Topics for this Lecture

- Version control systems
- GIT/git
- Git Workflow
- Main git commands

Version Control Systems (VCS)

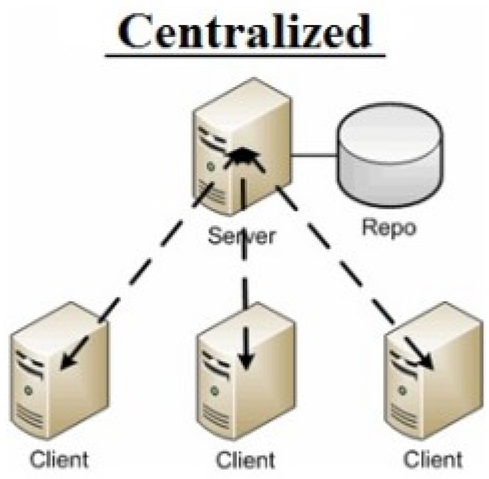
- Real-world projects are not a set of three .java files, or two .c files and two .h files
  - Real-world systems are complicated trees of source files, support files, documentation, test cases, and configuration files
  - Multiple developers work on the tree and make changes to it
    - May want to go back to an old version
    - May want to work on a file when you don’t have access to a shared network location
    - Dropbox/copying around a zipped version is clumsy and prone to disaster

Version Control Systems (VCS)

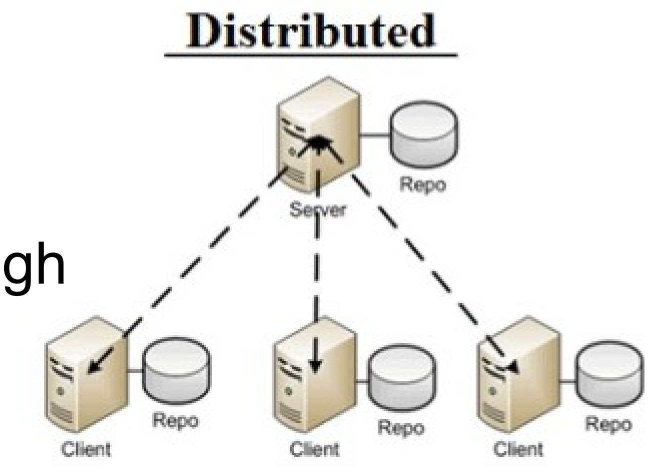
- What is a **version control system** (**VCS**)?
  - **A version control system (VCS)** (also known as revision control or source control) is a tool that software developers use to manage changes to source code over time.
  - Every modification/change made to the source is tracked, along with who made the change, why they made it, or references to problems fixed, by the change.
  - **VCS** is central to coordinating teams of contributors/developers to work on a single project at the same time.
    - **VCS** enables multiple developers to work on a single projects at the same time.
    - **VCS** integrates work done by different team members at the same time (i.e., Branching and merging).
    - **VCS** gives access to historical versions of your project.

Version Control Systems (VCS)

- Centralized VCS systems: developers perform various source control operations (get, commit, etc.) using a client installed on their local workstation, which then communicates and performs those operations against the server after some sort of security verification.
  - **Subversion** (e.g., Apache), **CVS**



- Distributed VCS systems :There’s no central server – every developer is the client, the server, and the repository. Source code changes are committed as per normal, but remain isolated unless a developer shares those changes with another repository through “push” and “pull” operations.
  - **GIT**(i.e., git) and **SQLite**

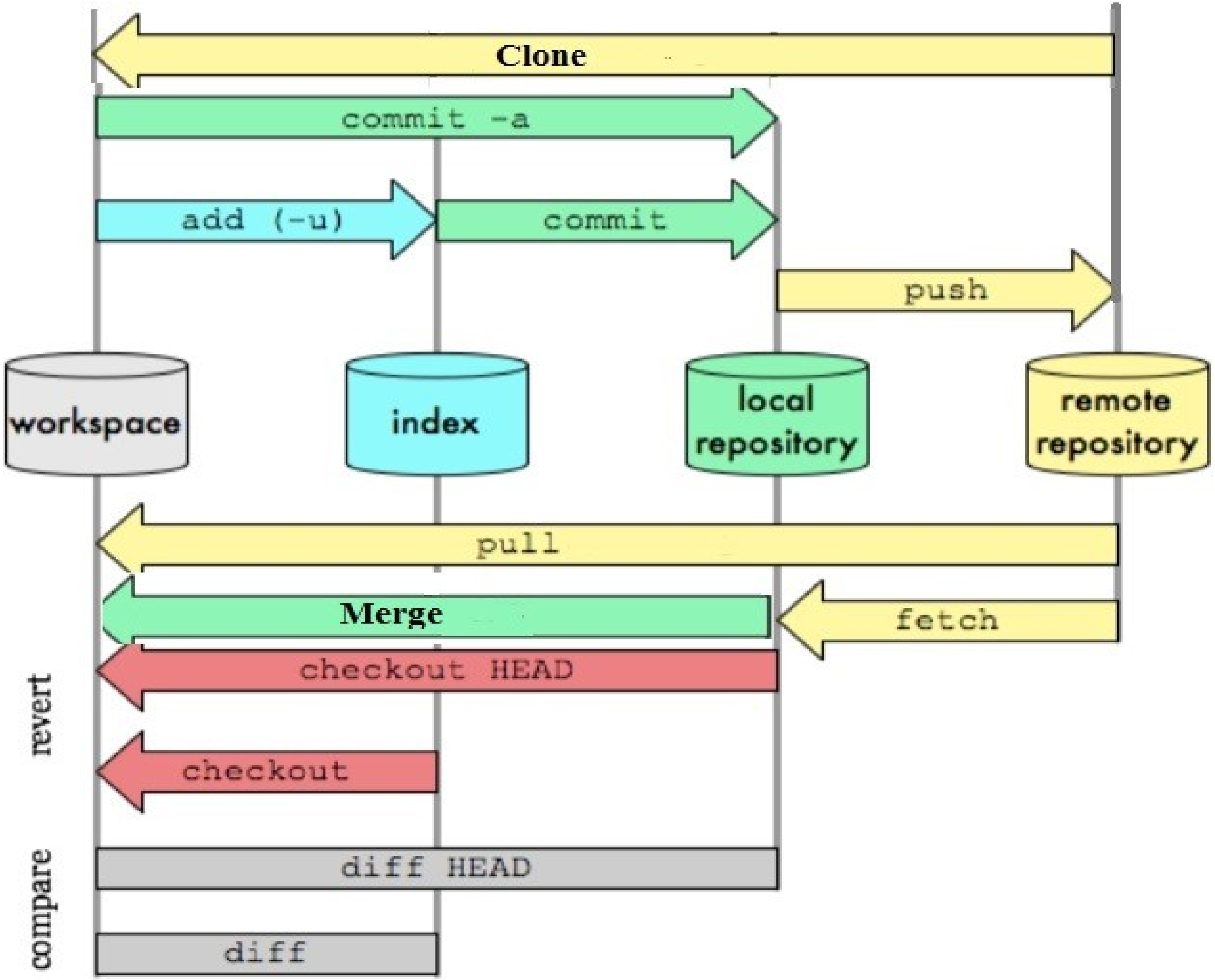


- We’re going to use a small subset of **git**, which I will introduce on the next few slides

Intro to GIT (git)

- One good representative of distributed version control systems, is **git**
  - it was initially designed and developed by **Linus Torvalds**.
  - It is open source.
  - **git** is a command-line tool.
- What is **GitHub**?
  - GitHub is a hosting service for Git repositories.
  - GitHub is “maybe” the world's largest code host where developers store their projects.
  - There's two ways you can use GitHub. You can use it publicly for open source and you can use it in private within your team.

GIT - The Simple Workflow



Main git commands

Here are some of the operations that we can perform on local repositories: (**Make them as your habit commands)**

Stage/Modify	
• **git clone [url]**	retrieve an entire repository from a hosted location via URL
• **git pull**	this updates all of the files and makes sure you are using the most current repo.
• **git add**	adds a specific file(s) or a complete directory to our index/stage for commit
• **git commit**	commits the changes that we made to those files to our local repository
• **git push**	pushes your projects/changes to your remote
• **git mv [file]** • **git rm [file]**	use git move and git rm or git remove to move files around and to remove files
Inspect	
• **git log**	to see all the commits of the repository.
• **git status**	shows modified files in working directory, staged for your next commit

Main git commands

Here are some of the operations that we can perform on local repositories: (**Make them as your habit commands)**

Branches	
• **git branch**	list your branches. a * will appear next to the currently active branch
• **git branch <branch-name>**	creates a new branch named <branch-name>
• **git checkout <branch-name>**	switches to the new branch <branch-name>.
• **git merge <branch-name>**	merge a remote branch into your current branch to bring it up to date

Here is the link to a cheat sheet that features the most important and commonly used Git commands for easy reference.