Cycling and source code management

Alexandre Mary, Stéphane Martinez

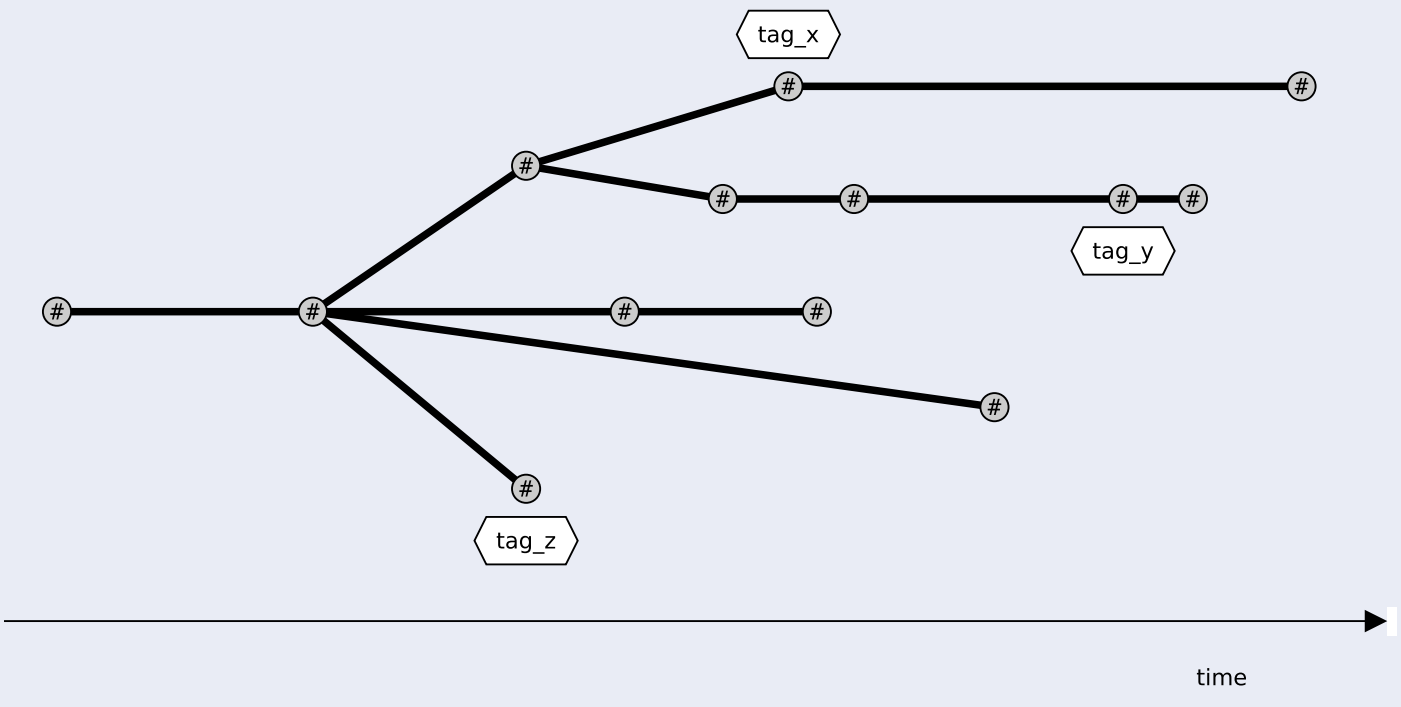
Code training days, 2019 Sept. 12th - Toulouse

Outline

Outline

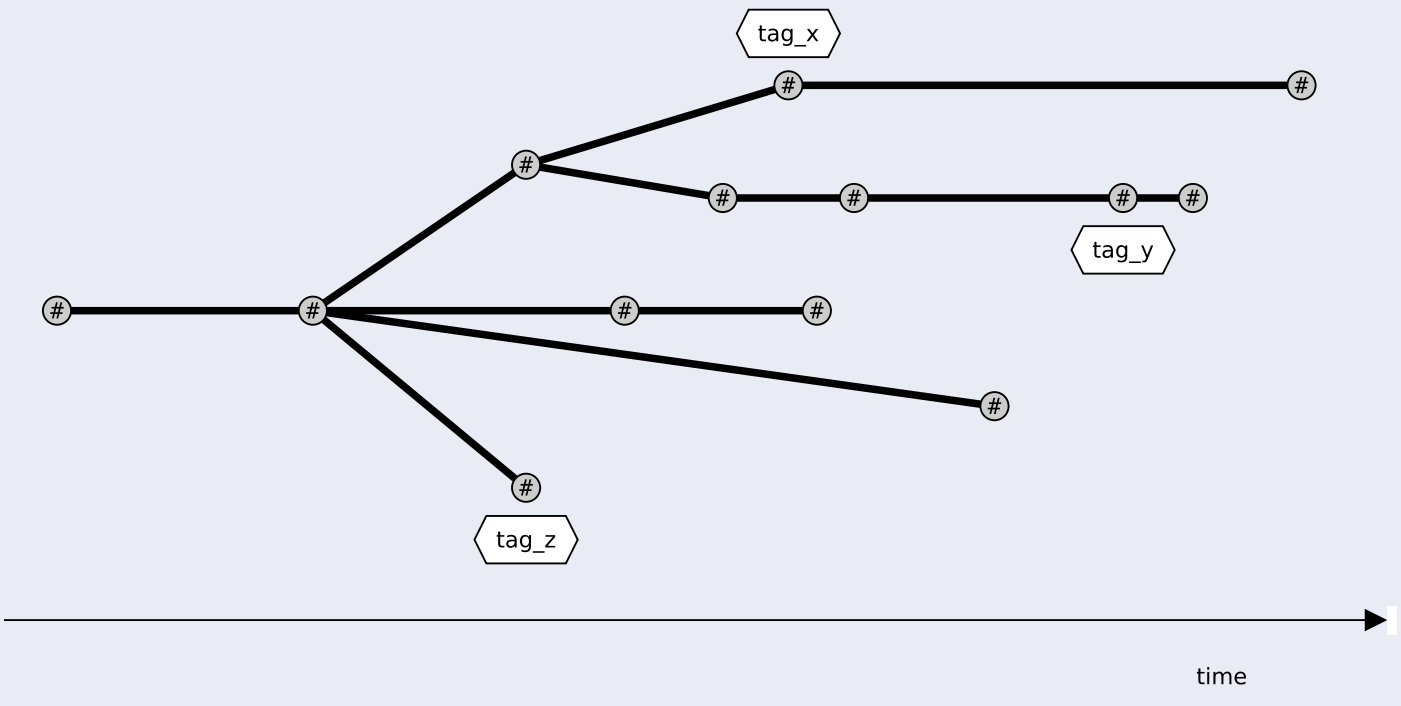
(Re)minders about GIT

- **commit** : save status of the code at a certain point (incl. ancestor), using a unique identifier



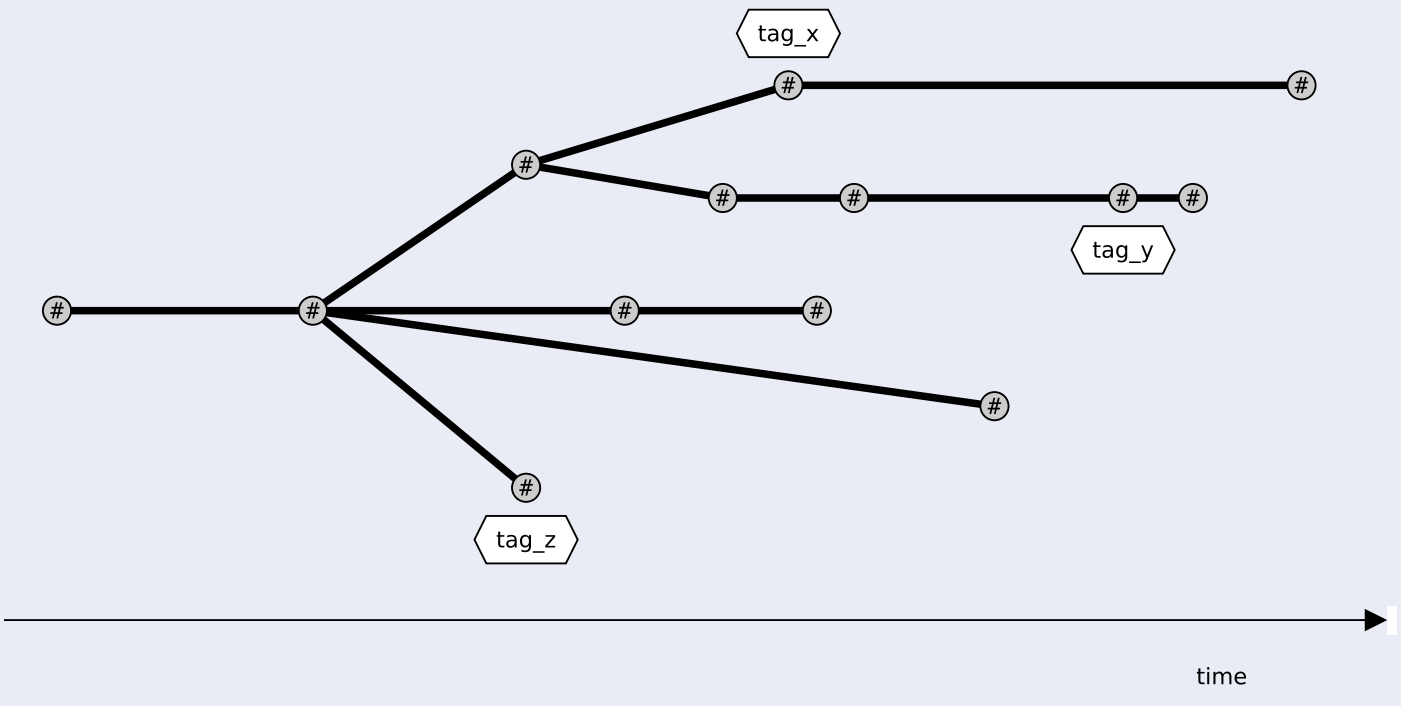
(Re)minders about GIT

- **commit** : save status of the code at a certain point (incl. ancestor), using a unique identifier
- **branch** : series of commits



(Re)minders about GIT

- **commit** : save status of the code at a certain point (incl. ancestor), using a unique identifier
- **branch** : series of commits
- **tag** : readable/tellable alias of a commit



Outline

What specific about GIT-GCO ?

- main cycles = tags on ***master*** branch

What specific about GIT-GCO ?

- main cycles = tags on ***master*** branch
- other official branches :
  - `_t1` (e.g. ***gco_CY46_t1***) : merging branch of all contributions based on CY46 to build CY46T1
  - `_r1` (e.g. ***gco_CY46T1_r1***) : merging branch of CY46T1 with CY46R1
  - `_bf` (e.g. ***gco_CY43T2_bf***) : bugfix branch, after declaration of main cycle
  - `_op` (e.g. ***gco_CY43T2_op2***) : MF operational branch

What specific about GIT-GCO ?

- main cycles = tags on ***master*** branch
- other official branches :
  - `_t1` (e.g. ***gco_CY46_t1***) : merging branch of all contributions based on CY46 to build CY46T1
  - `_r1` (e.g. ***gco_CY46T1_r1***) : merging branch of CY46T1 with CY46R1
  - `_bf` (e.g. ***gco_CY43T2_bf***) : bugfix branch, after declaration of main cycle
  - `_op` (e.g. ***gco_CY43T2_op2***) : MF operational branch
- user branches : `<user>_<maincycle>_<branchname>`
  e.g. **khatib_CY47_bc** = Ryad's *bound-checking* branch on CY47

What specific about GIT-GCO ?

- main cycles = tags on ***master*** branch
- other official branches :
  - `_t1` (e.g. ***gco_CY46_t1***) : merging branch of all contributions based on CY46 to build CY46T1
  - `_r1` (e.g. ***gco_CY46T1_r1***) : merging branch of CY46T1 with CY46R1
  - `_bf` (e.g. ***gco_CY43T2_bf***) : bugfix branch, after declaration of main cycle
  - `_op` (e.g. ***gco_CY43T2_op2***) : MF operational branch
- user branches : `<user>_<maincycle>_<branchname>`
  e.g. **khatib_CY47_bc** = Ryad's *bound-checking* branch on CY47
- a version of a *pre-cycle* = a numbered tag on a pre-cycle branch (e.g. `_t1` or `_r1` branch)

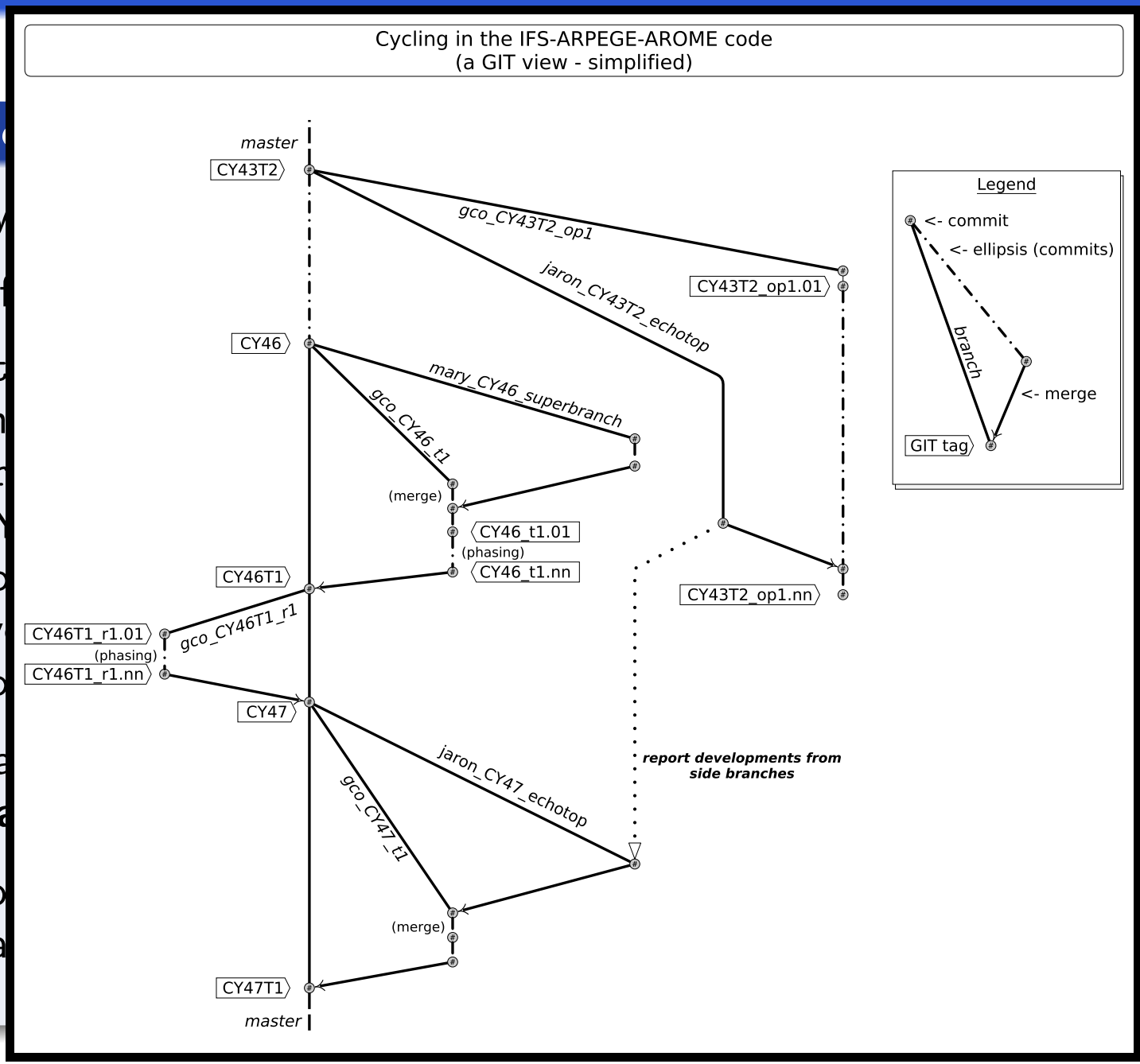
What specifi

- main cy
- other of
  - _t
    on
  - _r
    CY
  - _b
    cy
  - _o
- user bra
  e.g. **kha**
- a versio

based

of main

_t1 or



A toolbox

According to this organization, and in order to help the non GIT-native-speaking people, GCO (Stéphane) developed a toolbox of command-line tools, named `git_*`.
Especially :

- `git_branch -r CY46T1 -b r1 -v 03 -u nobugs`
  ⇒ **start** and **checkout** a **new** branch, based on tag `CY46T1_r1.03` ; the branch in GIT will be named `mary_CY46T1_nobugs`
- `git_commit -m "removal of all bugs"`
  ⇒ **commit** my modifications, with message describing the contents of the modifications since last commit
- `git_post`
  ⇒ **push** branch/commits to GCO's "central" repository
- `git_diff -h path/to/file.F90`
  ⇒ show **history** and **differences** in history of the file

More details in Stéphane's document.

A bunch of native GIT commands

- `git fetch`
  ⇒ update local repository with new branches/commit from origin repository ; do not modify current code
- `git pull`
  ⇒ update current branch to its latest commit