Revision control

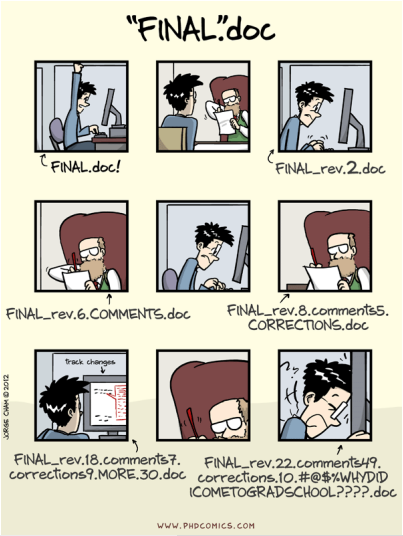
Marcello Vichi

Department of Oceanography
marcello.vichi@uct.ac.za

AOS

Outline

- You don't need to be a software engineer to appreciate the importance of maintaining different revisions of various objects. RCS have developed in the 80's for sequential and concurrent code writing, but are now being expanded to handle multiple types of files
- Revision Control is not just for BIG CODES (Source Code Management), but also for routine activities, especially in a shared professional environment
- The material in this section is adapted from http://swcarpentry.github.io/git-novice/



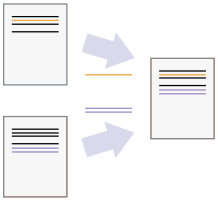
The concept

- Revision control systems (RCS) start with a base version of the document and then record changes you make each step of the way. You can think of it as a recording of your progress: you can rewind to start at the base document and play back each change you made, eventually arriving at your more recent version
- The terms **version** and **revision** are often used as synonyms. They have technical differences in software engineering. A version is a more generic term to indicate changes in a file or a set of files. Technically, it is often associated with **releases** of a software or document (e.g. v1.2). Revision is any recorded change or combination of changes.

- Once you think of changes as separate from the document itself, you can then think about "playing back" different sets of changes on the base document, ultimately resulting in different versions of that document.

Concurrent versioning, committing and merging

- Two users can make independent sets of changes on the same document. This operation creates two **concurrent** versions. A change to a file is a temporary modification in RCS: saving the file is not sufficient to acknowledge the generation of a new version. This further operation is called **committing** a version
- If the changes have been done to different parts, they are **complementary**, and you can incorporate two sets of changes into the same base document
- If the changes have been done to the same section of the document they generate a **conflict**.

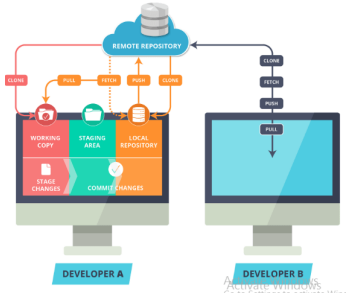


- The operation of combining the different committed versions is called **merging**. In the case of conflicts it can be a very complicated process.
- A RCS is a tool that keeps track of these changes for us, effectively storing different versions of our files. It allows us to decide which changes will be made to the next revision (each record of these changes is called a **commit**), and keeps useful metadata about them (when, who, what).
- The complete history of commits for a particular project and their metadata make up a **repository**. Repositories can be **distributed**. They are kept in sync across different computers (or your computer and a server in the cloud), facilitating collaboration among different people.

The GIT model: local and remote repositories

GIT is now the most used RCS, thanks to its distributed model. Git is a sophisticated software. It can get rather complex, but the basic commands and procedures are quite simple

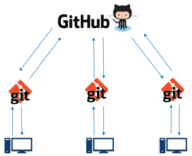
Term	Meaning
Workspace copy	Users' active directory. All files (including folders) in this directory will be tracked
Stage Area	It is a place where all the modified files marked to be committed are placed
Local Repository	User's copy of a remote repository. It must be synchronized manually
Remote Repository	It is a server where all the collaborators upload changes made to files
Clone Command	Creates a copy of an existing remote repository inside a local repository



- Commit: Commits all files from the staging area to local Repository
- Push: Pushes all the changes made in local to Remote Repository
- Fetch: Collects the changes from Remote Repository and copies them to Local Repository but doesn't affect our workspace
- Pull: Collects the changes from Remote Repository and copies them to Local Repository along with merges to the current directory or our workspace

Public services: GitHub

- GitHub is a web-based Git version control repository hosting service. It provides all of the distributed version control and source code management functionalities of Git for collaborative projects
- Git
  - It is a software
  - It is installed locally on the system
  - It is a tool to manage different versions of files
  - It is a command-line tool (some GUIs available but not recommended)
- GitHub
  - It is a service
  - It is hosted on the Web
  - It is a space to upload a copy of a Git repository
  - It provides a graphical interface through the browser
  - It enables to track issues and to create a collaborative development environment



- We will practice git using the tutorial on Software Carpentry. The practicals are in sections 1-9. The next sections are for information on collaborative development
- You are required to create an account on GitHub. https://guides.github.com/activities/hello-world/