Embedded computing for scientific and industrial imaging applications

Lecture 2 - Intro to version control systems

HanByul Yang
(Senior Engineer @ Samsung Medison)

For beginners

Learn DVCS

Distributed Version Control System

http://www.slideshare.net/ibare/dvcs-git

Outline

- Version control — main ideas
- Client-server version control, e.g., CVS, Subversion
- Distributed version control, e.g., git, Mercurial

Version control systems

Originally developed for large software projects with many developers.

Also useful for single user, e.g. to:

- Keep track of history and changes to files,
- Be able to revert to previous versions,
- Keep many different versions of code well organized,
- Easily archive exactly the version used for results in publications,
- Keep work in sync on multiple computers.

Server-client model

Original style, still widely used (e.g. CVS, Subversion) One central repository on server.

Developers' workflow (simplified!):

- Check out a working copy,
- Make changes, test and debug,
- Check in (commit) changes to repository (with comments). This creates new version number.
- Run an update on working copy to bring in others' changes.

The system keeps track of diffs from one version to the next (and info on who made the changes, when, etc.)
A changeset is a collection of diffs from one commit.

Server-client model

Only the server has the full history.
The working copy has:

- Latest version from repository (from last checkout, commit, or update)
- Your local changes that are not yet committed.

Note:

- You can retrieve older versions from the server.
- Can only commit or update when connected to server.
- When you commit, it will be seen by anyone else who does an update from the repository.

Often there are trunk and branches subdirectories.

Git

Created by Linus Torvalds in 2005 for development of the Linux kernel, with other kernel developers contributing to its initial development. - wikipedia

When you clone a repository you get all the history too,
All stored in .git subdirectory of top directory. Usually don't want to mess with this!

Ex: `$ git clone https://github.com/CSE6000/Fall2016 mydirname`

will make a complete copy of the class repository and call it mydirname. If mydirname is omitted, it will be called Fall2016.

This directory has a subdirectory .git with complete history.

DVCS (Distributed version control system)

Git uses a distributed model:

- git commit commits to your clone's .git directory.
- git push sends your recent changesets to another clone by default: the one you cloned from (e.g. bitbucket), but you can push to any other clone (with write permission).
- git fetch pulls changesets from another clone by default: the one you cloned from (e.g. bitbucket)
- git merge applies changesets to your working copy

Note: pushing, fetching, merging only needed if there are multiple clones.

Distributed version control

Advantages of distributed model:

- You can commit changes, revert to earlier versions, examine history, etc. without being connected to server.
- Also without affecting anyone else's version if you're working collaboratively. Can commit often while debugging.
- No problem if server dies, every clone has full history.

For collaboration will still need to push or fetch changes eventually and git merge may become more complicated.

GitHub - https://github.com

A web-based Git repository hosting service.
Provides a Web-based graphical interface. It also provides access control and several collaboration features such as bug tracking, feature requests, task management, and wikis for every project.

Almost everyone who works with computers must use version control system (Git and GitHub).

GitHub is not only development tool for programmers, but also SNS like facebook or instagram.
You can share your projects, follow celebrities and your friends. Most of users uses it for their codes but no restriction for any types of files.

Many open sources projects use it, (ex. Linux kernel, python)
Famous open source embedded computing projects use github,

e.g. • Rasberry Pi, Arduino

GitHub : Demo

Class repository : https://github.com/CSE6000/Fall2016

https://github.com/torvalds/linux

Other tools for git

The gitk tool is useful for examining your commits.

This will be installed on the PC, after Installing Git for windows.
Usage :

```
$ gitk
```

Many other tools, e.g.

- GitHub Desktop
- SourceTree
- TortoiseGit

Shell - command line interface (CLI)

- UW AMATH 483/583 class notes : Shell

Some important ones...

- `cd, pwd, ls, mkdir, mv, cp`

Commands are typed into a terminal window shell.

We will use “Git bash”. Prompt will be denoted $, e.g.

```
$ cd ..
```

Demo : Git for Windows

- $ git config --global user.name "name"
- $ git config --global user.email "email"

- $ mkdir ~/MyProject // make local directory
- $ cd ~/myproject // change directory
- $ git init // make git repository
- $ git status // check status of git repo
- $ git add filename // stage file
- $ git add . // stage all file including subdirectory
- $ git commit -m “comment” //

- $ git remote add origin https://github.com/username/myproject.git
  // connect
- $ git remote -v // verbose remotes
- $ git push origin master // push commits to remote “origin”

Demo : Git bash

Links

- Atlassian GIT tutorials
- Learn Git Brancing

- How to Use Git and GitHub @UDACITY

- Try Git: Git Tutorial
- GitHub Guides - Hello world
- GitHub Help
- Good Resources for Learning Git and GitHub