



# Version control with Git for scientists

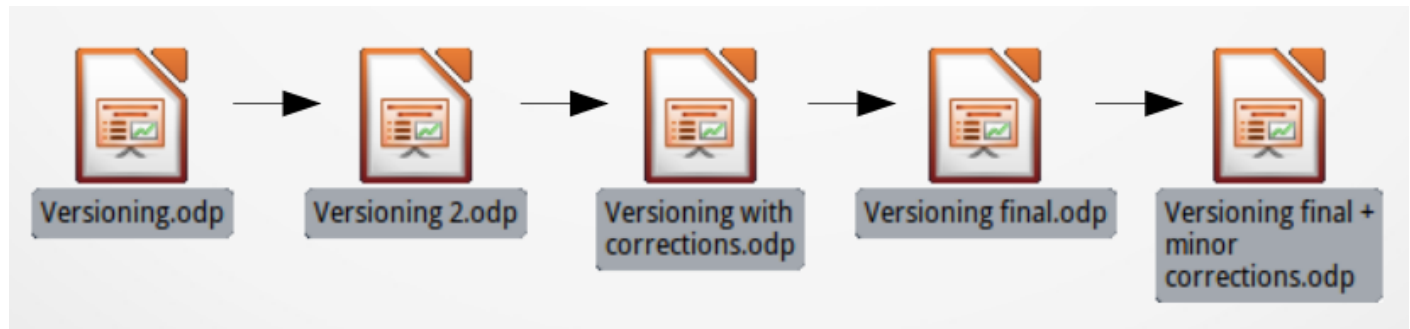
May 16, 2017  
PY Barriat & F. Massonnet

[http://www.climate.be:3000/TECLIM/Git\\_Training](http://www.climate.be:3000/TECLIM/Git_Training)

# Discuss

How do you manage different file versions ?

How do you work with collaborators on the same files ?



# Notions of code versioning

Track the history and evolution of the project  
think of it as a series of snapshots (commits) of  
your code

## Benefits:

- team work
- tracking bugs & recovering from mistakes

## Different usage:

- local
- client-server (SVN)
- distributed (Git)



# What is Git ?

## Version control system

Manage different versions of files

Collaborate with yourself

Collaborate with other people

## Why use Git

“Always remember your first collaborator is your future self, and your past self doesn’t answer emails”

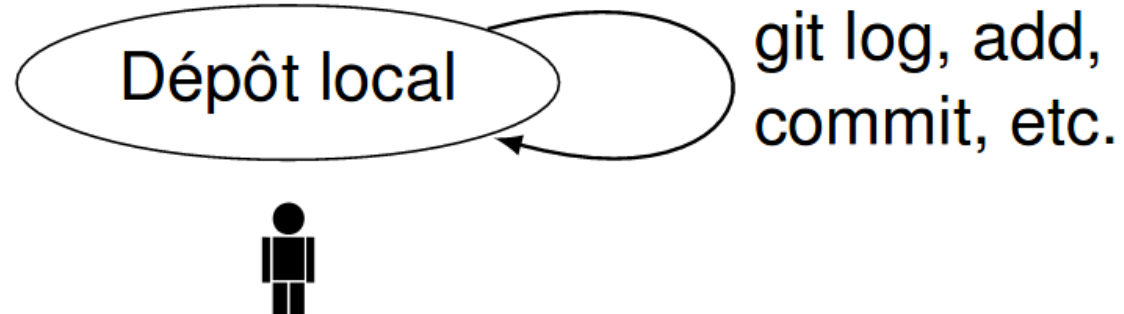
→ Christie Balhai



# What is Git good for ?

## Local

Backup, reproducibility



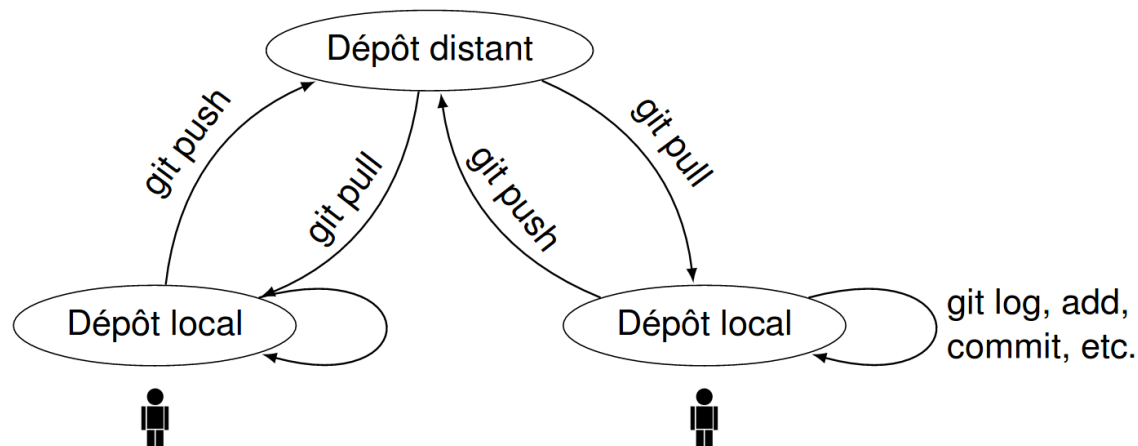
Utilisation locale



# What is Git good for ?

## Client-Server

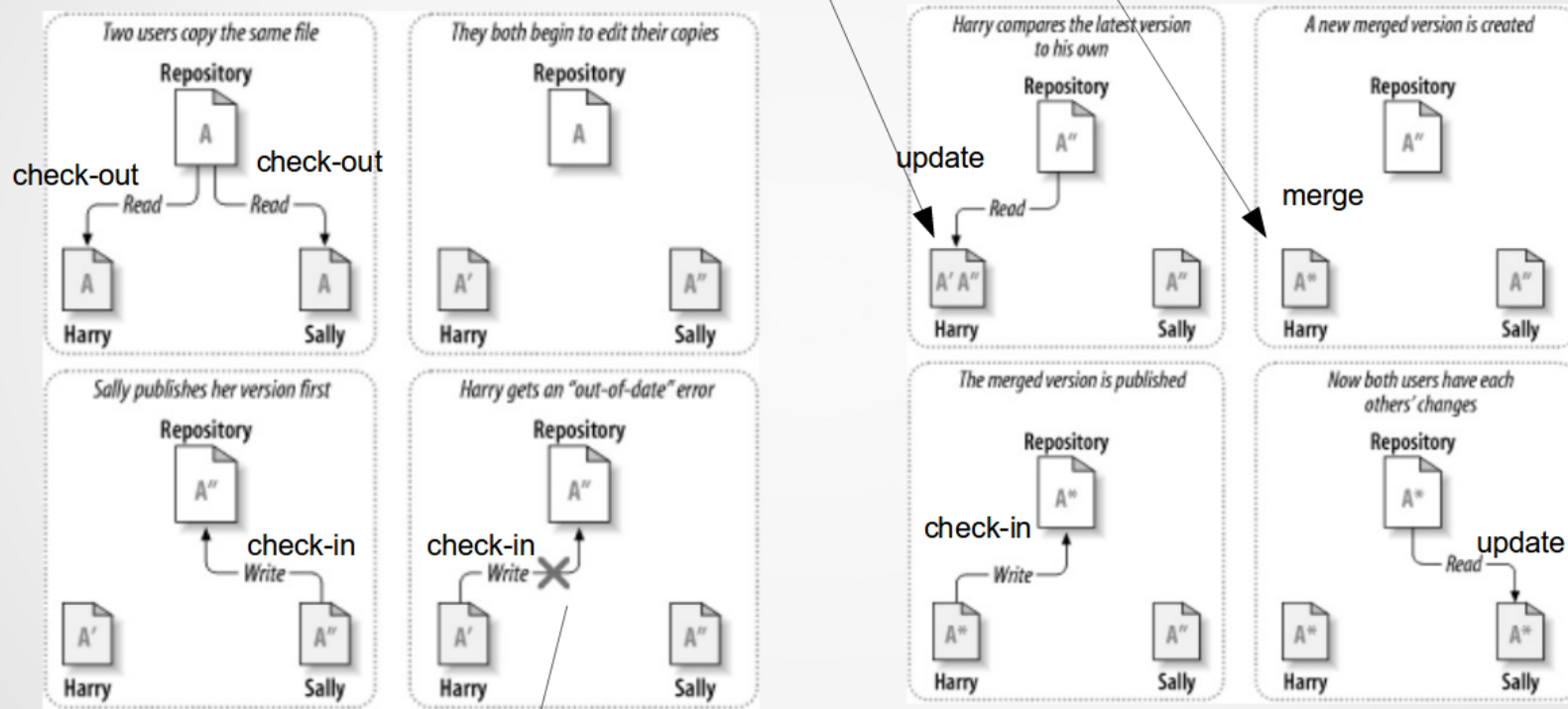
Backup, reproducibility, collaboration



Dépôt commun distant  
(Gitolite, Redmine, FusionForge, *GitHub*)

# Limitation

## copy-modify-merge solution

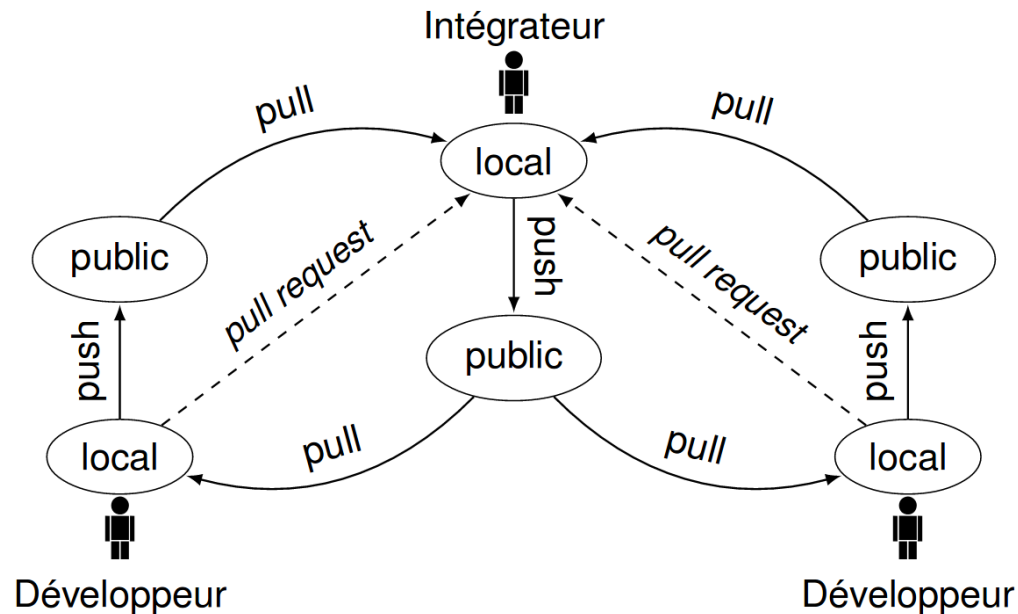


cvs commit: Up-to-date check failed for A

# What is GIT good for ?

## Distributed

Backup, reproducibility, collaboration, transparency



Utilisation distribuée (*GitHub*, *Linux*)



# Difference between Git & GitHub?

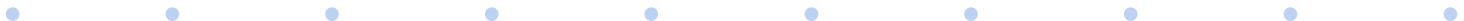
Git is the version control system service

Git runs local if you don't use GitHub

GitHub is the hosting service, a website

on which you can publish (push) your Git repositories and collaborate with other people

- It provides a backup of your files
- It gives you a visual interface for navigating your repos
- It gives other people a way to navigate your repos
- It makes repo collaboration easy (e.g., multiple people contributing to the same project)
- **It provides a lightweight issue tracking system**





## ... and GitLab vs GitHub vs others

GitLab is an alternative to GitHub

GitLab is free for unlimited private projects. GitHub doesn't provide private projects for free

And for ELIC, **Gogs** does the job

shares the same features (Dashboard, File browser, Issue tracking, Groups support, Webhooks, etc)

easy to install, cross-platform friendly,

uses little memory, uses little CPU power

... and 100% free



# Gogs: <http://www.climate.be:3000>

[Home](#)[Explore](#)[Help](#)[Sign In](#)

## Gogs - Go Git Service

### A painless self-hosted Git service



#### Easy to install

Simply **run the binary** for your platform. Or ship Gogs with **Docker** or **Vagrant**, or get it **packaged**.



#### Cross-platform

Gogs runs anywhere **Go** can compile for: Windows, Mac OS X, Linux, ARM, etc. Choose the one you love!



#### Lightweight

Gogs has low minimal requirements and can run on an inexpensive Raspberry Pi. Save your machine energy!



#### Open Source

It's all on **GitHub**! Join us by contributing to make this project even better. Don't be shy to be a contributor!

# Simple guide for getting started

## Checkout a remote repository

create a local working copy of a remote repository by running the command

```
git clone ssh://git@www.climate.be:3022/TECLIM/Git_Training.git
```

## Add & commit

You can propose changes (add it to the Index)

You can commit these changes to the HEAD

```
git add <filename>
```

```
git commit -m "Commit message"
```

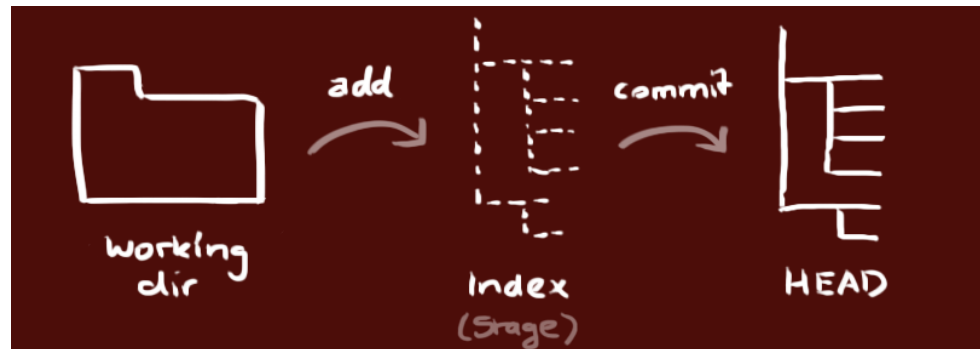
• • • • • • • • • •

# Git workflow

Your local repository consists of three "trees" maintained by git

the first one is your **Working Directory** which holds the actual files

the second one is the **Index** which acts as a staging area and finally the **HEAD** which points to the last commit you've made



# Simple guide for getting started

## Pushing changes

Your changes are now in the HEAD of your local working copy. To send those changes to your remote repository

```
git push
```

## Update

to update your local repository to the newest commit, execute

```
git pull
```

in your working directory to fetch and merge remote changes.



# Replace local changes

In case you did something wrong, which for sure never happens ;), you can replace local changes using the command

```
git checkout -- <filename>
```

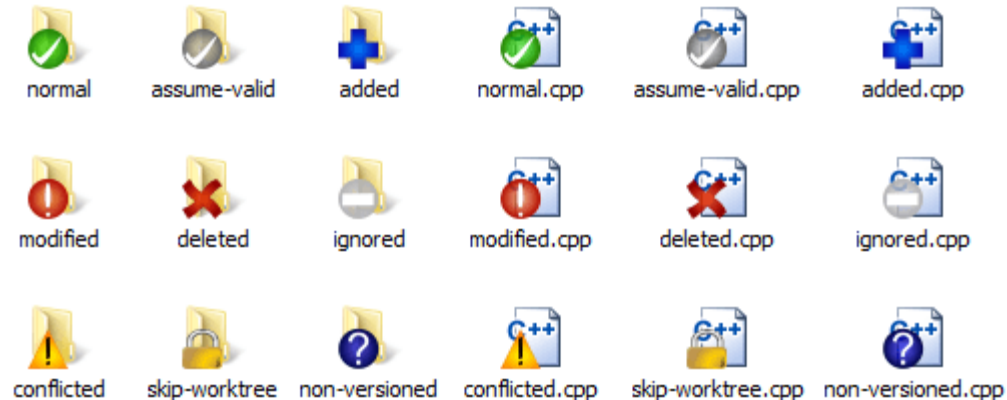
this replaces the changes in your working tree with the last content in HEAD. Changes already added to the index, as well as new files, will be kept



# Windows users

How commonly do programmers use Git GUIs instead of the command line?

use programs like SourceTree or TortoiseGit



But, to be familiar with Git, try the command line

(clone, push/pull, merge, rebase, log, tag, format-patch/am, bisect, blame, etc).

• • • • •



# Simple Git Exercises

First, configure your environment (just once)  
(on your laptop, on your ELIC account, etc)

```
git config --global user.name "Your Name"  
git config --global user.email "foo@bar.be"  
git config --global color.ui auto
```

Now, clone [http://www.climate.be:3000/TECLIM/Git\\_Training](http://www.climate.be:3000/TECLIM/Git_Training)  
Theses are very simple exercices to learn to manipulate  
git. In each folder, simply run `./create.sh` and follow the  
guide ;)

