Simple CVS Tutorial

ECE 379: Introduction to Computer Engineering
University of Wisconsin – Madison
Prof. Mikko H. Lipasti
TA: Payam Karbassi

CVS (Concurrent Versions System) is a simple software engineering tool that allows you to store multiple concurrent versions of a text file (usually a program source file, but not necessarily). It keeps track of multiple versions in an archive, and allows one or more users to check out specific versions, compare changes, commit updates with meaningful comment tags, and generally play nicely in an environment where more than one programmer is editing code in the same code base.

This tutorial focuses on using CVS in the unix environment. CVS is also available for other platforms (e.g. Windows) but those are beyond the scope of this tutorial.

This tutorial assumes you are already somewhat familiar with the unix command line, and are able to create directories, change directories, and edit files with a text editor (e.g. xemacs).

CVS stores code in a repository, which exists inside a single directory, usually ~/cvsroot (the '~' refers to your home directory).

CVS is most useful when more than one user is editing the code base; in this case both users must use the same repository, and file permissions must be set up to allow both users write access to that directory. Setting up file permissions for groups of users is described under Sharing Files on the CAE web pages (see http://www.cae.wisc.edu/site/public/?title=unixsharefile).

Follow these steps only once, to create a CVS repository for yourself (which you can also share with other users):

1. Set up a directory to store CVS entries
   $ mkdir ~/cvsroot
2. Optional (if sharing a repository with other users): Create a group and set permissions for ~/cvsroot appropriately, as described on the CAE help pages (http://www.cae.wisc.edu/site/public/?title=unixsharefile).
3. Set up an environment variable to point to this directory
   $ setenv CVSROOT ~/cvsroot
4. Recommended: add the above command to your ~/.cshrc file so it is invoked automatically every time you log in (otherwise you will have to type it in every time you log in). If you are sharing a CVS repository with another user, you will have to change this in your .cshrc, log out, and log back in (or source ~/.cshrc).
5. Run cvsinit to initialize the repository
   $ cvs init

Follow these steps once for each project you want to keep in CVS:

1. Create your project (this is just an example; projects should reside in their own directory or hierarchical set of directories):
   $ mkdir ~/myproject
   $ cd ~/myproject
   $ cp ~mikko/prog1/xyz.asm .
2. Import your project into CVS
   $ cvs import –m “My Project Name” myproject sample start

Follow these steps every time you want to start fresh with the archived project:

1. Check out an instance of your project in a different location
   $ mkdir ~/myproject2
   $ cd ~/myproject2
   $ cvs checkout myproject
2. Edit the new instance
   $ xemacs xyz.asm (then type in some new lines of stuff)
3. Compare against versions in the archive (study the output)
   $ cvs diff
4. Commit your changes. Remember to insert a useful comment after the –m option.
   $ cvs commit –m “Typed in some new stuff.” xyz.asm
5. Make some more changes
   $ xemacs xyz.asm (type in some new lines of stuff)
6. Compare against two versions in the archive (study the output)
   $ cvs diff –r1.1
   $ cvs diff –r1.2
7. Commit your changes again
   $ cvs commit –m “Typed in some new stuff.” xyz.asm

If another user is sharing your repository, the “cvs update” command will update your local versions of files to the most current ones in the repository.

Other features to try out include tags and branching, which allow you to associate a symbolic name (tag) with a particular revision, and branching, which allows you to track two different development paths. Read more about these in the online CVS manual:

$ man cvs
$ info cvs