what is netcdf?

And what are its plans for world domination?
NetCDF is a...

Software library

File format

- Store data model objects
- Persistence layer
- NetCDF-3, netCDF-4

API

An API is the interface to the Data Model for a specific programming language

Abstract Data Model

An Abstract Data Model describes data objects and what methods you can use on them
NetCDF is a...

- Stores scientific data
- Persistence layer
- NetCDF-3, netCDF-4

- Portable Format: *Machine, OS, application independent*
- Random Access: *fast subsetting*
- Simple: *self-describing, user accessible“flat file”*
- Documented: *NASA ESE standard, 1 page BNF grammar (netcdf-3)*
NetCDF-3 file format

- **Header**
- **Non-record Variable**
  - Variable 1
  - Variable 2
  - Variable 3 ...
- **Record Variables**
  - Record 0
  - Record 1
  - unlimited...

Variables:
- float var1(z, y, x)
- float rvar2(0, z, y, x)
- float rvar3(0, z, y, x)
- float rvar1(1, z, y, x)
- float rvar2(1, z, y, x)
- float rvar3(1, z, y, x)

Row-major order
NetCDF-4 file format

• Built on HDF-5
• Much more complicated than netCDF-3
• Storage efficiency
  – Compression
  – Chunking (can optimize for common I/O pattern)
  – Multiple unlimited dimensions
  – Variable length data
Row vs Column storage

• Traditional RDBMS is a row store
  – All fields for one row in a table are stored together
• Netcdf-3 is a column store
  – All data for one variable is stored together
• Netcdf-4 allows both row and column store
  – Row: compound type
  – Column: regular variable
• Recent commercial RDBMS with column oriented storage, better performance in some cases
• NetCDF-3 record variables are like a compound type
NetCDF is a... 

- Reference library in C
  - Fortran, C++, Perl, Python, Matlab, ...
- Independent implementation in Java
- others?

- Open Source
- Active community
- Supported

- No reference library, no user group for GRIB, BUFR
  - fragmented, not interoperable, difficult to use

Software library
NetCDF is a...

The Application Programming Interface (API) is the interface to the Data Model for a specific programming language.

- Clean separation of concerns
- Information hiding – user never does a seek()
- Stable, backwards compatible
- Easily understood – no surprises
- Interface has small “surface area”
NetCDF is a...

An **Abstract Data Model** describes data objects and what methods you can use on them.
NetCDF-3 data model

- Multidimensional arrays of primitive values
  - byte, char, short, int, float, double
- Key/value attributes
- Shared dimensions
- Fortran77
NetCDF-4 Data Model

A file has a top-level unnamed group. Each group may contain one or more named subgroups, user-defined types, variables, dimensions, and attributes. Variables also have attributes. Variables may share dimensions, indicating a common grid. One or more dimensions may be of unlimited length.
NetCDF, HDF5, OPeNDAP Data Models

- NetCDF (classic)
- NetCDF (extended)
- OPeNDAP
- Shared dimensions

HDF5
NetCDF-Java Library
(aka) Common Data Model
Status Update
C Library Architecture

Application

API

Dispatch

NetCDF-3

NetCDF-4

HDF5

HDF4

OPeNDAP

...
CDM Architecture

Scientific Feature Types → Datatype Adapter → NetcdfDataset → CoordSystem Builder

OpEnDAP
THREDDS
Catalog.xml

Remote Datasets

NcML

NetcdfFile

I/O service provider

NetCDF-3 → NIDS
NetCDF-4 → GRIB
HDF4 → GINI
Nexrad → DMSP

Local Files
<table>
<thead>
<tr>
<th>Id</th>
<th>Description</th>
<th>Reference URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUFR</td>
<td>WMO Binary Universal Form</td>
<td><a href="http://www.wmo.int/pages/prog/www/WMOCodes/OperationalCodes.html">www.wmo.int/pages/prog/www/WMOCodes/OperationalCodes.html</a></td>
</tr>
<tr>
<td>CINRAD</td>
<td>Chinese Level-II Base Data</td>
<td><a href="http://www.cinrad.com/">www.cinrad.com/</a></td>
</tr>
<tr>
<td>DMSP</td>
<td>Defense Meteorological Satellite Program</td>
<td><a href="http://dmsp.ngdc.noaa.gov/">dmsp.ngdc.noaa.gov/</a></td>
</tr>
<tr>
<td>FYSAT</td>
<td>Chinese FY-2 satellite image data in AWX format</td>
<td><a href="http://satellite.cma.gov.cn/">satellite.cma.gov.cn/</a></td>
</tr>
<tr>
<td>GempakGrid</td>
<td>GEMPAK Gridded Data</td>
<td><a href="http://www.unidata.ucar.edu/software/gempak/">www.unidata.ucar.edu/software/gempak/</a></td>
</tr>
<tr>
<td>GempakSurface</td>
<td>GEMPAK Surface Obs Data</td>
<td><a href="http://www.unidata.ucar.edu/software/gempak/">www.unidata.ucar.edu/software/gempak/</a></td>
</tr>
<tr>
<td>GINI</td>
<td>GOES Ingest and NOAAAPORT Interface</td>
<td><a href="http://weather.unisys.com/wxp/Appendices/Formats/GINI.html">weather.unisys.com/wxp/Appendices/Formats/GINI.html</a></td>
</tr>
<tr>
<td>HDF5</td>
<td>Hierarchical Data Format, version 5</td>
<td><a href="http://www.hdfgroup.org/HDF5/">www.hdfgroup.org/HDF5/</a></td>
</tr>
<tr>
<td>McIDASArea</td>
<td>McIDAS area file</td>
<td><a href="http://www.ssec.wisc.edu/mcidas/doc/misc_doc/area2.html">www.ssec.wisc.edu/mcidas/doc/misc_doc/area2.html</a></td>
</tr>
<tr>
<td>netCDF</td>
<td>NetCDF classic format</td>
<td><a href="http://www.unidata.ucar.edu/software/netcdf/index.html">www.unidata.ucar.edu/software/netcdf/index.html</a></td>
</tr>
<tr>
<td>netCDF-4</td>
<td>NetCDF-4 format on HDF-5</td>
<td><a href="http://www.unidata.ucar.edu/software/netcdf/index.html">www.unidata.ucar.edu/software/netcdf/index.html</a></td>
</tr>
<tr>
<td>NEXRAD-2</td>
<td>NEXRAD Level-II Base Data</td>
<td><a href="http://www.ncdc.noaa.gov/oa/radar/radarresources.html">www.ncdc.noaa.gov/oa/radar/radarresources.html</a></td>
</tr>
<tr>
<td>NEXRAD-3</td>
<td>NEXRAD Level-III Products</td>
<td><a href="http://www.ncdc.noaa.gov/oa/radar/radarresources.html">www.ncdc.noaa.gov/oa/radar/radarresources.html</a></td>
</tr>
<tr>
<td>NMCon29</td>
<td>NMC Office Note 29</td>
<td><a href="http://www.emc.ncep.noaa.gov/mmmb/data_processing/on29.htm">www.emc.ncep.noaa.gov/mmmb/data_processing/on29.htm</a></td>
</tr>
<tr>
<td>SIGMET</td>
<td>SIGMET-IRIS</td>
<td><a href="http://www.signet-metris.ch">www.signet-metris.ch</a></td>
</tr>
<tr>
<td>UAMIV</td>
<td>CAMx UAM-IV formatted files</td>
<td><a href="http://www.camx.com">www.camx.com</a></td>
</tr>
</tbody>
</table>
Coordinate System UML

- **Variable**
  - name
  - shape
  - dataType
  + read()

- **CoordinateSystem**

- **CoordinateTransform**
  - parameters : Attribute

- **ProjectionQt**
  - ucar.unidata.geoloc.Projection

- **VerticalCT**
  - ucar.unidata.geoloc.vertical.VerticalTransform

- **AxisType**
  - Time
  - Lat
  - Lon
  - Height
  - Pressure
  - GeoX
  - GeoY
  - GeoZ
  - RadialAzimuth
  - RadialElevation
  - RadialDistance
  - RunTime
  - Ensemble

- **CoordinateAxis**
  - axisType
  - 0..1
  - 1..*

- **Variable**
  - 0..*
  - 0..*
Conventions

- CF Conventions (preferred)
  - dataVariable:coordinates = “lat lon alt time”;
- COARDS, NCAR-CSM, ATD-Radar, Zebra, GEIF, IRIDL, NUWG, AWIPS, WRF, M3IO, IFPS, ADAS/ARPS, MADIS, Epic, RAF-Nimbus, NSSL National Reflectivity Mosaic, FslWindProfiler, Modis Satellite, Avhrr Satellite, Cosmic, ....
- Write your own CoordSysBuilder Java class
Projections

- albers_conical_equal_area (sphere and ellipse)
- azimuthal_equidistant
- lambert_azimuthal_equal_area
- lambert_conformal_conic (sphere and ellipse)
- lambert_cylindrical_equal_area (sphere and ellipse)
- mcidas_area
- mercator
- METEOSAT 8 (ellipse)
- orthographic
- rotated_pole
- rotated_latlon_grib
- stereographic (including polar) (sphere and ellipse)
- transverse_mercator (sphere and ellipse)
- UTM (ellipse)
- vertical_perspective
Vertical Transforms (CF)

- atmosphere_sigma_coordinate
- atmosphere_hybrid_sigma_pressure_coordinate
- atmosphere_hybrid_height_coordinate
- atmosphere_In_pressure_coordinate
- ocean_s_coordinate
- ocean_sigma_coordinate
- ocean_s_coordinate_g1, ocean_s_coordinate_g2
- existing3DField
NetCDF “Index Space” Data Access:
OPeNDAP URL:
http://motherlode.ucar.edu:8080/thredds/dodsC/NAM_CONUS_80km_20081028_1200.grib1.ascii?
Precipitable_water[5][5:1:30][0:1:77]

“Coordinate Space” Data Access:
NCSS URL:
http://motherlode.ucar.edu:8080/thredds/ncss/grid/
NAM_CONUS_80km_20081028_1200.grib1?
var=Precipitable_water&
time=2008-10-28T12:00:00Z&
north=40&south=22&west=-110&east=-80
Scientific Feature Types

- Classification of earth science data into broad categories.
- Take advantage of the regularities that are found in the data for performance.
- Scale to large, multifile collections.
- Support subsetting in Space and Time.
What’s in a file?

1. Feature Types
2. NetCDF File
3. OS File

Multidimensional Arrays

Bag of Bytes
Gridded Data

• **Grid**: multidimensional grid, separable coordinates

• **Radial**: a connected set of *radials* using polar coordinates collected into *sweeps*

• **Swath**: a two dimensional grid, *track* and *cross-track* coordinates

• **Unstructured Grids**: finite element models, coastal modeling (*under development*)
Point Data

- **point**: a single data point (having no implied coordinate relationship to other points)
- **timeSeries**: a series of data points at the same spatial location with monotonically increasing times
- **trajectory**: a series of data points along a path through space with monotonically increasing times
- **profile**: an ordered set of data points along a vertical line at a fixed horizontal position and fixed time
- **timeSeriesProfile**: a series of profile features at the same horizontal position with monotonically increasing times
- **trajectoryProfile**: a series of profile features located at points ordered along a trajectory
Discrete Sampling Convention

CF 1.6

• Encoding standard for netCDF classic files
  – Challenge: represent ragged arrays efficiently
• Classifies data according to connectedness of time/space coordinates
• Defines netCDF data structures that represent features
• Make it easy / efficient to
  – Store collections of features in one file
  – Read a Feature from a file
  – Subset the collection by space and time
Rectangular Array

A standard two-dimensional array is a rectangle.

Ragged Array

With Variant arrays, you need not waste space.

array2:
NetCDF Markup Language (NcML)

- XML representation of netCDF metadata (like ncdump -h)
- Create new netCDF files (like ncgen)
- Modify (“fix”) existing datasets without rewriting them
- Create virtual datasets as aggregations of multiple existing files.
- Integrated with the TDS
THREDDS Data Server

Servlet Container

 THREDDS Server

- catalog.xml
- NetCDF-Java library
  - NCSS
  - OPeNDAP
  - HTTPServer
  - cdmremote

Remote Access Client

Datasets

- IDD Data
  - motherlode.ucar.edu

configCatalog.xml
Remote Access

- OPeNDAP 2.0
  - index space access
  - Can transport full netCDF extended data model
  - Will replace with DAP 4 next year
- cdmremote
  - Full data model, index space access
- Netcdf Subset Service
  - coordinate space access to gridded data
  - Delivers netCDF files (also csv, xml, maybe JSON)
  - Now writes netcdf-4 (alpha test), with C library / JNI
- cdmrFeature Service
  - coordinate space access to point data
  - Feature type API
ncstream serialization
CDM Architecture

Scientific Feature Types

Datatype Adapter

NetcdfDataset

CoordSystem Builder

Application

NetCDF-Java/
CDM architecture

NetcdfFile

I/O service provider

NetCDF-3

NIDS

NetCDF-4

GRIB

HDF4

GINI

Nexrad

DMSP

Remote Datasets

Local Files

OPeNDAP

THREDDS

Catalog.xml

cdmremote

NcML

Remote Datasets

Local Files
CFSR timeseries data at NCDC

- Climate Forecast System Reanalysis
- 1979 - 2009 (31 years, 372 months)
- Total 5.6 Tbytes, 56K files
- Grib2 data
GRIB collection indexing

GRIB file

Index file name.gbx9

1000x smaller

Create

Collection Index
collectionName.ncx

1000x smaller
CDM metadata

TDS
What have we got?

• Fast indexing allows you to find the subsets that you want in under a second
  – Time partitioning should scale up as long as your data is time partitioned
• No pixie dust: still have to read the data!
• GRIB2 stores compressed horizontal slices
  – Decompress entire slice to get one value
• Experimenting with storing in netcdf-4
  – Chunk to get timeseries data at a single point
Big Data
Bigger Data

• CMIP5 at IPSL Climate Modelling Centre
  – 300K netCDF files, 300 Tb.
• Sequential read of 300 Tb @ 100Mb/sec
  – $3 \times 10^6$ sec = 833 hours = 35 days
• How to get that down to 8 hours?
  ➢ Divide into 100 jobs, run in parallel
Required: Parallel I/O Systems

- Shared nothing, commodity disks
- Fault tolerant, replicated data (3x)
- Google File System using map/reduce
  - Hadoop is open source implementation
- Wide industry use
- Cost
  - $3000 per node TCO per year
  - $300K per year for 100 node cluster
  - Cost will continue to fall
  - Not sure if you should rent or buy
Parallel File I/O

Google File System

Hadoop
Required:
Send User programs to server

• Need a query / computation language
  – easily parallelized
  – scientists can/will use
  – Powerful enough to accomplish hard tasks

• What language?
  – Not going to retrofit existing Fortran code
    • Remember, this is post-processing, not model runs
  – Not Fortran, C, Java (too low level)
  – Some subset of Python?
Send User programs to server

- Probably a Domain Specific Language (DSL)
  - Make it up for this specific purpose
  - But make it familiar!
  - So it could look like some subset of existing language
Existing Candidates

• SciDB just proposed ArrayQL:

“ArrayQL currently comprises two parts: an array algebra, meant to provide a precise semantics of operations on arrays; and a user-level language, for defining and querying arrays. The user-level language is modeled on SQL, but with extensions to support array dimensions.”

• Google Earth Engine is developing a DSL
Required: Parallelizable High Level Language

- **Scientific Data Management in the Coming Decade**, Jim Gray (2005)

- Now: File-at-a-time processing in Fortran
- Need: Set-at-a-time processing in HLQL

- Declarative language like SQL (vs. procedural):
  - Define dataset subset to work against
  - Define computation
  - Let the system figure out how to do it
NetCDF “Index Space” Data Access:

OPeNDAP URL:
http://motherlode.ucar.edu:8080/thredds/dodsC/
NAM_CONUS_80km_20081028_1200.grib1.ascii?
Precipitable_water[5][5:1:30][0:1:77]

“Coordinate Space” Data Access:

NCSS URL:
http://motherlode.ucar.edu:8080/thredds/ncss/grid/
NAM_CONUS_80km_20081028_1200.grib1?
var=Precipitable_water&
time=2008-10-28T12:00:00Z&
north=40&south=22&west=-110&east=-80
“Coordinate Space” Data Access:

http://motherlode.ucar.edu:8080/thredds/ncss/grid/
  NAM_CONUS_80km_20081028_1200.grib1?
  var=Precipitable_water&
  time=2008-10-28T12:00:00Z&
  north=40&south=22&west=-110&east=-80

Fake SQL:

SELECT Precipitable_water
FROM  NAM_CONUS_80km_20081028_1200.grib1
WHERE  time=2008-10-28T12:00:00Z
  AND space=[north=40,south=22,west=-110,east=-80]
More Elaborate

DATASET cfsr
FROM CFSR-HPR-TS9
WHERE month=April AND year >= 1990
AND space=[north=40,south=22,west=-110,east=-80]
AS Grid
SELECT precip=Precipitable_water, rh=Relative_Humidity,
    T=Temperature
FROM cfsr
CALC DailyAvg(Correlation( precip, rh) / Avg(T))
RETURN AS Grid
DATASET cfsr
FROM CFSR-HPR-TS9
WHERE month=April AND year >= 1990
AND space=[north=40,south=22,west=-110,east=-80]
AS Grid
CALCDEF myCalc (X,Y,DATA) {
    X ← 3 3⍴9 ⋄ Y ← DATA[⍋DATA] }
SELECT precip=Precipitable_water, rh=Relative_Humidity, T=Temperature
FROM cfsr
CALC myCalc(precip, rh, T)
RETURN AS Grid
Summary: Big Data Post-Processing

• Need parallel I/O System
  – Shared nothing, commodity disks, replicated data

• Need parallel processing system
  – Hadoop based on GFS (Map/reduce)

• Need to send computation to the server

• Need a parallelizable query / computation language
  – Possibly declarative
  – Must be expressive and powerful
  – Probably a new “domain specific” language
  – Need to capture common queries