered on the WarnGen wiki at the following location:

https://collaborate.nws.noaa.gov/trac/siteconfig/wiki/AddingMileMarkers

Another type of WarnGen location data is Point Markers. These are intended for temporary locations such as specific events (e.g., festivals and races). These are NOT included in the WarnGenLoc nationwide urban boundaries shape file. They are added to a different Postgres database table maintained by the site. Point marker configuration is detailed on the WarnGen wiki at the following link:

https://collaborate.nws.noaa.gov/trac/siteconfig/wiki/AddingPointMarkers

Each WarnGenLoc item can be treated as a polygon or as a single point. As a result, portions of the cities can be specified in the third bullet current storm position.
IMPORTANT: The portions of the cities are used in the third bullet ONLY if the WarnGenLoc shape file field "usedirs" is set to "1" (TRUE). Most locations will not use portions; therefore, WarnGen will treat them as simple point locations. For example, to reference "NORTHERN DENVER", the Denver usedirs value must be "1". If the Denver usedirs value is "0", then WarnGen will only refer to "DENVER" without any portions of the city in the third bullet storm location.

W.4.2 WarnGenLoc Shape File Fields

Table W.4.2-1 briefly defines the WarnGenLoc Shape File Fields. They are described in more detail in the WarnGenLoc QC section on the WarnGen wiki at the following link:

https://collaborate.nws.noaa.gov/trac/siteconfig/wiki/warnGenShapefileQC

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gid</td>
<td>System-assigned unique ID number</td>
</tr>
<tr>
<td>name</td>
<td>Location name</td>
</tr>
<tr>
<td>st</td>
<td>State abbreviation</td>
</tr>
<tr>
<td>state</td>
<td>Full state name</td>
</tr>
<tr>
<td>population</td>
<td>2010 Census population, where available</td>
</tr>
<tr>
<td>warngenlev</td>
<td>WarnGen priority levels from AWIPS I</td>
</tr>
<tr>
<td>cwa</td>
<td>CWA 3 letter ID</td>
</tr>
<tr>
<td>goodness</td>
<td>Controls zoom level when location appears on CAVE</td>
</tr>
<tr>
<td>lat</td>
<td>Latitude of location centroid</td>
</tr>
<tr>
<td>lon</td>
<td>Longitude of location centroid</td>
</tr>
<tr>
<td>usedirs</td>
<td>1=use portions of the location (e.g. NORTHERN DENVER), 0=use no portions</td>
</tr>
<tr>
<td>supdirs</td>
<td>Suppress these directions when usedirs=1 (values are NSEW or any combination with no delimiters)</td>
</tr>
<tr>
<td>landwater</td>
<td>codes to distinguish land and water locations</td>
</tr>
<tr>
<td>recnum</td>
<td>system assigned ID number unique for this CWA</td>
</tr>
<tr>
<td>the_geom</td>
<td>binary geometry data</td>
</tr>
</tbody>
</table>

W.4.3 Template Configuration Fields

The configuration items that control the selection of the third and fourth bullet cities are in the template xml file. Each xml section defines Java software objects that contain all the parameters needed to control the behavior of each object. Table W.4.3-1 identifies and defines the template objects related to configuring the urban boundaries functionality.
Table W.4.3-1. Template Configuration Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>areaSource object “areas”</td>
<td>Generates list of counties or zones in the first bullet.</td>
</tr>
<tr>
<td>pointSource object “closestPoints”</td>
<td>Generates the first city listed in the third bullet storm location.</td>
</tr>
<tr>
<td>pointSource object &quot;otherClosestPoints&quot;</td>
<td>Generates the &quot;other&quot; city listed in the third bullet storm location.</td>
</tr>
<tr>
<td>pathcastConfig</td>
<td>Specifies the general pathcast configuration</td>
</tr>
<tr>
<td>pointSource object &quot;cityList&quot;</td>
<td>Generates the fourth bullet locations in the list of cities and the pathcast.</td>
</tr>
<tr>
<td>pointSource object &quot;otherPoints&quot;</td>
<td>Generates the fourth bullet other locations (only used for the pathcast).</td>
</tr>
<tr>
<td>geospatialConfig</td>
<td>Specifies whether the product codes counties, forecast zones or marine zones.</td>
</tr>
</tbody>
</table>

Table W.4.3-2 identifies and briefly describes some of the important WarnGenLoc-related configurations. More detailed descriptions can be found in the WarnGen Velocity Documentation on the WarnGen wiki at the following link:

https://collaborate.nws.noaa.gov/trac/siteconfig/wiki/WarnGen

Scroll to the section titled AWIPS 2 WarnGen Documentation and Localization, and select the link for “Current WarnGen Velocity Documentation.”

Table W.4.3-2. Template Configuration Fields

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>City inclusion</td>
<td>Defines how much of the location must fall within the warning polygon to be included in the warning. If set to zero, locations with boundaries barely touching the warning polygon will be included. These are the fields inclusionPercent, inclusionArea, and inclusionAndOr. They apply to both counties/zones in the first bullet and cities in the third bullet (only larger cities with userdirs = &quot;1&quot;).</td>
</tr>
<tr>
<td>City position relative to the storm</td>
<td>Locations can be chosen according to distance from the storm centroid or distance from the storm track (fields searchMethod, distanceThreshold).</td>
</tr>
<tr>
<td>City filters</td>
<td>Used to select desired locations such as the warngenlev (AWIPS ONE WarnGen priority) and the landwater fields.</td>
</tr>
<tr>
<td>Number of cities</td>
<td>Refers to the number of locations; can be limited using the maxResults field.</td>
</tr>
<tr>
<td>Polygon</td>
<td>withinPolygon field can be set to 'false' to allow reference locations outside the warning polygon.</td>
</tr>
<tr>
<td>Pathcast</td>
<td>Various pathcast aspects can be configured such as the first pathcast entry, time interval, number of locations at each time and total number of time intervals (fields delta, interval, maxCount and maxGroup).</td>
</tr>
<tr>
<td>Watches</td>
<td>IncludeWatchAreaBuffer field sets the area within which nearby watch information will be included.</td>
</tr>
<tr>
<td>Counties vs. zones</td>
<td>geospatialConfig and areaSource can be used to list counties in the first bullet of zone-based products.</td>
</tr>
</tbody>
</table>

W.4.4 Filters and Constraints

Constraint values control the filters. The valid constraints are identified in Table W.4.4-1.
Table W.4.4-1. Constraint Values Use to Control Filters

<table>
<thead>
<tr>
<th>Constraint</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EQUALS</td>
<td>Matches exactly the constraint value (only one value should be supplied)</td>
</tr>
<tr>
<td>NOT_EQUALS</td>
<td>Does not match exactly the constraint value</td>
</tr>
<tr>
<td>GREATER_THAN</td>
<td>Greater than the constraint value</td>
</tr>
<tr>
<td>GREATER_THAN_EQUALS</td>
<td>Greater than or equal to the constraint value</td>
</tr>
<tr>
<td>LESS_THAN_EQUALS</td>
<td>Less than or equal to constraint value</td>
</tr>
<tr>
<td>BETWEEN</td>
<td>Range value set in constraint value (constraint value must be in the format</td>
</tr>
<tr>
<td></td>
<td>of &quot;a--b&quot;)</td>
</tr>
<tr>
<td>IN</td>
<td>True if one of the values in the constraint value. constraint values</td>
</tr>
<tr>
<td></td>
<td>separated by commas (i.e., L,LW,W)</td>
</tr>
<tr>
<td>LIKE</td>
<td>True if one of the values contains a value that contains the constraint value</td>
</tr>
<tr>
<td></td>
<td><em>(BEWARE: THIS MAY AFFECT PERFORMANCE)</em></td>
</tr>
</tbody>
</table>

The landwater field values from the WarnGenLoc shapefile are used to control locations in land and marine products. These values are identified in Table W.4.4-2.

Table W.4.4-2. Landwater Field Values Used to Control Locations in Land and Marine Products

<table>
<thead>
<tr>
<th>Landwater Field Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>L</td>
<td>Land only (default value for AWIPS I LocalCitiesInfo file and cities shapefile)</td>
</tr>
<tr>
<td>W</td>
<td>Marine only, for use in the third bullet (storm location) and fourth bullet (pathcast) of SMW and MWS products (this is the default value for locations from the AWIPS I MarineInfo file).</td>
</tr>
<tr>
<td>C</td>
<td>Used only as reference points in the 3rd bullet of SMW and MWS products</td>
</tr>
<tr>
<td>LW</td>
<td>Combo of &quot;L&quot; and &quot;W&quot; (use land location as 3rd or 4th bullet marine reference)</td>
</tr>
<tr>
<td>LC</td>
<td>Combo of &quot;L&quot; and &quot;C&quot; (use land location as 3rd bullet marine reference point)</td>
</tr>
</tbody>
</table>

The following are the required landwater values for WarnGen products. These are the baseline template coded values:

- **LAND PRODUCTS**: third bullet - L, LW, LC and fourth bullet - L, LW, LC
- **MARINE PRODUCTS**: third bullet - W, C, LW, LC and fourth bullet - W, LW

W.4.5 Third Bullet Example

This section lists the OB13.4 SVR template xml configuration that controls the selection of locations in the third bullet storm location. In OB13.5.2 these items are in file geospatialConfig_COUNTY.xml:

```xml
<!-- pointSource object to generate third bullet first city -->
<pointSource variable="closestPoints">
  <pointField>NAME</pointField>
  <type>AREA</type>
  <searchMethod>POINTS</searchMethod>
  <filter>
    <mapping key="WARNGENLEV">
      <constraint constraintValue="1,2" constraintType="IN" />
    </mapping>
  </filter>
</pointSource>
```
In the preceding third bullet example, the closestPoints object returns warngen level one or two land locations within 100 miles of the current storm location, sorts by distance, warngenlev and population and returns at most one location. This is the first location coded in the third bullet.

The storm location is the WarnGen estimated storm location when “create text” is used. It is not the storm centroid at the time of the latest radar frame.

The distance to the storm location is measured from the city centroid (lat/lon in the WarnGenLoc). If the storm location falls within an urban boundary (i.e. “drag me to storm” is within the city polygon), the effective distance/direction becomes zero and a portion of the city will be used if useDirs is “1.”

When the storm distance is <= 2 miles, the template codes "OVER" the city. When the distance is > 2 miles and <= 6 miles, the template codes "NEAR" the city. When the distance is > 6 miles, the template codes a direction and distance.

The otherClosestPoints object returns warngen level one land locations within 100 miles of the current storm location, sorts by distance and returns at most one location. The closest location is used as the "other" city in the third bullet. This is the closest level one city to the storm location.
In the following third bullet example, closestPoints provided Westminster and otherClosestPoints provided Downtown Denver:

\begin{verbatim}
AT 1206 PM MDT . . . DOPPLER RADAR INDICATED A SEVERE
THUNDERSTORM CAPABLE OF PRODUCING DAMAGING WINDS IN EXCESS
OF 60 MPH. THIS STORM WAS LOCATED NEAR SOUTHERN WESTMINSTER
. . . OR 8 MILES NORTHWEST OF DOWNTOWN DENVER . . . AND
MOVING NORTHEAST AT 5 MPH.
\end{verbatim}

The following city portions are coded (if WarnGenLoc usedirs = 1): NORTHERN, NORTHEASTERN, EASTERN, SOUTHEASTERN, SOUTHERN, SOUTHWESTERN, WESTERN and NORTHWESTERN. Also, the following portions containing CENTRAL are coded: NORTH CENTRAL, EAST CENTRAL, SOUTH CENTRAL, and WEST CENTRAL.

For asymmetrically shaped locations, directional portions can be excluded. This is set in the WarnGenLoc supdirs field. The values are "NSEW" or any combination with no delimiters. For example, "NS" would produce no NORTHERN or SOUTHERN portions for this city.

**W.4.6 Fourth Bullet Example**

The fourth bullet can contain a list of cities or pathcast. This section lists the OB13.4 SVR template xml configuration (file geospatialConfig_COUNTY.xml) that controls the selection of locations in the fourth bullet:

\begin{verbatim}
<!-- pathcast configuration -->
<pathcastConfig>
  <inclusionPercent>1</inclusionPercent>
  <withinPolygon>true</withinPolygon>
  <distanceThreshold>8.0</distanceThreshold>
  <interval>5</interval>
  <delta>5</delta>
  <maxResults>10</maxResults>
  <maxGroup>8</maxGroup>
  <pointField>Name</pointField>
  <type>AREA</type>
  <areaField>COUNTYNAME</areaField>
  <!-- <areaField>NAME</areaField> -->
  <parentAreaField>STATE</parentAreaField>
  <areaNotationField>STATE</areaNotationField>
  <areaNotationTranslationFile>countyTypes.txt</areaNotationTranslationFile>
  <sortBy>
    <sort>warngenlev</sort>
    <sort>population</sort>
    <sort>distance</sort>
  </sortBy>
  <filter>
    <mapping key="WARNGENLEV">
      <constraint constraintValue="1,2" constraintType="IN" />
    </mapping>
    <mapping key="LANDWATER">
      <constraint constraintValue="L,LW,LC" constraintType="IN" />
    </mapping>
  </filter>
</pathcastConfig>
\end{verbatim}
The pathcast configuration above specifies that the pathcast begins 5 minutes after the current storm location (delta tag). Locations will be listed at five minute intervals (interval tag). There will be a maximum of eight pathcast intervals (maxGroup tag). Each interval will have a maximum of ten locations (maxResults tag). If no locations are found for a particular time period, then that interval is skipped in the pathcast.

The pathcast contains only level 1 and 2 cities (warngenlev tag) within 8 miles of the storm track (distanceThreshold tag) and within the warning polygon (withinPolygon tag). Only land locations are selected (landwater tag). The location is included if at least one percent of the location polygon is inside the warning polygon (inclusionPercent tag). The inclusion threshold applies only for locations with usedirs set to 1 in the WarnGenLoc shape file. They are sorted by warngenlev, population, followed by the distance from the storm track. Many cities do not contain population and the NWS is researching this issue.

The list of cities contains a maximum of 20 level one, two, three and four locations within the polygon and within 10 miles of the storm track. The location is included if at least one percent of the location polygon is inside the warning polygon. They are sorted by warngenlev, population, and then distance from the storm track.
When pathcast is used, “OTHER LOCATIONS IMPACTED” is also coded after the pathcast. The “otherPoints” object contains these locations. These are a maximum of 10 level 3 and 4 locations, ten miles or less from the storm track and within the warning polygon, sorted by distance from the storm track.

The third and fourth bullet logic is similar for the TOR, SVR, SPS, NOW, EWW and SMW products. The hydrologic products generally do not track specific storm cells and do not contain specific storm location and movement information. In general, the hydro products only use the WarnGen locations shape file in the fourth bullet list of affected locations.

W.5 Template Customization, Configuration and Testing

The OB13.4.1 and OB13.5.2 WarnGen templates contain changes to several thousand lines of code – achieving numerous error fixes, enhancements, and syntax clean-up as well as the addition of the urban boundaries functionality and automated Tornado/Flash Flood Warning Emergency headlines. Unfortunately, it was necessary to bundle all the changes together. That means that the urban boundaries changes and other changes cannot be separated. The best strategy for customizing the templates is to begin with the baseline OB13.5.2 templates and add the site customizations at the AWIPS user level.

The baseline templates were developed by NWS operational forecasters over the past two years. They have tried to implement as many features as possible that are typically used at field sites. The need for local customizations should be minimal.

The OB13.5.2 templates add Tornado and Flash Flood Emergency logic. This involves adding an emergency headline, third bullet emergency statement and emergency CTA to the following templates: Tornado and SVS follow-up statement, Flash Flood Warning (both convective and non convective) and associated FFS follow-up statements. Also, the use of impact based warning products may become more widespread, requiring more baseline template changes.

For the above reasons, try to keep the number of local customizations to a minimum. Otherwise, the task of maintaining local template logic changes will become burdensome when baseline template changes are delivered. It is anticipated that after the TOR/FFW
Emergency and impact based template changes are completed, the baseline templates will stabilize. At that time, it would be more prudent to make local customizations.

There may be some local template configuration changes needed in the template xml files. These most likely configuration changes are summarized in Section W.4.

Template customization should be done using the CAVE localization perspective at the user level.

**Note:** Never make any changes to templates at the AWIPS base level (dx3:/awips2/edex/data/utility/common_static/base/warngen). Future AWIPS software installs will overwrite these files.

The site-level file config.xml contains the WarnGen site configuration information and should be preserved at the site level. Don't copy the baseline version of config.xml to the site level.

Template testing should be done in workstation PRACTICE mode. Be sure to test a full warning lifecycle for each product. For example, complete the following for every WarnGen product issued at the office. Verify that all geographic processing and template formatting work correctly:

1. Issue a new warning for several counties.
2. Issue a correction.
3. For hydrologic products, issue an extension in time.
4. Issue a follow-up statement partial cancellation (cancel one or more counties and continue the warning in other counties.)
5. Issue a follow-up statement continuation.
6. Issue a follow-up statement cancellation.
7. Issue follow-up statement expiration.

For your reference, the WarnGen Template wiki contains a great variety of information and can be found at:

[https://collaborate.nws.noaa.gov/trac/siteconfig/wiki/WarnGen](https://collaborate.nws.noaa.gov/trac/siteconfig/wiki/WarnGen)
Appendix X
Configuring metarToShefFilter.xml
Appendix X. Configuring metarToShefFilter.xml

Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>X.1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>X.2 XML Elements</td>
<td>1</td>
</tr>
</tbody>
</table>

List of Exhibits

Exhibit X.2-1. Sample metarToShefFilter.xml use <metarToShefFilter> ........................................ 4
Exhibit X.2-2. Sample metarToShefFilter.xml use <pluginDataObjectFilter> .......................... 5
Exhibit X.2-3. Sample METAR file ............................................................................. 5
Appendix X Configuring metarToShefFilter.xml

X.1 Introduction

EDEX uses the file **metarToShefFilter.xml** to determine if METAR stations’ data should be SHEF-encoded. As a METAR product is processed within EDEX, each station is checked against all filters provided in each element in this file to determine SHEF-encoding eligibility. Once a station becomes eligible, its contents are encoded in SHEF according to switches and configuration settings.

The file normally is located at
/awsips2/edex/data/data/utility/edex_static/site/XXX/plugin-filters at workstation dx3/4, where XXX is the uppercase 3-character site ID. Two root elements are available for metarToShefFilter.xml: <metarToShefFilter> and <pluginDataObjectFilter>. An XML file can only have one root element. [For more information about XML syntax, see http://www.w3schools.com/xml/xml_syntax.asp.] Any violation of XML syntax will result in an error thrown upon EDEX restart. Element <pluginDataObjectFilter> is designed for a single METAR configuration file with one set of METAR to SHEF options to be applied to all METAR stations. Element <metarToShefFilter> is designed for one or more METAR configuration files with a different set of METAR-to-SHEF options.

Once a station is deemed eligible for SHEF encoding by a filter element, filter element processing for that station stops. General information pertaining to the METAR-to-SHEF process follows.

- The METAR-to-SHEF process is controlled by a combination of switches and a configuration file’s contents. Additional notes on how switches are provided are explained in this appendix.

- The configuration files are to be stored in
/awsips2/edex/data/data/utility/common_static/site/XXX/shef. They should be edited using the CAVE Localization perspective (Hydro -> SHEF).

- The METAR-to-SHEF document referenced in this appendix can be found here: http://www.nws.noaa.gov/om/water/WFO_support/Metar2Shef/metar_ver6.9.doc or in documents/metar_to_shef/metar_ver6.9.doc.txt on dx3/4 in
/awsips2/edex/lib/plugins/com.raytheon.uf.common.dataplugin.shef.jar

X.2 XML Elements

- **<metarToShefFilter>:** Root element. It should have one or more <metarToShefRun> elements.

  *Example: See the sample xml file shown in Exhibit X.2-1, at the end of this section. This file has two <metarToShefRun> elements.*

- **<pluginDataObjectFilter>:** Root element. It should have one or more <filterElements> element, and one <filterName> element. It always uses the configuration file metar.cfg
and switches set in the token “metar2shef_options” to apply to the METAR to SHEF conversion.

Example: See the sample xml file shown in Exhibit X.2-2, at the end of this section.

<metarToShefRun> : It should have one <configFileName> element, one <metarToShefOptions> element, one or more (any type combination)<filterElements> element, and one <filterName> element.

<configFileName>: It should have a METAR-to-SHEF configuration file name, located in /awips2/edex/data/utility/common_static/site/XXX/shef directory. The default is metar.cfg if the specified file not found or none is given.

Example: <configFilename>metar_asos.cfg</configFilename>

<metarToShefOptions>: It should have the options for a METAR-to-SHEF conversion. [Refer to the general information provided in section X.1 for documentation location.]

Example: <metarToShefOptions>-a –b –p1 -v</metarToShefOptions>

<filterElements>: There are four types of <filterElements> elements – wmoHeaderFilterElement, stationIdFilterElement, radiusFilterElement, and rectFilterElement. The type of element should be specified in the open tag. All filterElements in an element must be matched in order for a station to be processed.

Example: <filterElements xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="wmoHeaderFilterElement">

WmoHeaderFilterElement: This type of <filterElement> element should have one <filterElementName> element, one <filterType> element, and one or more <patterns> elements.

stationIdFilterElement: This type of <filterElement> element should have one <filterElementName> element, one <filterType> element, and one or more <patterns> elements.

RadiusFilterElement: This type of <filterElement> element should have one <filterElementName> element, one <filterType> element, one <radius> element, one <pointLat> element, and one <pointLon> element.

RectFilterElement: This type of <filterElement> element should have one <filterElementName> element, one <filterType> element, one <upperLeftLat> element, one <upperLeftLon> element, one <lowerRightLat> element, and one <lowerRightLon> element.

<filterElementName>: Name for this filter. Can be any name.

Example: <filterElementName>Filter by station ID</filterElementName>

<filterName>: Name for a METAR-to-SHEF run. Can be any name.

Example: <filterName>Synop METAR to SHEF</filterName>

<filterType>: INCLUDE or EXCLUDE only.
Example: `<filterType>INCLUDE</filterType>`

`<patterns>`: Pattern to filter a METAR product.

- In a wmoHeaderFilterElement type `<filterElement>` filter, the pattern is matched against the WMO header of a METAR product. (See the red boxed text in Exhibit X.2-3, at the end of this section.)

- In a stationIdFilterElement type `<filterElement>` filter, the pattern is matched against the station IDs contained in the METAR product. (See the green boxed text in Exhibit X.2-3.)

- One or more sets of `<patterns>` tags are allowed.

For more information on how a pattern works see, “Summary of regular-expression constructs” at http://docs.oracle.com/javase/7/docs/api/javax/xml/soap/Node.html

Example: `<patterns>KHUT</patterns>`

`<radius>`: Radius in nautical miles.

Example: `<radius>1.2</radius>`

`<pointLat>`: Center Latitude point in decimal degrees.

Example: `<pointLat>40.0</pointLat>`

`<pointLon>`: Center Longitude point in decimal degrees.

Example: `<pointLon>-94.90</pointLon>`

`<upperLeftLat>`: Upper left latitude in decimal degrees.

Example: `<upperLeftLat>44.0</upperLeftLat>`

`<upperLeftLon>`: Upper left longitude in decimal degrees.

Example: `<upperLeftLon>-100.0</upperLeftLon>`

`<lowerRightLat>`: Lower right latitude in decimal degrees.

Example: `<lowerRightLat>40.0</lowerRightLat>`

`<lowerRightLon>`: Lower right latitude in decimal degrees.

Example: `<lowerRightLon>-94.90</lowerRightLon>`

Notes:

- Refer to the general information in section X.1 of this appendix for the location of METAR-to-SHEF switch and config file documentation.

- AWIPS II does not use the first three lines in metar.cfg, but the three lines must be kept in there.

- METAR data is auto fed to the METAR-to-SHEF process.
Log information on the METAR-to-SHEF process can be found in 
/awips2/edex/logs/edex-ingest-yyyyymmdd.log.

Output is auto fed to the SHEF decoder. No SHEF output files are created by 
default by the METAR-to-SHEF process. To see what SHEF is created, use the 
"-v" switch, which will display SHEF in the edex-ingest-yyyyymmdd.log file.

Two tokens are available to set SHEF output files aside – “archive_enable” set to 
ON, and “archive_shefdata_dir” set to a directory location visible on dx3/4. Note 
that at RFCs archive_shefdata_dir typically is set to the directory monitored by 
the RAX shefdecode program for RAX DB storage.

AWIPS II does not use token metar_config_dir to locate file metar.cfg.

If any changes made to metarToShefFilter.xml, any metar.cfg file, or any metar-
to-shef token, EDEX must be restarted to pick up the revisions.

Exhibit X.2-1. Sample metarToShefFilter.xml use <metarToShefFilter>
Appendix X

Configuring metarToShefFilter.xml

Exhibit X.2-2. Sample metarToShefFilter.xml use <pluginDataObjectFilter>

Exhibit X.2-3. Sample METAR file

AWP.MAN.SMM.A2  X-5

– Hard copies uncontrolled. Verify effective date prior to use. –
Appendix Y
What’s New in SMM OB14.1.1
This edition of the AWIPS System Manager’s Manual includes a number of additions and changes. The key changes, additions, deletions, enhancements, and corrections found in the manual are described in this appendix.

The purpose of this appendix is not to provide a detailed and comprehensive listing of every change made to this new release (such as corrections of typographical or grammatical errors). Rather, it describes those changes that have been made in an effort to accurately reflect the current state of AWIPS II.

The AWIPS Documentation Team would like to express its appreciation to SMM users who have contacted us with questions or suggestions. Without their input, we might not have been able to identify and change some of the items listed here. If you spot a potential error in this edition, please contact us at nws.hq.awips.smm.um@noaa.gov.

Chapter 2, System Architecture

In Section 2.2.1.1, Satellite Broadcast Network, the examples of CPSBN mount points for the WFOs and RFCs have been updated.

In Section 2.2.1.4.1.1, File Systems on the Linux Data Server (DX), the examples of mount points for dx1/2, dx3/4 at the WFOs and the RFCs have been updated.

In Section 2.2.1.4.2, AX File Systems, the examples of mount points for ax at the WFOs and RFCs have been updated.

In Section 2.2.1.5.1, LDAD Server File Systems, the examples of mount points for ls at the WFOs and RFCs have been updated.

In Section 2.2.1.6.2, Linux Workstation File Systems, the examples for the 64-bit WFO workstations have been updated.

In Section 2.2.1.7.2, PX File Systems, the examples of mount points for px at WFOs and RFCs have been updated.

Table 2.5.3.1-1, COTS Software and Freeware Used in AWIPS II, has been updated to reflect new additions and version changes.

Section 2.15.2, AWIPS II Raw Data Storage Purge Architecture, has been updated to reflect the fact that the AWIPS II Raw Data Storage is purged by EDEX based on entries in RAW_DATA.xml.

Chapter 4, Data Flow Overview

In Section 4.7, General Monitoring of the EDEX System, the events captured from the EDEX file have been updated.
Chapter 17, AWIPS System Monitor

OB14.1.1 simplifies the process for accessing the AWIPS System Monitor, i.e., simply select Firefox Web Browser from the AWIPS Start menu. Section 17.1, System Monitor Overview, has been updated to reflect the new instructions.

Chapter 25, AWIPS II/EDEX Administration Guide

In Section 25.12.1, AWIPS II/EDEX Software Dependencies, Exhibit 25.12.1-1 has been updated to provide more accurate information about the LDM.

In Section 25.13-1, Troubleshooting Satellite Data Flow, Exhibit 25.13-2 has been updated to provide more accurate information about the LDM.

Examples have been updated throughout this chapter with recent results from the testbed, which reflect changes identified in Chapter 2.

Appendix A, Acronyms

RHEL and GNOME were added.

Appendix C, Directories for /awips2/edex/data and /data/fxa

Several entries under these directories have been updated to reflect the revised format of the data files.

Appendix E, Mass Storage Design

In Sections E.1, LS Drive Allocations, E.2, Linux LX-Drive Allocations, E.3, Linux XT Drive Allocations, E.4, PX Drive Allocations, and E., Linux DX drive Allocations, the volume sizes have been updated.

Appendix I, Administrative Tips for Monitoring System Performance

Several examples used in this appendix have been updated with recent results from the testbed.