Distributed Version Control
Who? When? Why?

Leon Tabak
l.tabak@ieee.org

Cornell College

12 September 2017

Version control is a means of...

- simply and safely preserving writing projects
- simply and safely moving writing projects
- collaborating (sharing) with teammates
- recovering from errors
- understanding what happened before, so that we can wisely add to the work that has already been done

Synonyms

- RCS (Revision Control System)
- SCM (Source Code Management)
- VCS (Version Control System)

Version control works by...

- keeping a record of which files belong to a project
- keeping a record of which files were changed at each step in project's development
- keeping a record of which lines in a changed file were changed

Challenges of doing this on our own

- a project may contain many documents
- a document may require many files
- any system of tagging files with version numbers is likely to be complicated
- my way surely differs from yours
- lots of potential for tragic mishaps if one of us fails to follow any detail of the discipline!

Change sets

- each element tells us. . .
  - who?
  - when?
  - what or why?
- set shows ordering of changes
- picture could be a single straight sequence of line segments, or parallel lines, forks, and joins (like railroad tracks)

History

- a list of changes
- a means to go back in time by un-doing changes
- a means to go forward in time by re-doing changes
- original document + ordered list of all changes makes a smaller archive than a collection of all versions of the document!
- a collection of all versions might contain all of the information that we need to construct a history, but that info might be very hard to find, assemble, and use!

Distributed version control

- everyone has a complete copy of the project
- everyone can work online or offline (anywhere, anytime)
- everyone can work concurrently
- no single point of failure

Actions

- init
- add <filename>
- commit -m <description of change>
- status
- log
- update

More actions (especially important for teams)

- fork
- clone
- push
- pull
- pull request
- merge

Merging files

Two authors may simultaneously edit copies of the same document. They (or a third) person can later merge the two updated versions of the original document.

- software merges the changes automatically where only one author has changed a given line
- merging requires a person to make a choice where the authors have each changed the same line

Branches

A writer or team of writers might wish to develop and maintain several branches, where each branch has its own history.

- development
- testing — periodically merged with latest development version
- release — periodically merged with latest testing version

Distributed version control systems

There are two popular distributed version control systems. Both are free and available for the Windows, Macintosh, and Linux platforms.

- Git (https://git-scm.com/)
- Mercurial (https://www.mercurial-scm.org/)

Hosting services

- Bitbucket (https://bitbucket.org/product)
- GitHub (https://github.com/)

Why Mercurial?

- simpler
- works better with Windows (support from the start)
- Unix philosophy: each command should do one thing and do it well
- simplicity makes it easy to find the right command for the task at hand
- designers gave priority to clarity (rather than cleverness)
- deliberately hard to alter project's history and data (safer?)
- popular GUIs (TortoiseHg)

Why Git?

- more popular
- commands have many options, interact in complex ways
- experts can accomplish much with concise expressions
- easy to rewrite project's history (clean up with clearer ordering and explanation)
- made for Linux
- better branching

Caveat

- version control systems are designed to work with text files
- word processing programs typically store files in a binary (non-text) format
- "save-as" option might allow you to store files in a text (e.g., XML) format
- version control systems cannot merge changes in binary files, but can still provide other useful functions

Related tools

Some other widely used software works with Mercurial and Git.

- Software development — Integrated Development Environments (IDEs) including Eclipse and Netbeans
- Project management — including Jira (from Atlassian, same company that created Bitbucket)

How to get started

1. on your own small project (no team), on one computer
2. on your own small project (no team), on your computer and a cloud-based service
3. on a shared project with a partner, on your two computers and your two accounts with a cloud-based service
4. on a large collaboration, set up your own server, in open source context, etc.

Take-aways

- software engineers use version control software
- software engineers have moved to distributed version control in this decade
- people in other professions who also work with documents could find advantages in the use of version control software
- you can use free version control software in personal projects
- what you have learned here might help you guide your organization's efforts to improve its methods of managing changes in documents