Lecture 1 - Version Control

Two types of Version control:

1. Centralized — Code is stored online and maintained/kept at one place
2. Decentralized — Stored at several places

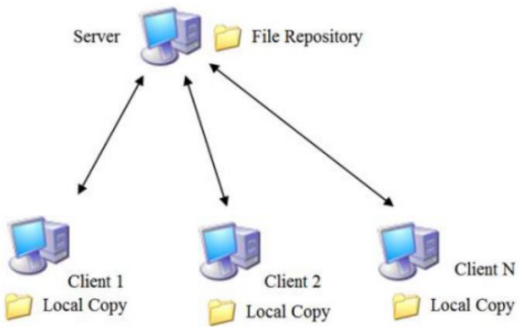
Centralized Version control :

Kept at centralized location (Repositories)

This copy is called master Copy

To make changes we have to get a copy on your machine(Client machine) — Called working copy

When changes are made on local copy, "Commit " the change the repository



Question 1 - what is every client makes a commit which one is sent to repositories?

That is called conflict. (Version control can resolve the conflict to some extend)

Terms to remember

Repository/ Repo

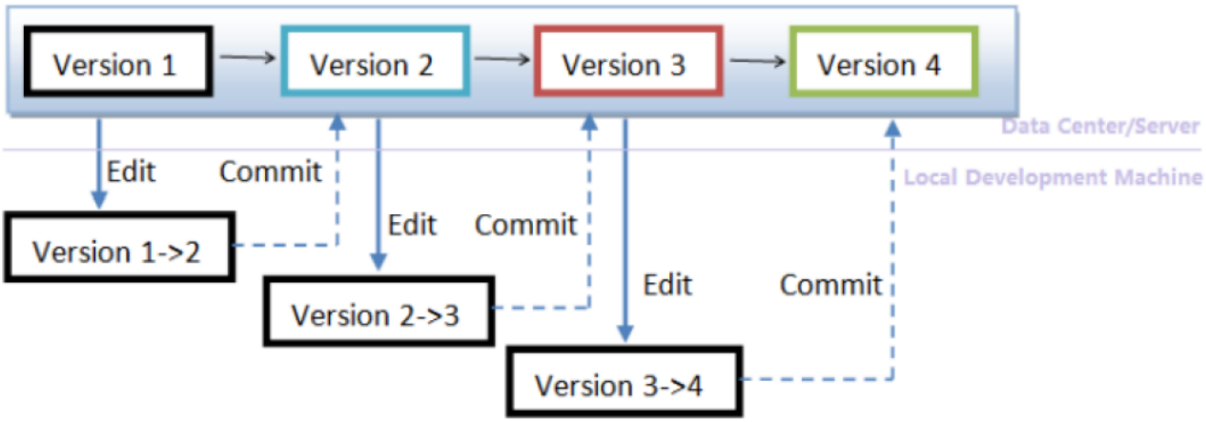
Client program (Clients working)

Working Copy (Copy you get of the latest repo)

Checkout

Commit (Sending the changes to repo)

Tracking Changes when working alone



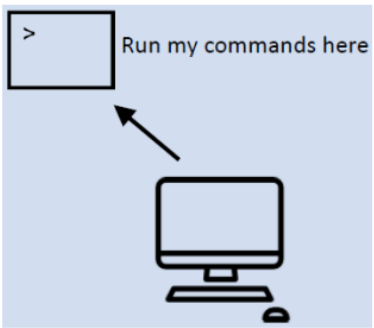
Sub VersionN (SVN)

- SVN is the successor to Concurrent Versions System (CVS), and was built to help fix many issues in CVS
- Other source control systems include: Git, Mercurial, ClearCase, Perforce, etc

Not VCS

Secure Shell (SSH)

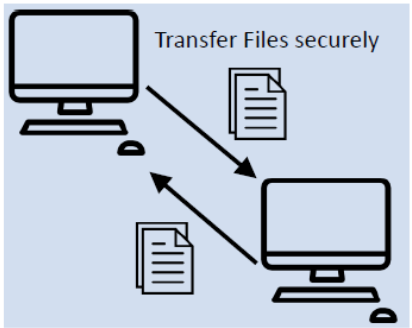
- Connect to remote computer and work in a shell on that computer



Run my commands here

Secure CoPy (SCP)

- Way to securely copy files from one computer to another
- Transfers a copy of the files
- Does not version files

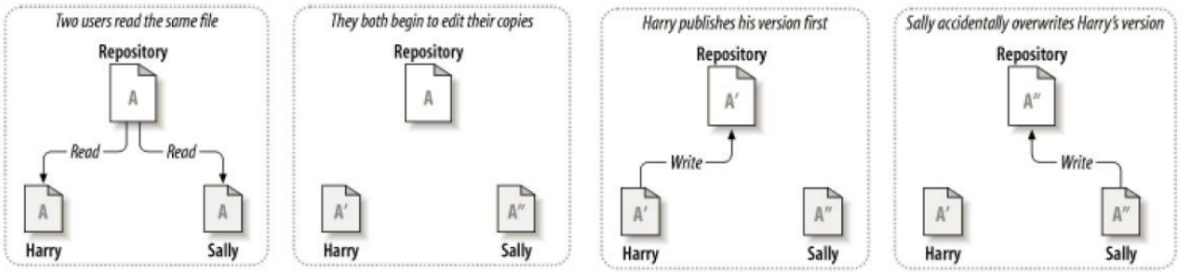


SVN vs GIT

SVN	Git
It's a Centralized version control system	It's a distributed version control system.
It is revision control.	Git is an SCM (source code management).
It does not keep a cloned repository.	It has a cloned repository.
Branches in SVN are a folder that takes place in the repository. Some special commands are required For merging the branches.	The Git branches are familiar to work. The Git system helps in merging the files quickly and also assists in finding the unmerged ones.
It has an internationalized revision number.	It does not have a Global revision number.
SVN does not contain any cryptographically hashed contents.	It has cryptographically hashed contents that protect the contents from repository corruption taking place due to network issues or disk failures.
SVN stores content as files.	Git stored content as metadata.
SVN's content is less secure than Git.	Git has more content protection than SVN
SVN's content is less secure than Git.	Git has more content protection than SVN.
CollabNet, Inc developed SVN.	Linus Torvalds developed git for Linux kernel.
SVN is distributed under the open-source license.	Git is distributed under GNU (General public license).

Managing Concurrency

What if two or more people want to edit the same file at the same time?



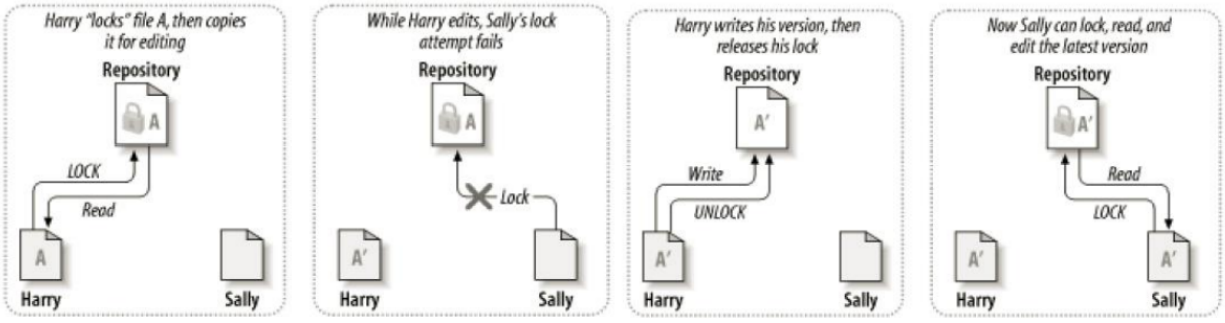
Option 1: Prevent it

- Only allow one writeable copy of each file
- Known as pessimistic concurrency
- E.g. Microsoft Visual SourceSafe, Rational ClearCas

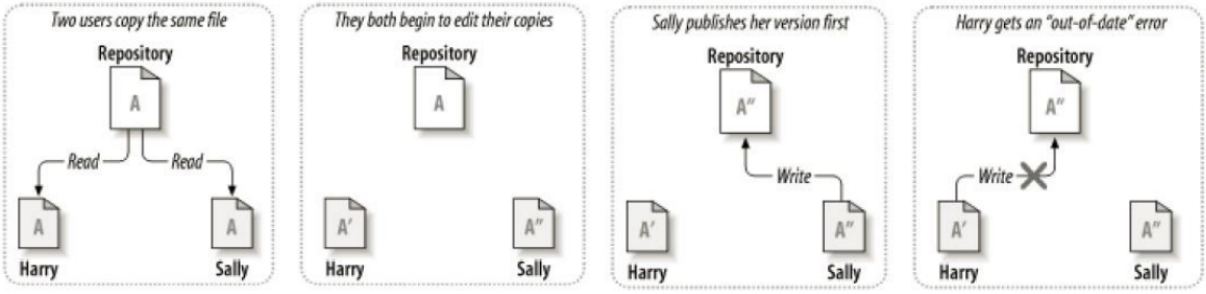
Option 2: Allow it, fix issues afterwards

- Optimistic concurrency
- E.g. Subversion, CVS, Perforce

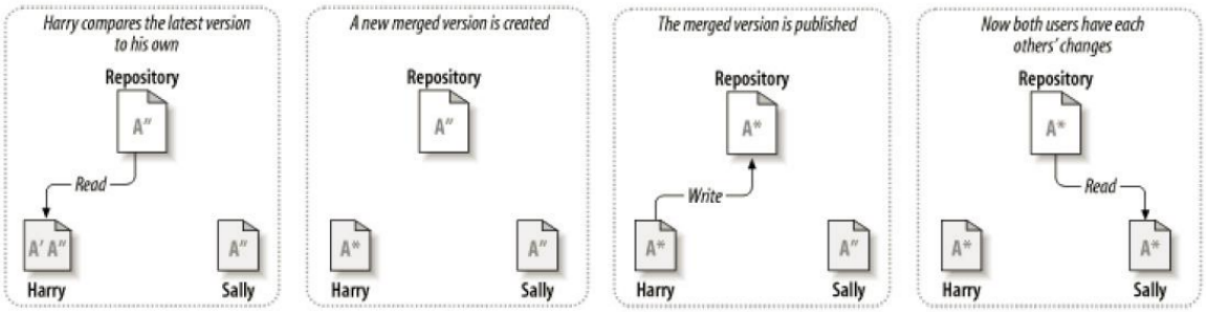
Pessimistic Concurrency



Optimistic Concurrency (1/2)



Optimistic Concurrency (2/2)



Optimistic Concurrency - Merging

Two Scenarios-

1. SVN is able to merge without help from user
2. Conflict: SVN needs the user to resolve the conflict

Options

```
Select: (p) postpone, (df) diff-full, (e) edit,
    (mc) mine-conflict, (tc) theirs-conflict,
    (s) show all options: s

(e)  edit          - change merged file in an editor
(df) diff-full     - show all changes made to merged file
(r)  resolved      - accept merged version of file

(dc) display-conflict - show all conflicts (ignoring merged version)
(mc) mine-conflict    - accept my version for all conflicts (same)
(tc) theirs-conflict  - accept their version for all conflicts (same)

(mf) mine-full     - accept my version of entire file (even non-conflicts)
(tf) theirs-full   - accept their version of entire file (same)

(p)  postpone      - mark the conflict to be resolved later
(l)  launch        - launch external tool to resolve conflict
(s)  show all      - show this list
```

Integrating the code

What causes merge conflict:

Communication issue

Complex code bases

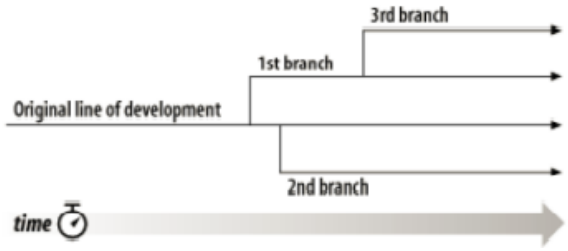
Experimental features being built

Two features being built in the same class be different developers

Branching (VCS)

Branching strategies include

1. No branching
2. Feature Branching - (Not sure if the code will end up in the main branch, if successful then)
3. Release Branching - (A release branching strategy involves creating a branch for a potential release that includes all applicable stories.)



Storage Scheme

• Storing every copy of every file we generated over the course of a project is not practical

• Version control systems store incremental differences in files/folder structures

  • These differences store enough information to re-construct previous versions, without storing every single copy ever made of the file.

What is Stored where

Server Side: This is out of the scope of this course
• Your local copy contains a special directory .svn
• It stores (locally) the information subversion needs to keep track of your files, version numbers, where the repository is, etc.
• Needless to say, you should not mess with the contents of this directory. Let subversion do its job.

General Rules

1. Update and commit frewuently
2. Never break main branch
3. always comment changes made by you
4. Test all code before accepting merge
5. Communicate with your team

Commands

 SVN Commands

https://cscb07-tips.github.io/