SVN Starter's By Ankit Panwar

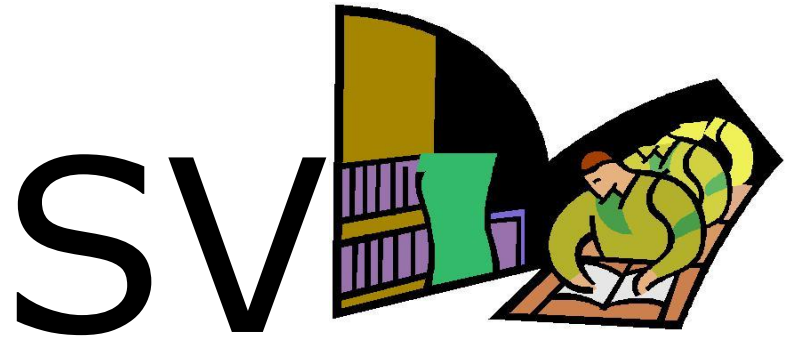


Table of Contents

1) What is SVN?

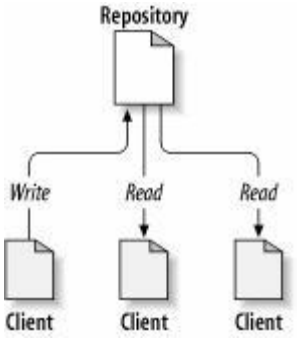
SVN is a centralized system for sharing information. At its core is a repository, which is a central store of data.

The following is a comparison between SVN and basic server file-sharing:

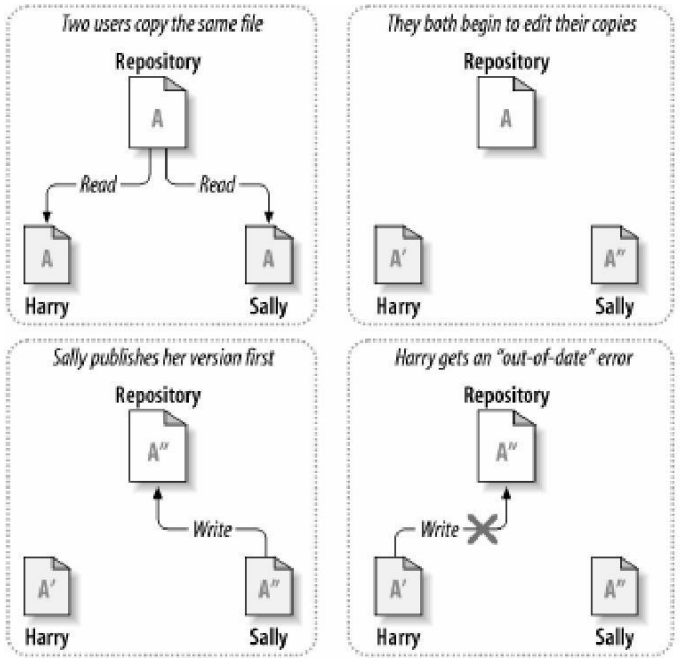
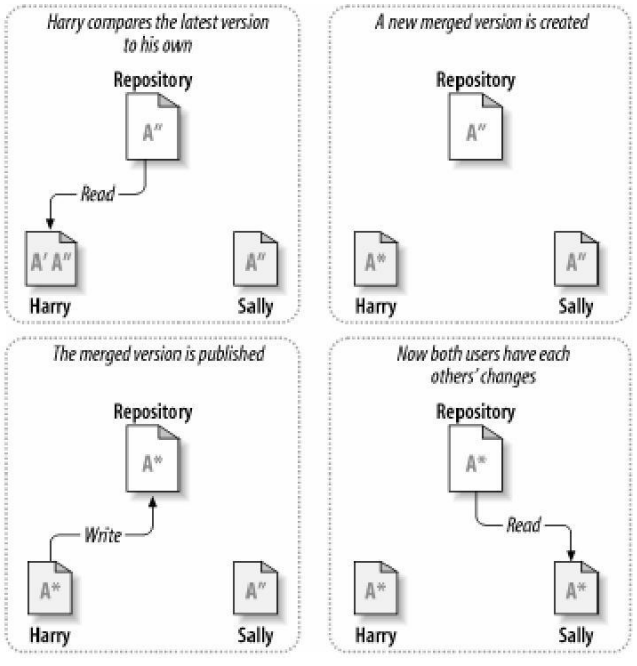
Basic server file-sharing	SVN
	
One person can use a file at a time; other users are locked out.	Any number of users can use a file at one time, and then read and write to these files.
Users can only view the current state of the file system; there is no record of changes to data.	Users can view previous states of the files system and track changes to data.
Users cannot tell who modified a file.	Users are notified when a file has been modified, and by whom.
Users do not comment on changes they make to files, or the file system; other users may have to manually review files in order to determine what has been modified.	Users comment on changes they make to files or the file system.
Users can only replace or copy whole files, or manually make changes to files.	Users can integrate changes to a file that is in the repository, to their own copy.

2) SVN Architecture

A typical client/server system



The Copy-Modify-Merge Solution

Part 1	Part 2
	

3) Creating a Working Copy

What is a Working Copy?
A working copy is a directory that is under version control.

Steps for Creating a Working Copy

1) Import the directory into the repository. If you want to import a directory called "Robot" into the "robby" repository, go one level outside the Robot directory and type:

svn import Robot http://svn:8080/repos/robby/Robot –m "your comment"

> Interpretation: You are importing files from a directory called "Robot," into the robby repository; the files will be in a newly created repository directory called "Robot." For –m "your comment," the comment should describe what you are doing; this will let other users know what modifications you made to the file or directory.
>
> Note: If you forget to type –m "your comment," then you must type 'i' to begin typing your comment, and then press "Esc" Shift-zz when you are finished commenting.

The Robot project is now in the repository, but you do not yet have a working copy—it must be checked-out first.

2) To checkout "Robot" from the repository, you should be in the directory where you want your working copy, and type:

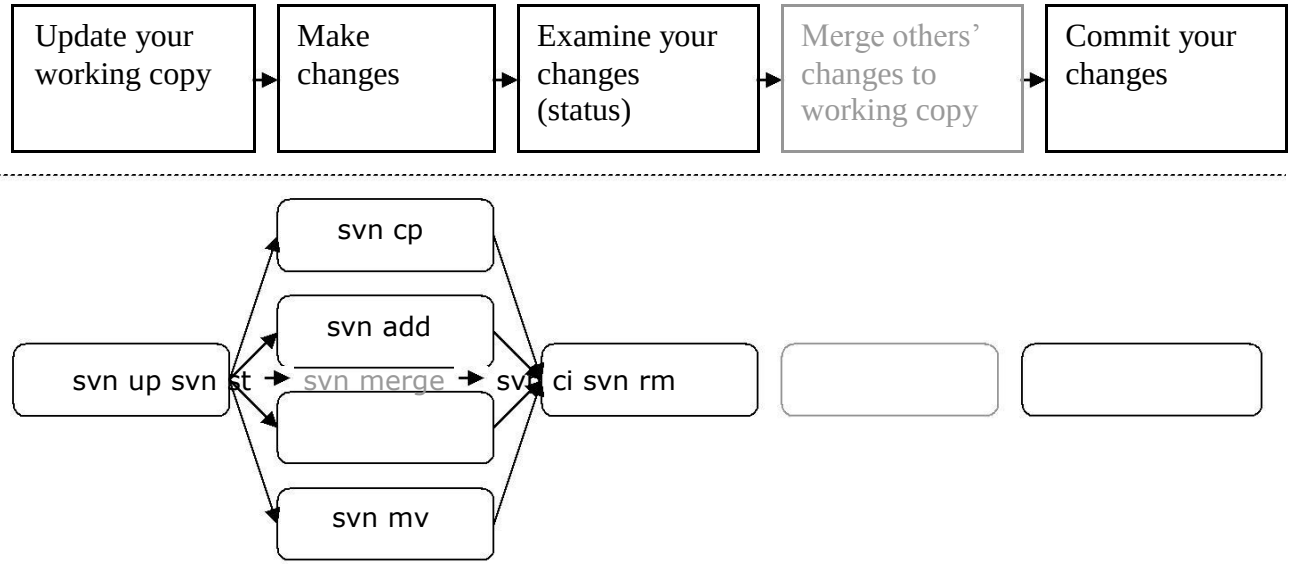
svn co http://svn:8080/repos/robby/Robot Robot

> Interpretation: You are checking out "Robot" into a newly created folder called "Robot" (your working copy); it is now under version control.

You have just created a working copy of "Robot."

4) Basic Work Cycle

This is the basic work cycle you will follow once you have a working copy.



For example, if you would like to add a file to the repository, go to your working copy directory and type:

1) svn up
2) svn add filename
3) svn st
4) svn ci –m "your comment"

Merging is discussed in Section 7, "Extra Commands."

Note:

- If you are in a working copy, you can list the files that are under version control by typing "svn ls".
- To list what is in the robby repository outside of the working copy, type "svn ls http://svn:8080/repos/robby."
- For remote access to the repository, you may need to use port 8888.
- When you add a file to the repository, it will not appear in the repository file list until you run an update (svn up).

5) Status Symbols

Example of What You Will See

Type: svn status	svn status --verbose	svn status -u --verbose
A abc.c M bar.c D baz.c	A 44 23 Sally abc.c M 44 20 Harry bar.c D 43 35 Harry baz.c	A * 44 23 Sally abc.c M 44 20 Harry bar.c D * 43 35 Harry baz.c

Show Updates (-u) Notes: The second column shows the working-revision of the item. The third and fourth columns show the revision in which the item last changed, and who changed it.

First Column	Indicates items were added, deleted, or changed.
A	Item is scheduled for Addition.
D	Item is scheduled for Deletion.
M	Item has been modified.
C	Item is in conflict w/ updates received from repository.
X	Item related to an externals def.
I	Item is being ignored.
?	Item is not under version control.
!	Item missing / co interrupted.
~	Versioned as dir., replaced by file.

Second Column	Tells the status of a file's or dir.'s properties.
M	Properties for this item have been modified.
C	Properties are in conflict with updates from the repo.

Third Column	Populated only if working copy dir. is locked.
L	Item is locked.

Fourth Column	Populated if item sched. for addition-w/-history.
+	History scheduled with commit.

Fifth Column	Populated if item switched relative to its parent.
S	Item is switched.

Eighth Column	Out-of-date info. (use --show-updates switch)
*	A newer revision of the item exists on the server.

6) Important Notes

1) Think before deleting a file that is under version control!

When you delete a file from your working copy using "svn rm", that file will be removed from the repository, and from the working copy folder.

Solution 1: Remove the file using the full file path (e.g., svn rm http://svn:8080/repos/robby/filename).

Solution 2: Create a folder within the working copy, where you can keep a copy of the file; you can delete this file without causing disruption to the group.

2) svn import Robot http://svn:8080/repos/robby/Robot –m "your comment"

Include the name of the repository folder that your files will be imported into, even though it does not yet exist; the folder will be created automatically once you enter the command.

3) Do not add media files (e.g., pictures and movies) to the repository—it is unnecessary and takes up repository space.

4) svn import –- Recursively commits a copy of PATH to URL.

5) If you remove a repository file without typing "svn rm" or "svn del," then the "!" status symbol will appear next to the filename when you run a status check (svn st). This is because the file still exists in the repository, but not in your working copy.

Solution: Type "svn rm filename." The file will then be scheduled for deletion from the repository. When you run the status check this time, the 'D' status symbol will appear next to the filename.

7) Extra Commands

svn log — Displays commit log messages.

```
svn log type this if you are within the current working copy. svn
log [PATH] type this if you are outside of the working copy. svn
log –r 24 type this to see a log for a specific revision.

*24 is the revision number.
```

svn cleanup — Recursively cleans up the working copy.

```
svn cleanup [PATH...]

"Recursively cleans up the working copy, removes locks, and resumes
unfinished operations. If you ever get a 'working copy locked' error,
run this command to remove stale locks and get your working copy into
a usable state again."
```

svn di — Displays the differences between two paths.

```
svn di -r N:M [URL]
svn di [-r N[:M]] [--old OLD-TGT] [--new NEW-TGT]
[PATH...] svn di [-r N[:M]] URL1[@N] URL2[@M]

For example, typing svn di –r 21:24 abc.txt, will show any
differences between revision 21 and 24 for the file abc.txt.

To compare the BASE and your working copy, type: svn di abc.txt.

Note: svn di is the same as svn diff.
```

svn resolved — Removes "conflicted" state on working copy files or directories.

If you get a conflict on an update, your working copy will sprout three new files:

```
$ svn ls
filename.c
filename.c.mine
filename.c.r30
filename.c.r31
```

*You can also remove the conflict files and then commit (svn ci).

svn lock — Locks a file. svn unlock — Unlocks a file.

8) Switches

"While Subversion has different switches for its subcommands, all switches are global—that is, each switch is guaranteed to mean the same thing regardless of the subcommand you use it with."

--force

> Forces a particular command or operation to run. There are some operations that Subversion will prevent you from doing in normal usage, but you can pass the force switch to tell Subversion "I know what I'm doing as well as the possible repercussions of doing it, so let me at 'em". This switch is the programmatic equivalent of doing your own electrical work with the power on—if you don't know what you're doing, you're likely to get a nasty shock.

--non-recursive (-N)

> Stops a subcommand from recursing into subdirectories. Most subcommands recurse by default, but some subcommands—usually those that have the potential to remove or undo your local modifications—do not.

--recursive (-R)

> Makes a subcommand recurse into subdirectories. Most subcommands recurse by default.

--show-updates (-u)

> Causes the client to display information about which files in your working copy are out-of-date. This doesn't actually update any of your files—it just shows you which files will be updated if you run svn update .

--verbose (-v)

> Requests that the client print out as much information as it can while running any subcommand.

9) Glossary of Basic Commands

svn checkout or svn co

This command is used to pull a directory from the repository from the server, to create a working copy.

svn add

When you are creating a new file or directory, you need to tell the SVN server about it. This does that. Note that the file won't appear in the repository until you do a svn commit.

svn delete, svn del, or svn rm

This does what it says! When you do an svn commit (svn ci) the file wil be deleted from your local sandbox immediately as well as from the repository after committing.

svn status

This command prints the status of working directories and files. If you have made local changes, it'll show your locally modified items. If you use the --verbose switch, it will show revision information on every item. With the --show-updates switch, it will show any server out-of-date info.

You should always do a manual svn status --show-updates before trying to commit changes in order to check that everything is alright and ready to go.

svn update or svn up

This command syncs your local sandbox with the server. If you have made local changes, it'll try and merge any changes on the server with your changes on your machine.

svn commit or svn ci

This command recursively sends your changes to the SVN server. It will commit changed files, added files, and deleted files. The -m option should always be used to pass a log message to the command. Please don't use empty log messages (see later in this document on the policy which governs them).

svn move SRC DEST or svn mv SRC DEST

This command moves a file from one directory to another or renames a file. The file will be moved on your local sandbox immediately as well as from the repository after committing.

10) Reference

1. Collins-Sussman, B., Fitzpatrick, B. W., & Pilato, C. M. (2004). Version Control with Subversion. Retrieved June, 2005, from http://svnbook.red-bean.com/