13 CVS

13.1 What is CVS?

CVS stands for Concurrent Version System, and used for version control.
Special features of CVS are:

- Non-proprietory, and can be downloaded from the internet;
- Allows users to track their changes by revision, tag and date;
- Can obtain an earlier version easily;
- Allows you to track the supplier's software releases while making your code change locally. Will enable you to merge code changes between your version and supplier's automatically and complain if merge presents contradictions;
- A user of CVS needs only to know a few basic commands to use the tool.

For more information on CVS, visit our Web page:

http://www.mmm.ucar.edu/mm5/dhansen/cvs.html

13.2 How to Use CVS?

There are two ways you can use the CVS:

1. Use CVS to keep up to date with the mesouser changes. This will require a SCD/meeker account. If you would like to do this without loggin on to meeker, you need CVS installed on your local machine.

2. Use CVS to track both mesouser's releases and your own change. Again you can do this either on meeker or on your local machine (with your own CVS installation).

13.3 Use CVS to Keep Up to Date with Mesouser's Changes

This method requires access to mesouser's repository.

The master repository name on NCAR/SCD's meeker machine is

/fs/othrorgs/home0/mesouser/repository

To access this repository, you need to set the following:

setenv CVSROOT /fs/othrorgs/home0/mesouser/repository

** Note that if you want to access a different repository, you need to change the environment variable CVSROOT.

Currently only the MM5 directory is present. We will later add Terrain, Datagrid, Rawins, Interp and Graph to the repository. The repository is for MM5 modeling system version 2 only.

To use CVS on meeker, you must have an access to the machine. If you are an SCD user, you should be able to open an account on meeker by contacting SCD.

When you are on meeker, edit your .cshrc file to add

~mesouser/bin/cvsbin/bin

in your path. This makes sure that you have access to CVS software.

There are a few simple commands you need to use: (for complete CVS documentation, see our Web page: *http://www.mmm.ucar.edu/mm5/dhansen/cvs.html.*) for information on how to download).

cvs export [-D today] [-r tag] MM5

This will give you exported version of MM5, the same as you get when you un-tar the tar file we provide you. The things in [] are options. '*-D today*' gives you the latest version of the code. You can also specify " *-D 'June 24, 1997'* " (note that the date is in single quote) for a version of that day, or use '-r release-2-5' for release 2.5 (release-2-5 is a CVS tag).

cvs checkout MM5

This will give you, in addition to the exported version, CVS information. With such information, users will be able to keep up-to-date with our release automatically with a simple *cvs update* command (instead of having to manually insert the changes we broadcast). Once you check out a version of the code, you form a 'working directory'.

cvs update MM5 (only works if a user has cvs-checked-out version)

This will bring the changes made in the master repository to your working directory. An example of print out from this command:

cvs update: Updating domain/io
U domain/io/conmas.F
M domain/io/initsav.F
C domain/io/savread.F
? domain/io/diff

A letter 'U' implies a subroutine has been updated, and a letter 'M' implies that you have made changes to a subroutine with respect to the subroutine in the master repository. A '?' indicates that the file does not exist in the master repository. If there is any conflict between the changes in the master repository and your working directory, a letter 'C' appears and an message will appear to warn you to resolve the differences. In the subroutine where conflict occurs, the related code is surrounded by >>> ... <<<.

As an example, say you checkout a copy of MM5 on meeker by doing:

cvs checkout MM5

This will create a copy of MM5 in your own directory (you probably want to do this in your home directory so that it is not purged) with CVS information. Say you have made some changes to your copy of MM5, and then the next MM5 release arrives, you can simply do a **cvs update** to bring the new changes in the new release into your copy:

cd MM5 (all the files are under MM5 after you checkout)
cvs update

A list is printed on your screen to let you know that which files are updated (a 'U' in front of the file) from the new release, and which files are modified by you (a 'M' in front of the file) and any conflict that may result from this update. It is a good practice to pipe the '**cvs update**' output to a file.

Note that doing **cvs update** under the MM5 directory will automatically update the entire code. You can update individual directory or file by going into the directory and do '**cvs update**' - which updates that directory and any sub-directories, or '**cvs update filename**' - which updates only that file.

After this, you can tar up the entire MM5 directory and move the tar file to the machine you want to run the model. Next time when a new release arrives, you can move your tar file back to meeker, un-tar it, and do a '**cvs update'** to bring more changes to your working directory.

Note that the cvs-checkout version of code has CVS directory in each of the directories in the tar file. Do not remove these directories. The command '**cvs update**' relies on these CVS directories to work.

This method only allows you to update your code with mesouser's, but does not allow you to track your own change.

cvs diff filename

This does differencing between the file in your working repository with the one you check out from the repository. Example:

cvs diff -r release-2-2 file1

cvs log filename

This lists the log messages and status of the master repository.

The above information applies to meeker users. If you have CVS installed on your local machine, and you have access to meeker, you can set up so that you can do the above CVS commands remotely without login onto meeker. In this case, you need to set the following line in your local .cshrc file:

setenv CVSROOT meeker.ucar.edu:/fs/othrorgs/home0/mesouser/repository

You will need a proper .rhosts file on meeker to make this work.

13.4 Use CVS to Track Both New Releases and Your Changes

This method does not require the access to mesouser's repository. It requires CVS software and a tar file from, for example, /fs/othrorgs/home0/mesouser/MM5V2/MM5.

If you want to maintain your own code and keep track of the changes from mesouser, what you should do is to create your own repository and use the 'vendor branch' concept in CVS. You may do this on SCD's meeker, or on your local machine, on which CVS is available. If you need to do this on meeker (because you don't have CVS installed on your local machine), you need to set the environment variable CVSROOT to YOUR directory:

setenv CVSROOT ~yourname/your-repository
(e.g. setenv CVSROOT ~fred/repository)

You also need to add **~mesouser/bin/cvsbin/bin** to your path in your .cshrc file so that you have access to CVS software we have installed.

If you will do this on your local machine, set

setenv CVSROOT some-home-directory-on-your-local-machine
(e.g. setenv CVSROOT ~fred/repository)

in your .cshrc file.

To initialize the repository, cd to your repository directory (in the example, cd to ~fred/repository), type

cvs init

Now, say you start your MM5 V2 adventure with release-2-5. First you obtain the mm5v2.tar file, bring it to your favorite machine and in a temporary working space,

mkdir temp (in your temporary directory - to be eliminated later)
cd temp
tar -xf mm5v2.tar (this is mm5 v2 release-2-5)

then type

cvs import -m "Import of MM5 V2 release-2-5" MM5 NCAR release-2-5

If the import is successful, the last message CVS prints is

No conflicts created by this import

(After this you can remove the temporary directory temp.) This will create a repository directory by the module name MM5 in your 'some-home-directory-on-your-local-machine' and with vendor tag 'NCAR' and release tag 'release-2-5' (the sentence in " "is a log message). Note that by using a vendor tag you have created a 'vendor branch'.

After this you can use this repository to checkout MM5 code (by doing so, you are creating a working directory), make changes and bring your changes to your repository - you are doing these in the working directory.

Say if you have made some changes and you are happy with them, and want to bring them to your repository for keep. You should do the following:

cvs diff > output.diff

(at the top directory where configure.user exists, and direct the output to a file to output.diff so that you can check;)

cvs update

(you can do this at the top directory where configure.user exists. Or you can do this in the subdirectory where you have made changes to)

cvs commit -m 'message for the commit'

(this will send all your changes from the current directory down recursively to YOUR repository)

After this, you can ‘**cvs export**’ a copy and bring it to the machine where you want to run the model.

Later when mesouser’s release-2-6 becomes available, you can get the new tar file, bring it to your local machine/or your directory on meeker, and again in a temporary working space,

mkdir temp (in your temporary directory - to be eliminated later)
cd temp
tar -xf mm5v2.tar (this is mm5 v2 release-2-3)

then type

**cvs import -m “Import of MM5 V2 release-2-6” MM5 NCAR release-2-6 **
>& ../import.log

(again after this you can remove the temporary directory temp.) It is important to keep a log of CVS print message when you import, because this tells you what CVS did during the import process, and send the print message to a file in a upper directory (otherwise CVS may think the import.log file is a new file and will import it to your repository).

Note that in this import, the only change is the release tag (release-2-6 now) and import log message. The import.log file contains messages like:

U MM5/CHANGES - for straight update (no conflict)
N MM5/ReleaseNotes/diff.24FEB97 - for a new file
C MM5/domain/io/rdinit.F - for files that conflicting changes exist if you have made local changes

If you have made one change to a line of code that vendor has also changed in the new release, at the end of this import it would say

1 conflicts created by this import. Use the following command to help the merge:
cvs checkout -jNCAR:yesterday -jNCAR MM5

If new release comes more than once per day, you should use the following to merge the changes

cvs checkout -jrelease-2-5 -jrelease-2-6 MM5 >& merge.log

You can use this command with or without a working directory. Note though if you have been working in your working directory, make sure that you have committed all the changes to the repository before you acquire a new release from the vendor (in this case, mesouser) so that your working directory is up-to-date with the repository (doing **cvs update** will tell you whether your working directory is up to date with the repository). Also the above command should be executed at a directory above your working directory. Executing the above command will create or update a working copy of MM5 that includes the changes you made since release-2-5, and changes in new release-2-6. The merge.log file should tell you in which files that conflicts exist.

An example of the output from the above command:

U MM5/domain/io/outtap.F
RCS file: /mmmtmp/weiwang/test/MM5/domain/io/rdinit.F,v
retrieving revision 1.1.1.1
retrieving revision 1.1.1.2
Merging differences between 1.1.1.1 and 1.1.1.2 into rdinit.F
rcsmerge: warning: conflicts during merge

The last line warns you that conflicting changes exist in file *rdinit.F.* You should then edit the file, resolve the differences, and commit the changed file.

An example of conflicting changes in a file:

```
<<<<<<< rdinit.F
      STOP 'STOP IN RDINIT'                                             RDINIT.221
  910 PRINT *,'ERROR IN READING 2D FIELDS. PROBABLY LESS 3D FIELDS'
      STOP 'STOP IN RDINIT 910'
=======
     *      ,' CHECK IF INPUT FILE EXISTS'                              24FEB97.83
      STOP 'STOP 999 IN RDINIT'                                         24FEB97.84
  998 PRINT *,'ERROR IN READING HEADER. CHECK INPUT FILE, UNIT ',IUNIT  24FEB97.85
     STOP 'STOP 998 IN RDINIT'                                          24FEB97.86
  991 PRINT *,'ERROR IN READING INPUT FILE. CHECK THE FILE, UNIT ',IUNIT 24FEB97.87
     STOP 'STOP 991 IN RDINIT'                                          24FEB97.88
>>>>>>> 1.1.1.2
```

The lines above ======= are changes you have made, and below ======= are changes from the new release.

NOTE: even there is no conficit occurring during the merging, you should always check the part of code you have changed to make sure that your changes are compatible with mesouser's, therefore will do what you expect it to do.

13.5 Where to Obtain CVS on the Internet?

Please go to our Web page:

http://www.mmm.ucar.edu/mm5/dhansen/cvs.html

and click on 'CVS Bubbles'.