Introduction to Using Git & Github for Version Control

Ashish Singh

Ashish.Singh@pennmedicine.upenn.edu

December 2020

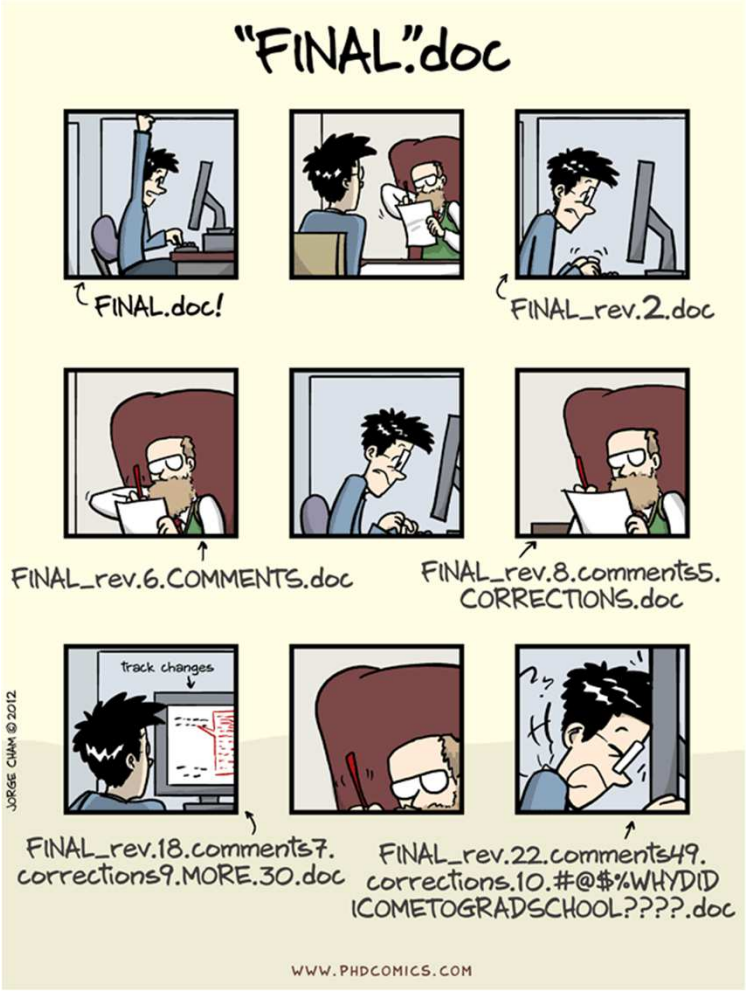
Center for Biomedical Image Computing & Analytics

On today’s menu

- Problematic Scenarios
- Version control systems
  - What are they?
  - History & Types
  - Centralized versus Distributed version control
- About Git
  - Commonly used Git commands
  - Simple Git workflow
- About Github
- Collaborative workflow with Git & Github
- Git Workflows
- Git GUI Clients
- Git Commits
- A short demo of Git & GitHub

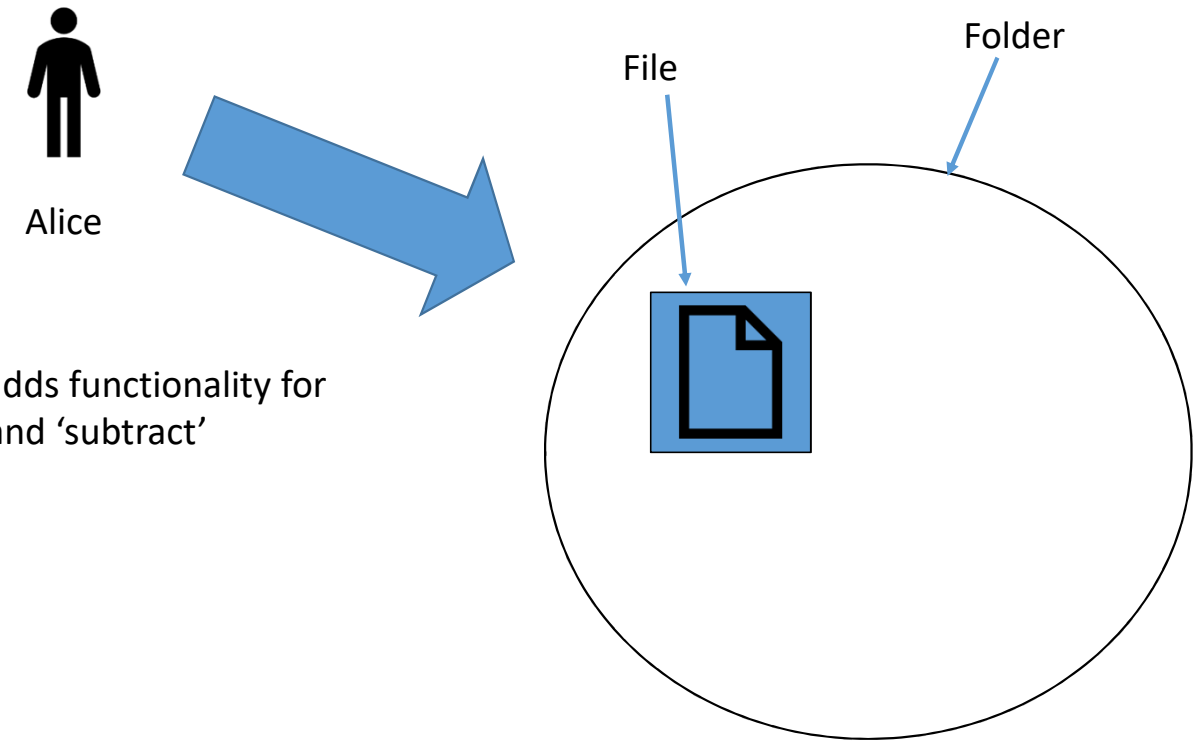
Problem: Manual Copies?

- Flaws:
  - Manual = fallible
  - Backup = copies of copies
  - Labelling
- We need metadata:
  - Datestamps
  - Timestamps
  - Annotations
  - Attribution
- Tools make this stuff quick and easy.



http://phdcomics.com/comics/archive_print.php?comicid=1531

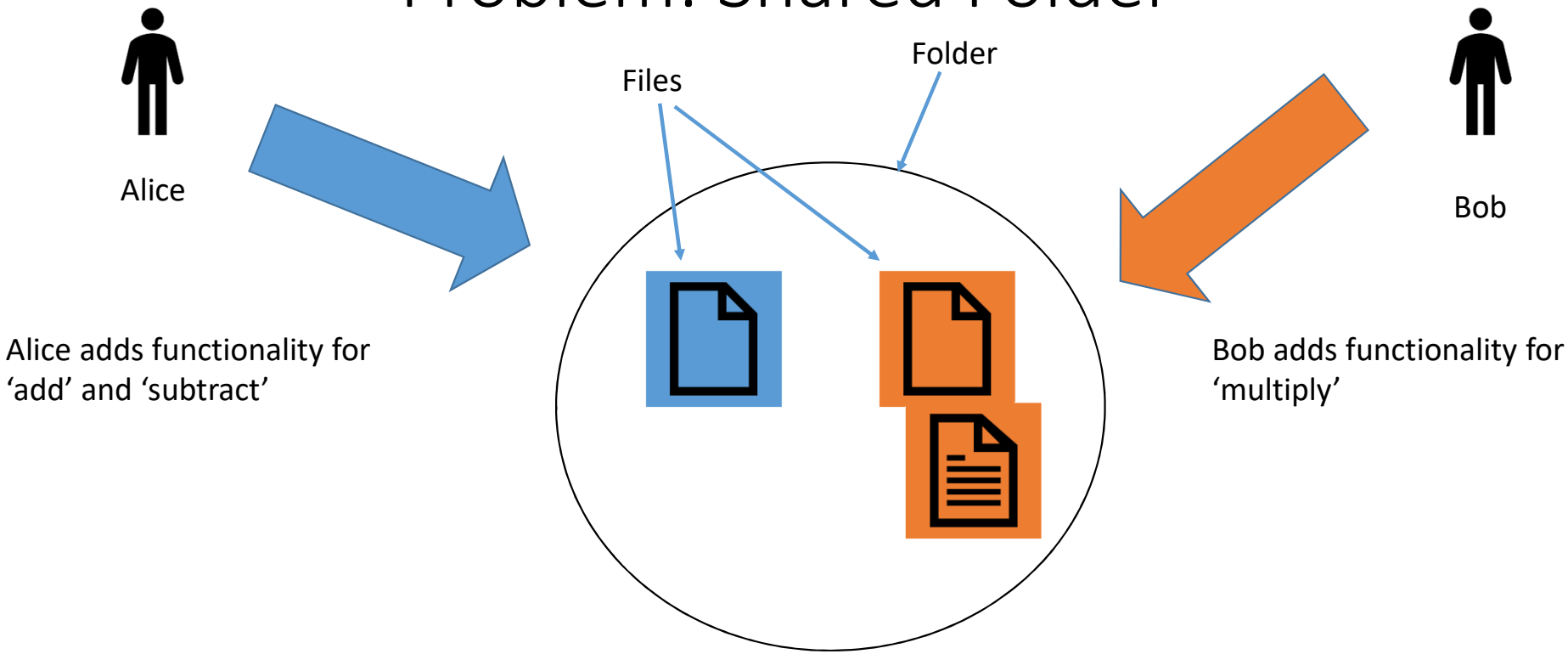
Problem: Shared Folder



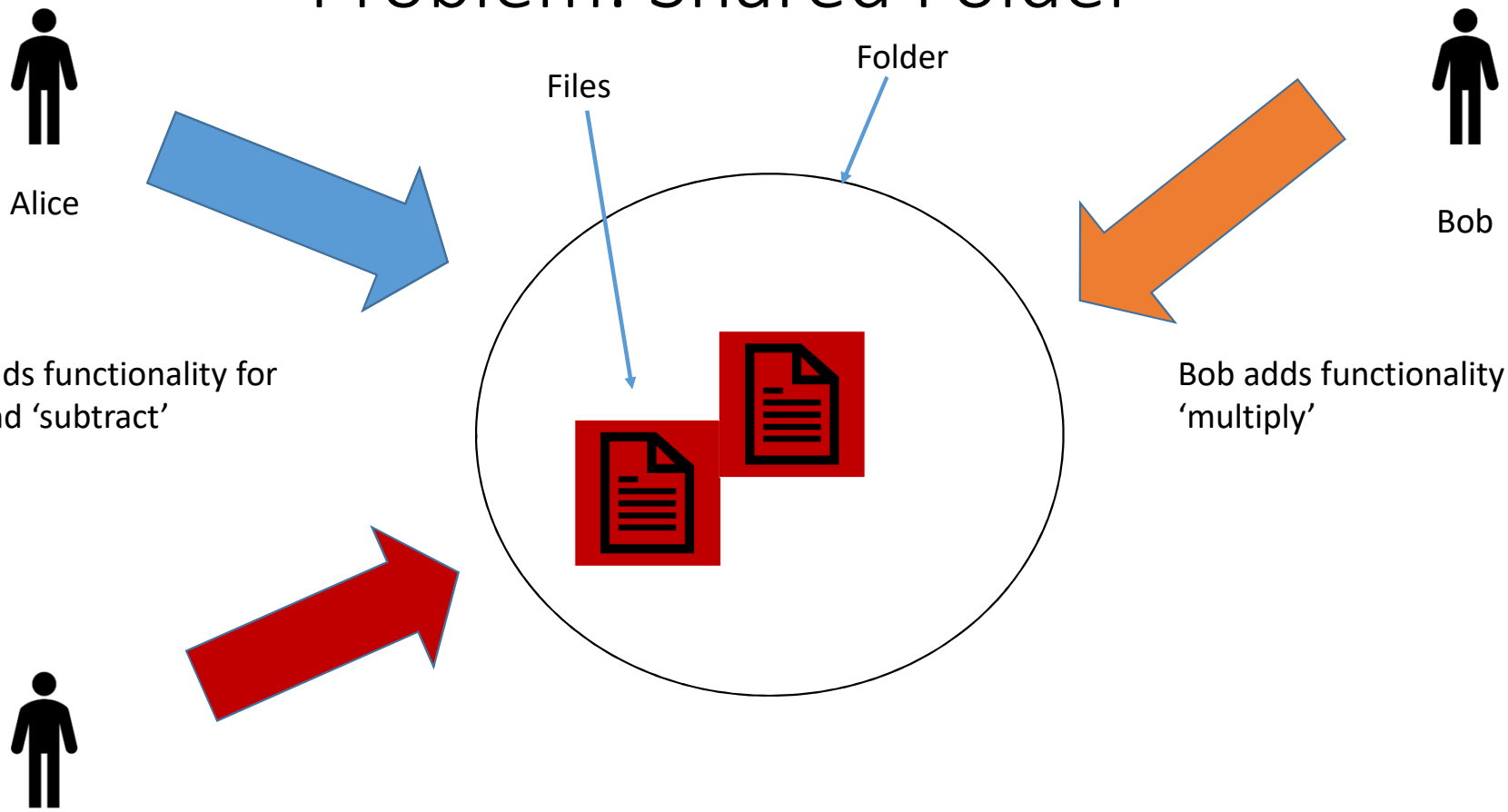
Alice adds functionality for
'add' and 'subtract'

Team Project: 3 people working on a 'calculator' application

Problem: Shared Folder



Problem: Shared Folder



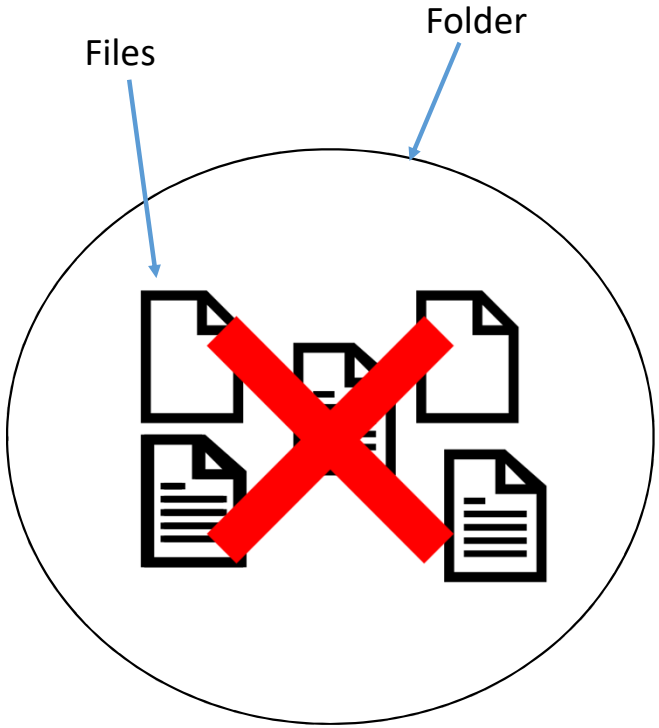
Alice adds functionality for 'add' and 'subtract'

Bob adds functionality for 'multiply'

Joe deletes everything and adds new files that he thinks is the correct code. His code however does not work correctly.

Problem: Shared Folder

Alice



Bob

Joe

Who replaced the files? When ? How do I get working code back?

Bigger projects

Visualization Toolkit

Analyzed 12 days ago

5.65M lines of code

The Visualization ToolKit (VTK) is an open source, freely available software system for 3D computer graphics, image processing, and visualization used by thousands of researchers and developers around the world. VTK consists of a C++ class library, and several interpreted interface layers including ... [More]

118 current contributors

11 months since last commit

77 users on Open Hub

Mostly written in C++

Licenses: BSD-3-Clause

Tags 3d graphics toolkit visualization vtk

Qt 5

Analyzed 12 months ago

8.19M lines of code

Qt is a cross-platform application and UI framework. Using Qt, you can write applications once and deploy them across many desktop and embedded operating systems without rewriting the source code.

292 current contributors

12 months since last commit

119 users on Open Hub

Mostly written in C++

Licenses: Commercial, GNU_Free_..., gpl3

Tags api bsd c++ coding cplusplus cross-platform cross-plattform database dbus desktop development embedded

MITK

Analyzed 3 months ago

1.45M lines of code

The Medical Imaging Interaction Toolkit (MITK) is a free software library with the aim of simplifying the development of interactive medical image processing programs. Through enhanced combinability and reusability of interactive components as well as the avoidance of redundant developments, an ... [More]

31 current contributors

3 months since last commit

9 users on Open Hub

Mostly written in C++

Licenses: BSD-3-Clause

Tags 3d c++ image itk medical registration segmentation visualization vtk

https://www.openhub.net/

Version Control to the rescue

What is a version control system?

aka: revision control or source control

- A way to manage files and directories
- Track changes over time
- Synchronization
- Short-term undo
- Long-term undo
- Track Ownership
- Branching and merging

Shared folders are quick and simple, but can't beat these features.

More uses of Version Control (1)

- *Change Management:*
  - Professional software will have bugs. Customers will find them. How do we know if a bug has been fixed?
  - Check-outs of code usually controlled.
    - A bug report will identify where the bug is in the code.
    - The fixed code (patch) is checked in and linked to bug report
    - Hence we can see exactly what changes were made in response to a specific bug. Good for accountability

More uses of Version Control (2)

- Code responsibility & Code audits.
  - You stole my code!
  - Who is responsible for this module?
  - Legal stuff
- Metrics (Managers only!)
- Version control is not just useful for collaborative working, essential for quality source code development

History of source control

- (1972) Source Code Control System (SCCS)
  - closed source, part of UNIX
- (1982) Revision Control System(RCS)
  - open source
- (1986) Concurrent Versions System (CVS)
  - open source
- (2000) Apache Subversion (SVN)
  - open source
- (2000) BitKeeper SCM
  - closed source, proprietary, used with source code management of Linux kernel
  - free until 2005
  - distributed version control

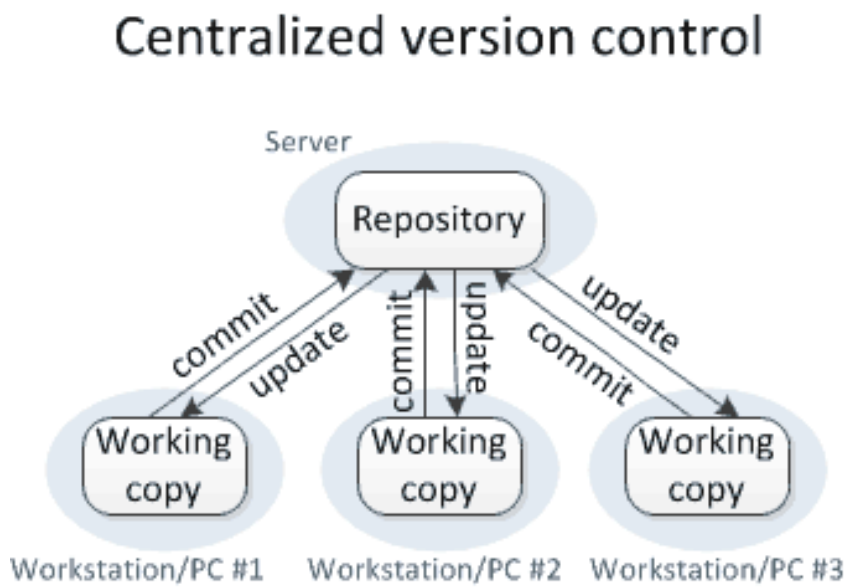
Some other well known version control systems

Types of version Control Systems

- Centralized
- Distributed

Centralized Version Control

- There is just one central repository.
- Each user gets his or her own working copy
- As soon as you commit, it is possible for your co-workers to update and to see your changes.
- For others to see your changes, 2 things must happen:
  1. You commit
  2. They update

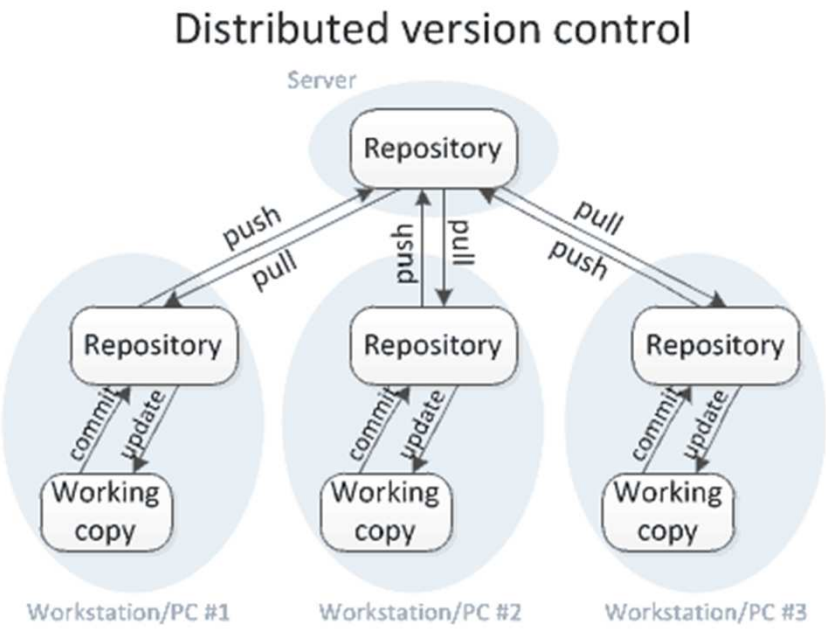


Example: CVS and Subversion

https://homes.cs.washington.edu/~mernst/advice/version-control.html

Distributed Version Control

- There are multiple repositories
- Each user gets his or her own repository *and* working copy.
- After you commit, others have no access to your changes until you push your changes to the central repository.
- When you update, you do not get others' changes unless you have first pulled those changes into your repository.
- For others to see your changes, 3 things must happen:
  1. You commit
  2. You push
  3. They pull



Example: Git

https://homes.cs.washington.edu/~mernst/advice/version-control.html

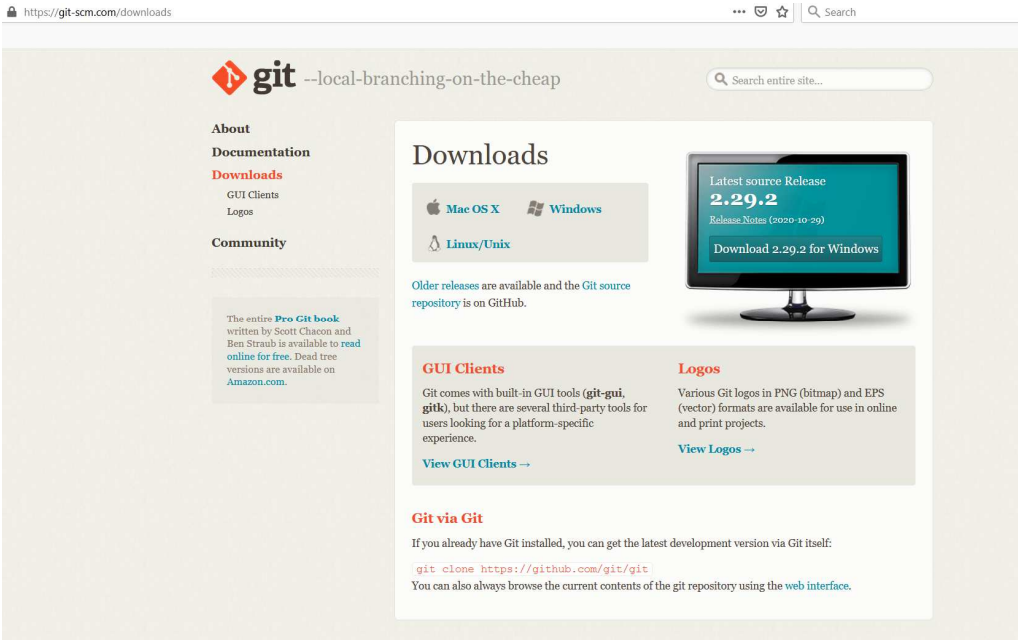
About Git

- Created by Linus Torvalds, creator of Linux, in 2005
  - Came out of Linux development community
  - Designed to do version control on Linux kernel
  - Distributed version control
  - Can work without internet connection
  - Cross-platform
  - Open source/free

Download and install Git

- Installers available from: http://git-scm.com/downloads

- Note: Git is primarily a command-line tool



Git has a lot of commands

- Learn a core subset of them
- And one of the GUI tools (e.g. gitEye, gitkraken, etc.)
- Then learn the rest as you need them

Some git commands

- Make a repo
  - git init
- Update a repo
  - git add
  - git commit
  - git push
- Create a copy of a remote repo
  - git clone
- Update from the remote repo
  - git pull

- Git Command CheatSheets
  - https://confluence.atlassian.com/bitbucketserver/basic-git-commands-776639767.html
  - https://www.atlassian.com/git/tutorials/atlassian-git-cheatsheet

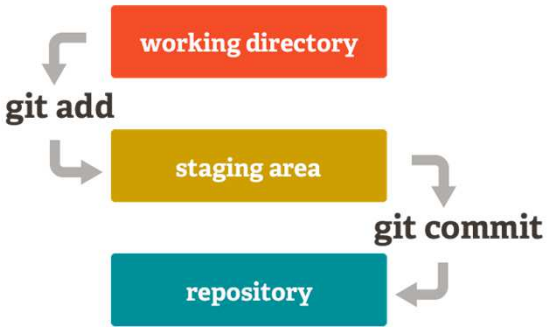
Some useful commands

- Initial Configuration
  - git config --global user.email you@example.com
  - git config --global user.name "Your Name"
- Check Status
  - git status
  - Use this frequently
- Check logs
  - git log
  - Shows the last commit on the top

Creating a Repository

1. To create a new local repository
   - git init myproject (creates directory ‘myproject’ and initializes a new repository there)
   - cd myproject
   - To add and commit new files to the repository
     - git add source.cpp
     - git commit -m “source file added” source.cpp
     - If you don’t provide file after the comment, everything you have done will be committed.
2. To clone a remote repository
   - git clone url localDirectoryName
   - This will create the given local directory, containing a working copy of the files from the repo, and a .git directory (used to hold the staging area and your actual local repo)

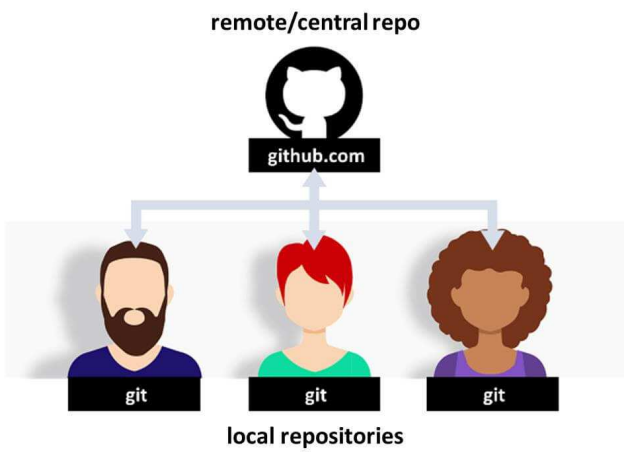
You need to do any one of these!

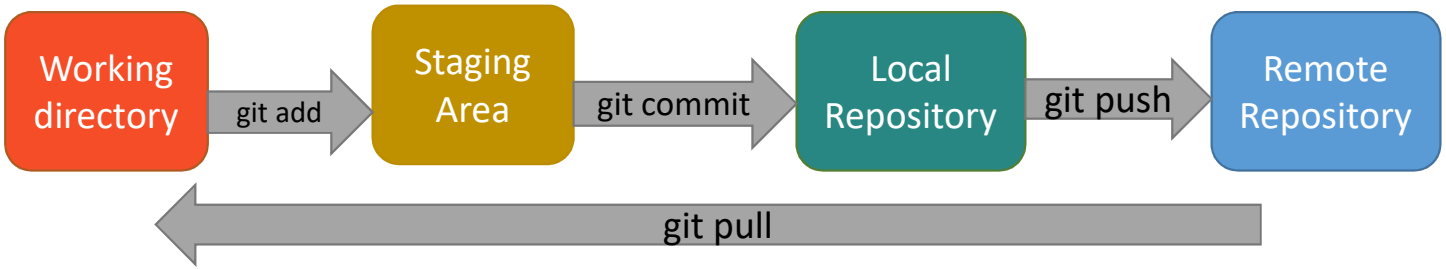


Simple workflow for working with a local repo

Simple workflow for working with remote repository

- git clone
- git add
- git commit
- git push





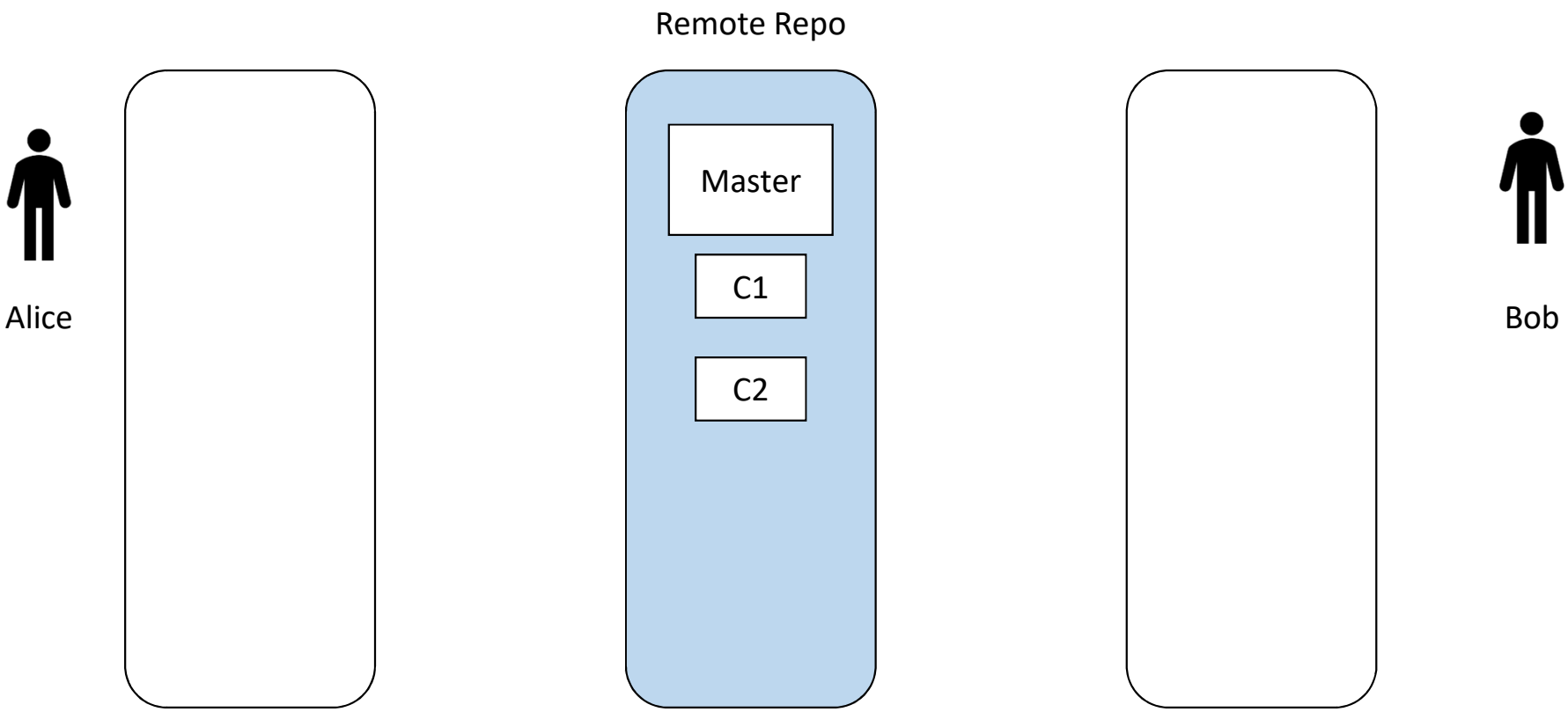
About Github

- It’s a hosting medium/website for your Git repositories
- Free to use – allows creation of unlimited repositories.
- Offers powerful collaborative abilities
- A good indicator of what you code/how much you code/quality of your code

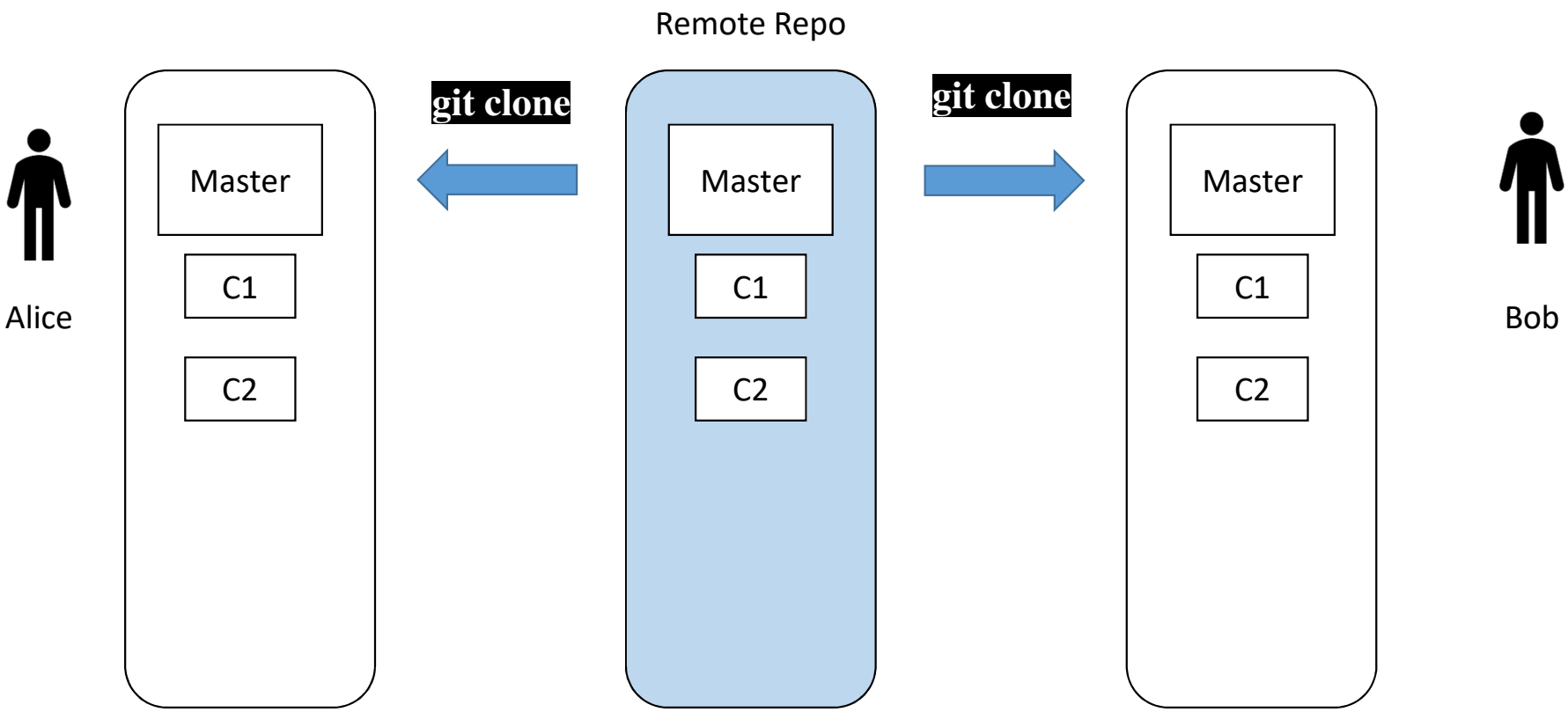
Working with others

- Here's what you normally do:
  - Clone the central repository
  - Make your changes
  - Commit your changes to your local repository
  - Check to make sure someone else on your team hasn't updated the central repository since you got it
  - Push your changes to the central repository
- If the central repository *has* changed since you got it:
  - It is *your* responsibility to **merge your two versions**
    - This is a strong incentive to commit and upload often!
  - Git can often do this for you, if there aren't incompatible changes

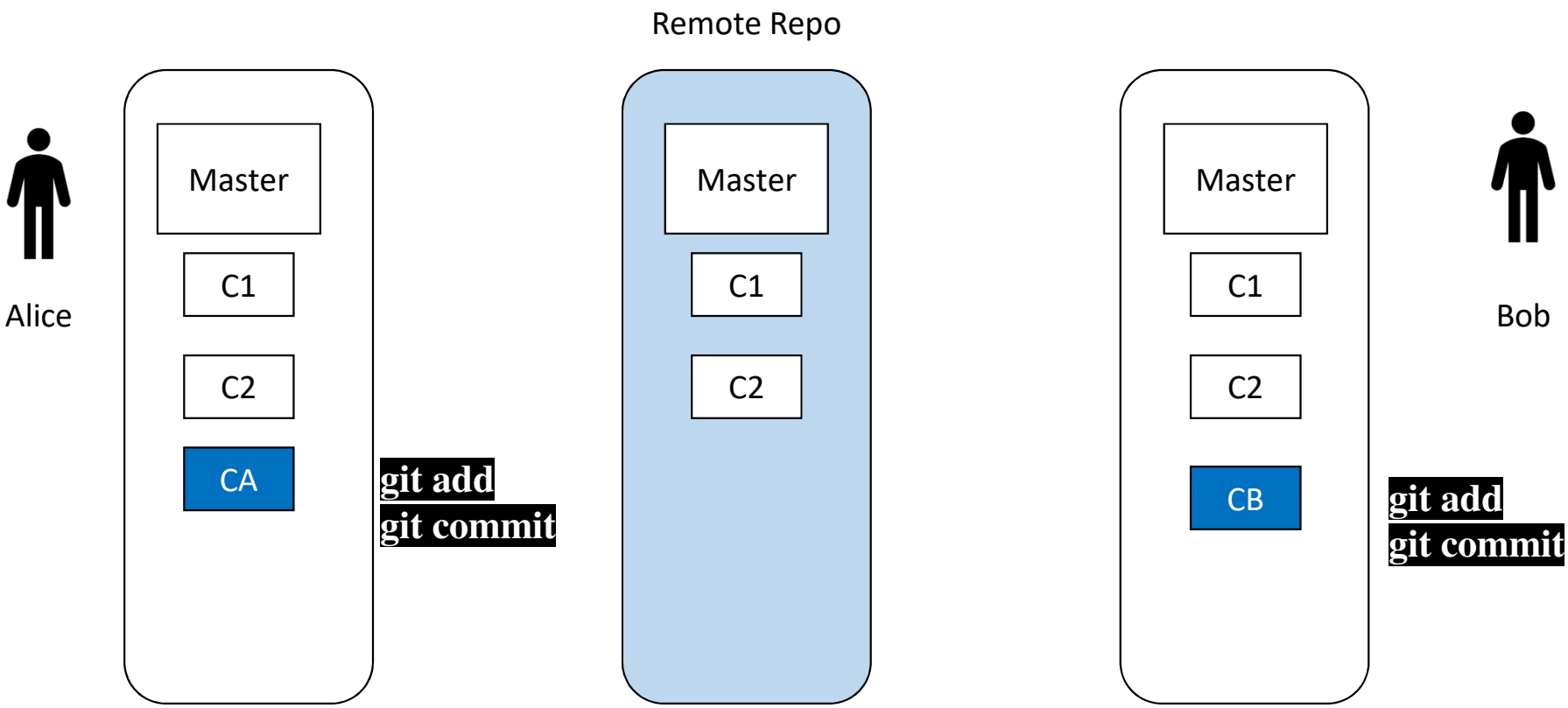
Collaboration with Git & GitHub



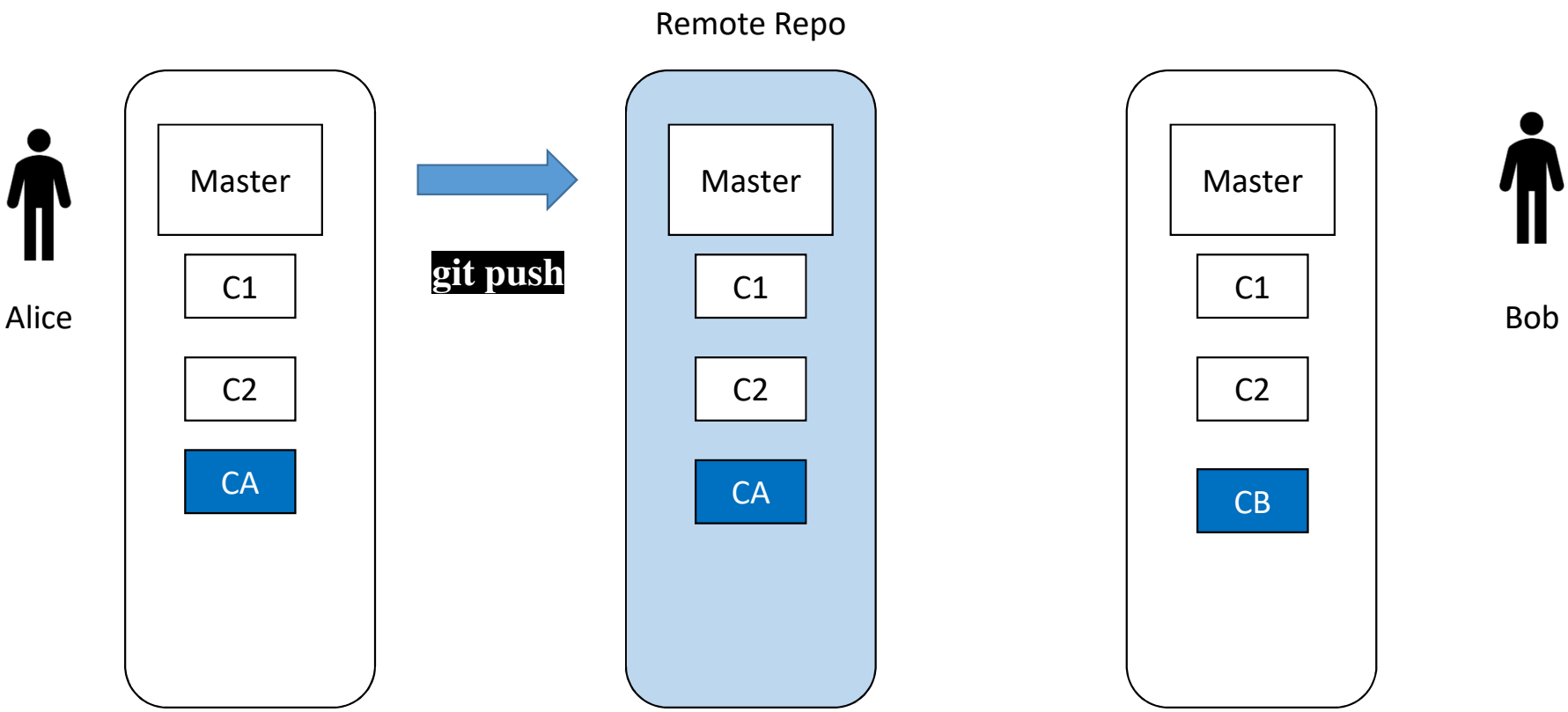
Collaborate



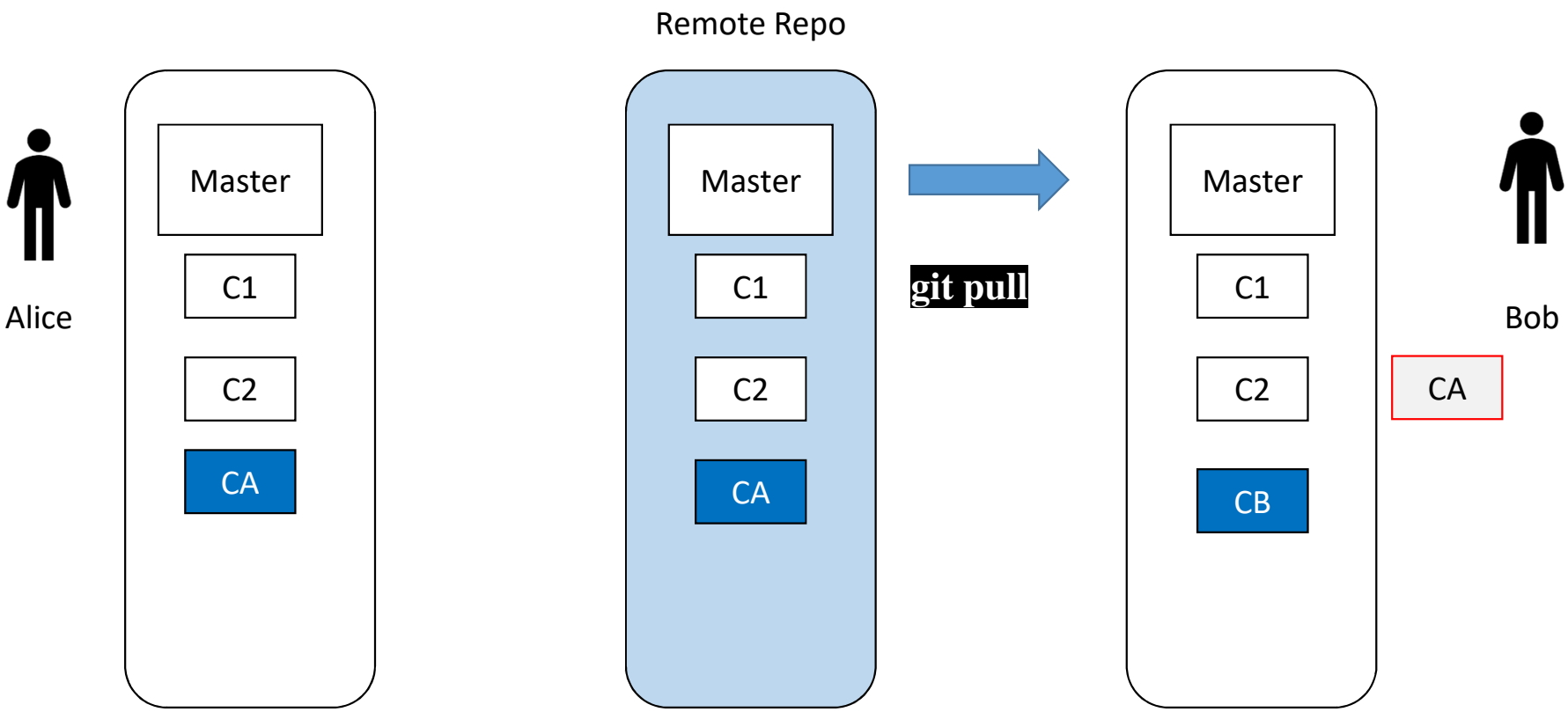
Collaborate



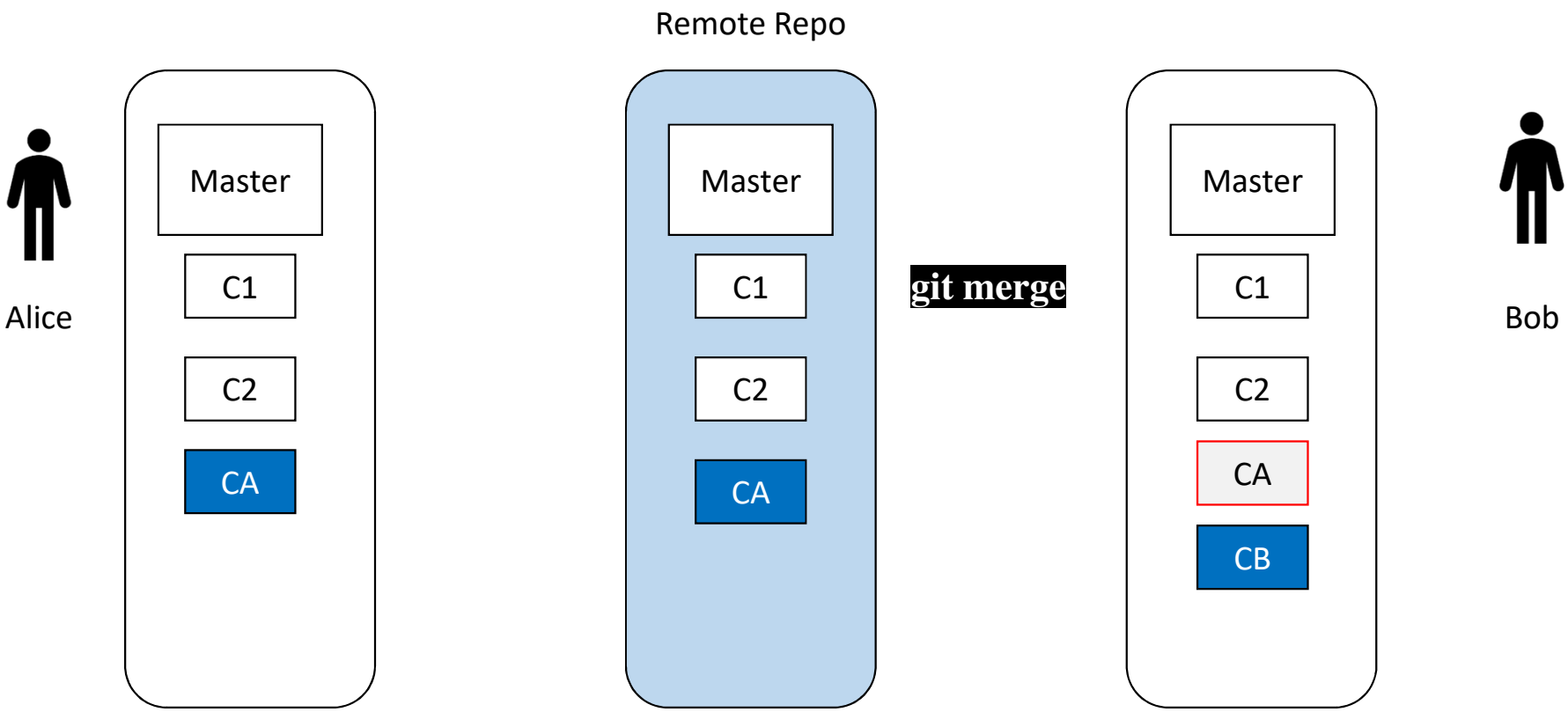
Collaborate



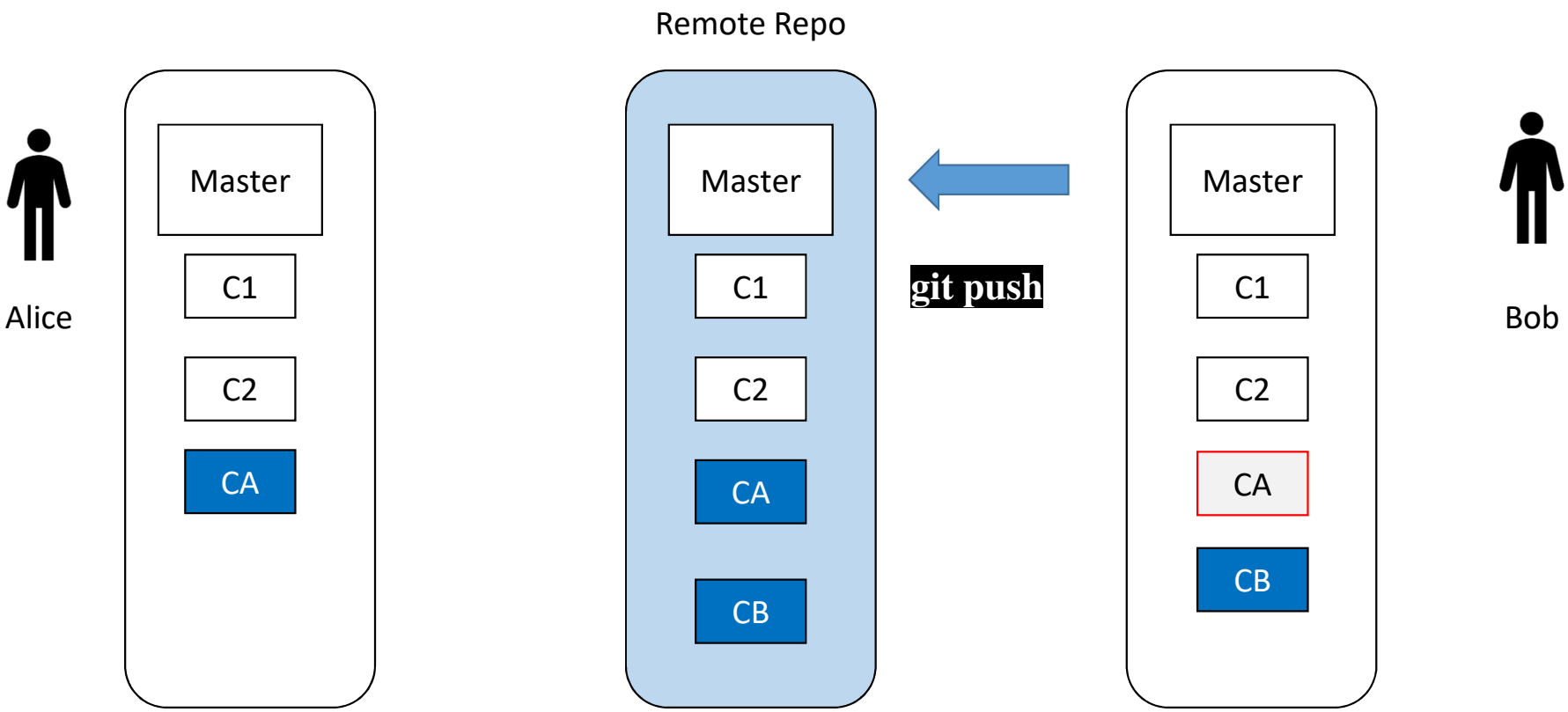
Collaborate



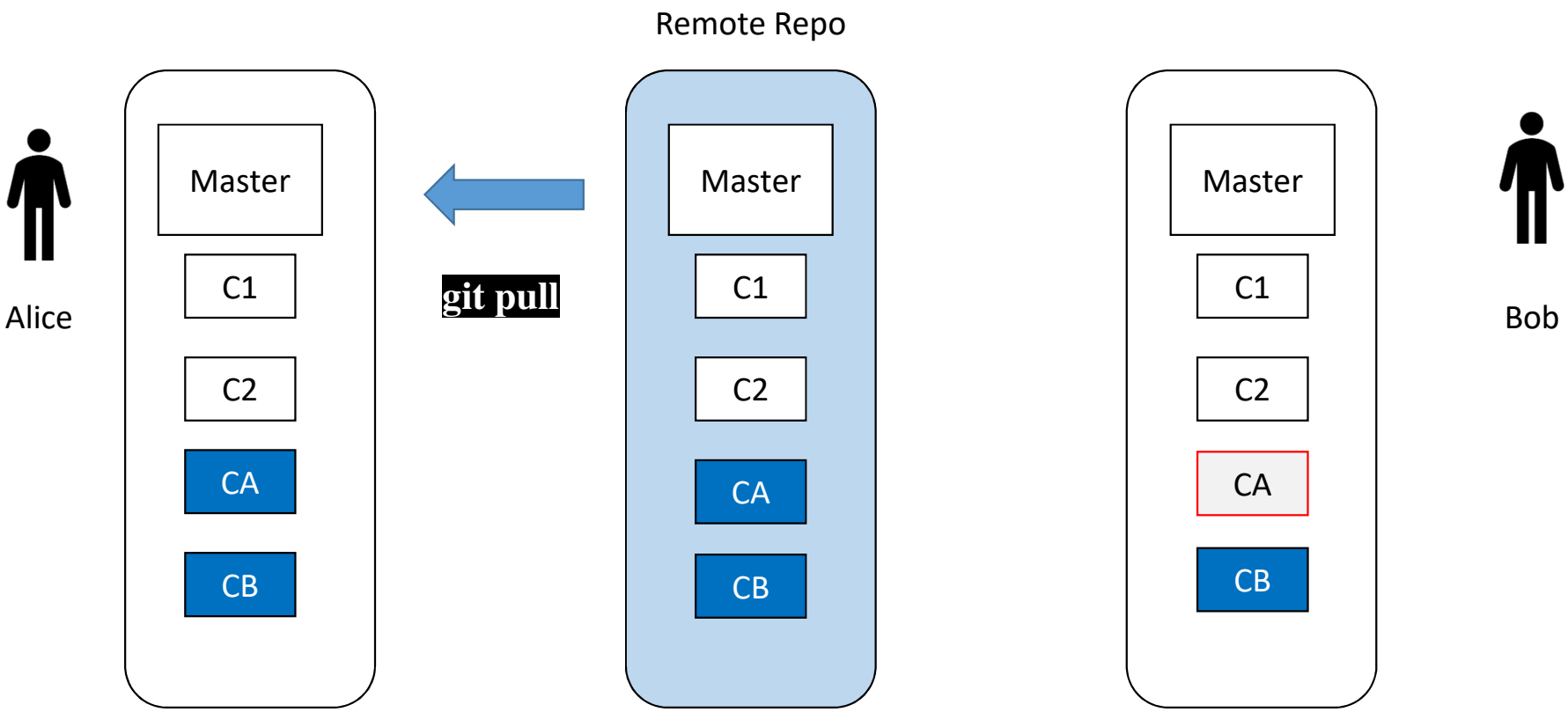
Collaborate



Collaborate



Collaborate

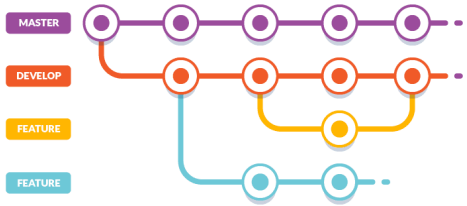


Git Workflows

- Centralized workflow

- Forking Workflow (server side clone)
  - https://github.com/CBICA/CaPTk

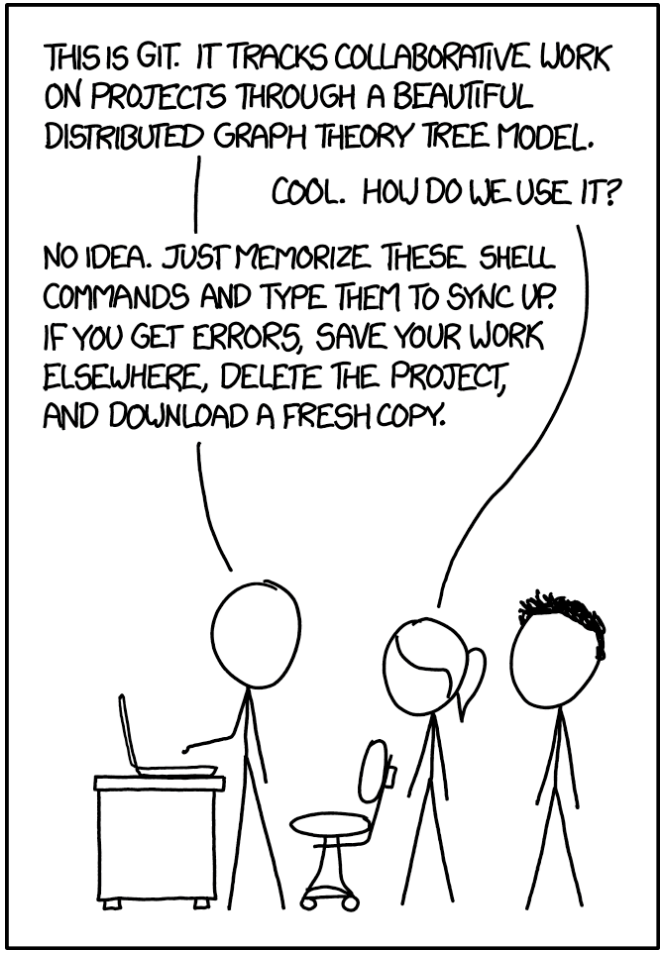
- Gitflow Workflow
  - https://github.com/MITK/MITK



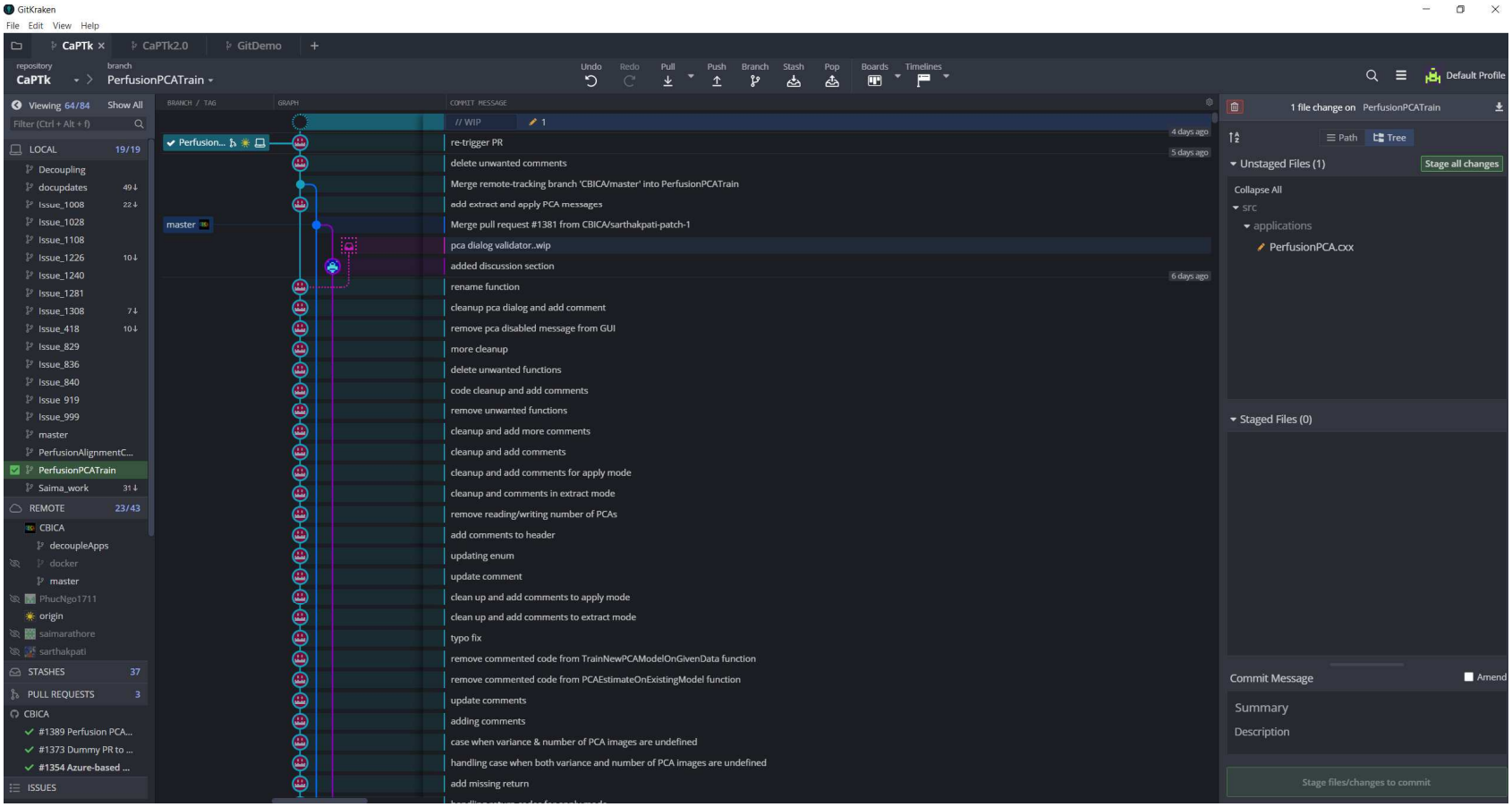
- More information
  - https://www.atlassian.com/git/tutorials/comparing-workflows

Git GUI Clients

- No worries. We can now use git from a graphical front end.
  - GitEye
  - GitCola
  - GitKraken
  - TortoiseGit
  - Basic GUI also available via IDE such as VS, Eclipse, QtCreator, etc.



GitKraken - https://www.gitkraken.com/

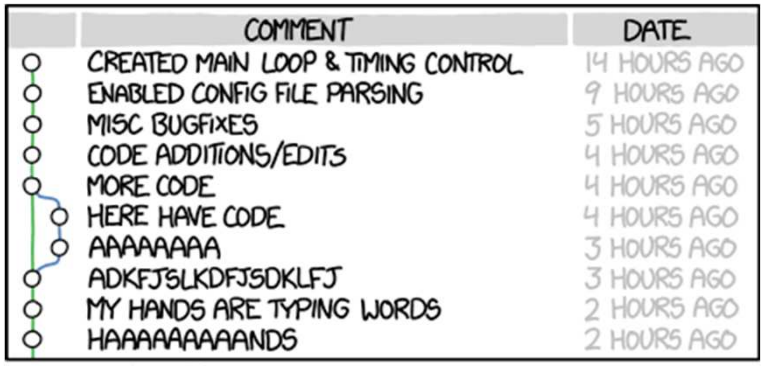


Git commits

- In Subversion each modification to the central repo increments the version # of the overall repo.
- In Git, each user has their own copy of the repo, and commits changes to their local copy of the repo before pushing to the central server.
- So Git generates a unique SHA-1 hash (40 character string of hex digits) for every commit.
- Refers to commits by this ID rather than a version number.
- Often we only see the first 7 characters:
  - 1677b2d Edited first line of readme
  - 258efa7 Added line to readme
  - 0e52da7 Initial commit

Commit Messages

- Many conventions
- Choose one
- Whatever you do make sure your messages are meaningful and descriptive
  - Your future self and contributors will thank you!
  - Especially as you move on to bigger and better projects



AS A PROJECT DRAGS ON, MY GIT COMMIT MESSAGES GET LESS AND LESS INFORMATIVE.

Demo

Take home message

- Start using version control and preferably Git & Github
- Benefits
  - For working Individually
    - Gives you a “time machine” for going back to earlier versions
    - Gives you great support for different versions (standalone, web app, etc.) of the same basic project
  - For working in a team:
    - Greatly simplifies concurrent work, tracking changes, answer who did what – ~~blame/~~praise, merging changes, etc.
  - For getting an internship or job:
    - All companies use some kind of version control
    - Showcasing your work as in GitHub

Advanced Reading/Videos

- https://www.atlassian.com/git/tutorials
- https://www.atlassian.com/git
- https://git-scm.com/doc
- https://git-scm.com/book/en/v2
- https://git-scm.com/doc/ext
- https://try.github.io/
- https://www.gitkraken.com/learn/git/tutorials

For any questions or for creating remote repositories on CBICA GitHub organization, contact us at: developers@cbica.upenn.edu

Thank You!