NetCDF Build and Test System

Ed Hartnett, 1/25/8

Outline

NetCDF Repository

Building NetCDF

Testing NetCDF

NetCDF Code Repository

We use cvs for netCDF code repository.

The cvs repository module is called **netcdf-3**, but should just be called netcdf, since it holds versions 2, 3, and 4 of the API.

Someday we would like to move to subversion.

Repository Suggestions

Coding spheres of influence for each developer. (Russ: ncdump, ncgen, libsrc. Ed: top level, libsrc4, fortran, f90. Dennis: new directory). Test dirs open to all.

Don't beak the build, but if you do, fix it soon. If it becomes a problem, other developers can roll back code changes to make the build work.

Check in frequently (passing **make check** on your machine is good enough to check in).

No cvs banches without serious thought.

Notification of cvs updates by team members is very helpful, and will become more so with Dennis working on the code.

Autotools Background

Autotools are GNU standard, and (probably) the only ones that meet strict requirements for maximum portability.
NetCDF has used the GNU autoconf technology since 2004.
GNU automake and libtool were used starting with the 3.6.0 (2006).
Code does not contain dependencies on platform, but on features. The configure script runs tests for each feature without regard to platform.
It is hard in the short term, but leads to wide portability, even to new platforms.

Autotools puts burden on developers to make it easy for the user.

Complications: Fortran

The F77 API is just the C API, with function names changed, and parameters munged to look more Fortrany.

To work the F77 API runs some tests in configure script, and also makes a compiler-based choice of how to handle calling Fortran from C.

The F90 API is F90 wrapper code, which calls the F77 API.

This is the biggest source of support and portability problems. We hope to replace it with fortran's portable new way of calling C (Fortran 2003).

Fortran introduces many complexities into the build system. These will be ignored in this presentation.

Development Using Autotools

Developer modifies configure.ac in top level directory, and Makefile.am in every directory.

Only developer files are checked into the repository. Generated files are not checked in.

Each developer runs **autoreconf -i** to generate configure script and other files.

Then run the configure script and use **make** to build the software.

Developers must have autotools installed, but user only needs POSIX2 commands, not autoconf, automake, and libtool.

The configure.ac File

The configure.ac file is fed to autoconf to generate the shell script called configure.

The configure.ac file can be pretty annoying to code.

Most everything can (and should) be accomplished with pre-existing macros.

What configure.ac Looks Like

```
AC_MSG_CHECKING([whether netCDF-4 is to be built])
AC_ARG_ENABLE([netcdf-4],
              [AS_HELP_STRING([--enable-netcdf-4],
                              [build with netcdf-4 (HDF5 is required)])])
test "x$enable_netcdf_4" = xyes || enable_netcdf_4=no
AC_MSG_RESULT($enable_netcdf_4)
AM_CONDITIONAL(USE_NETCDF4, [test x$enable_netcdf_4 = xyes])
if test "x$enable_netcdf_4" = xyes; then
   AC_DEFINE([USE_NETCDF4], [1], [if true, build netCDF-4])
   AC_DEFINE([H5_USE_16_API], [1], [use HDF5 1.6 API])
fi
```

Some NetCDF configure Options

--help Show all options.

--enable-shared Turns on shared library build.

--prefix=/someplace Where to install netCDF.

--enable-netcdf-4 Turns on netCDF-4.

--with-hdf5=/someplace Where to find HDF5.

--with-zlib=/someplace Where to find zlib.

Output of configure to stdout

```
configure: netCDF 4.0-snapshot2008012802
checking build system type... x86_64-unknown-linux-gnu
checking host system type... x86_64-unknown-linux-gnu
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
```

etc.

Output of configure for Build

When the configure script is run, a C header file conig.h is created.

The Makefile.in files are used to construct the user Makfiles.

The config.h File

Tests in the configure script can set pre-processor macros.
Configure writes config.h in the top-level directory.
All netCDF code files **must** include config.h. Otherwise they may break tests in non-obvious ways.
The include for config.h must come **first** in the code.
Users do not include config.h. It is an internal netCDF file.
config.h is produced by configure, but may be edited by hand for debugging or developer convenience. It is not checked in.

What config.h Looks Like

```
/* use HDF5 1.6 API */
#define H5_USE_16_API 1

/* Define to 1 if you have `alloca', as a function or macro. */
#define HAVE_ALLOCA 1

/* Define to 1 if you have the <dlfcn.h> header file. */
#define HAVE_DLFCN_H 1

/* Define to 1 if you have the <hdf5.h> header file. */
#define HAVE_HDF5_H 1

/* Define to 1 if you have the <inttypes.h> header file. */
#define HAVE_INTTYPES_H 1
```

The Makefiles

NetCDF developers edit the Makefile.am in each directory.

The automake utility converts these to Makefile.in files to include in the distribution.

The configure script converts Makefile.in files to Makefiles. The user then runs their local make to do the build.

Makefile.am can contain conditionals which are set in the configure script. Example for conditional BUILD_F77:

```
if BUILD_F77
F77_DIR = fortran
F77_TEST = nf_test
endif
```

Using Autotools - the Cook Book Recipie

In summary: modify the **configure.ac** in the top level code directory, and the **Makefile.am** in each directory.

Check out from cvs, run autoreconf, then configure, then make:
cvs co netcdf-3 && cd netcdf-3 && autoreconf -i && ./configure && make check
Some important standard make targers: all, check, install, clean, distclean, maintainer-clean, dist, distcheck.

Complication: NetCDF Has Two Release Tracks

Users may get either netCDF-3.x or netCDF-4.x. Eventually we anticipate ending the 3.x branch.
The 4.x branch can build the 3.x branch as well.
The two distributions are the same except for the **configure.ac** file.

The **configure.40** is for netCDF-4.x **configure.ac** is for netCDF-3.x.

If this were to be permanent, or involved more than one file, I would use a cvs branch.

Building NetCDF-4 from Repository

```
cvs co netcdf-3
cd netcdf-3
cp configure.40 configure.ac
autoreconf -i
./configure --enable-netcdf-4 --with-hdf5=/upc/share/ed/local/buddy
make check
```

Suggestions for Autotools Development

There are lots of quirks to using autotools. As they say in their documentation: "autoconf is a mess because the world is a mess."

Most simple programming tasks can be easily handled in the local Makeile.am. (Then just run **make** and it will be handled.)

The most common configure.ac edit is to add an option, which can be done with cut-and-paste.

The full test suite will turn up some subtle mistakes, which Ed can fix.

Nightly Build and Test

Both netCDF-3.x and netCDF-4.x are built and tested nightly on Unidata machines.

If **make distcheck** passes on shecky, a daily snapshot is released to the ftp and web sites.

Snapshot release gets version changed to include the date and time, for example:
3.6.2-snapshot2007073018

Test Scripts

Three bash scripts run tests: **test_cvs_module.sh**, **test_one_machine.sh**, and **test_lib.sh**.

test_cvs_module.sh will check out from cvs, build and test, put out snapshot distribution, and then call **test_one_machine.sh** for a list of test machines.

test_one_machine.sh will get the snapshot dist, unpack, and test it. (Just like a user would).

Test scripts are checked into cvs. Do **cvs co builder** to get test system. (But I've only every run it on shecky).

Running the Test Scripts

The test scripts are run each night (at 1 a.m.) but can also be run (by Ed) from the command line.

Various options help make this a tool for checking completeness of distribution and cross-platform portability.

-v verbose mode
-n netCDF-4 build
-m machine1 machine2 run tests on these machines only
-q run quickly by only doing one test on each machine.

Verbose Test Script Output

```
bash-3.2$ ./test_cvs_module.sh -vnqm buddy
release =  all =  local =  machine_spec = 1 machine_list = buddy quickly = 1 netcdf4 = 1
setting up netcdf-4 build
package netcdf-4.0-snapshot2008012608 dist_dir /content/software/netcdf/builds/snapshot/netcdf-4 docdir
/content/software/netcdf/netcdf-4/newdocs
testing netcdf-4.x
getting the code from CVS...
output to /content/software/netcdf/builds/snapshot/netcdf-4 and /content/downloads/netcdf/netcdf-
snapshot
autoreconf in /shecky/n4_test, output to /upc/share/ed/output/nc4_autoreconf.txt...
building 4.0...
shecky building with make check distcheck, options --enable-netcdf-4 --with-
hdf5=/upc/share/ed/local/shecky --with-zlib=/upc/share/ed/local/shecky
copying /upc/share/ed/output/nc4_make_distcheck_out.txt to ed@conan:
/content/software/netcdf/builds/snapshot/netcdf-4
shecky packaging docs for website...
netcdf_docs.tar.gz                          100% 9509KB   9.3MB/s   00:00
 *** FAILURE on Linux (shecky), gcc/g++/gfortran 4.1.1 with make check distcheck with --enable-netcdf-4
--with-hdf5=/upc/share/ed/local/shecky --with-zlib=/upc/sh\
```

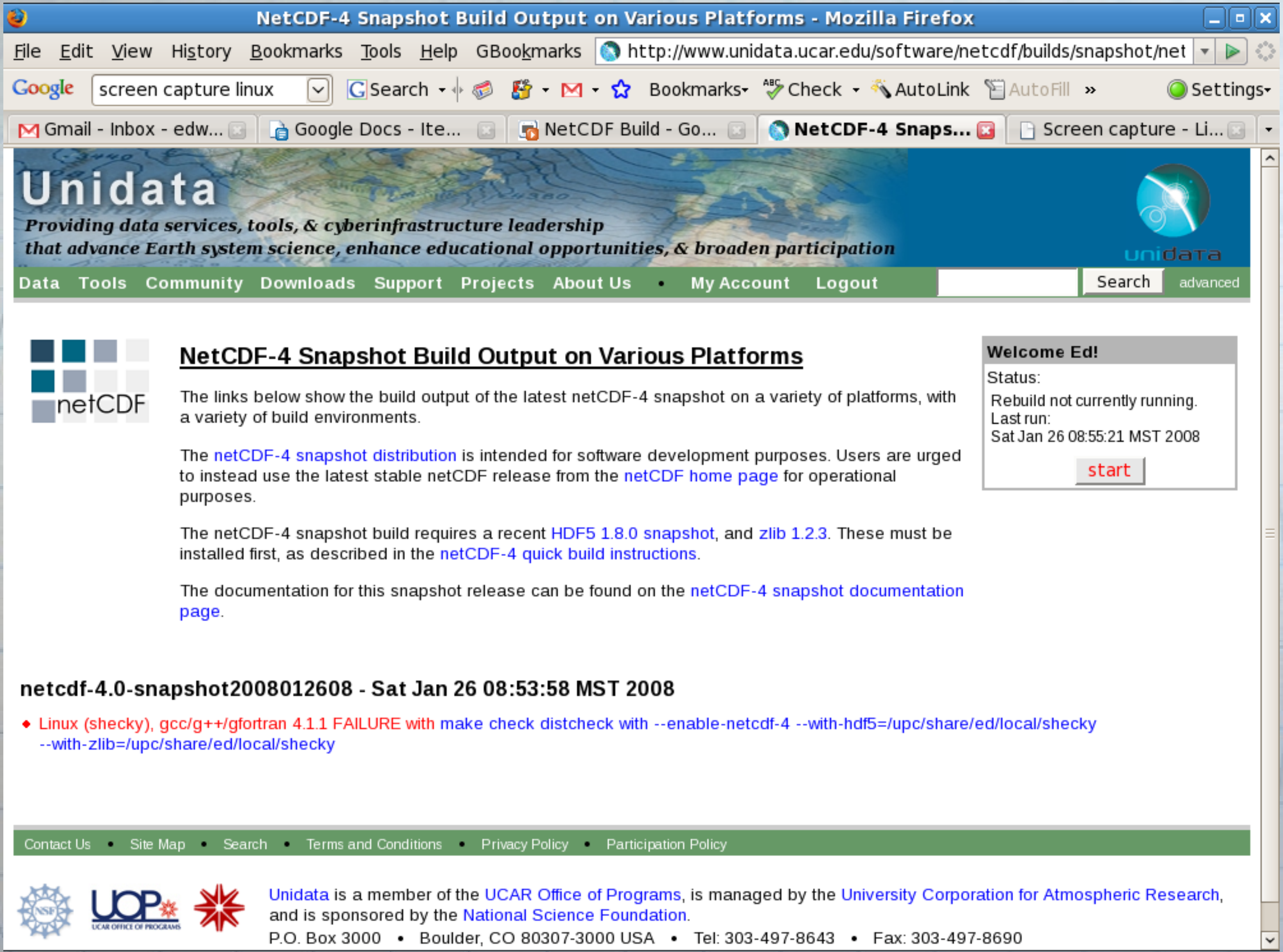
Test Output

HTML pages are generated and copied to conan with test results.

XML pages are generated and copied to conan with pointers to daily snapshot and binary distributions.

Text summary of results emailed to netCDF developers.

Documentation is generated in it's many formats and copied to web site.


NetCDF-4 Snapshot Build Output on Various Platforms - Mozilla Firefox
File Edit View History Bookmarks Tools Help GBookmarks
http://www.unidata.ucar.edu/software/netcdf/builds/snapshot/net
screen capture linux
Search
Bookmarks
Check
AutoLink
AutoFill
Settings
Gmail - Inbox - edw...
Google Docs - Ite...
NetCDF Build - Go...
NetCDF-4 Snaps...
Screen capture - Li...
Unidata
Providing data services, tools, & cyberinfrastructure leadership
that advance Earth system science, enhance educational opportunities, & broaden participation
Data Tools Community Downloads Support Projects About Us • My Account Logout
Search
advanced
netCDF

NetCDF-4 Snapshot Build Output on Various Platforms

The links below show the build output of the latest netCDF-4 snapshot on a variety of platforms, with a variety of build environments.

The netCDF-4 snapshot distribution is intended for software development purposes. Users are urged to instead use the latest stable netCDF release from the netCDF home page for operational purposes.

The netCDF-4 snapshot build requires a recent HDF5 1.8.0 snapshot, and zlib 1.2.3. These must be installed first, as described in the netCDF-4 quick build instructions.

The documentation for this snapshot release can be found on the netCDF-4 snapshot documentation page.

Welcome Ed!

Status:
Rebuild not currently running.
Last run:
Sat Jan 26 08:55:21 MST 2008

start

netcdf-4.0-snapshot2008012608 - Sat Jan 26 08:53:58 MST 2008

- Linux (shecky), gcc/g++/gfortran 4.1.1 FAILURE with make check distcheck with --enable-netcdf-4 --with-hdf5=/upc/share/ed/local/shecky --with-zlib=/upc/share/ed/local/shecky

Contact Us • Site Map • Search • Terms and Conditions • Privacy Policy • Participation Policy

Unidata is a member of the UCAR Office of Programs, is managed by the University Corporation for Atmospheric Research, and is sponsored by the National Science Foundation.
P.O. Box 3000 • Boulder, CO 80307-3000 USA • Tel: 303-497-8643 • Fax: 303-497-8690

Test System Future Work

The test system is in a constant state of development as I tinker with it. Some imminent features:

- a button on the snapshot results page to launch testing.
- test scripts portable across developer machines so that each developer can run them from the command line.
- generate named release instead of just snapshot release, and use cvs tag to tag the source automatically.
- it would not be hard to make this script work on any cvs module, not just netcdf; libcf is the next target, when development kicks off there again.

Feel free to also tinker with thee test scripts.

Adding Tests

Add tests in directory with the code they test.

Test programs must accept no arguments, and return 0 for success, something else for failure.

I always name them tst_something, like tst_compounds.c.

I have them create test data files with the same name as the program (that is, tst_compounds.c produces data file tst_compounds.nc).

It's easier to create test files as needed rather than create them and ship them with the dist, but when shipped, I call them ref_tst_something.nc. (ref = reference).

Test Requirements

Test requirements:
www.unidata.ucar.edu/software/netcdf/netcdf-4/req_4_0.html

- All tests are automated and can be run from makefile targets.
- Some tests make take excessive time. Make test may skip very lengthy tests...
- All supported language interfaces are tested.
- Test programs do not accept or require command line options.
- Test programs provide feedback in accordance with netCDF test look and feel.

etc.

Suggestions for Tests

Do not take any tests away, or even make modifications. Take a copy of the test and make modifications to that.

Repeating code in tests is OK, if it allows simpler tests to co-exist with more complicated tests. The simpler a failed test is, the easier to debug.

Some macros are defined in nc_tests.h to help with testing:

- ERR - report "unexpected result" and continue.
- ERR_RET - same, but exits program immediately.
- SUMMARIZE_ERR - summarizes result of one set of tests.
- FINAL_RESULTS - adds up errors and exits with 0 (no errors) or 2 (one or more errors).

More Test Suggestions

Don't write throw-away tests, always add them to the repository and build.

Build up to complex tests with a series of simple tests to aid debugging.

Shell scripts can also be used as test, but they add complexity. However for testing ncgen/ncdump they are needed.