Universität Zürich[UZH]

Version Control Systems: Git and SVN

A practical guide

Livio Sgier (livio.sgier@uzh.ch)

Outline

- Motivation/Problem Description
- Version Control System (VCS)
  - Git
  - SVN
- Git Live Presentation
- Useful Links

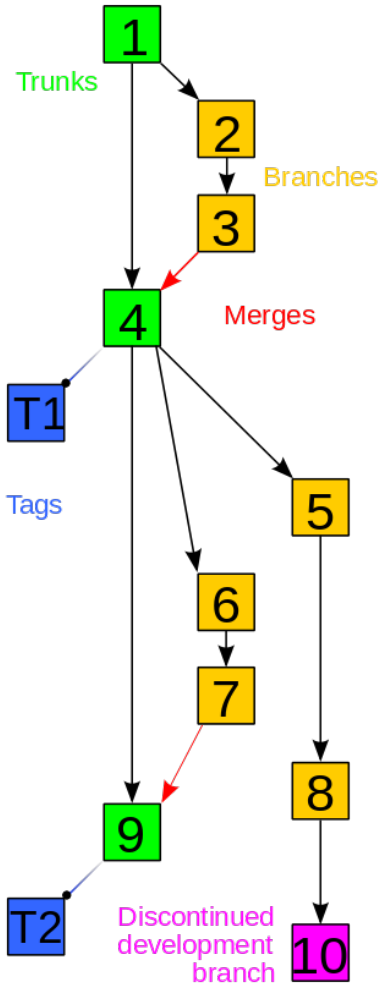
Problem description

- Motivating example
  - 5 people work on a software project
  - Source files are stored on a cloud system (e.g., Dropbox)
  - Files are edited by multiple people
  - How can we know:
    - Who did edit what?
    - How do we know what was edited?
    - What happens if two people work on the **same file** at the **same time**?
    - How can we go back in time?
    - How can we plan releases?

      → We can‘t do any of that!

What are Version Control Systems?

- Management of changes to information
- Features
  - Distributed Development (local repository copies)
  - File locking vs. version merging (first check-in succeeds)
  - History Preservation
  - Branching
  - Tagging
- Most notable systems (for our purpose „equal“ capabilities)
  - Git
  - SVN (Apache Subversion)



Source: wikipedia.org

Git

- Original author: Linus Torvalds
- Decentralised revision control model
- Most widely used VCS in industry, GitHub uses Git
- Graphical User Interfaces
  - GitHub Desktop (Windows, Mac)
  - GitX-dev (Mac)
- Live Demo introduces basics from installation to development

SVN (Apache Subversion)

- Maintained by Apache Software Foundation
- Centralized revision control model
- No equvalent to GitHub for hosting repos
- Graphical User Interface
  - TortoiseSVN (Windows)
  - SmartSVN (Windows, Mac, Linux)
- Comparable workflow (for what we use it)
  - svn checkout
  - svn status
  - svn add
  - svn commit
  - svn log
  - … etc.

Live Git Demo using command-line (1)

- Summary
  - Create repository on GitHub
  - git clone (clone the repo on the local disk)
  - git status (check for changes)
  - git add (adding changes)
  - git commit (-m „Comment“, every commit generates a hash)
  - git log (shows past commits)
  - git pull (sync with remote repo, always perform before starting work)
  - git push (sync with remote repo, always perform after work)

Useful links

- **https://try.github.io/** (interactive GitHub tutorial)
- **https://git-scm.com/doc** (good documentation)
- **https://en.wikipedia.org/wiki/Comparison_of_Subversion_clients** (SVN GUI Clients)
- **https://git-scm.com/downloads/guis** (Git GUI Clients)