Tecniche e architetture avanzate per lo sviluppo del software

A.A. 2023-2024

Universita' degli Studi Di Torino - Dipartimento di Informatica

Git for beginners

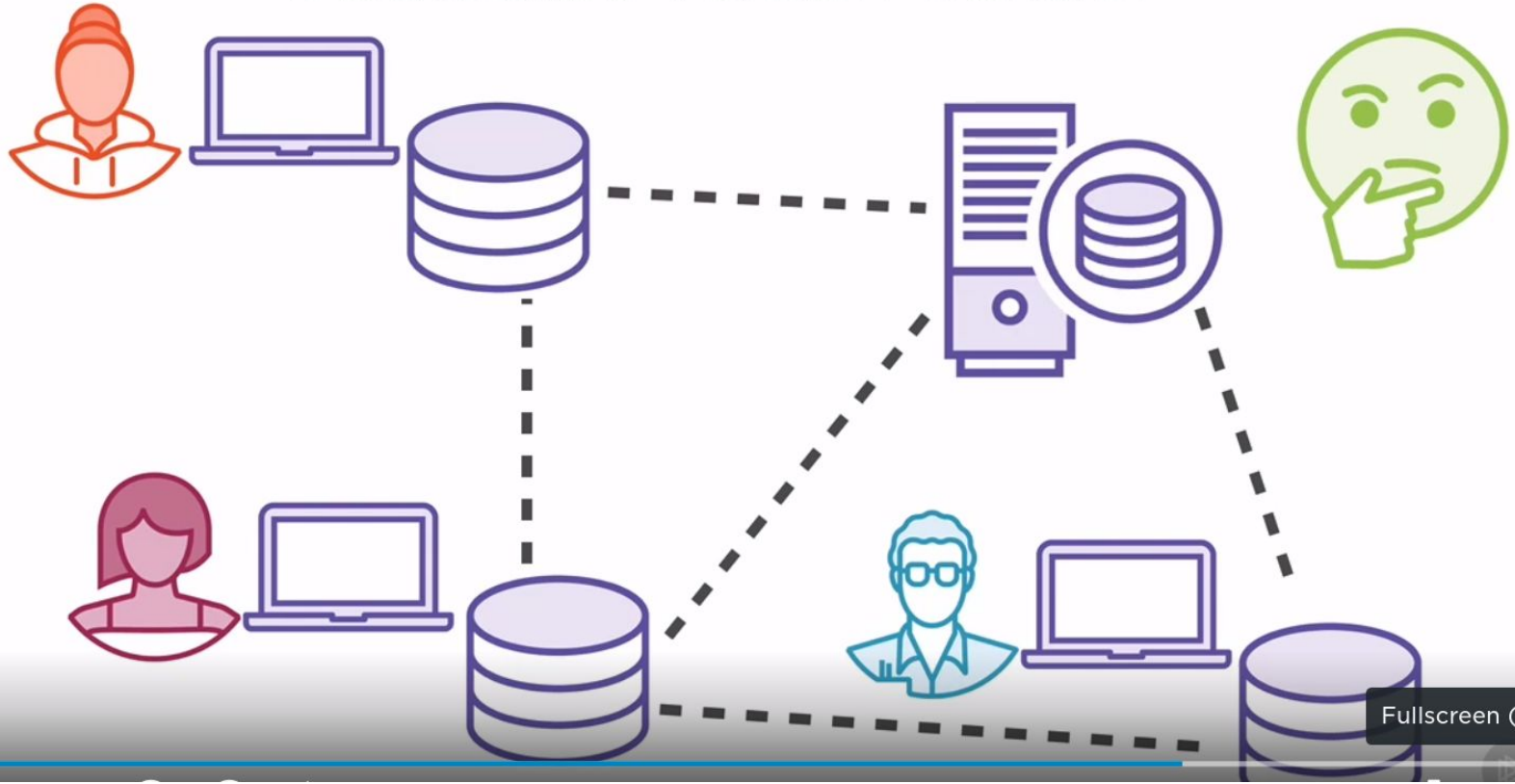
Klas Mellbourn

klas@mellbourn.net github.com/mellbourn

What is Git?

- *Distributed* Version Control System
- each repository independent and has complete version history
- communication at well defined points
- most commands local
  - commit
  - branch
  - merge
  - history
- fast
- offline is the normal case
- each clone is a backup

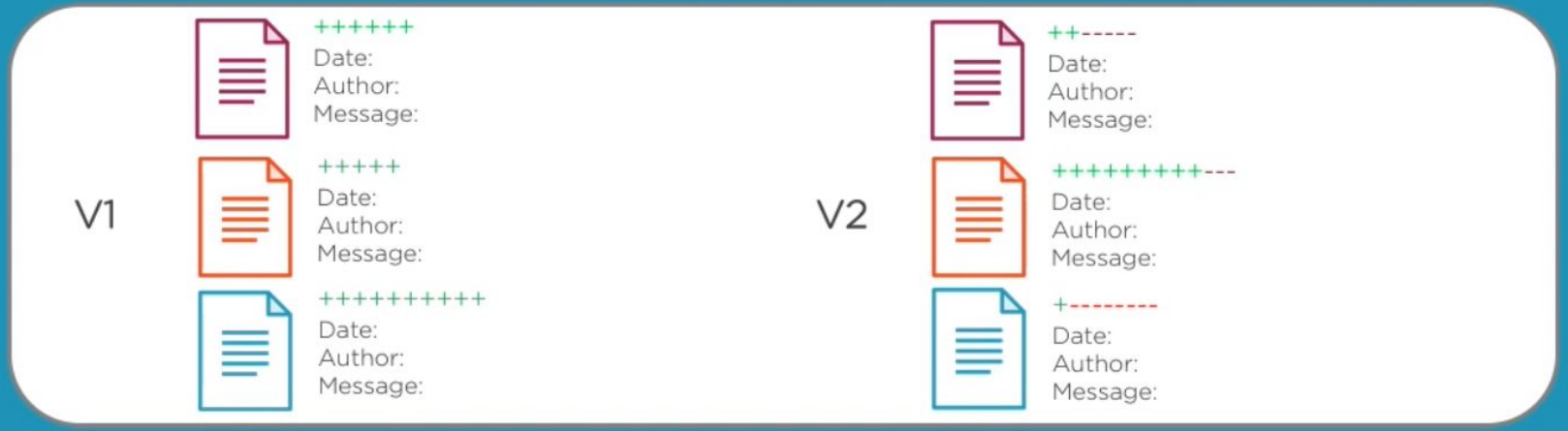
Distributed Version Control



What is Git?

Version Control System

- Software designed to record changes made to files over time.
- Ability to revert back to a previous file version or project version.
- Compare changes made to files from one version to another.



Git Clients

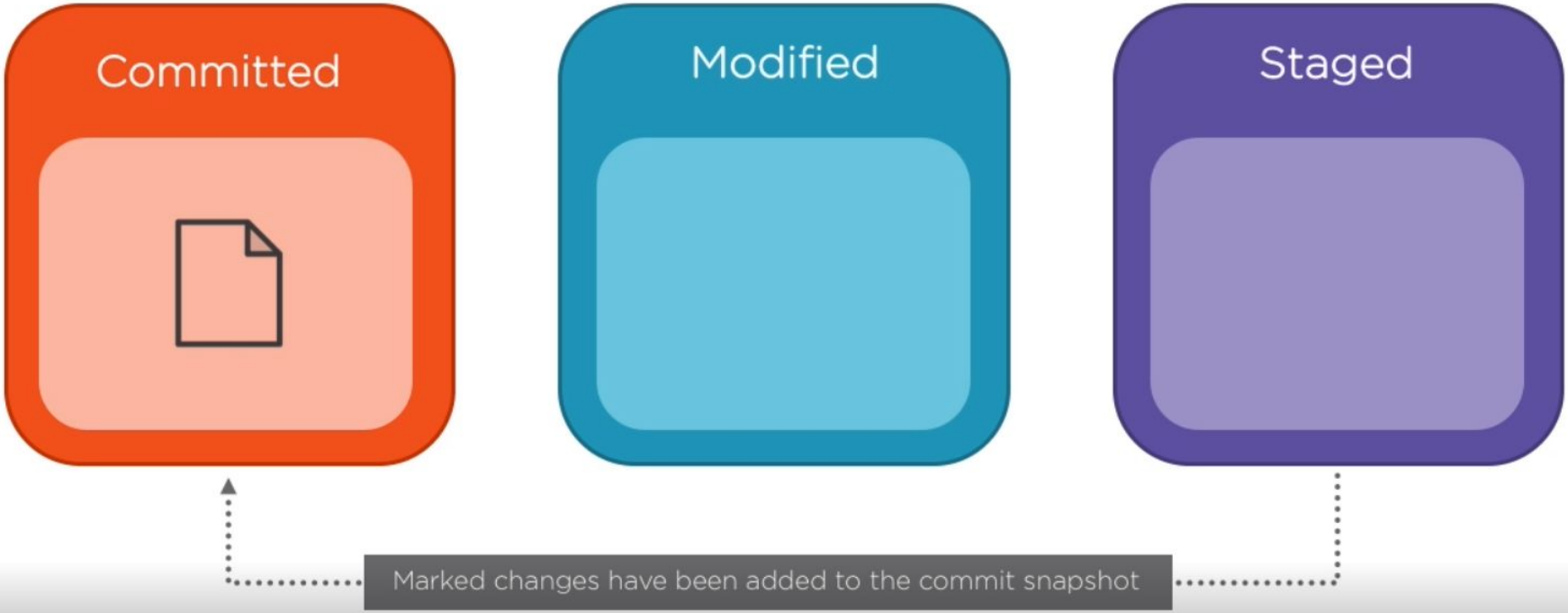
- Open source
- many clients
  - git command line
    - with powershell module: posh-git
  - Git Extensions (stand alone gui client)
  - TortoiseGit (stand alone gui client, shell integration)
  - SourceTree (cross platform gui client)
  - Visual Studio Microsoft Git source control Provider
    - built in to all versions
    - VS2015 even has method level git history

Absolute basics

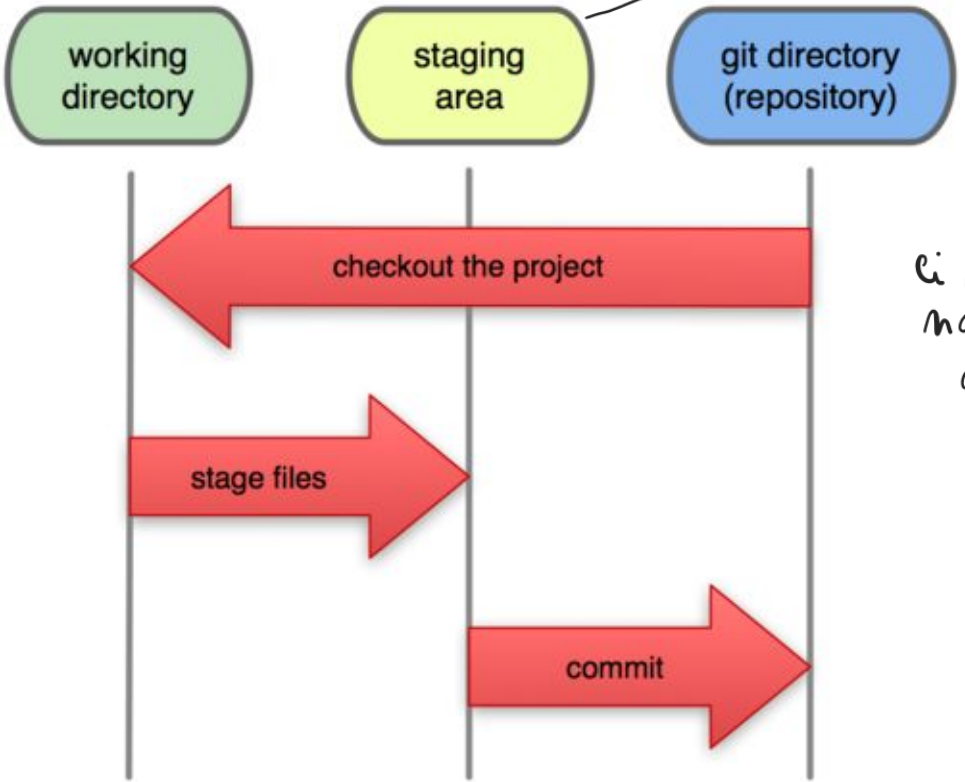
- create a file

- create a repository
- set username
- commit
- show history
- change
- show history

The Three Stages of a File



Local Operations



test locali prima di fare una commit

↓

step intermedio tra working dir e commit

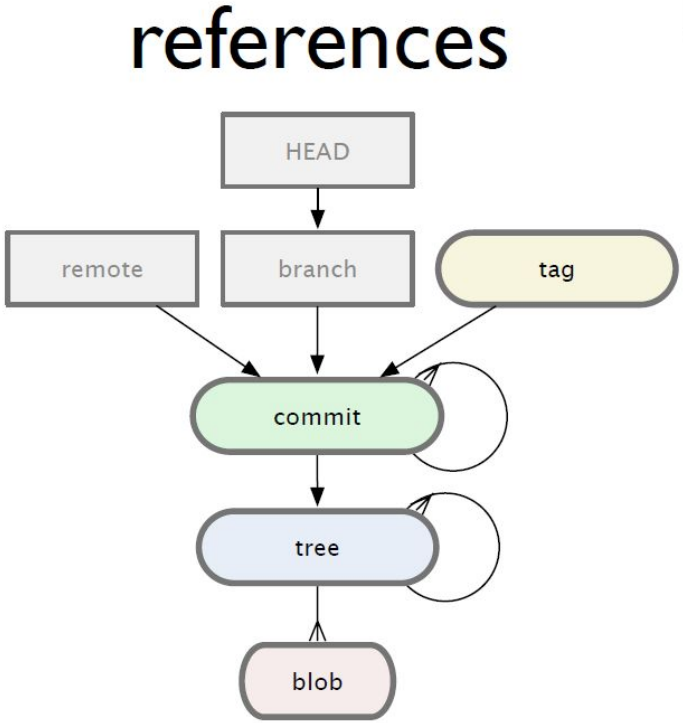
↓

ci metti del codice ancora non pronto che deve essere ancora testato

More changes

1. add files
2. stage one, commit it
3. stage another, commit it
4. change file
5. partially stage
6. commit
7. view history
8. note that you can see the file tree in each commit

References



Undoing things

- resetting working directory
- reverting commits
- reset - move branch to a commit
  - hard
  - soft
- amending commits

.gitignore

- Some files you don’t want to version control
- Put them in .gitignore
- note that the tree of the commit does not contain all files

Branching

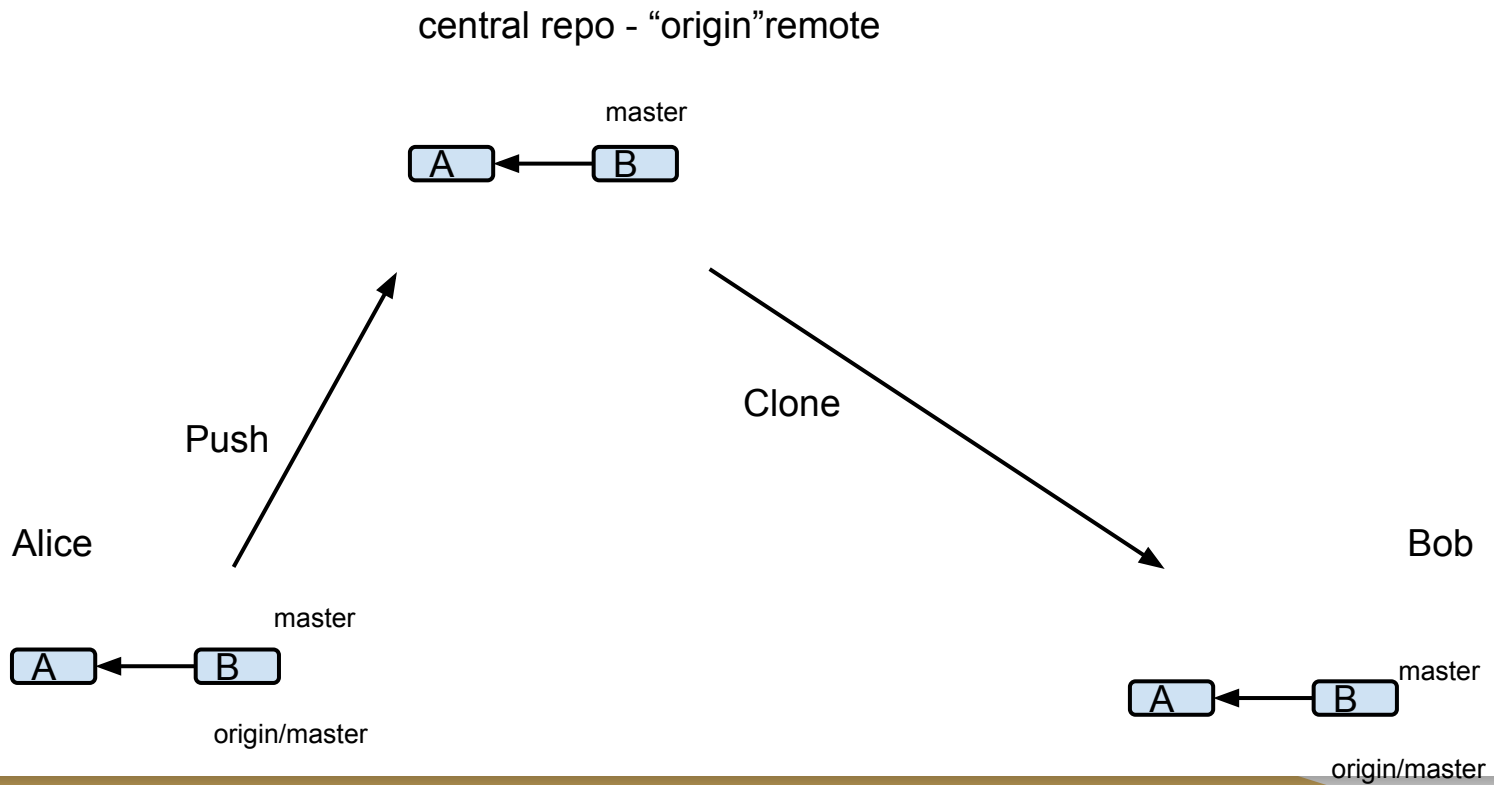
In git you branch a lot!

- branch
- checkout (switch branch)
- merge
  - reintegration merge
  - conflict resolution

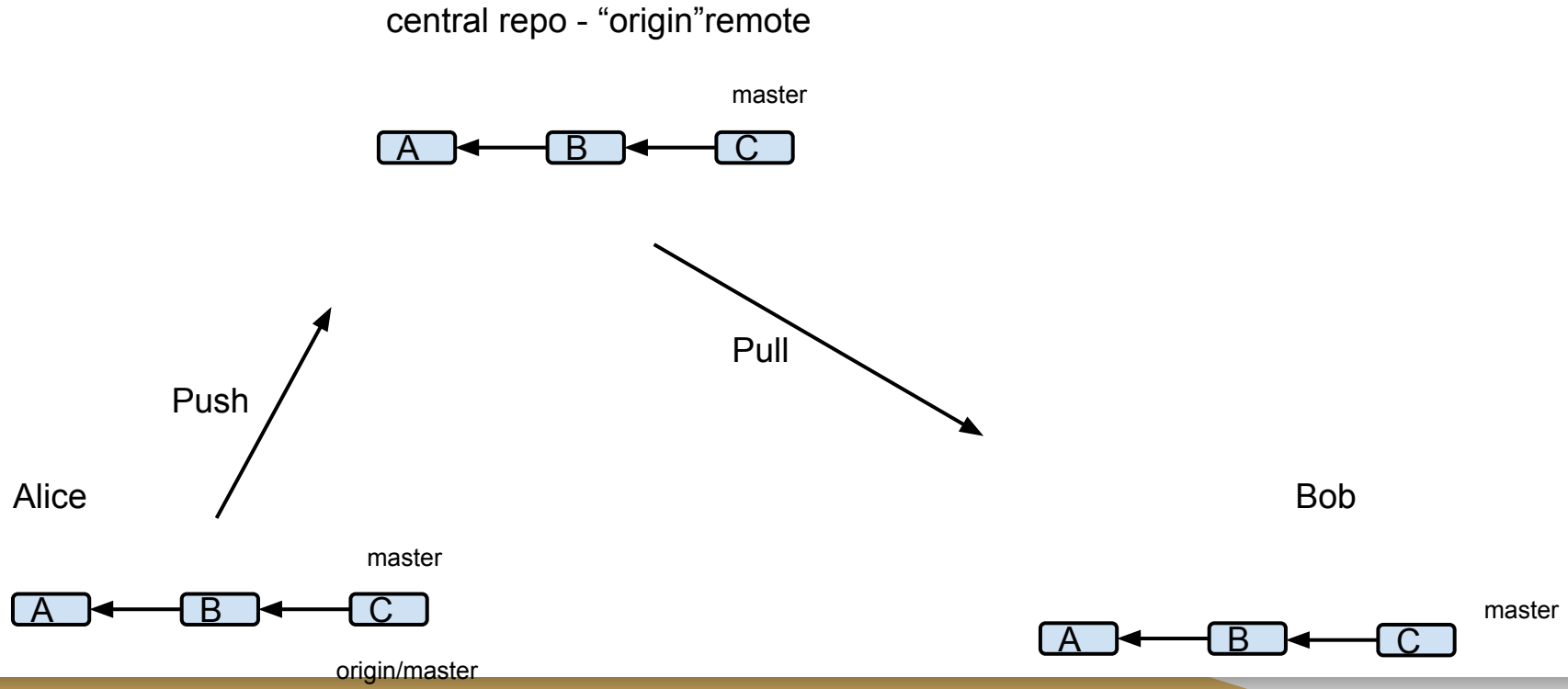
Remotes

- git **clone**
- remotes
- remote branches vs local branches
- push
- pull

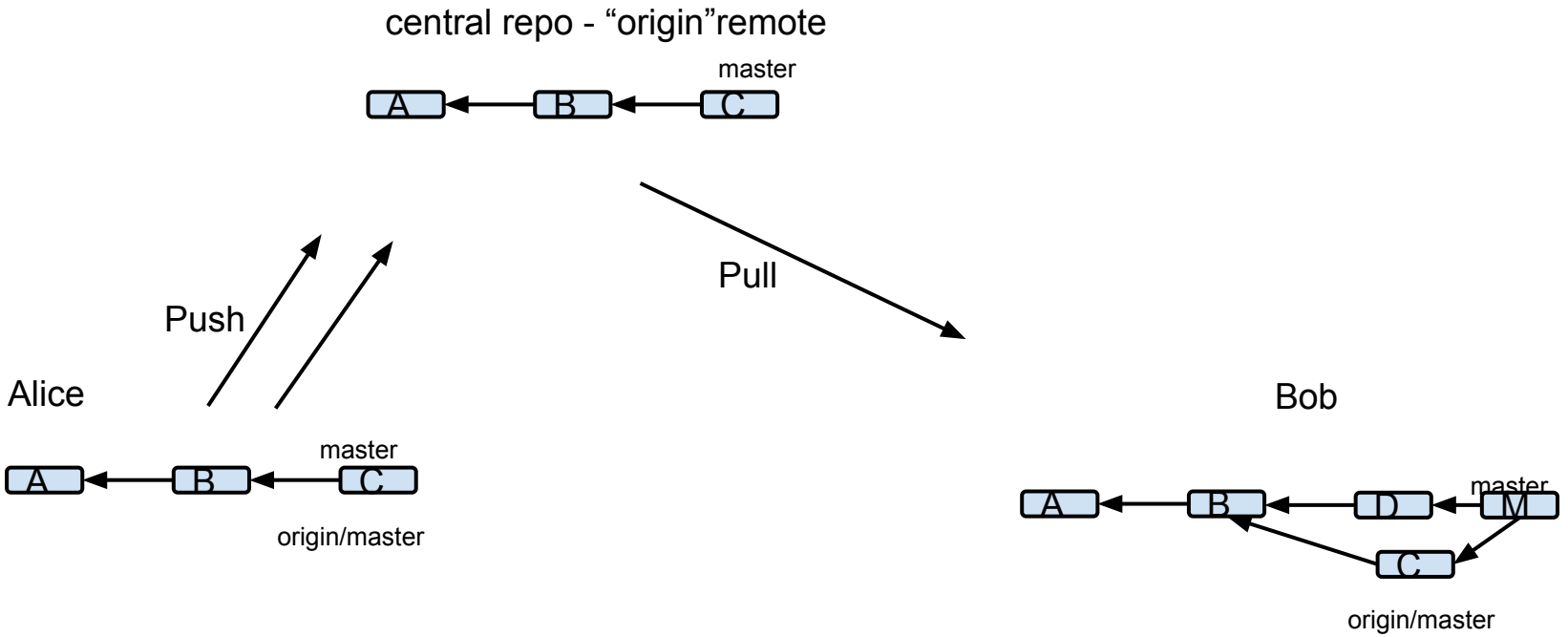
Clone


central repo - “origin”remote
master
A
B
Push
Clone
Alice
Bob
master
A
B
origin/master
master
A
B
origin/master

push/pull - the simple case



Push/pull with merging

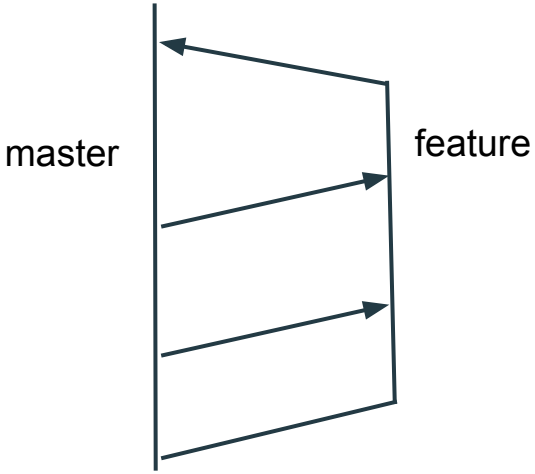


Feature branch workflow

1. create a feature branch
2. checkout feature branch
3. make commits until done with feature
4. checkout master
5. pull master
6. merge feature into master
7. make sure everything works
8. push master

Update your feature branch

- Merge from master to integrate latest changes
- fishbone pattern:



Remote branches

- You can push and pull other branches than master
- This way you can share work on a feature branch
- Branches that are local on the remote are reproduced as “remote branches” in your repository
- Remote branches cannot themselves be checked out
- Local branches can *track* remote branches

Tags

- label a version with a tag
- pointer to a commit
- if annotated it also has
  - comment, tagger, date
  - hash
- does not move (like branches do)
- cannot be checked out (like branches)
- must be pushed specifically

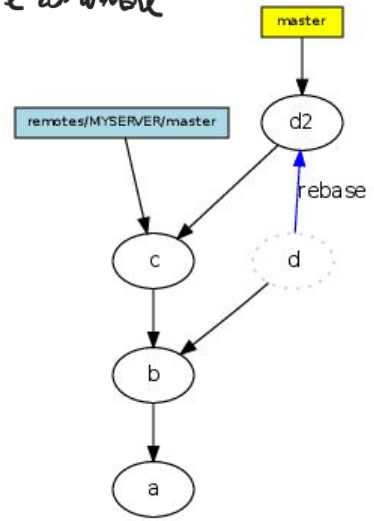
Rebase

cambiare la base di un branch da una commit a un'altra

- A rebase replays the differences introduced by commits, creating new “copied” commits.

il processo di muovere e combinare sequenze di commit

- cleaner history
- fewer commits
- don’t do it with pushed commits



References

gitref.org - quick walkthrough

git-scm.com/book - compete book, especially Chapter 2 (and preferably also chapter 3)

youtube.com/watch?v=ZDR433b0HJY - excellent video on basics (but very command line oriented)

youtube.com/watch?v=ig5E8CcdM9g - advanced video

How to use Git on GitHub

https://git-scm.com/book/en/v2/Getting-Started-About-Version-Control