# Introduction

Although Subversion (SVN) was originally created as a tool for software programmers to manage their source code, it can also be used for a variety of other purposes. For the Design Lab, users also apply SVN for collaboration and version control for:

* Word documents
* PowerPoint presentations
* SolidWorks / NX CAD files
* Electronic cad files
* Simulations
* And more!

A Subversion server and client are designed to organize and provide version control for your team’s files, and to ease sharing and updating them. The server software manages the repository. Both the server software and the repository reside on the Electronic Design Notebook (EDN) computer. The client runs on your PC and helps to link windows explorer and the repository.

There are several benefits to using version control software such as subversion. Because of these benefits, the use of subversion is preferable to the use of simple shared folders:

* Changes can be made by multiple users
* Changes can be tracked to each user
* Previous versions are saved and recoverable

For Capstone, we have found Subversion to be preferable than the other popular version control system, Git. Most students have never used any version control tool before taking Capstone Design. Subversion was chosen as it requires less steps to upload a file to an existing remote repository and it also eliminates the risk that users will forget to perform the final “push” step required by Git. Subversion is also more efficient when handling binary files such as CAD models. It seems that those who learn to use Subversion are very quickly able to later learn how to use Git.

While Git certainly offers a number of features not available in Subversion, we have found those to rarely be needed for our multidisciplinary Capstone program. The purpose of using these tools is to manage our files. It is NOT about teaching the specific tools.

# The Subversion Workflow

The general workflow when using a version control system can be seen in Figure 1 - The Subversion Workflow. As the team works, the portion in the red box is repeated continually over the course of a project. The remainder of the diagram is essentially performed only one time by others.

****

Figure 1 - The Subversion Workflow – Only the portion indicated is repeated throughout the semester

# Installation and Setup

This guide steps you through the process to link your team’s existing SVN repository on the server with a folder on your drive.

1. Download the Tortoise SVN Client at <http://tortoisesvn.net/downloads.html> and run the installer.
   1. You can verify that it installed successfully by right clicking on any folder on your desktop or in Explorer. You should see an item for TortoiseSVN.
2. Verify that you can access the repository (i.e. your team’s server).
   1. Right click anywhere in windows explorer, and select TortoiseSVN🡪Repo-Browser.
   2. It asks for a path. Enter “https://designlab.eng.rpi.edu/svn2/**[repository name]**”. The repository name is the unique identifier given to all projects. If you go to your project’s home page and look at the URL, it is of the form: [https://designlab.eng.rpi.edu/edn/projects/repository name](https://designlab.eng.rpi.edu/edn/projects/repository%20name)

In Figure 2 - Sample URL Showing the Project Identifier below, the project identifier is “sample-ied-fall-2014” so that is the repository name.



Figure 2 - Sample URL Showing the Project Identifier: sample-ied-fall-2014

* 1. You should have to enter your EDN username and password. Check on the “remember this password” box so that you don’t have to keep re-entering it.
  2. A window should appear (Figure 3 - TortoiseSVN Repository Browser Window) with two main panes: on the left is the folder hierarchy and on the right side are the files and sub folders of the currently selected folder from the left. Congratulations, SVN is working! This does NOT require VPN access to campus.  
     If this does not happen and/or you get an error, check with a team mate. If you both cannot access the repository, contact your Project Engineer.

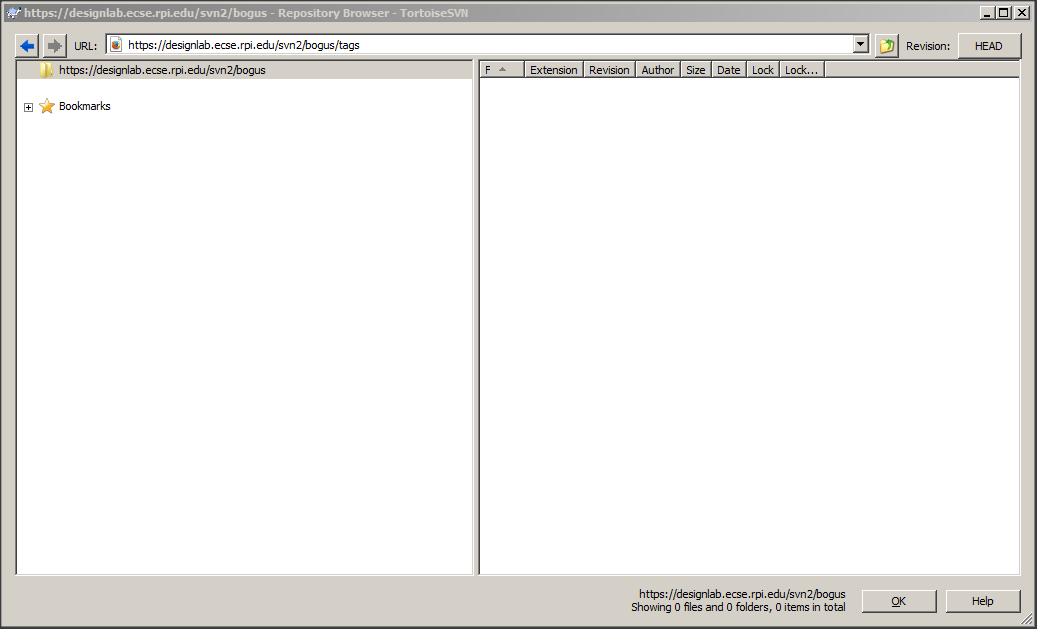


Figure 3 - TortoiseSVN Repository Browser Window

1. Back in Windows Explorer, create your own “working” folder somewhere on your hard drive. The folder can be called whatever you want. All of your files for Capstone will end up in this folder.
2. Right click on your new folder, and select “SVN Checkout”, just above the TortoiseSVN option. See Figure 4 - Tortoise Right Click Menu for reference

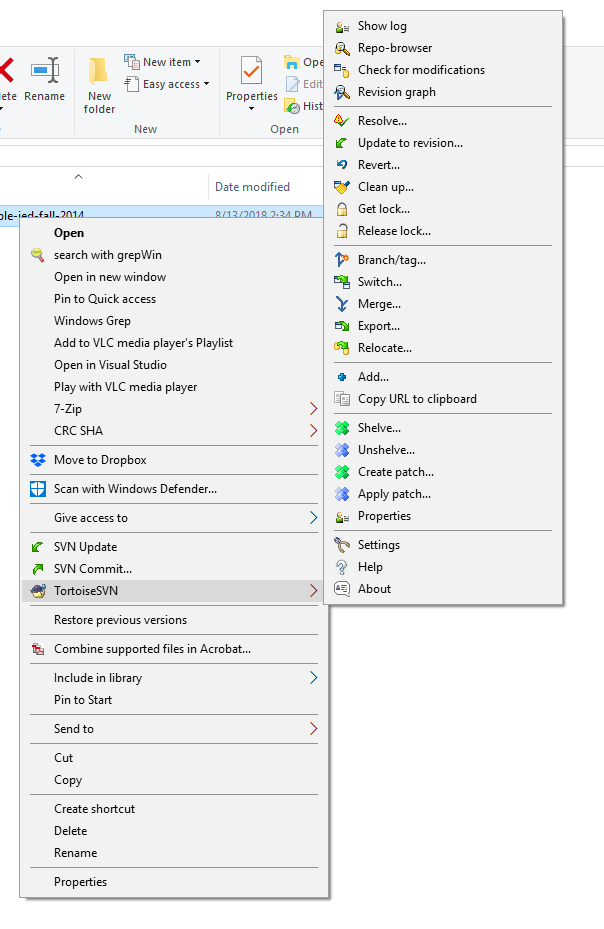


Figure - Tortoise Right Click Menu

1. In the “URL of Repository” line, enter in the location of the working folder on your repository. For instance “https://designlab.eng.rpi.edu/svn2/**[repository name]**/working”. For the sample project shown earlier this URL would be: https://designlab.eng.rpi.edu/svn2/**sample-ied-fall-2014**/working.  
   The checkout directory should be the path of your own working folder.  
   Checkout Depth: Fully recursive  
   Revision: Head
2. Click OK. If prompted, enter your EDN username and password. The server will take some time to download the history and files of the server to date. Click OK.
3. Your “working” folder might have a check-mark graphic on it now. If not, don’t worry. Double-click the folder. Inside you will see all the other folders and files that were within the Repo’s working folder, available for you to edit.

**DO NOT CREATE A REPOSITORY ON YOUR PC! There is only 1 and it’s on the server!!**

# Using Your Repository

In Windows’ Explorer, right click on a folder associated with your SVN server; you will see a few options.

SVN Update – This will update your computer with all the revisions that have been made to the server since the last time you updated. You should do an update before you start working on shared files – in case someone else has already modified them.

SVN Commit – This brings up options to commit any folder or files you have selected up to the server. You should provide a short comment for every commit so that it’s clear what the purpose is for each commit. If you are updating a batch of different files, it might be wise to update them individually or in chunks such that the revisions are easier to separate, and you can write a short, relevant note about each one. Be sure to only commit “good” versions of your files. Your team mates will be using them in their work! If you commit a file with known errors and a team mate updates that to their PC, they will have problems in their work.

TortoiseSVN Options – Most of these, with the exception of Repo-Browser, Show Log, and Clean-Up, are irrelevant to your needs. Read the Tortoise documentation online if you are really curious.

## Graphic Status Indicators

On Windows 8 and earlier, Tortoise marks files and folders with a graphic status indicator. Unfortunately, these apparently stopped working as of Windows 10 due to issues within Windows. On Windows 10, a right click, Properties will show a Subversion tab with useful information (Figure 4).

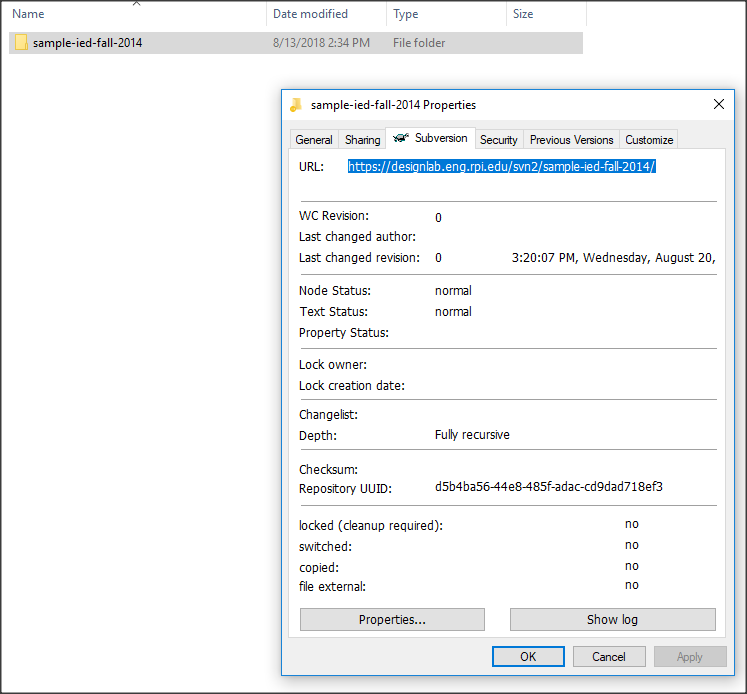


Figure - Right Clicking to see the Subversion tab on a folder's Properties

In Windows 7 explorer, the primary graphic indicators are:

* Check mark – Your folder and file set match those on the server.
* Exclamation point – Something about your folder and file set is different than the server. Commit what you have changed, *then* Update which will download whatever is new. It is ok if you accidentally do the update first as Subversion will NOT overwrite the files you changed! Simply do a “Commit” followed by another “Update”.

# Other Platforms

Because Subversion is a mature and widely used package, support is available for all of the platforms we use in Capstone. Since we are running a standard Subversion server, any compatible SVN client can be used. Many platforms include a command line client.

## Linux

On Ubuntu you may download and install RabbitVCS which is essentially a Linux implementation of Tortoise and works the same way.

## Mac

Neither Tortoise nor RabbitVCS is currently available for Mac however there are other free SVN clients. Mac users have been successfully working with svnx although there are occasional issues. A number of Mac users simply use the command line “svn” interface instead.

## Using Subversion at the Command Line

If you are accustomed to using Subversion at the command line, then you may continue to do that – you do NOT have to use a GUI. Note that when using the command line you need to first change to the proper directory! The key commands are:

* Checkout: svn co https://designlab.eng.rpi.edu/svn2/ied-sample-fall-/working
* Update: svn up
* List: svn list
* Add: svn add filename (must follow with a commit)
* Delete: svn delete filename (must follow with a commit)
* Commit: svn commit –m “log message, why are you committing this”
* Help: svn help

# Things to Avoid

The Capstone Support wiki page for Subversion contains all of the current guidelines but we include a few here to help you avoid some of the common errors.

* Do not use file or folder names that include version information. Subversion will be responsible for all version info – so you do not have to be! A given memo might be called “Memo on Topic X” but should not be called:
  + “Memo on Topic X – version 1”
  + “Memo on Topic X - Tuesday”
  + “Memo on Topic X 0 2014-08-26”
  + “Memo on Topic X – Mark’s Updates”
  + Etc.
* Always include all documents in their “source” format not just a PDF of the output. For example, **DO** include a PDF of a SolidWorks / NX CAD models but ALSO be sure to include the actual CAD files. **DO** include a PDF of something that you processed using a Finite Element Analysis package but ALSO include the raw data and program command files. Your own or the following team may want to reuse your information and/or edit it!
* Software folders should include all source code and “make” files or instructions on how to rebuild the program. For microcontroller types of programming be sure to include full instructions on how to obtain / load their development environment.
* Every folder should include a README file that **explains** what is in that folder and what software tools are required to access/use it. Don’t just list the files!
* Typically, you only need the working folder contents – not the branches or tags folders. For some projects you might not need all of the working folder – LOOK before you check out the entire folder!
* Do not create a *repository* on your own PC. The repository lives on the server. You will checkout a copy of the repository into a ***working*** copy that lives on your PC/Mac.

# Revision History

The original author was Capstone student Ian Keyworth, Class 2010, MECL. All subsequent revisions can be tracked in the capstone Support repository.