Source Control

By Rachel Sheldon & Stuart Nelis

Source Control - the management of changes to documents, programs and other information stored as computer files.

The idea is that a team of people can remotely edit and develop the same set of files. The Source Control program will track the changes and save the individual drafts.

Reasons for Using Source Control

1. Source Control allows a team of programmers to collaborate on a project from anywhere in the world. The programs can be accessed, amended and then checked back into the repository.
2. It is easier to fix bugs in a program if you can retrieve and run older versions of the file.
3. Source Control allows you to keep track of what changes have been made to a program and who made the changes.

The format of a Source Control system is shown in Figure 1. The copies of the files are stored in the *Subversion repository*. Here, it is possible to view the changes made to the stored files and folders and all the previous versions.

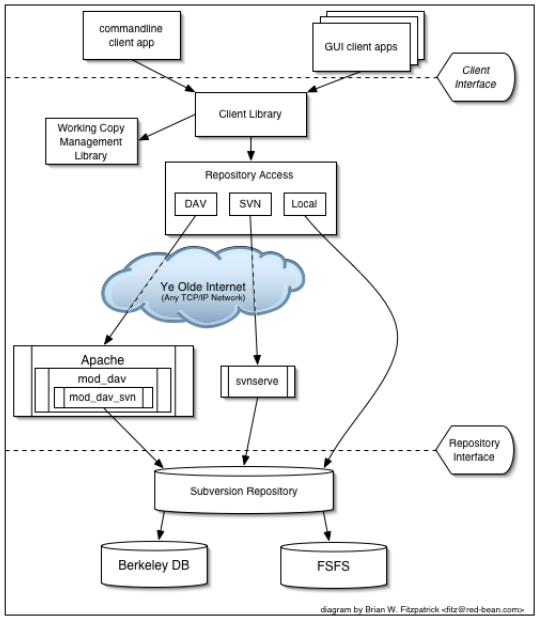

Figure 1: Example Source Control System.

The *Subversion client system* is where the user sees a local reflection of portions of the data. Files are

downloaded from the Subversion repository to the client system, edited and then re-uploaded to the repository.

There are multiple routes through a Repository Access layer. Some go across computer networks and through network servers, which then access the repository. Others bypass the network and access the repository directly.

Figure 2 shows an example history tree of a revision-controlled project.

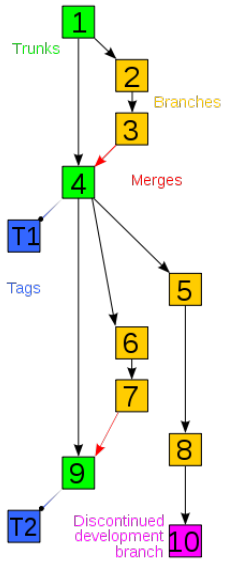

Figure 2: History Tree of a Revision-Controlled Project.

Square 1 represents the initial file. It then is split into 2 branches as 2 different people access the file for amending. It is merged again at square 4 and tagged as an important update. This might represent the stage at which the file has made a significant development or has been sent for publishing. The files are numbered by their revision number sequentially. Therefore file 10 is a later draft than file 9 but was a discontinued development branch, *i.e.* nothing more was done with it.

There are different circumstances where you may need to merge files.

1. A user, working on a set of files, uploads his working copy to the source control. The program merges the updates with the version stored in the source control.

2. A user tries to check in files that have been updated by others since the files were checked out. The software automatically merges the two files. Discrepancies will arise if 2 users have edited the same line of code, bringing up an error message.

3. A set of files is branched. A problem that existed before the branching is fixed in one branch,

and the fix is then merged into the other branch.

4. A branch is created. The code in the files is independently edited and the updated branch is later incorporated into a single, unified trunk.

Setting up Subversion, svnX & MATLAB integration

1. Subversion is a cross platform software tool, suitable for both Window and UNIX like systems. The Subversion software can be downloaded from COLLABNET at `http://www.open.collab.net/downloads/community/`. You will need the correct download for you version of Mac OS X or windows. Once you have downloaded and installed Subversion you will need to set it up through Terminal. A very useful guide to doing this is available at `http://www.rubyrobot.org/tutorial/subversion-with-mac-os-x`. Alternatively a full free online manual is available at `http://svnbook.red-bean.com/`.

2. svnX is a useful graphical user interface (GUI) front end to Subversion which can simplify using Subversion for users less comfortable with the command line interface (Subversion still needs to set through Terminal though). svnX is available at `http://www.lachoseinteractive.net/en/community/subversion/svnx/download/`. svnX runs directly from the disk image, on the first use however, you will need to go into the Preferences and change the "Path to svn binaries folder" to `/usr/bin` so that svnX can find your Subversion intallation.

3. To integrate Subversion with MATLAB you will first need to download the plugin from the MATLAB Central File exchange which can be found at `http://www.mathworks.com/matlabcentral/fileexchange/11596-subversion-interface-for-matlab`, this plugin is only for Mac OS X systems.

   Once you have download and extracted the zip file, you will need to edit the included MATLAB file "`customverctrl.m`". Open this file in a text editor and find the two instances of "`/sw/bin/svn`" (you can use the find tool to find these quickly) and replace them with just "`svn`", save and close the file. You will now need to place this file in the `verctrl` MATLAB directory, replacing the file already there (don't worry the original file is blank, you won't damage your MATLAB installation). This directory can be found by going into your applications folder in Finder, right clicking on the MATLAB icon and selecting "Show package contents" then looking under `/toolbox/matlab/verctrl/`.

   Once you have done this, open MATLAB, go into the Preferences (`File -> Preferences`), under `General -> Source Control` change the "Source Control Option" to "Custom", click apply and your system should be set up.

You will however, still need to set up your Subversion repository and upload your initial files in Terminal (explained in the guide mentioned in point 1).