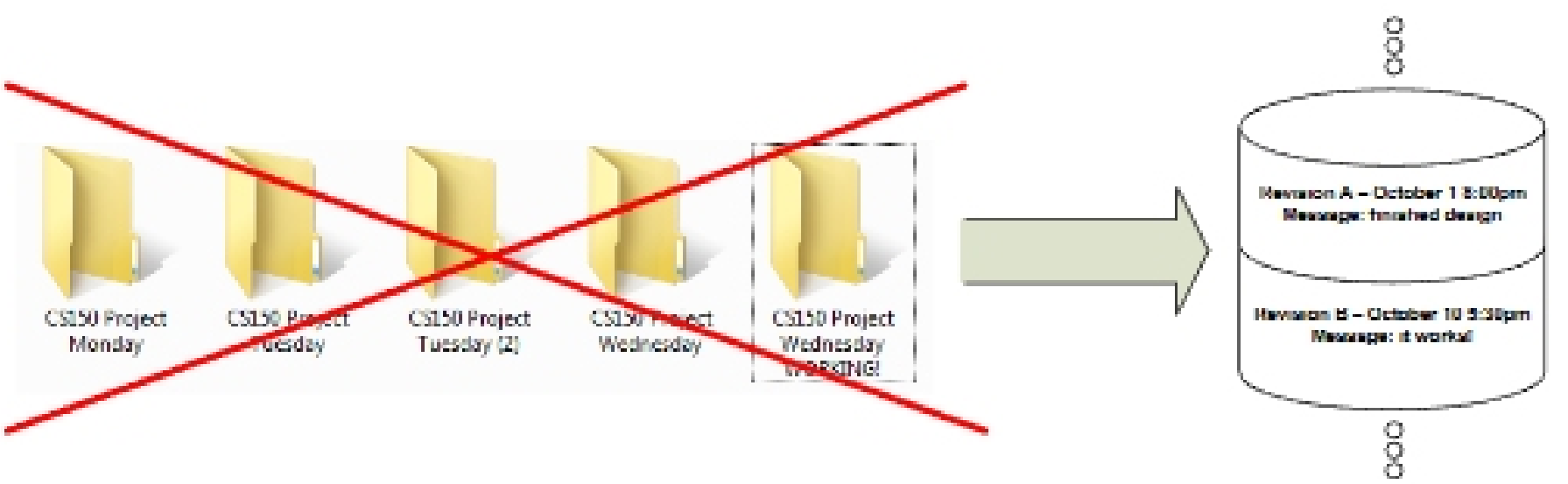
UNIVERSITY OF CALIFORNIA AT BERKELEY
COLLEGE OF ENGINEERING
DEPARTMENT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

Using Subversion (SVN)

Overview

Subversion (SVN) is a version control system that allows you to track every change (or revision) made to different files, compare a file from one revision to another, and even revert to an old revision if necessary. In EECS150, benefits to using SVN are that:

1. Your data will be safe.
   a. SVN is hosted on the inst server.
   b. Anything you store in SVN will be accessible on any computer.
   c. You won't have to worry about moving files back to your U:/ drive manually (and losing data from forgetting to do this).
2. You can store all of your work in one place.
   a. Never again will you have to make different copies of your code when you want to keep a revision history of them. See below:



3. You will be able to safely work on the same code with your project partner.
   a. If your partner breaks the project source, you will be able to see exactly what change introduced the bug (and eliminate it).
   b. Both you and your partner can modify the same file at the same time on different computers.
   c. If you decide that work done by either you or your partner made things worse, you can revert to any old version of whichever file at any earlier point in time.
4. You will get access to class library files
   a. Each project group directory in SVN has a copy of familiar files (`Counter.v`, `Register.v`, etc) inside them.

b. As we distribute new material, you will have access to it locally and immediately, without having to search through the website.

This document covers the following:

1. Important concepts and jargon related to SVN.
2. Using these concepts with an SVN client (SmartSVN).
3. Getting access to your EECS150 SVN repository.

Important Concepts and Jargon Related to SVN

SVN stores your data in a **repository** (or database) which is situated either on your personal computer or on a server. In EECS150, repositories are hosted on the inst server. This allows you to **check out** (or download) your code (or whatever you have saved in the repository) onto any computer you wish.

The code that you check out is known as your **working copy**. Your working copy consists of all of the files that you checked out onto your computer. The files in your working copy behave exactly like any other files on your computer. You may modify them just like any other files. The only difference between files in your **working copy** and ordinary files is that you can **commit** changes made to files in your working copy back to the repository.

When you **commit** changes made to your working copy, you save a new revision of those files for all time. At any time in the future, you can look back on the revisions you made to a.) see what changes have occurred from then to now, and b.) **revert** (or move back to) to an old revision if necessary. When you commit a change back to the repository, you can write a **commit message** describing the change you made. Whenever you get a module working, you should commit that module and type the message: "I got it working!" or similar. This allows you to easily see what revision to revert to in case you introduce a bug at some later time.

Before you commit changes made to your working copy back to the repository, the repository only has your data as it was last committed. This means that if both you and your partner have different working copies (which is normal as each of you will need to check out your own copy), you will potentially have different versions of the same file. To keep both working copies as in-sync as possible, you will **svn update** as soon as you start working on your working copy, every time you start work on your working copy. **Again, the first thing you should do whenever you sit down to work on SVNed files is run SVN UPDATE.** If you forget to run svn update and modify a file that your partner was also modifying, you might run into a **conflict** when you and your partner both commit your changes back to the repository. You can resolve conflicts my looking though the file and deciding which pieces of your work and which pieces of your partner's work you really want to keep. Run **svn update** often to avoid conflicts; know, however, that none of your work will be lost if you have a conflict.

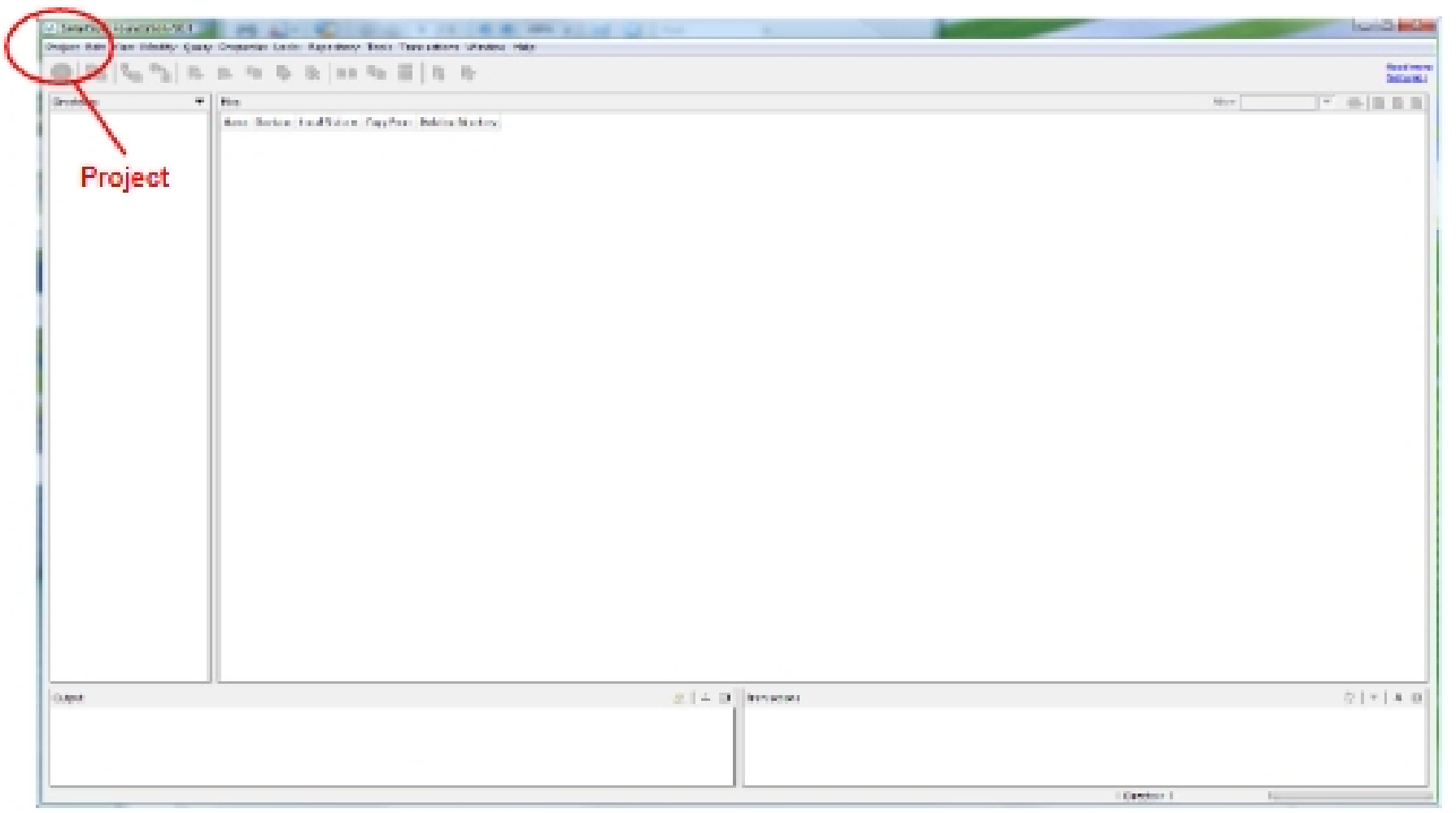
There are countless other features associated with SVN. The above are most important. Ask your TA if you have any questions.

Using an SVN Client: SmartSVN

Without a client program, you must work with SVN through the command line. This is extremely unwieldy and time-consuming, so we will now discuss how to use **SmartSVN**, an SVN client installed on all machines in 125 Cory.

1. From your desktop, open **Start→Programs→SmartSVN.**
   a. SmartSVN might be labeled with its version number (for example: "SmartSVN 4").
   b. The desktop icon for SmartSVN looks like this:

2. Open SmartSVN. You should see the following screen:



3. First, you must **check out** your part of the repository onto your local machine (this will be your **working copy**).
   a. Open **Project→Check Out…→Quick Checkout**
4. You should now be prompted with the **Check Out Project** dialog:
   a. For **URL** type:
   https://isvn.eecs.berkeley.edu/cs150/Project/Groups/<GROUP_NAME>