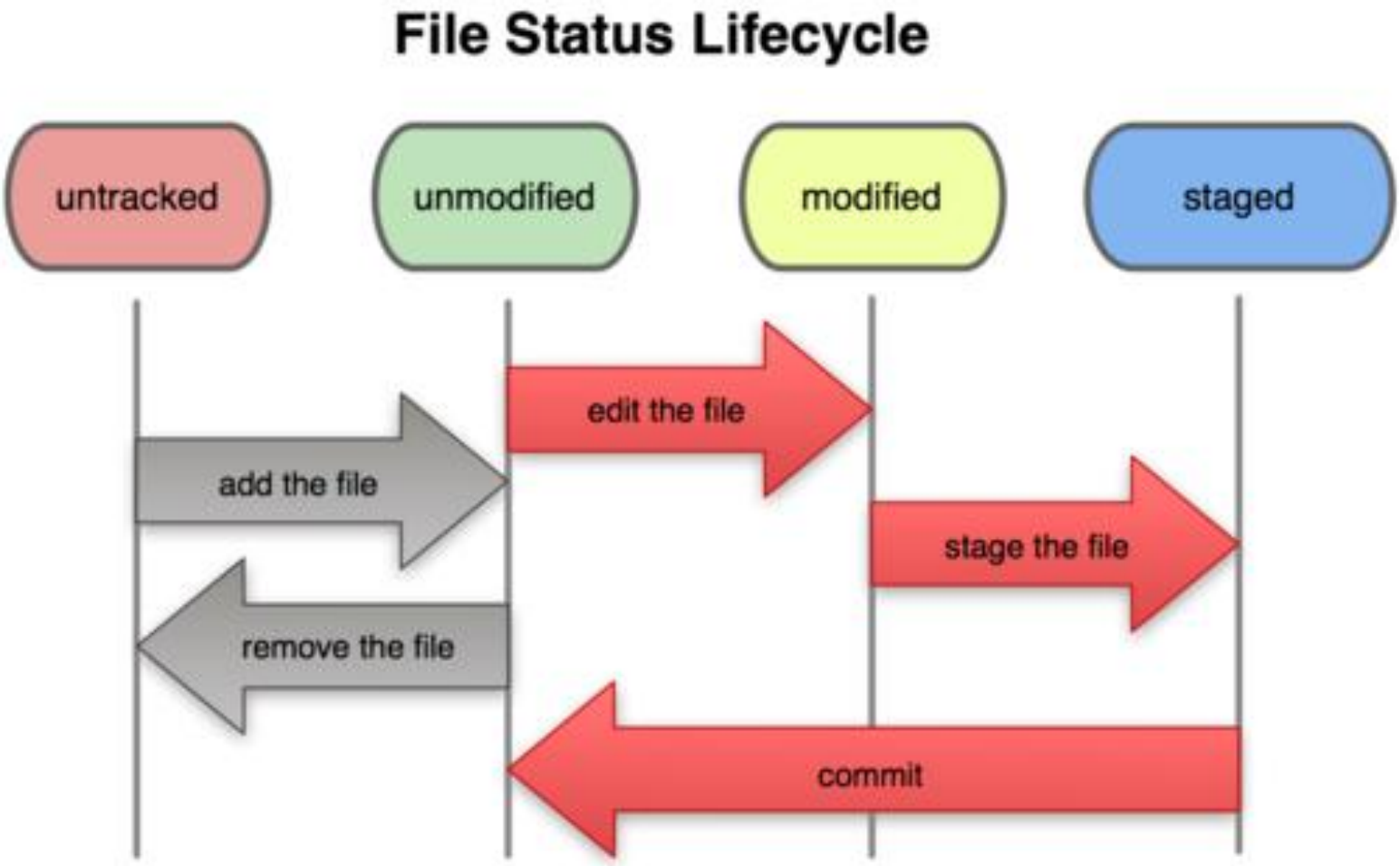
Hands-on Introduction to the git Source Code Management System

Mark Jones

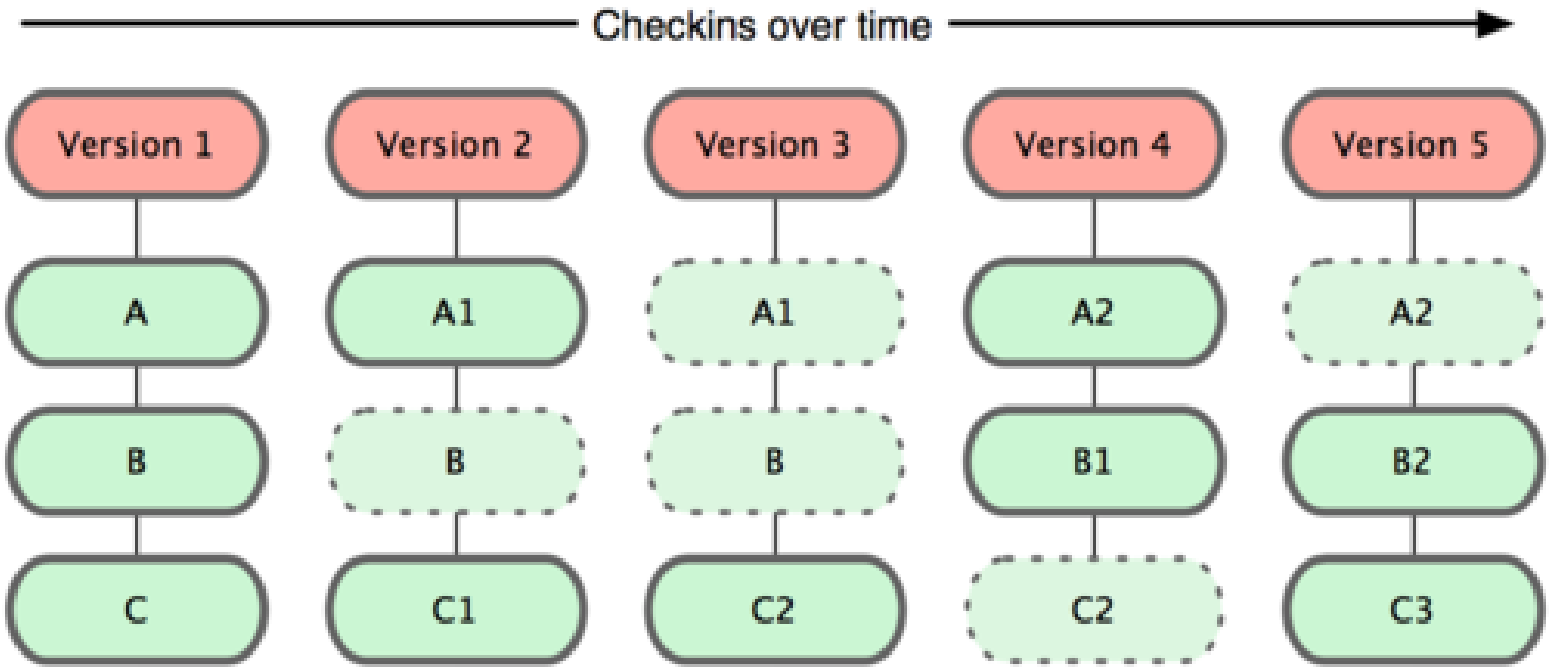
Overview of git concepts

- Easy to setup. No repository is special. Only a workflow model would designate a repository as central or special.
- Each repository is standalone and has the entire history of revision.
- Easy to make commits. Changes to code is marked by “commits”. A “commit” could be one file or many files. Each commit has an unique name.
- Merges from other repositories are special commits.
- Easy to make branches: When switching branches, the files are modified. Cannot switch branches if files have been changed but not committed.

File Status Lifecycle



Git stores snapshots of files



Personalize git

- Important to use “git config” to keep track of author of changes.
  - `git config --global user.name` "*Firstname Lastname*"
  - `git config --global user.email` “your_email@youremail.com”
- Need to set the editor for commits
  - `git config --global core.editor` "emacs“
- See config settings
  - `git config --list`

Starting a local repository

- Create git repository in working directory:
  - `git init`
  - `git status :` will list all files as untracked
  - If you do not want to track all files: Edit file: `.gitignore`
  - `git add .  :` stages all files in the directory and subdirectories
  - `git commit –m " Initial files" :` Repository is updated
  - The initial repository is the "master" branch

Working in local repository

- Modify two files
  - `git status :` will list files as modified
  - Need to decide to commit both at same time or two separate commits
  - `git checkout -- filename` : go back to previous version
  - `git add filename` : stages one file
  - `git commit –m “ why the change”`
  - `git diff` : difference between working directory and staged
  - `git diff –staged` : difference between staged and current repo state
  - `git commit –a` : stage and commit in one step
- History
  - `git log :` history of commits
  - `gitk` : GUI form of the log

Branches in local repository

- `git branch exp` : Creates a new branch called “exp” based on present repo
- `git branch` : List branches in the repo
- `git checkout exp` : Switch to “exp” branch. Files in working directory modified.
- Modify file while in branch exp. Add and commit to branch exp of repo.
- Merge into master branch
  - `git checkout master` : Switch to “master” branch of repo
  - `git merge exp` : Merge changes into “master” branch with commit history of the “exp” branch.

Create Shared Repository

- Create bare repository for sharing
  - `git clone --bare directory name.git`
  - Others can clone with: git clone stored_directory/name.git
- `git remote add origin name.git` : makes a shortcut for pushing and fetching commits from the bare repository.
- `git remote -v` : list the remote names
- Make changes to file and commit to local repo
- `git push origin` : pushes commits to shared repository.

Accessing Shared repository

- `git clone directory/repo.git` : Get repository and automatically create remote shortcuts called “origin”
- `git remote -v` : Show the shortcut “origin” for remote repos
- `git branch -a` : Shows local and tracking remote branches
- `git pull origin` : fetches and merges changes from remote repo and puts them in local branches
- `git fetch origin` : fetches changes from remote to tracking branch. Can see changes but not incorporated locally.
- `git merge origin` : merge the fetched changes into local repository.

Accessing Hall C Analyzer HCANA

•Follow directions at https://hallcweb.jlab.org/wiki/index.php/ROOT_Analyzer/Git

• Need to send Steve Wood your public ssh key (id_dsa.pub) to be install on server.

• For development work set up your own local branch

- `git checkout develop` : Be sure to be on develop branch
- `git checkout –b mybranch` : new branch
- `git push origin mybranch` : put on remote repository
- `git checkout -b branch origin/branch` : To get remote branch to local repository

Websites for git

- Git web page: http://git-scm.com/
- Git book: http://git-scm.com/book
- Local resource : https://hallcweb.jlab.org/wiki/index.php/Git_Howto