Version Control – Agile Workflow with Git/GitHub

19/20 November 2019 | Guido Trensch (JSC, SimLab Neuroscience)

Content

Content

Motivation

- Version control is one aspect of configuration management (CM).

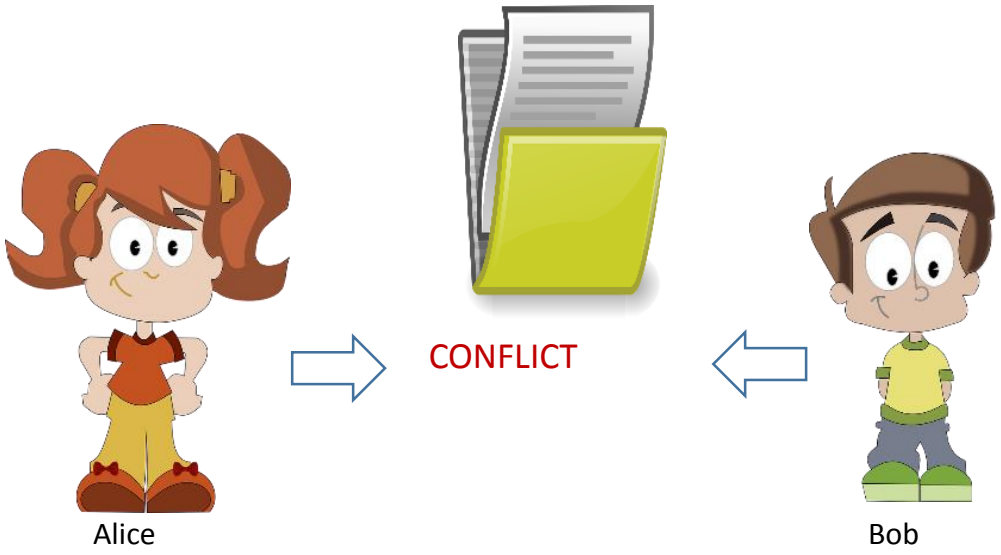
  The main CM processes are concerned with:

  - System building
    - *Preparing software for releases and keeping track of system versions.*
  - Change management
    - *Keeping track of requests for changes, working out the costs and impact.*
  - Release management
    - *Preparing software for releases and keeping track of system versions.*
  - Version control
    - *Keep track of different versions of software components and allow independent development.*

[Ian Sommerville, *"Software Engineering"*]

Motivation

- Keep track of different versions of software components
- Identify, store, organize and control revisions and access to it
- Essential for the organization of multi-developer projects is independent development
- Ensure that changes made by different developers do not interfere with each other
- Provide strategies to solve conflicts



Content

Version Control (VCS)

There are two types of version control systems:

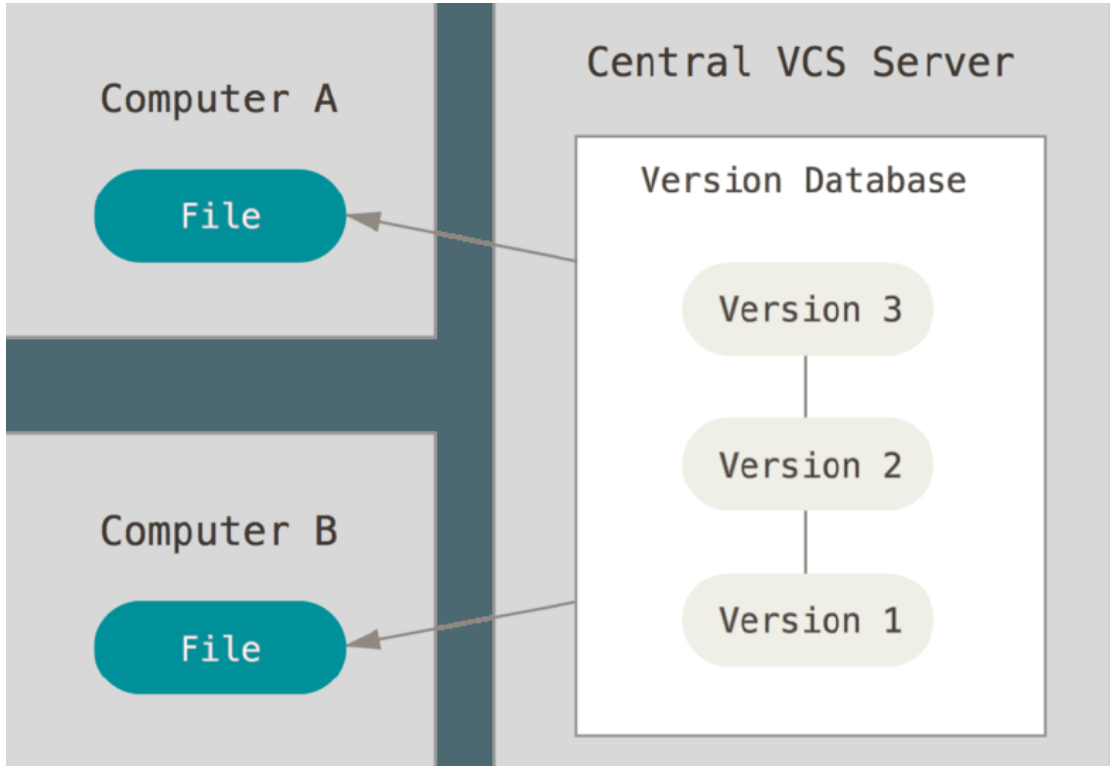
- Centralized systems
- Distributed systems

Version Control (VCS)

Centralized systems

- Maintain a single master repository

- *Revision Control System (RCS, 1982)*
- *Concurrent Versioning System (CVS, 1986)*
- *Subversion (SVN, 2000)*

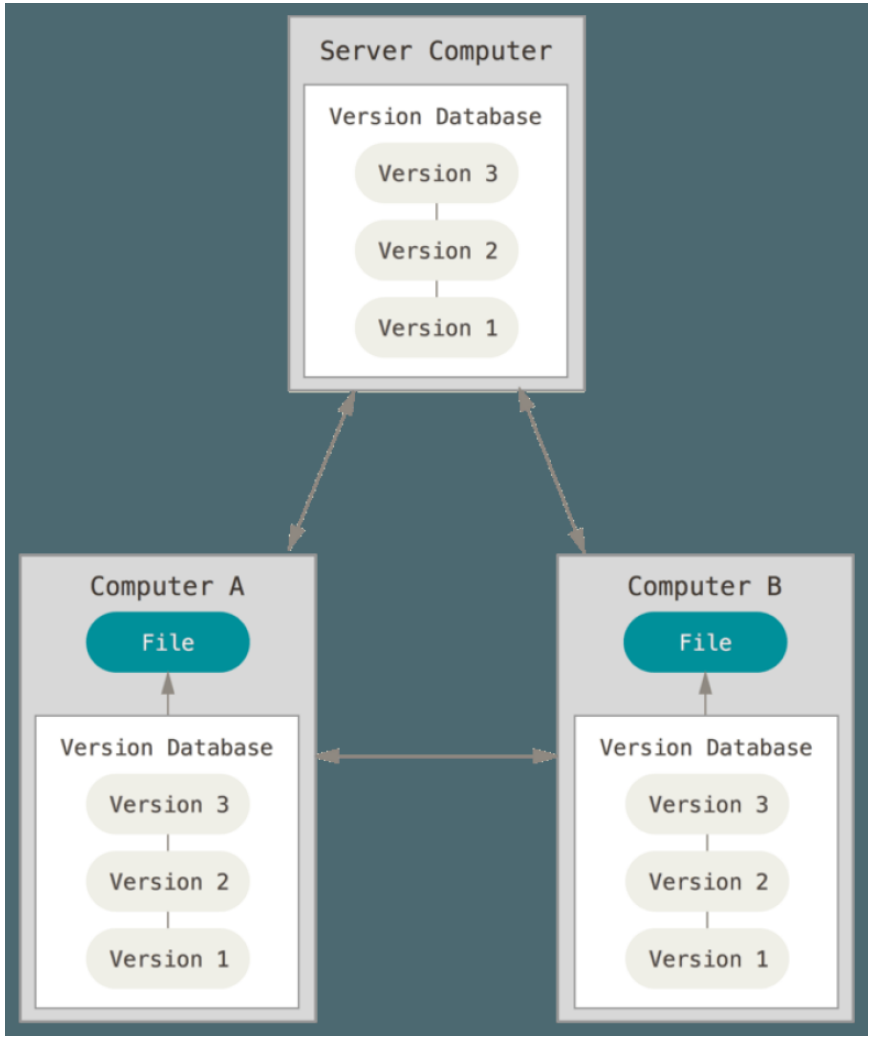


[Scott Chacon and Ben Straub, “Pro Git”]

Version Control (VCS)

Distributed systems

- Multiple versions of the component repository exist at the same time
- *Git (by Linus Torwalds, 2005)*



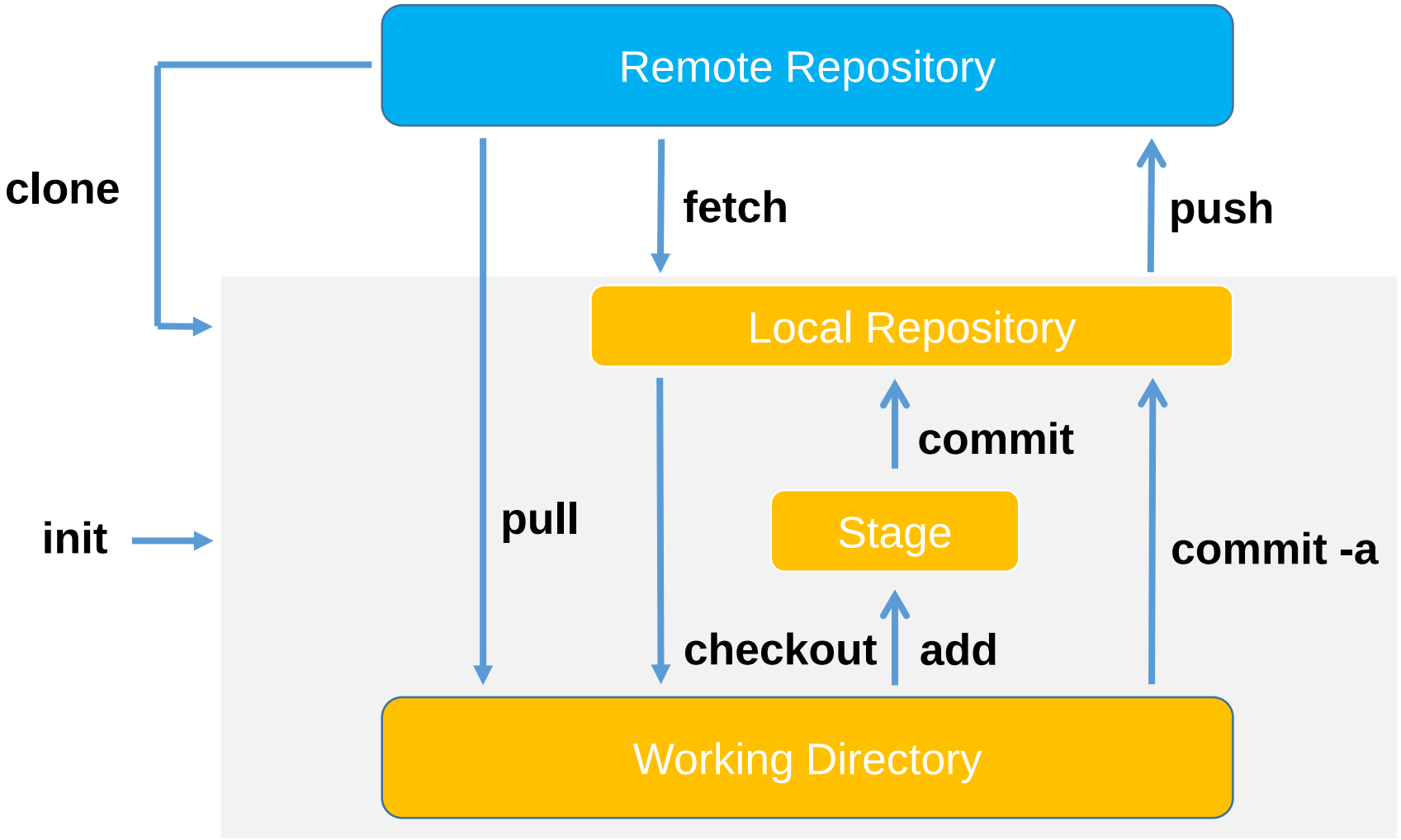
[Scott Chacon and Ben Straub, “Pro Git”]

“**Distributed version control** is essential for open-source development where several people may be working simultaneously on the same system without any central coordination.”

[Ian Sommerville, *“Software Engineering”*]

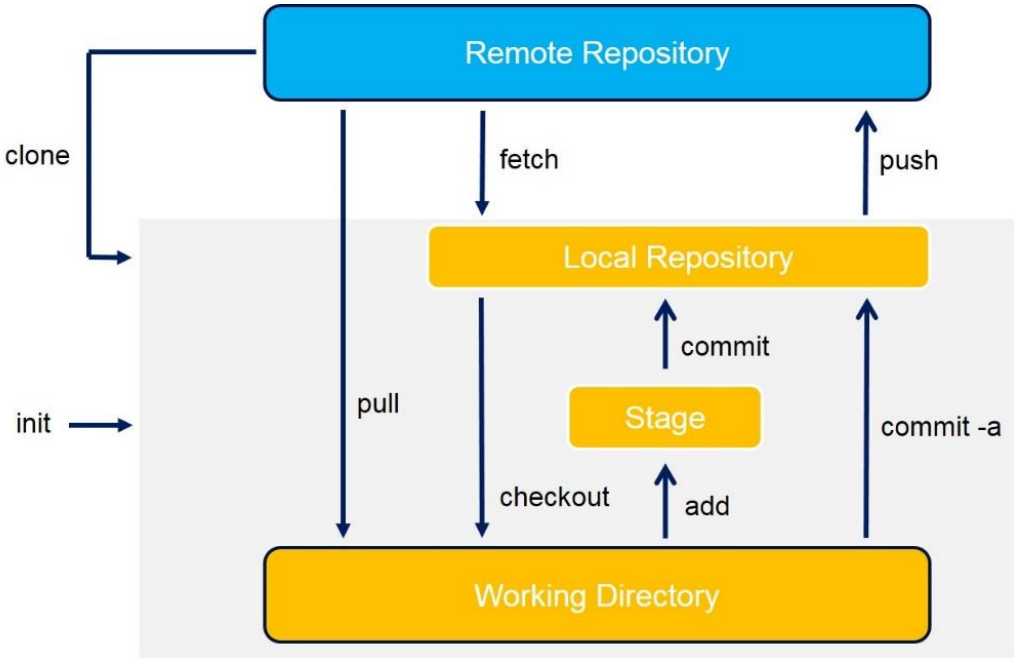
Content

Understanding Git



Understanding Git

Obtain a repository



- **git init**

  Create an empty Git repository or reinitialize an existing one.

- **git clone <repository>**

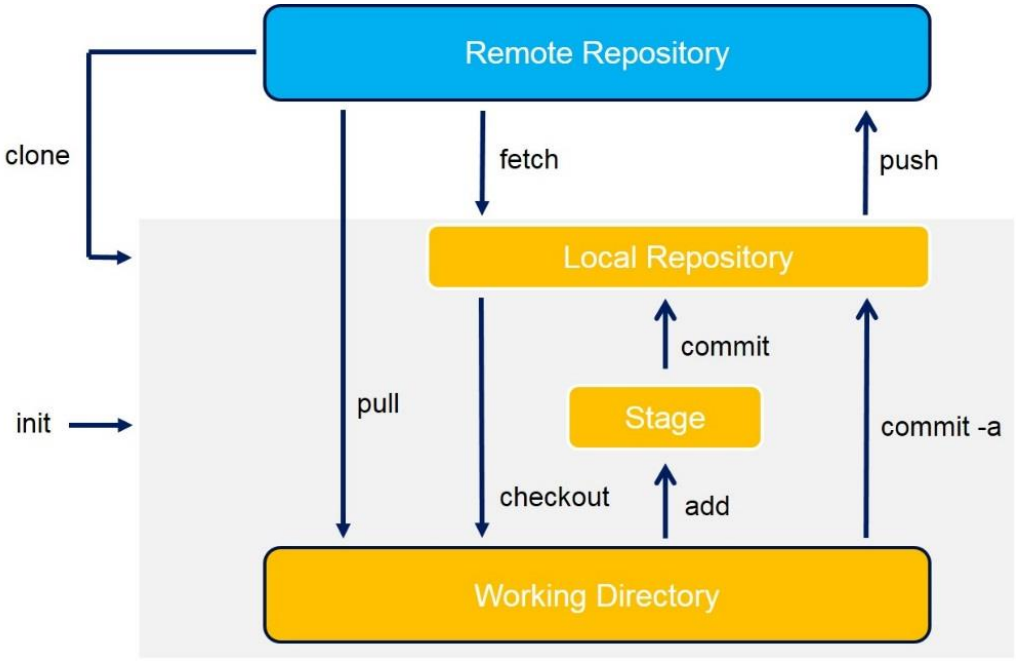
  Clone a repository into a new directory.

Example:

git clone https://github.com/gtrensch/SoftwareDevInScience2019.git

Understanding Git

Get changes from a remote repository



- **git fetch**

  Download objects and refs from another repository.

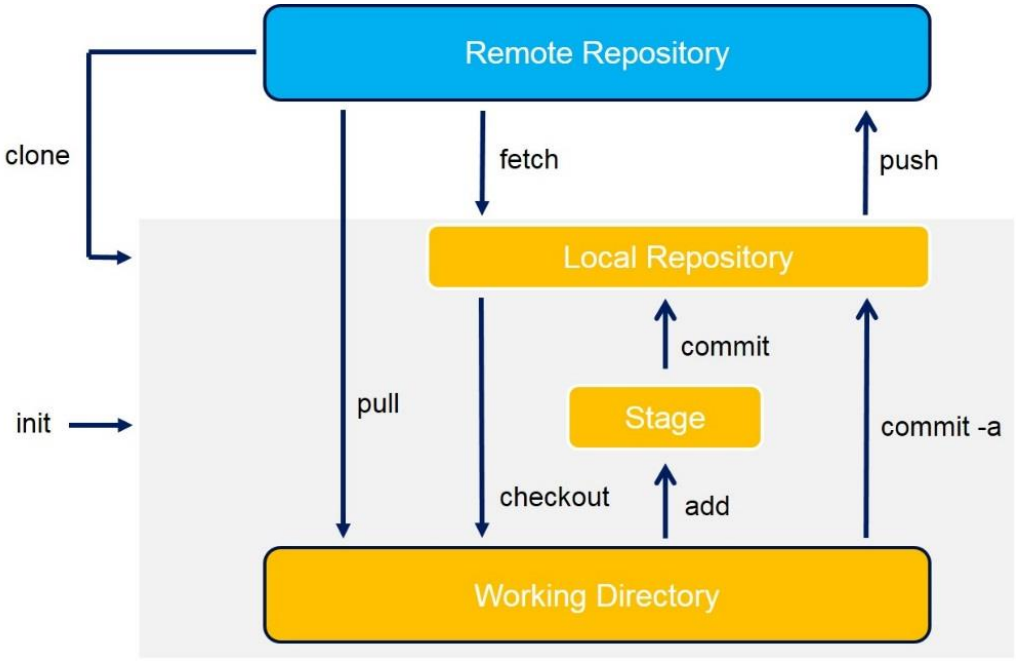
- **git pull <repository>**

  Fetch from and integrate with another repository or a local branch.

  (shorthand for *git fetch* followed by *git merge FETCH_HEAD*)

Understanding Git

Push changes to a remote repository



- **git add <file_name(s)>**

  Add file contents to the index (stage).
- **git commit -m <message>**

  Record changes to the repository.
- **git push <repository>**

  Update remote refs along with associated objects.

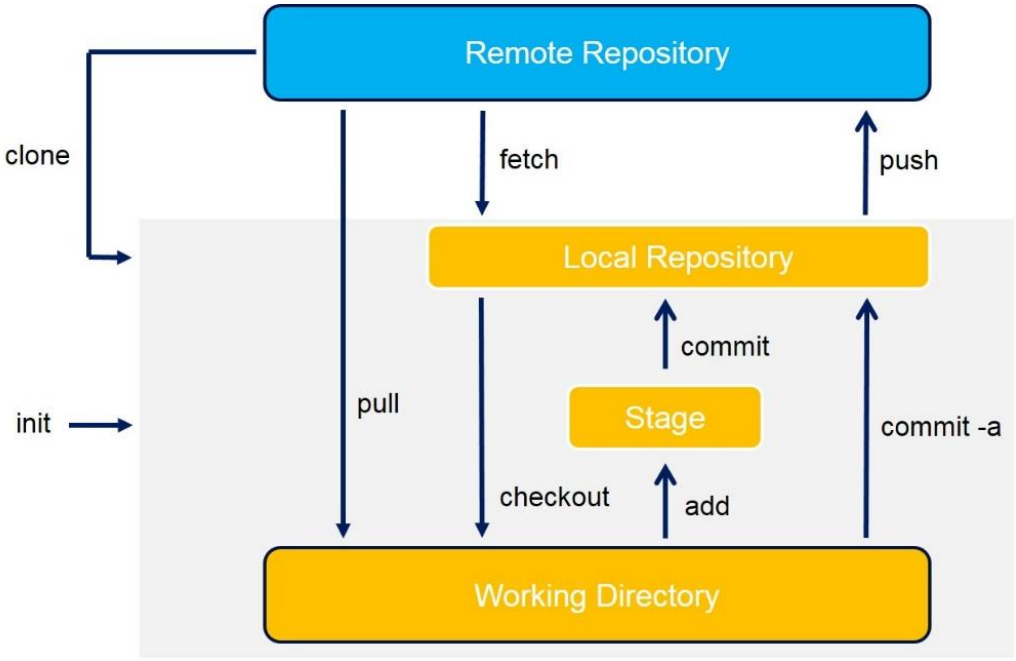
Understanding Git

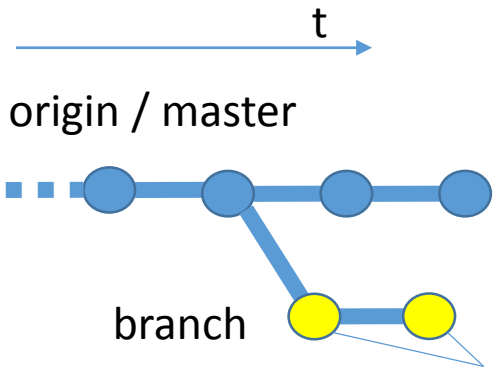
Branching

- **git checkout –b <new_branch>**
- **git branch –c <new_branch>**

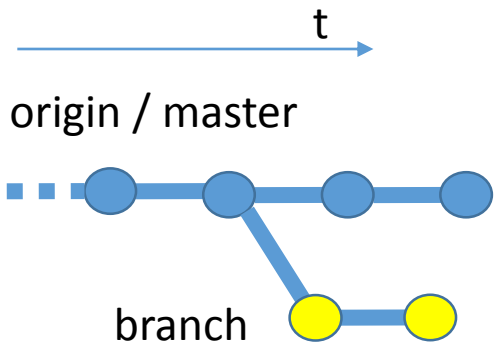
  Both commands creates a new branch.
- **git branch –d <old_branch>**

  Deletes a branch.





Commits waiting for review and merge into master.



Don‘t work on your master!

Understanding Git

Useful Git commands

- **git status**

  Show the working tree status.

- **git reset**

  Reset current HEAD to the specified state.

- **git diff**

  Show changes between commits, commit and working tree, etc.

- **git merge**

  Join two or more development histories together.

- **git remote**

  Manage set of tracked repositories.

Content

GitHub (Agile Workflow)

What is GitHub?

- Web-based Git repository hosting service
- Platform to share open-source projects
- ~ 5 million registered developer accounts
- ~10 million hosted projects
- Supports agile practices:
  - Code review workflow
  - Continuous Integration and Delivery (CI/CD)
    - Coupling with Travis CI
  - Basic project management

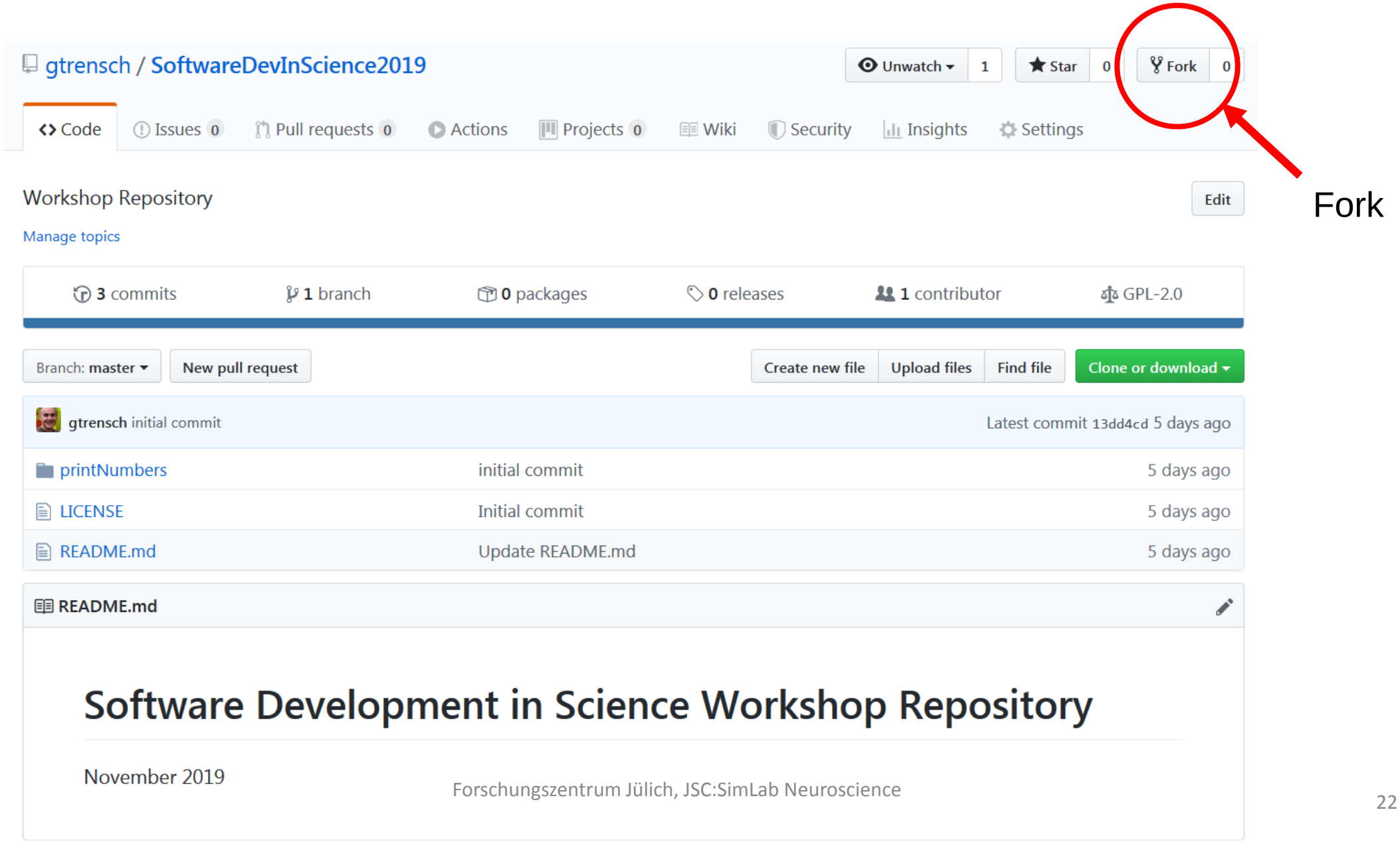
GitHub

www.github.com

GitHub (Agile Workflow)

- Issue tracker
- Wiki
- GitHub **”forking”**:
  - Enables you to copy a repository from one user‘s account.
  - You can make changes under your own account and share your work by and issue a so called **”pull request”**.

GitHub (Agile Workflow)

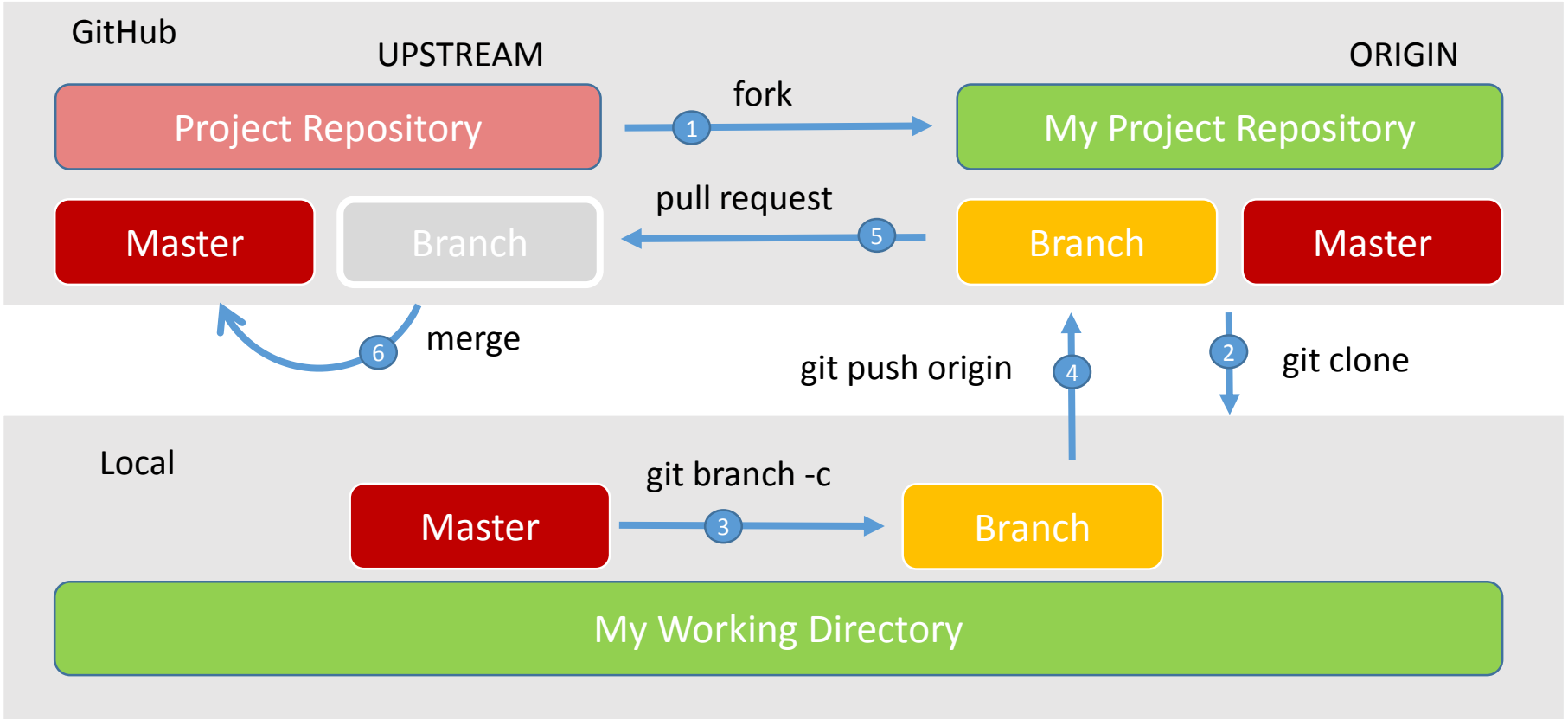


Fork

GitHub (Agile Workflow)

GitHub – Git – Workflow

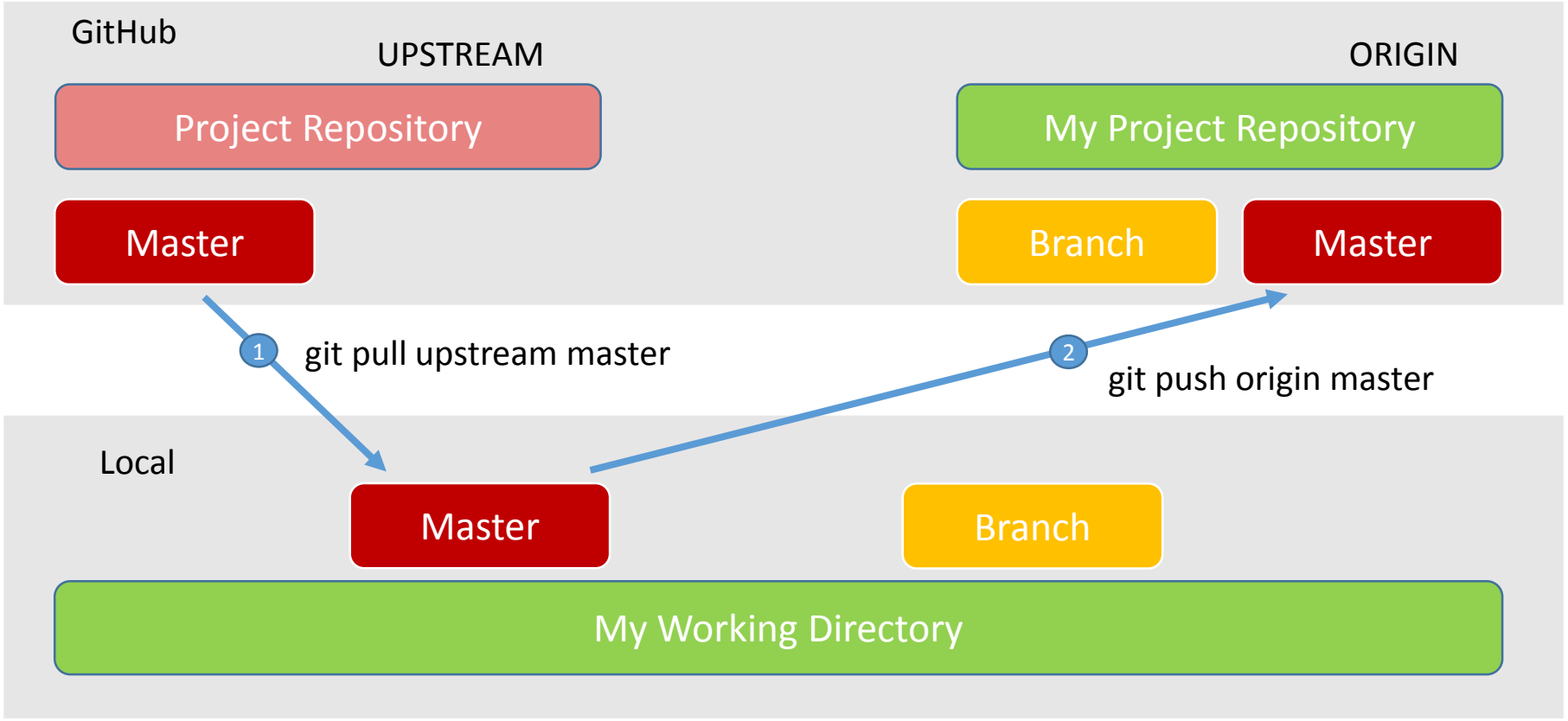
- Not rules, but guidelines developers can follow.



GitHub (Agile Workflow)

GitHub – Git – Workflow

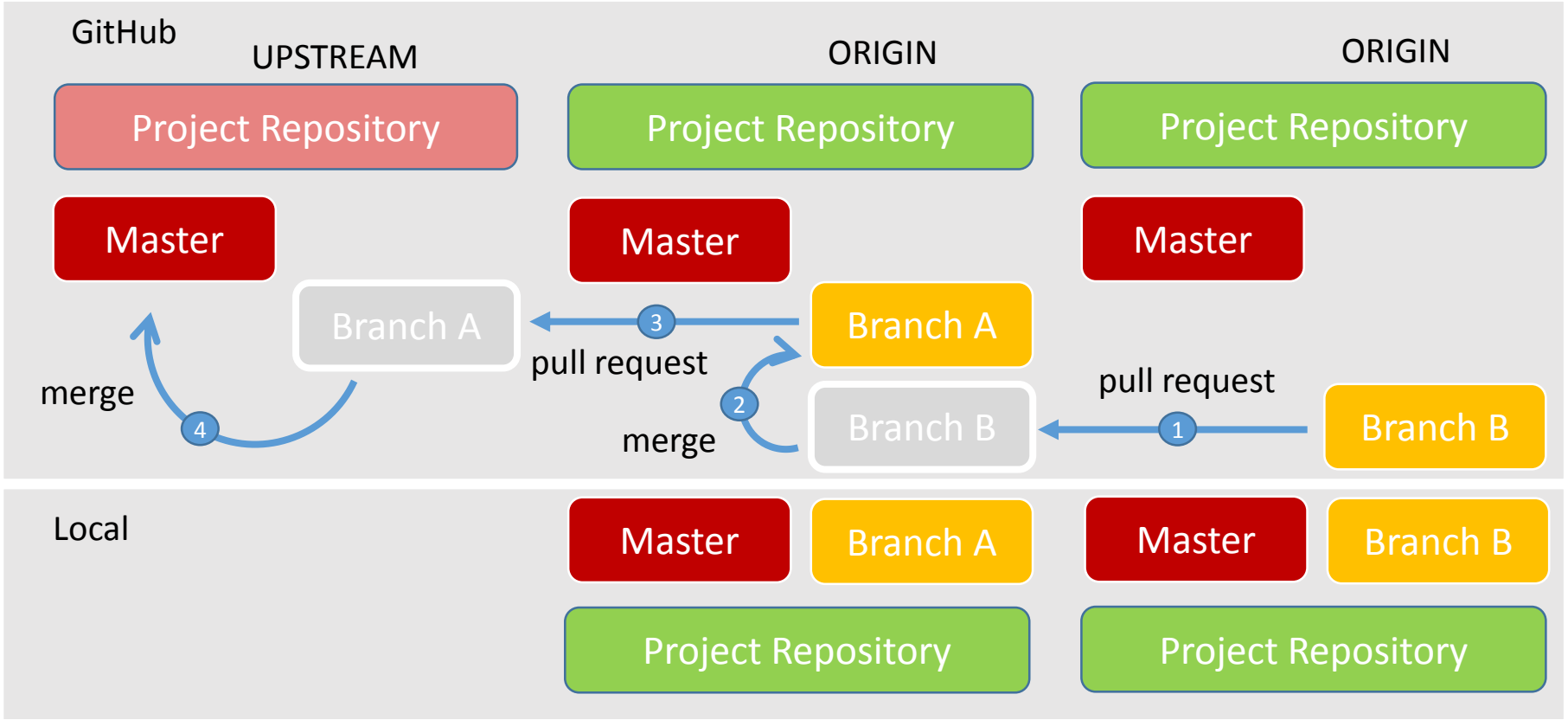
- Actualize the Master



GitHub (Agile Workflow)

GitHub – Git – Workflow

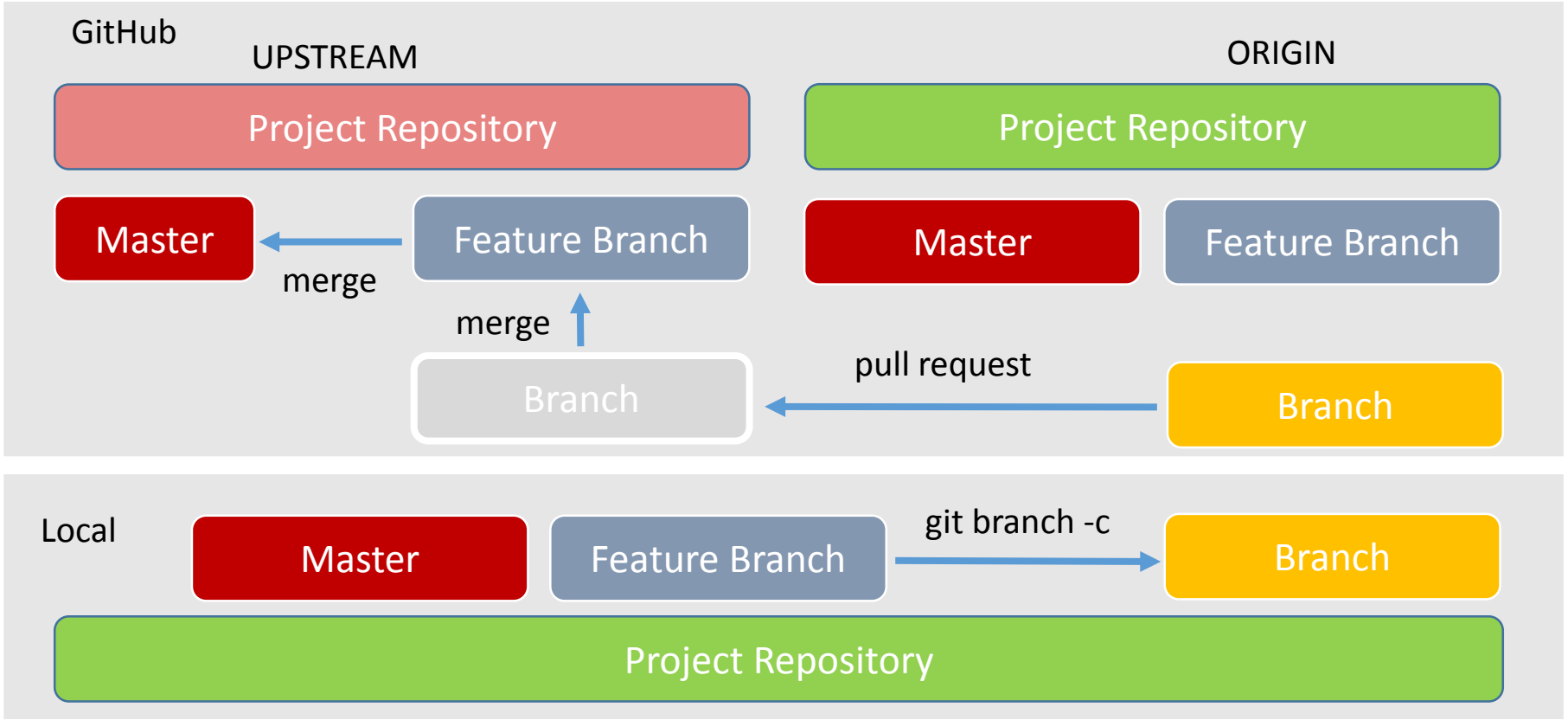
- Collaborate



GitHub (Agile Workflow)

GitHub – Git – Workflow

- Feature Branch



Content

References

- Everything you need to know about Git.

  https://git-scm.com/book/en/v2

- Git Reference

  https://git-scm.com/docs

- GitLab

  https://about.gitlab.com/