Change Management: Part 2

Week 4

Version Control

What is a Versioning System?

- The process of recording and being able to retrieve changes to a project
- Benefits
  - Any stored revision of a file can be retrieved, viewed, and changed.
  - The differences between any two revisions can be displayed.
  - Patches can be created automatically.
  - Multiple developers can work simultaneously on the same project or file without overwriting one another's changes.
  - The project can be branched to allow simultaneous development along varied tracks. These branches can be merged back into the main line of development.
  - Distributed development is supported across large or small networks. (subversion offers a variety of authentication mechanisms.)

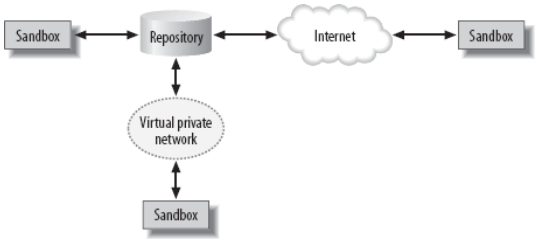
Terminology

- 'Repository' - Location on SVN server where versioned files are kept.
- 'Revision' - A certain version of all files in the project.
- 'Trunk' – The main project being developed (e.g. not a branch)
- 'Branch' - A 'fork' of the trunk.

Repository

- File-based database called the SVN repository
- Contains all historic data about your project
- Developers work from a sandbox which contains working copies of the files in your project
- Changes from a sandbox are committed to the repository
- Changes in the repository, but not in your sandbox, are updated from the repository
- Each file is associated with a revision number

Repository with Multiple Users



Creating Your First Repository

- Create a directory somewhere. This is where your repository will be and referred to as the ***repository_root_directory***
- Initialize the repoistory:
  - svnadmin create **repository_root_directory**
  - It creates a bunch of subdirectories in your repository
  - Remember you should never edit the repository directly

Importing Projects

- You have an existing projected called FooBar stored in a directory called FooBar-1.1
- From within FooBar-1.1, you import the project with the command:
  - svn import **directory_to_import** file://**repository_path**
- It'll open an editor and you can add notes to the SVN log regarding your changes/imports.
- SVN import will complete after you quit and save the log
- You can also import/access projects on other machines!

Checking Out Files

- SVN stores projects and files in a central repository, but you work from a working copy, the sandbox.
- SVN creates the sandbox as a subdirectory of your current working directory:
  - svn checkout file://***repository_path***

Editing Files

- Once you've checked out your files into a sandbox, you can edit them with your favorite editors.
- Some SVN functions won't work on binary files like MS Word files or JPEG images.

Committing Changes

- Changes to your sandbox are not synchronized with the repository until you run the **svn commit** command
- Its best run from the root directory of your sandbox and must be run within the sandbox
- When you commit, SVN examines each directory and subdirectory under the current working directory. It searches for files that it is tracking that have changes and commits all changes to the repository.

Updating Sandboxes

- **svn update** command checks your sandbox against the repository and downloads any changed files to the sandbox.
- There can be "conflicts" if you've changed a file in your sandbox and someone else has changed the same file in the repository

Adding Files

- To add a file to the a project in the repository
  - Create the file in your sandbox
  - svn add ***filename***
  - svn commit

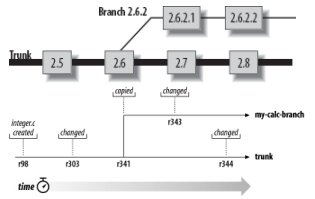
Removing Files

- To remove a file from the repository
  - Remove the file from the sandbox
  - svn remove ***filename***
  - svn commit

Comparing File Revisions

- You can call ***svn diff*** with filenames, directories, or module names.
- Usually, you call *svn diff* with at least one *-r* ***version*** specified. If you invoke it with a single *-r* ***N***, svn compares the current copy in the working directory with the version in the repository
- If you invoke *svn diff* with two versions in the *–r* option (e.g. *svn diff –r* ***N:M***), svn compares the two revisions against each other and generates a patch to go from ***N*** to ***M***.

Branching



- Branch is a forked line of development in your project, with the line that has been forked off called the branch, and the main line the trunk.
- SVN builds the branch and the trunk from the same source data, up until the point at which they diverge, which is called the *base* of the branch.
- From that point, SVN stores the changes made to the branch separately from the changes in the trunk.

Uses for Branches

- Variations on a Theme
- Bugfix management
- Experimental Work
- Major changes like complete code rewrite
- Release candidates for testing
- Standards compliance

Making a Branch

- Unlike CVS (svn's predecessor) branches are simply files in svn
- To make a branch, assuming your files are in the *trunk* directory, simply copy it: *svn copy trunk myBranch*

Add/Removing Files in a Branch

- When *svn add* or *svn remove* are applied to files in a branch sandbox, the addition or removal applies only to the branch and does not affect the trunk.

Merging Branches

- Merging a branch to the trunk applies the differences created during the life of the branch to the most recent revisions of the trunk code.
- Merging from branch to trunk:
  svn merge ***branch trunk***
- Merging from trunk to branch:
  svn merge ***trunk branch***
- Merging from branch to branch
  svn merge ***branch1 branch2***

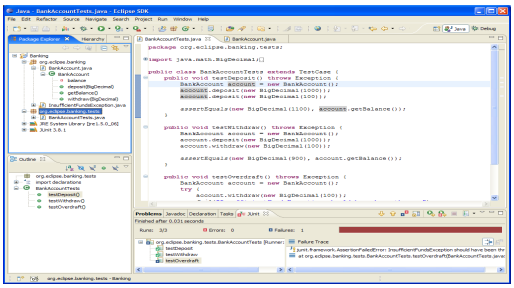
When svn Goes Bad

- Merge conflicts (e.g., two users editing the same line in the same file)
- Deleting files in the repository can be dangerous: next *svn update* a different user does could delete uncommitted changes

svn in History

- svn is slowly replacing its predecessor CVS
- svn is greatly simplified compared to CVS (be happy you're doing this lab now and not 10 years ago)
- svn is intended to be a complete replacement to CVS, so it uses almost exactly the same commands

Subversion with Eclipse



To Get Started

- Create an svn repository in your home directory
  - mkdir home/knoppix/svnrepository
  - svnadmin create /home/knoppix/svnrepository
- Let's say you want to add coreutils to your svn repository
  - Go inside the coreutils directory
  - svn import . file:///home/knoppix/svnrepository