Aurélie Névéol, LIMSI-CNRS

Tools for Reproducibility & Collaborative Information Management

dépasser les frontières

Outline

- Challenges
  - Version management
  - Synchronous editing
  - … Beware of partial/relative information
- Solutions
  - Metadata
  - Cloud and collaborative editing
  - Versioning file systems
  - Version control systems

Tracking the Whole Information

Information integrity:

- Information can be distributed over multiple document sources
  - Reference to other documents
    - « see attached », .bib file in a latex document
    - library dependancies in code
  - Use of document or other IDs in corpus collections
    - Separate corpus file and gold standard key

    (Database management systems, e.g. EMS, Workbench)

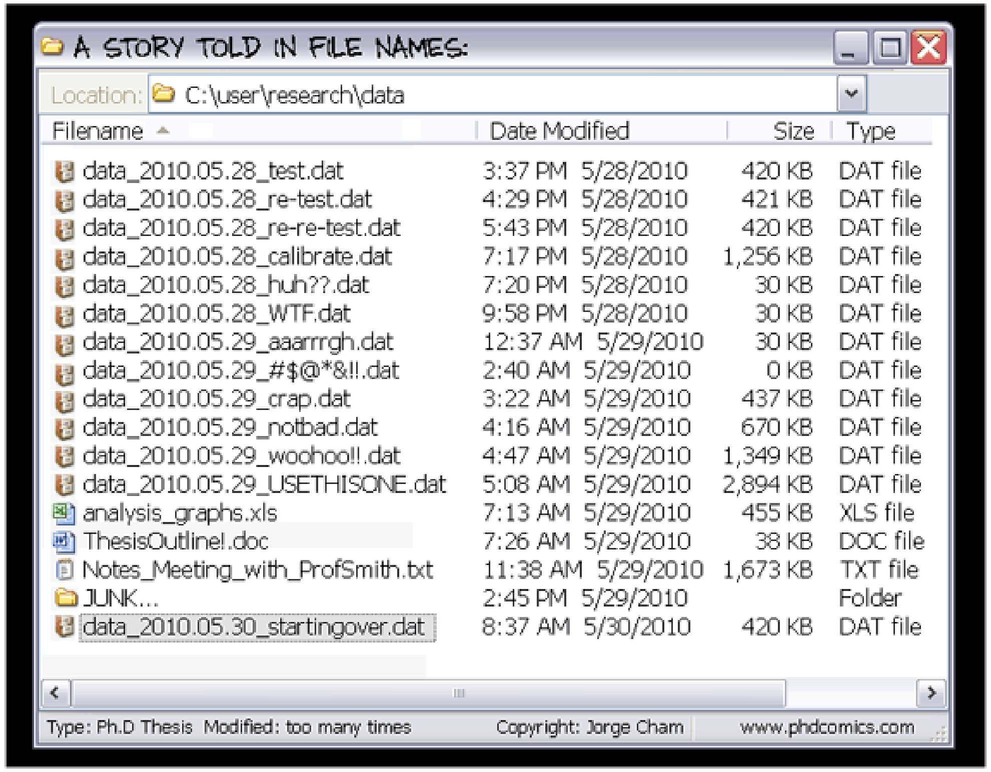
➯ Need to preserve integrity in updates

➯ Need to question integrity over time

Metadata

Common approaches to data management…

(from PhD Comics: A Story Told in File Names, 28.5.2010)



Filename	Date Modified	Size	Type
data_2010.05.28_test.dat	3:37 PM 5/28/2010	420 KB	DAT file
data_2010.05.28_re-test.dat	4:29 PM 5/28/2010	421 KB	DAT file
data_2010.05.28_re-re-test.dat	5:43 PM 5/28/2010	420 KB	DAT file
data_2010.05.28_calibrate.dat	7:17 PM 5/28/2010	1,256 KB	DAT file
data_2010.05.28_huh??.dat	7:20 PM 5/28/2010	30 KB	DAT file
data_2010.05.28_WTF.dat	9:58 PM 5/28/2010	30 KB	DAT file
data_2010.05.29_aaarrrgh.dat	12:37 AM 5/29/2010	30 KB	DAT file
data_2010.05.29_#$@*&!!.dat	2:40 AM 5/29/2010	0 KB	DAT file
data_2010.05.29_crap.dat	3:22 AM 5/29/2010	437 KB	DAT file
data_2010.05.29_notbad.dat	4:16 AM 5/29/2010	670 KB	DAT file
data_2010.05.29_woohoo!!.dat	4:47 AM 5/29/2010	1,349 KB	DAT file
data_2010.05.29_USETHISONE.dat	5:08 AM 5/29/2010	2,894 KB	DAT file
analysis_graphs.xls	7:13 AM 5/29/2010	455 KB	XLS file
ThesisOutline!.doc	7:26 AM 5/29/2010	38 KB	DOC file
Notes_Meeting_with_ProfSmith.txt	11:38 AM 5/29/2010	1,673 KB	TXT file
JUNK...	2:45 PM 5/29/2010		Folder
data_2010.05.30_startingover.dat	8:37 AM 5/30/2010	420 KB	DAT file

Type: Ph.D Thesis Modified: too many times Copyright: Jorge Cham www.phdcomics.com

Source: http://www.phdcomics.com/comics.php?f=1323

Metadata management

Simple work practices

- File and document editing policies

With advantages

- Quick and easy, no learning curve

… and drawbacks

- Metadata piles up over time and user input
- Version control is hazardous
- Stability over time is questionable

Versioning file systems

A versioning file system is any computer file system which allows a computer file to exist in several versions at the same time

- Sample tool: RCS
- Different from backup systems

https://www.**wikipedia**.org/

Versioning file systems

Advantages

– Easy and transparent to the user

– Changes are dated and old versions available

Drawbacks

– Some training required to use the tool

Cloud, collaborative web

A series of tools

- Doodle, Dropbox, Framapads, GoogleDocs, Skype, etc.

With advantages

- Ubiquitous access (from multiple places and devices)
- Multiple users can edit, merge seamless

… and drawbacks

- Connexion required
- Key aspects are provider dependent
  - Security: data is physically stored by a provider
  - Privacy: who has access to the data?
  - Stability over time

Version control systems

A *version control system* is a software tool that manages changes to documents, computer programs, large web sites, and other collections of information.

- Changes are identified by a number or letter code.
- Each revision is associated with a timestamp and the person making the change.
- Revisions can be compared, restored, and with some types of files, merged.
- Sample tools: sccs, cvs, rcs, svn, git

*https://www.**wikipedia**.org/*

Version control systems

- Advantages
  - Ubiquitous access (from multiple places and devices)
  - Multiple users can edit via local/distant copies
  - Does not require a connexion at all times
  - Key aspects user dependent: security, privacy, stability
- Drawbacks
  - Technical complexity: need a system administrator
  - Github ~cloud
  - Merge – dealing with concurrent editing sometimes tricky

In practice, what to use when?

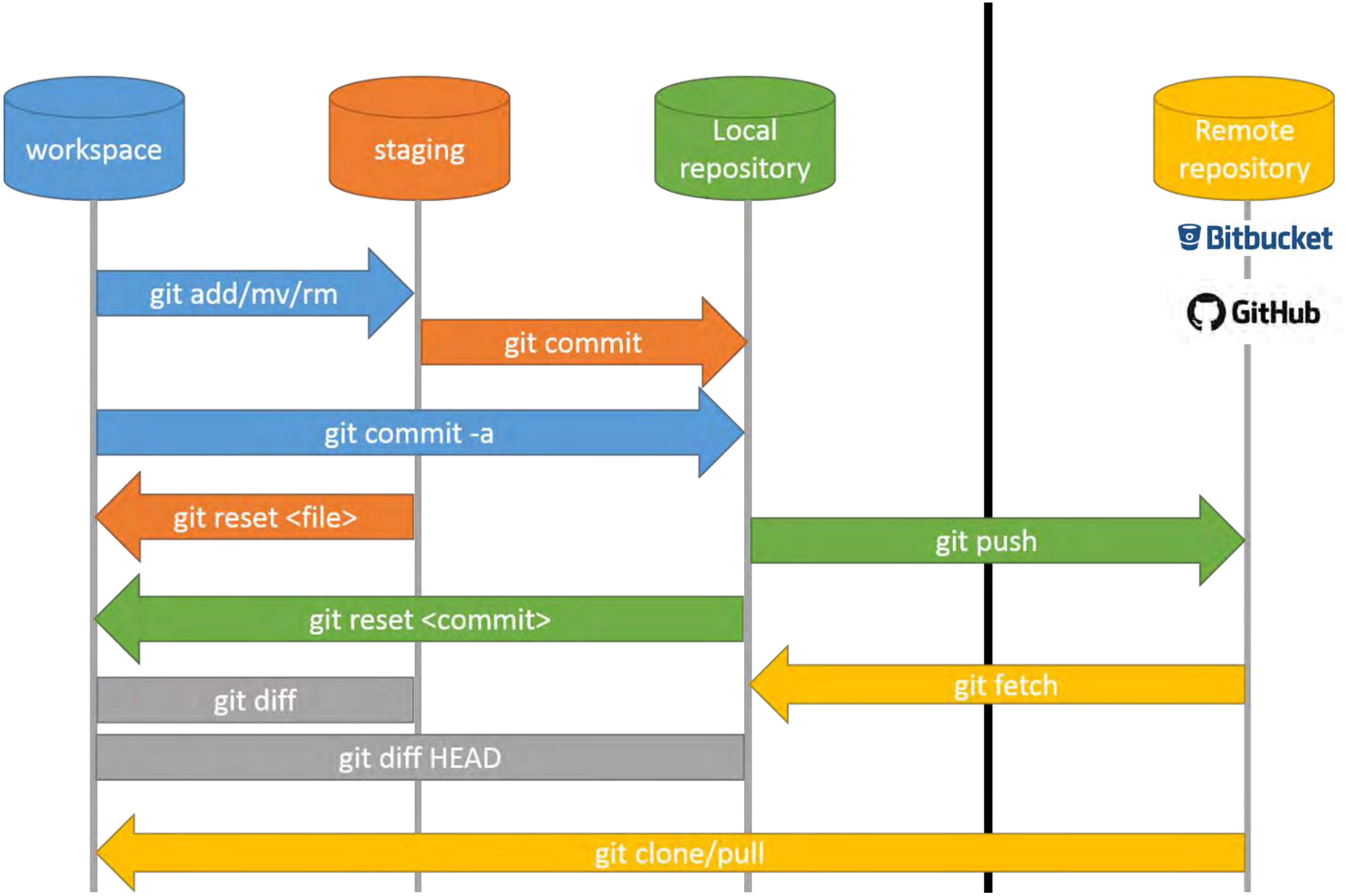
- By yourself - rush, small scale need): metadata
- By yourself - all needs: RCS, GIT, SVN
- Multiple users - low privacy or security requirements: cloud
- Multiple users - all needs: GIT/SVN

- GIT/SVN all the way…
  - More details now
  - Pointers to tools & how to install

A Brief Introduction to Version Control

Created by Linus Torvalds, in 2005

git (noun) : [*british informal*] An unpleasant or contemptible person



http://www.moxie.io/images/git-operations.png

Download Git

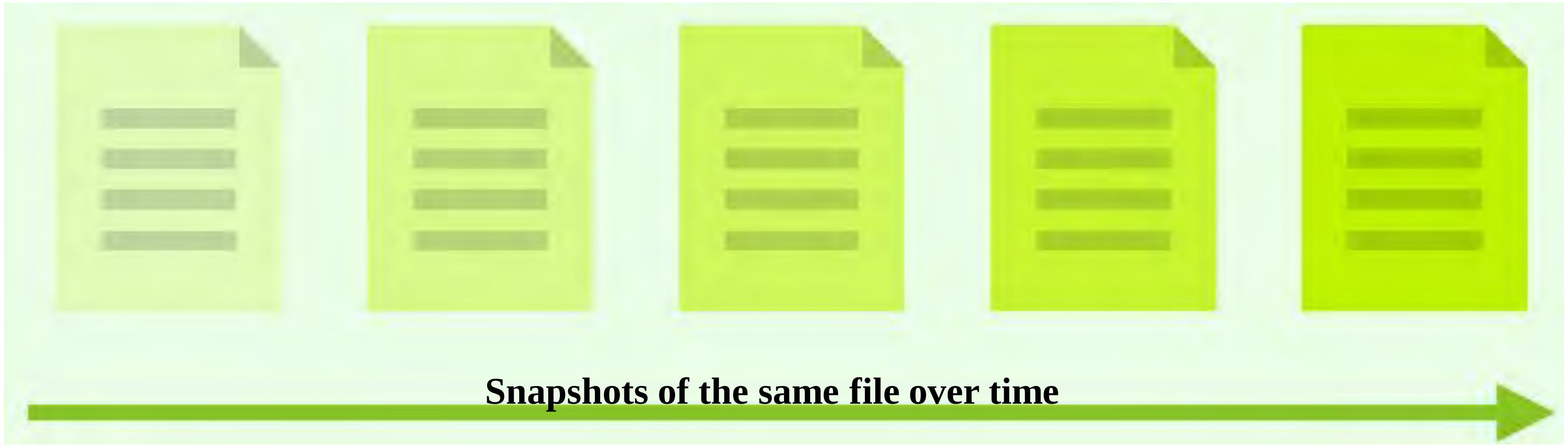
https://git-scm.com/download

Available for all major platforms
OS X, Windows, Linux

Git Graphical User Interfaces

https://git-scm.com/download/guis

Git organises snapshots for you



https://kevinchen.co/assets/blog/git-for-hackathons/git-for-hackathons-slides.pdf

How does it work?

`git` <u>`command`</u> <u>`options`</u>

(or equivalent button clicking
in your favorite GUI)

Repository / « Repo »

A folder where Git is tracking changes

Make a new repository

```
cd my_projcet
git init
```

commit

- Create a snapshot of your repository
- Commit the changes you have made

Steps to *commit*

`git status` (which files changed?)

`git diff` (which lines changed?)

`git add my_report.tex` (I want this file in my next commit)

`git add pic.jpg papers.bib` (**these** files too!)

`git commit` (OK, save a snapshot of what I just added)

`git log` (Show me the commit history)

Commit message should have a concise summary.

Put it on the first line. 70 characters or less.

Commit message should have a detailled explanation.

Think lab notebook. Wrap your text at 70 characters

Informative message

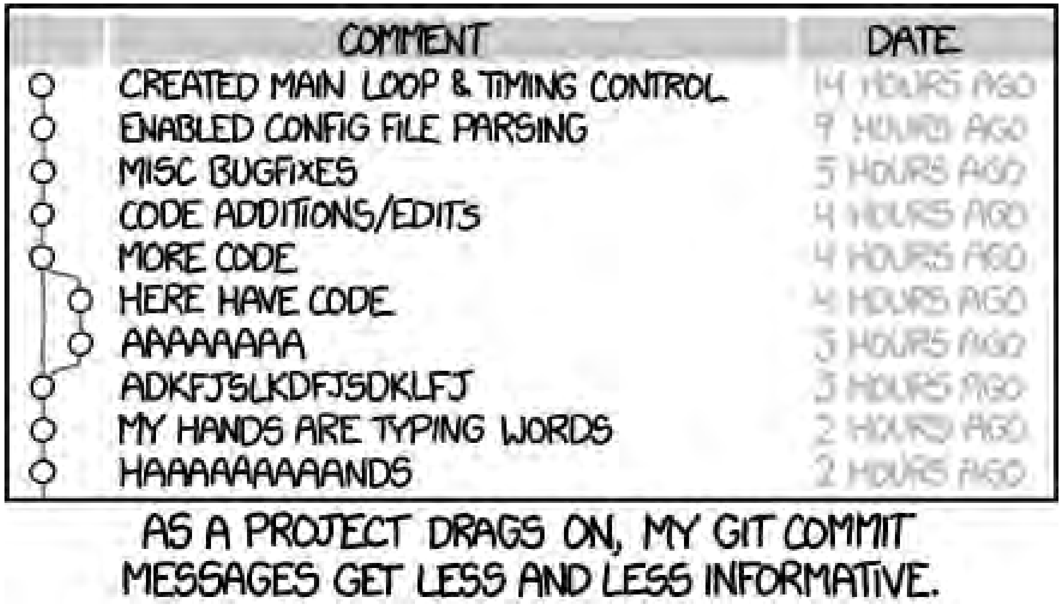
< 70 characters

```
Updated article classification features

The function compute_features now produces additional features related
to token characteristics and external clusters.
```

Limsi

Beware of commit message drifting

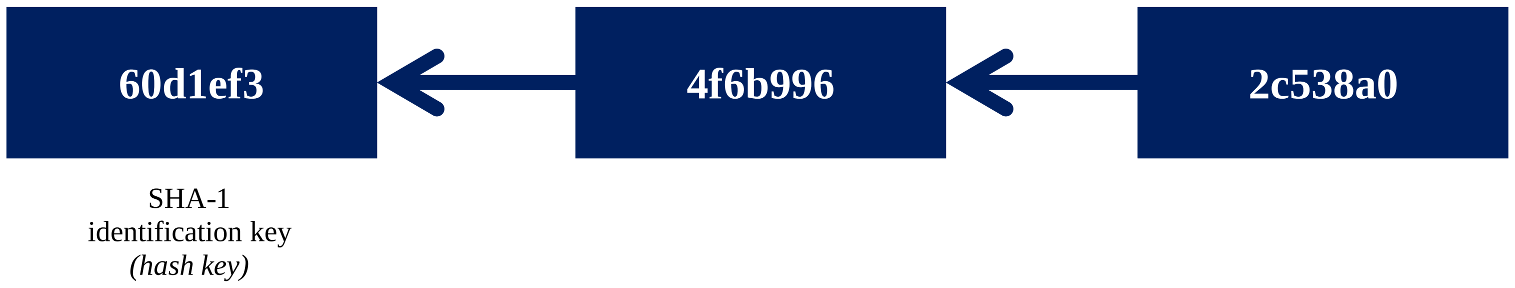


	COMMENT	DATE
○	CREATED MAIN LOOP & TIMING CONTROL	14 HOURS AGO
○	ENABLED CONFIG FILE PARSING	9 HOURS AGO
○	MISC BUGFIXES	5 HOURS AGO
○	CODE ADDITIONS/EDITS	4 HOURS AGO
○	MORE CODE	4 HOURS AGO
○	HERE HAVE CODE	4 HOURS AGO
○	AAAAAAAA	3 HOURS AGO
○	ADKFJSLKDFJSDKLFJ	3 HOURS AGO
○	MY HANDS ARE TYPING WORDS	2 HOURS AGO
○	HAAAAAAAANDS	2 HOURS AGO

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

https://m.xkcd.com/1296/

A History of your *commits*



You messed something up and you want to go back

```
git checkout – my_file
```

Go back to my_file version per most recent *commit*

```
git checkout 60d1ef3
```

Go back to the *commit* labeled as 60d1ef3
(and then you can branch out from there – coming up)

You broke something and you want to change history

```
git reset --hard 60d1ef3
```

Revert everything to the commit labelled 60d1ef3

```
git reset --hard
```

Revert everything to the most recent commit

What if you wanted that code but not at that moment?

Branching

Track separate versions of your code

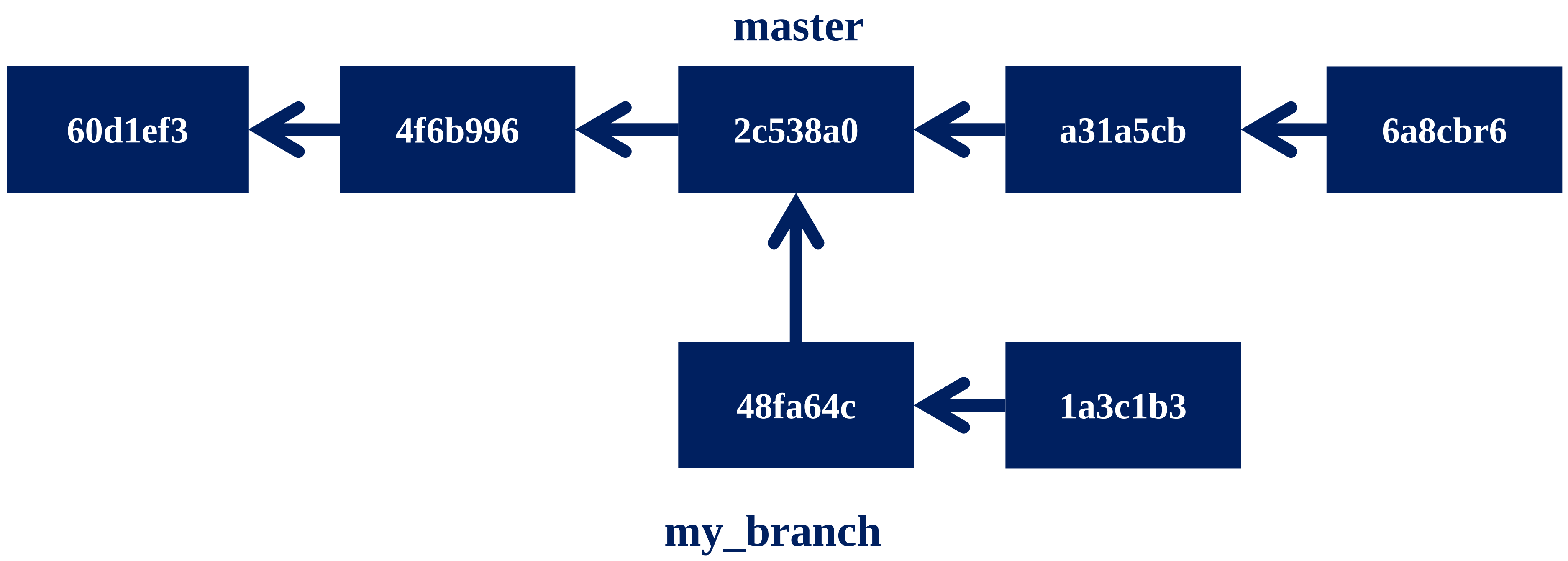
Making a branch

`git branch` (list the branches of the repo)

`git branch my_branch` (make a branch called `my_branch`)

`git checkout my_branch` (switch to `my_branch`)

Now you can *commit* changes to that branch.


master
60d1ef3
4f6b996
2c538a0
a31a5cb
6a8cbr6
48fa64c
1a3c1b3
my_branch

Take Home Messages

Git helps you
organize snapshots
of your projects

These snapshots are called *commits*

If you mess up,
you can always go back
as long as there’s been a commit.

Branches let you
try out new ideas
without losing access to the
version that works.

90 % of the time : `status / log / commit / push / pull`

8 % : `checkout / merge`

2 % : other commands

Further documentation

gitimmersion.com An interactive tutorial

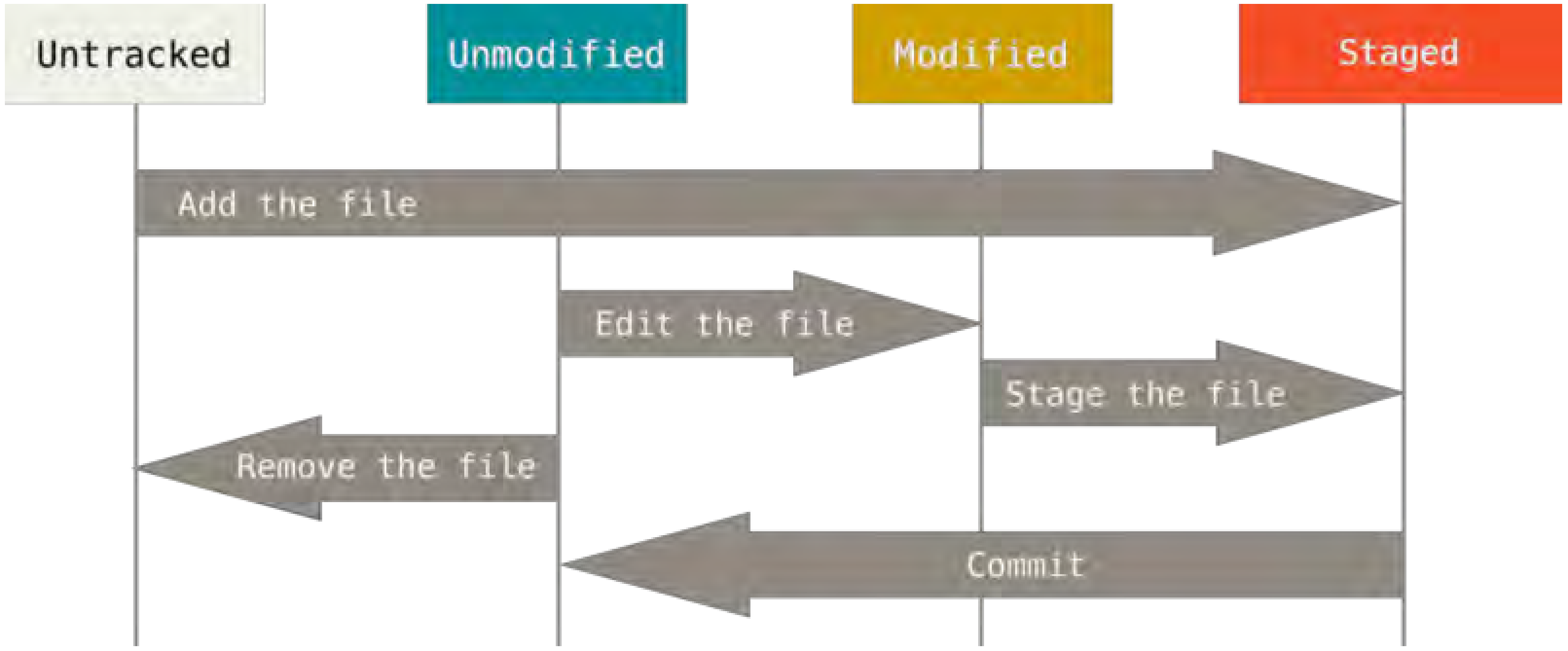
book.git-scm.com A detailed text book on Git

think-like-a-git.net Advance use of Git

nvie.com/posts/a-successful-git-branching-model/ on using branches

Acknowledgements

- Kevin Chen (Columbia University)
- Nicolas Grenier, Thomas Lavergne (LIMSI-CNRS)
- Patrick Paroubek (LIMSI-CNRS)
- Pierre Zweigenbaum (LIMSI-CNRS)



https://git-scm.com/book/en/v2/Git-Basics-Recording-Changes-to-the-Repository

Methods in Research on Research

A presentation delivered at the

first MiRoR training event
October 19-21, 2016
Ghent, Belgium

This project has received funding from the EU Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie Grant Agreement #676207

www.miror-ejd.eu @MirorProject