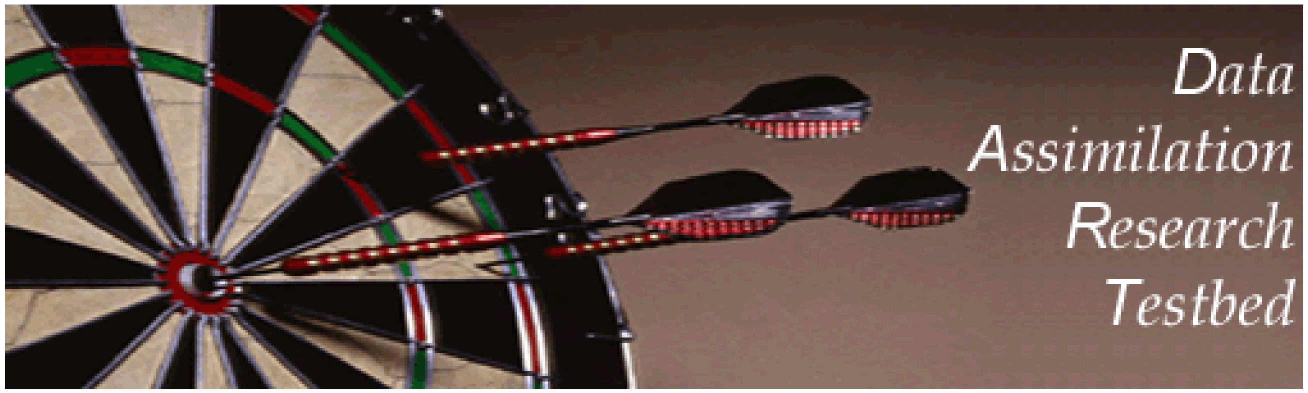
Data Assimilation Research Testbed Tutorial



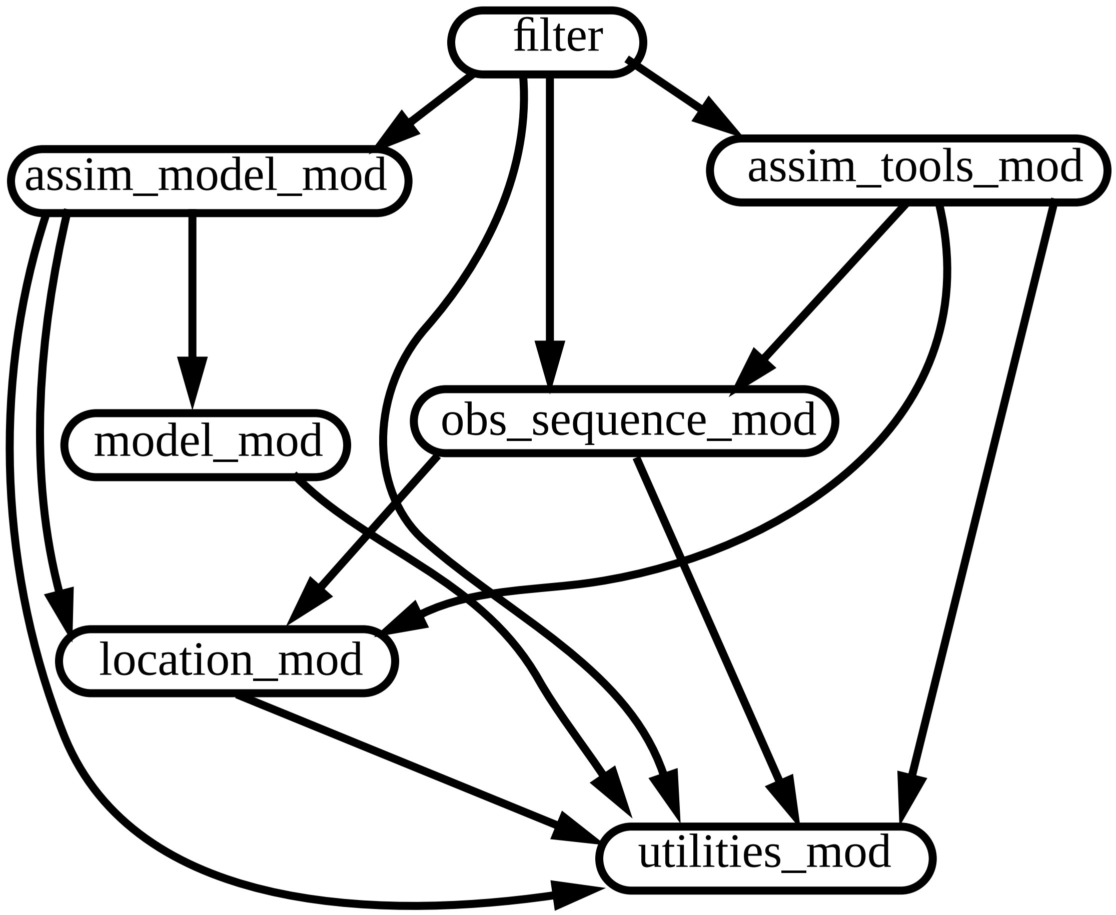
Section 11: Creating DART Executables

Version 1.0: June, 2005

Fortran 90 ‘Use’ Trees:

DART requires use of F90 *use module_mod, only:*

No other mechanism for use of ‘external’ routine.



Program viewed as Directed Acyclic Graph (tree with shared leaves).

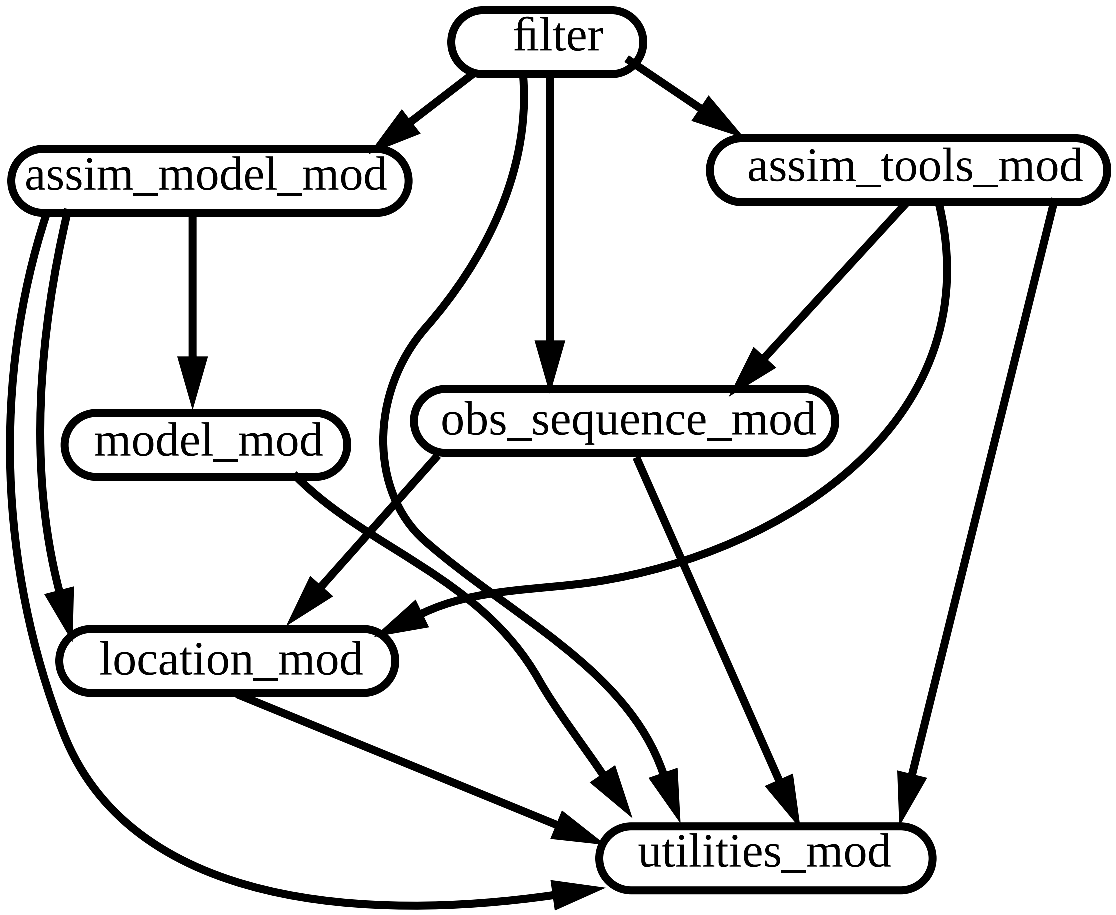
No cycles!

Partial tree for filter is shown.

Vector represents *use* of target module.

Generating a makefile with mkmf:

DART requires use of F90 *use module_mod, only:*



Use mkmf perl script to *make* a *makefile*

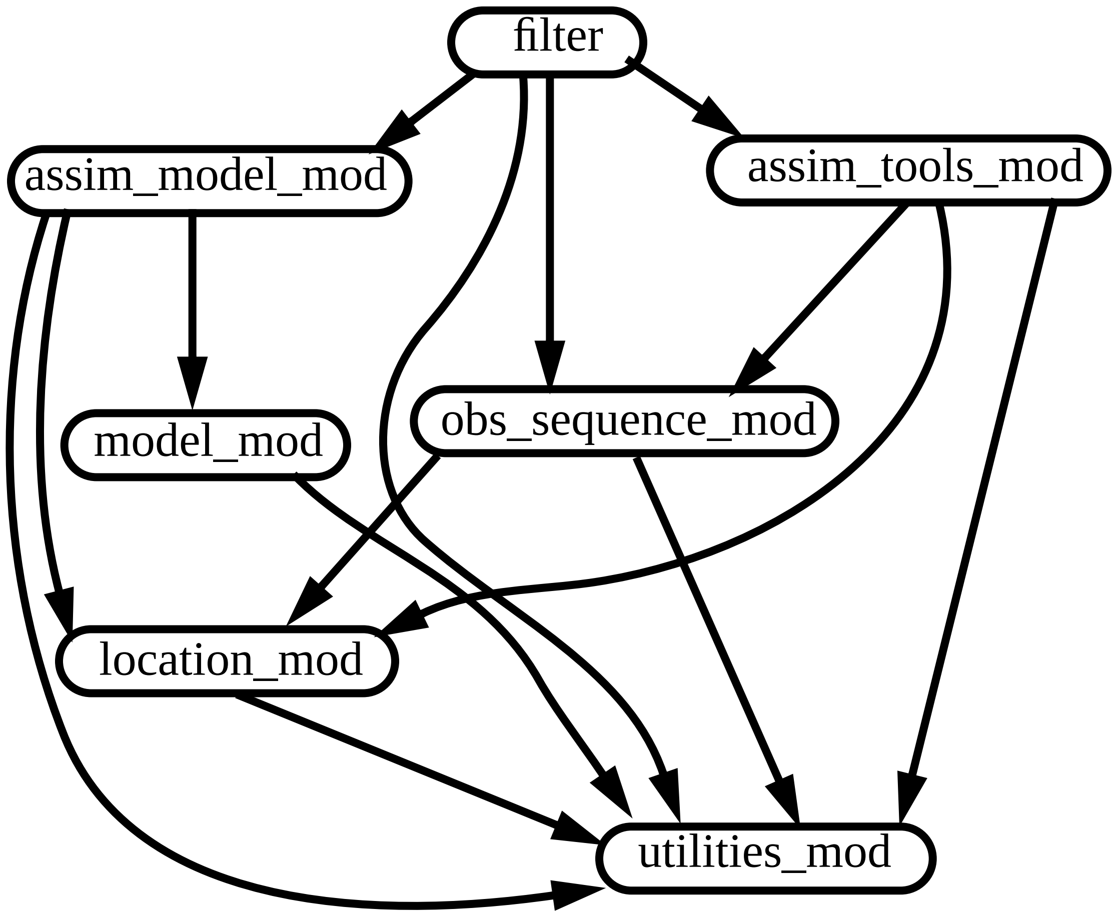
mkmf requires a list of files to search for the main program and modules that are used.

This is called a *path_names* file.

See *path_names_filter* in *models/lorenz_63/work*.

Generating a makefile with mkmf:

DART requires use of F90 *use module_mod, only:*



mkmf searches files in path_names for one that contains *program filter.*

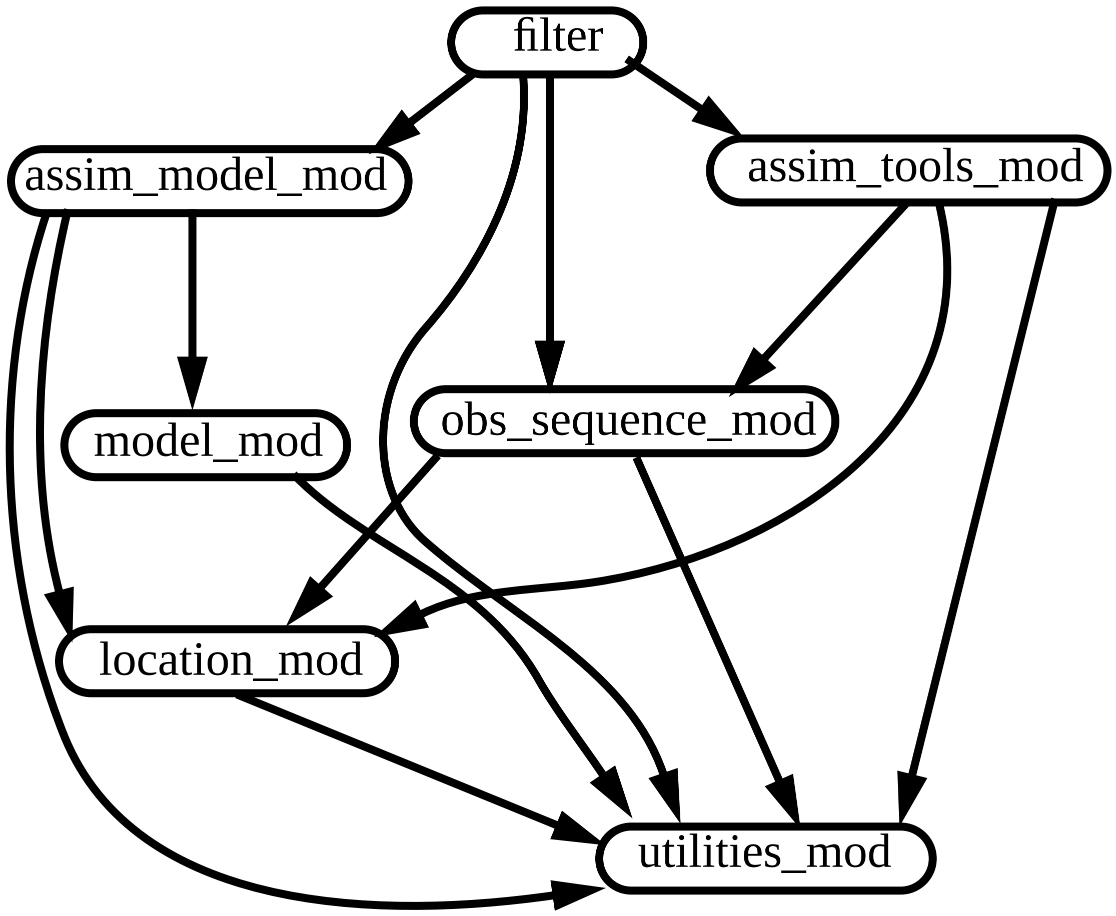
Finds first *use only* in filter.

Searches path_names files for this module recursively.

Builds a dependency graph like one at left.

Generating a makefile with mkmf:

DART requires use of F90 *use module_mod, only:*



From dependency graph, mkmf generates a standard *Makefile*.

Also creates a default namelist file, *input.nml.filter_default*.

Copy this to *input.nml* to use it.

Enter *make* to create filter executable.

mkmf details:

To run mkmf for filter, execute shell script *mkmf_filter*.

File *mkmf_filter* contains (default values in parentheses):

1. Relative location of the mkmf perl script (../../../mkmf/mkmf),
2. Name of executable program to create (-p filter),
3. Compiler options file (-t ../../../mkmf/mkmf.template),
4. Additional libraries needed (-c"-Duse_netCDF"),
5. Relative base location for file search (-a "../../.."),
6. Name of the path_names file (path_names_filter).

Each DART program has its own *mkmf_* and *path_names_ file_*.
Can see a selection of these in *models/lorenz_63/work*.

mkmf details:

A variety of mkmf templates are available in directory *mkmf*.
(You can also see the perl script *mkmf* here).

mkmf templates specify:
What compiler to use,
Where to find netCDF libraries,
Where to find udunits library (used by netCDF),
Compile and link command line options.

Templates are available for an array of architectures and compilers.

Two ways to change your template:
1. Change the template name in your *mkmf_filter* file,
2. Copy the appropriate mkmf filter to *mkmf/mkmf.template*
(a default location).

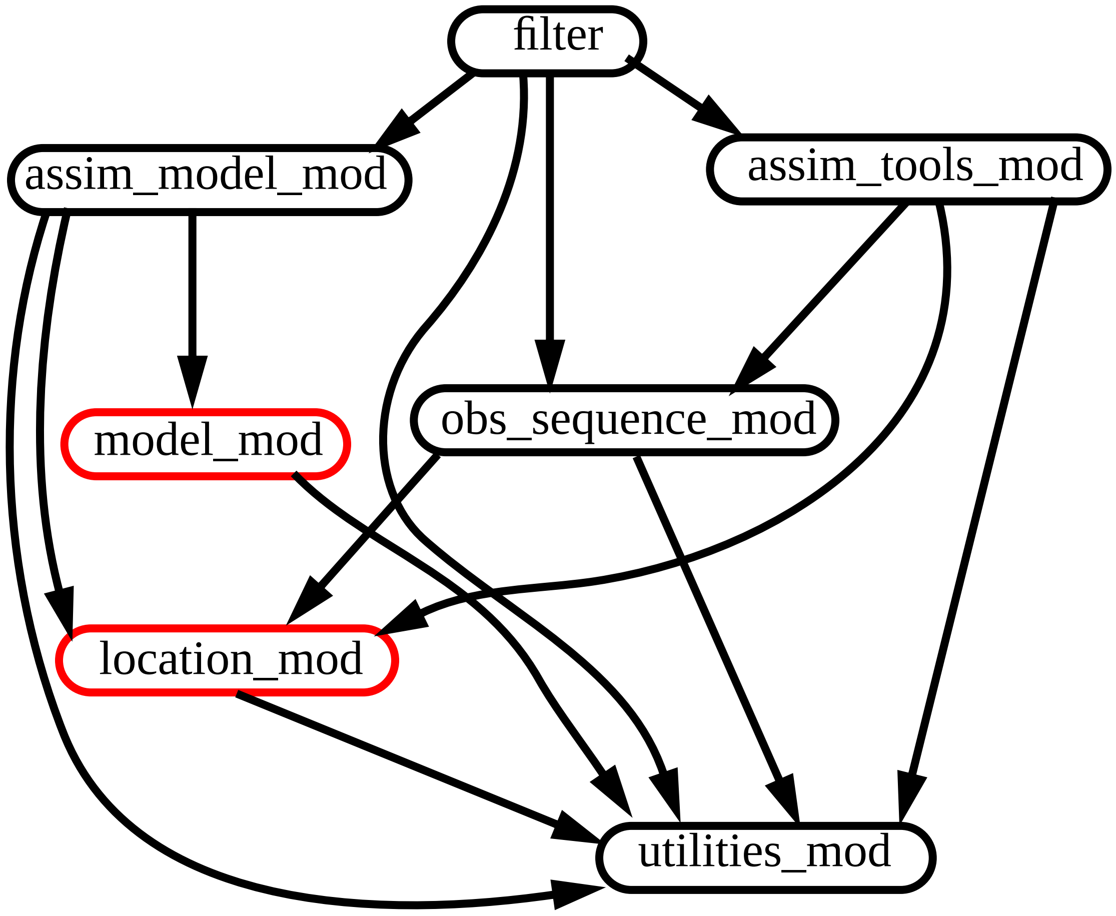
The Dart Modular Philosophy

Can swap modules with same names and public interfaces:

Changing the following paths in path_names_filter

models/lorenz_63/model_mod.f90 -> models/cam/model_mod.f90

location/oned/location_mod.f90   -> location/threed_sphere/location_mod.f90



Switches from lorenz-63 to CAM GCM!

Modules with multiple implementations have second directory level (see paths above).

Compare path_names files in models/lorenz_63 with those in models/cam

Exercise: Compiling lorenz_63 filter program

1. Go to *models/lorenz_63/work*.

2. Remove all files with .o and .mod extensions.
   (Depending on compiler you've been using, existence will vary).

3. Generate a *Makefile* and *input.nml.filter_default*
   Enter *csh mkmf_filter*

4. Generate program filter:
   Enter *make*