

Advanced versioning

27/09/2016

Cécile Camillieri



Previously...

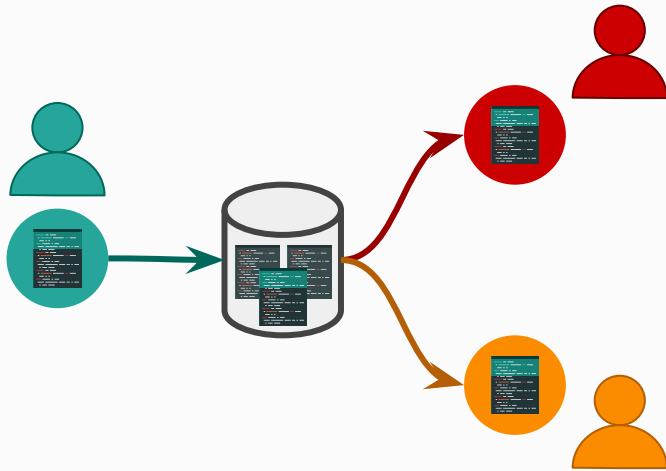
Source Code Versioning

Share changes

Trace changes

Rollback changes

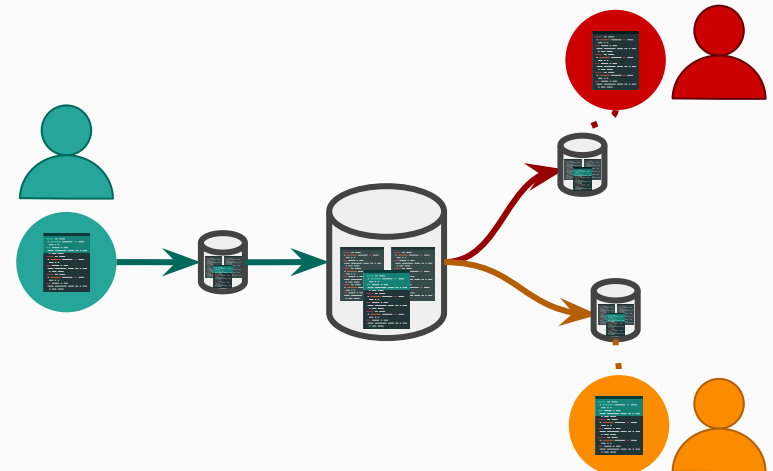
Centralized vs Distributed



add -> commit

1 Repository

> CVS, SVN, ...



add -> commit -> **push**

N repositories : 1 per user

Offline work, branches, and more...

> Git, Mercurial, ...

Today...

Outline

Using **git** (101)

Handling **conflicts**

Using **branches**

Git 101

Git basic commands

- `git clone {url}` -> Get an existing git repository
- `git init` -> Create a new local repository

```
glc@Stagiaire-PC MINGW64 ~/Desktop/git101
$ git clone https://github.com/[redacted]/lp-geo-shapes-ws.git
Cloning into 'lp-geo-shapes-ws'...
remote: Counting objects: 79, done.
remote: Total 79 (delta 0), reused 0 (delta 0), pack-reused 79
Unpacking objects: 100% (79/79), done.
Checking connectivity... done.
glc@Stagiaire-PC MINGW64 ~/Desktop/git101
$ cd lp-geo-shapes-ws/
glc@Stagiaire-PC MINGW64 ~/Desktop/git101/lp-geo-shapes-ws
$ ls -a
./ ../ .git/ .gitignore pom.xml site/ src/
```

clone

init

```
glc@Stagiaire-PC MINGW64 ~/Desktop/git101/newrepo
$ git init
Initialized empty Git repository in C:/Users/glc/Desktop/git101/newrepo/.git/
glc@Stagiaire-PC MINGW64 ~/Desktop/git101/newrepo
$ ls -a
./ ../ .git/
```

All information about the repository

Git basic commands

- `git add {file}` -> Stage a file for the next commit
- `git status` -> Status of the local repository

```
glc@Stagiaire-PC MINGW64 ~/Desktop/git101/lp-geo-shapes-ws
$ git add pom.xml

glc@Stagiaire-PC MINGW64 ~/Desktop/git101/lp-geo-shapes-ws
$ git add newFile.txt

glc@Stagiaire-PC MINGW64 ~/Desktop/git101/lp-geo-shapes-ws
$ git add site/script.js
```

add

status

Up to date with remote

Added for next commit

Not added for next commit

Not known by git and not added for next commit

```
glc@Stagiaire-PC MINGW64 ~/Desktop/git101/lp-geo-shapes-ws
$ git status
On branch master
Your branch is up-to-date with 'origin/master'.
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    new file:   newFile.txt
    modified:   pom.xml
    deleted:    site/script.js

Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

    modified:   pom.xml
    deleted:    site/index.html
    modified:   src/main/java/fr/unice/iut/shapes/services/ShapesService.java

Untracked files:
  (use "git add <file>..." to include in what will be committed)

    newFile2.txt
```

Git basic commands

- `git add {file}` -> Stage a file for the next commit
- `git status` -> Status of the local repository

```
glc@Stagiaire-PC MINGW64 ~/Desktop/git101/lp-geo-shapes-ws
$ git add pom.xml

glc@Stagiaire-PC MINGW64 ~/Desktop/git101/lp-geo-shapes-ws
$ git add newFile.txt

glc@Stagiaire-PC MINGW64 ~/Desktop/git101/lp-geo-shapes-ws
$ git add site/script.js
```

add

status

Can add only part of a file

```
glc@Stagiaire-PC MINGW64 ~/Desktop/git101/lp-geo-shapes-ws
$ git status
On branch master
Your branch is up-to-date with 'origin/master'.
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    new file:   newFile.txt
    modified:  pom.xml
    deleted:   site/script.js

Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

    modified:  pom.xml
    deleted:   site/index.html
    modified:  src/main/java/fr/unice/iut/shapes/services/ShapesService.java

Untracked files:
  (use "git add <file>..." to include in what will be committed)

    newFile2.txt
```

Git basic commands

- `git commit -m {msg}` -> Create a new (local) commit
- `git push` -> Send commits to the remote repository
- `git pull` -> Get changes from the remote repository

One commit to push →

```
g1c@Stagiaire-PC MINGW64 ~/Desktop/git101/lp-geo-shapes-ws
$ git commit -m "an example commit"
[master 396d291] an example commit
3 files changed, 1 insertion(+), 226 deletions(-)
create mode 100644 newFile.txt
delete mode 100644 site/script.js

g1c@Stagiaire-PC MINGW64 ~/Desktop/git101/lp-geo-shapes-ws
$ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
(use "git push" to publish your local commits)
Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        deleted:    site/index.html
        modified:   src/main/java/fr/unice/iut/shapes/resources/Point.java

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        newFile2.txt

no changes added to commit (use "git add" and/or "git commit -a")
```

Git basic commands

- `git log` -> See commits history (local)

```
glc@Stagiaire-PC MINGW64 ~/Desktop/git101/lp-geo-shapes-ws
$ git log
commit 9e21c9a2ec3aed06e98c64f6232ffacea19eedc5
Author: Cecile Camillieri [REDACTED]
Date: Mon Sep 26 14:08:51 2016 +0200

    some commit

commit 396d29160a02aa34d0445bd4373c24a19422cc01
Author: Cecile Camillieri [REDACTED]
Date: Mon Sep 26 13:56:00 2016 +0200

    an example commit
```

latest



oldest

Git basic commands

- `git clone {url}` -> Get an existing git repository
- `git init` -> Create a new local repository

- `git add {file}` -> Stage a file for the next commit
- `git commit -m {msg}` -> Create a new (local) commit
- `git push` -> Send commits to the remote repository
- `git pull` -> Get changes from the remote repository

- `git status` -> Status of the local repository
- `git log` -> See commits history of the local repository

For more : <http://git-scm.com>

Only commit what is necessary

- `.gitignore` file to define files ignored by git
 - > not shown when doing 'git status'

- Should not push :

- hidden files (most of the time)
- compiled code
- IDE settings

```
.*  
*.class bin/ target/ etc.  
.eclipse/ *.iml etc.
```

- You should push :

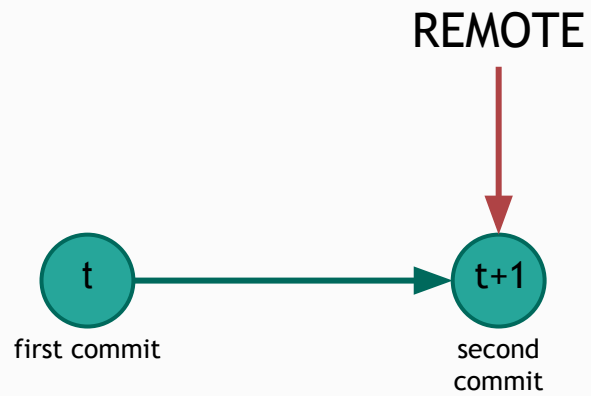
- your `gitignore` file
- source, resources, doc, etc.

Remote

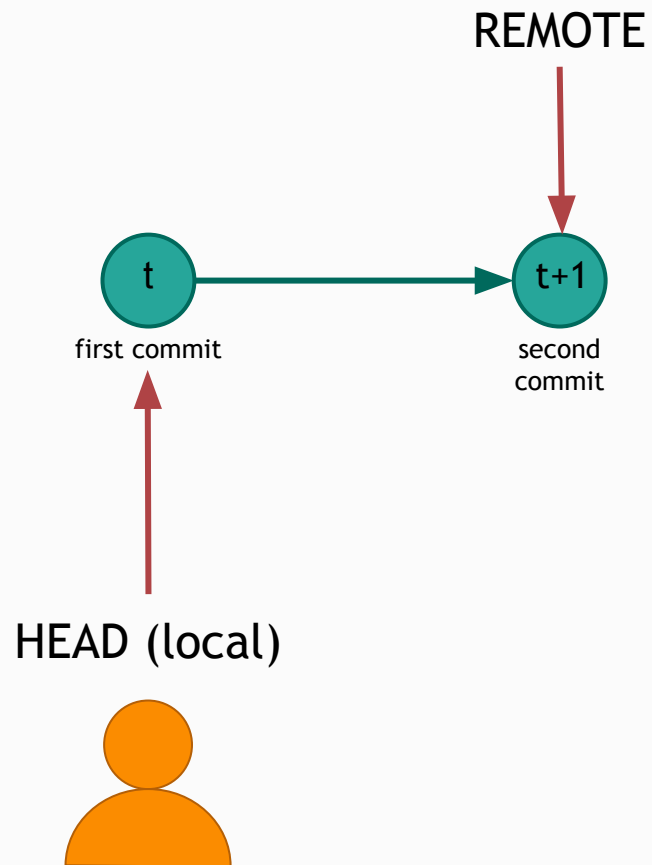
vs

local

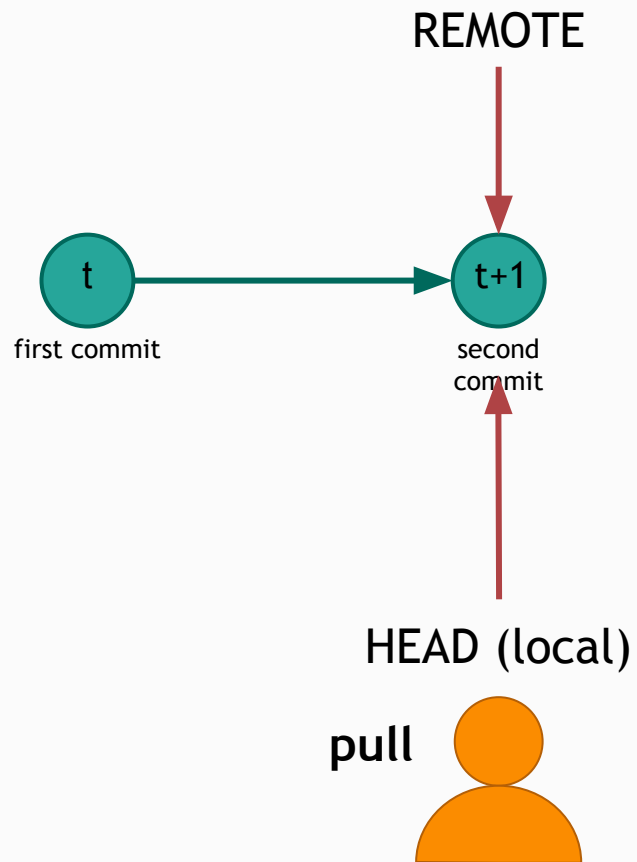
History



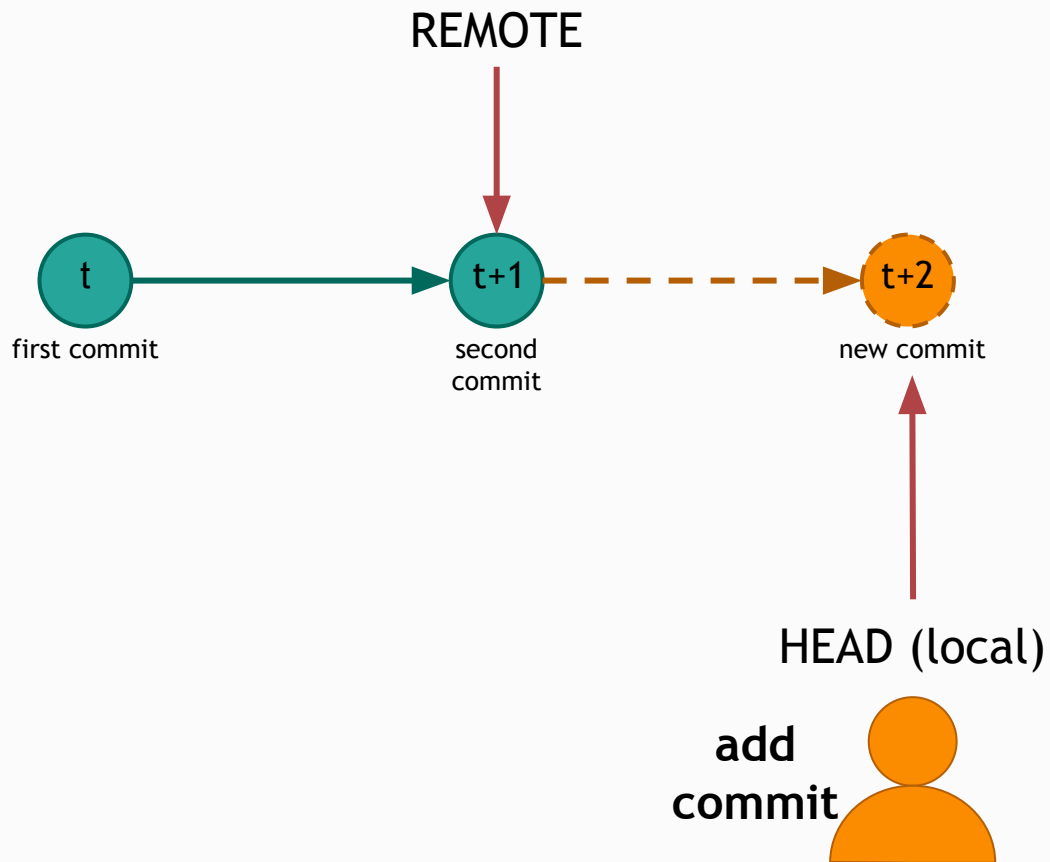
History



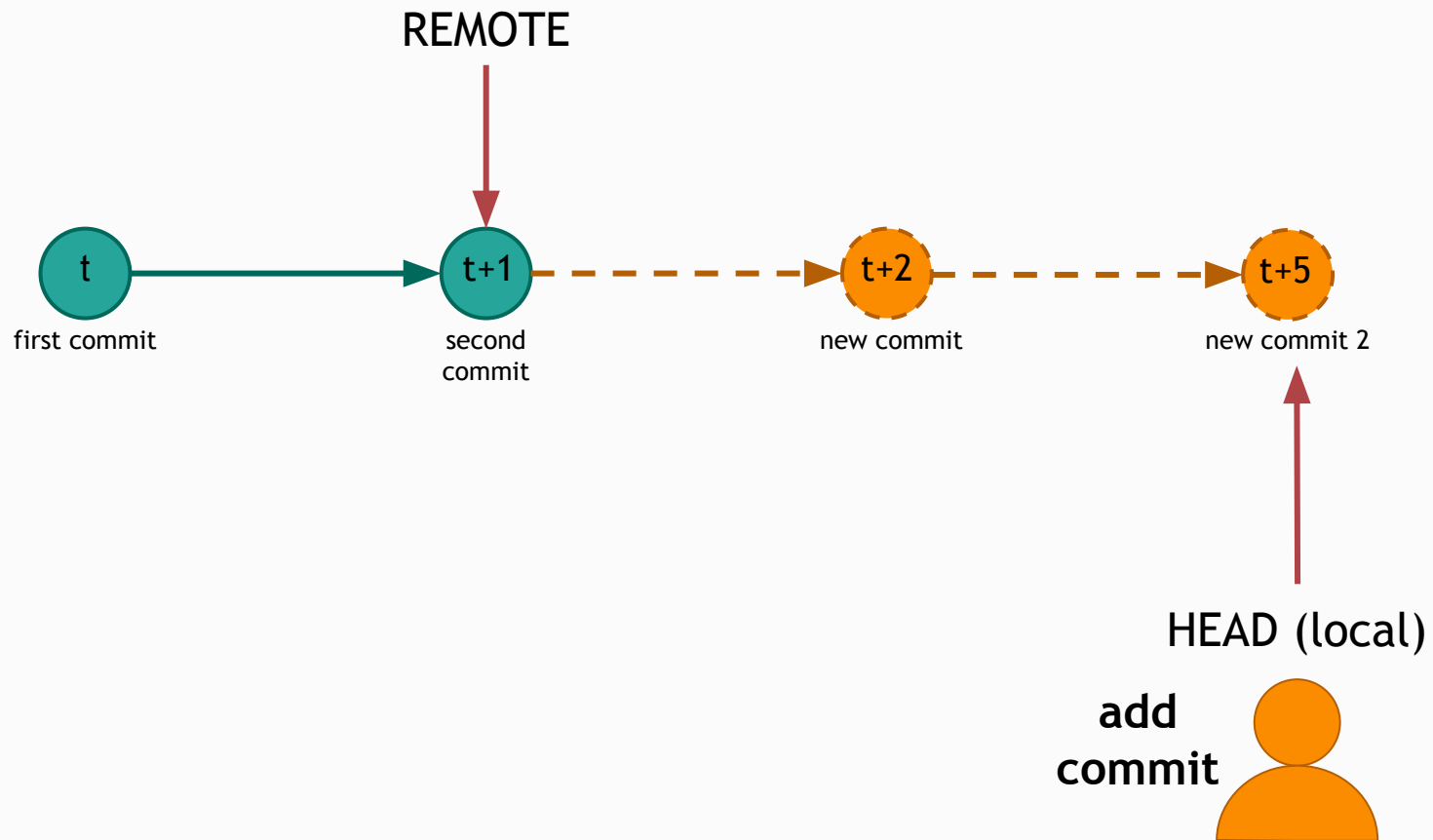
History



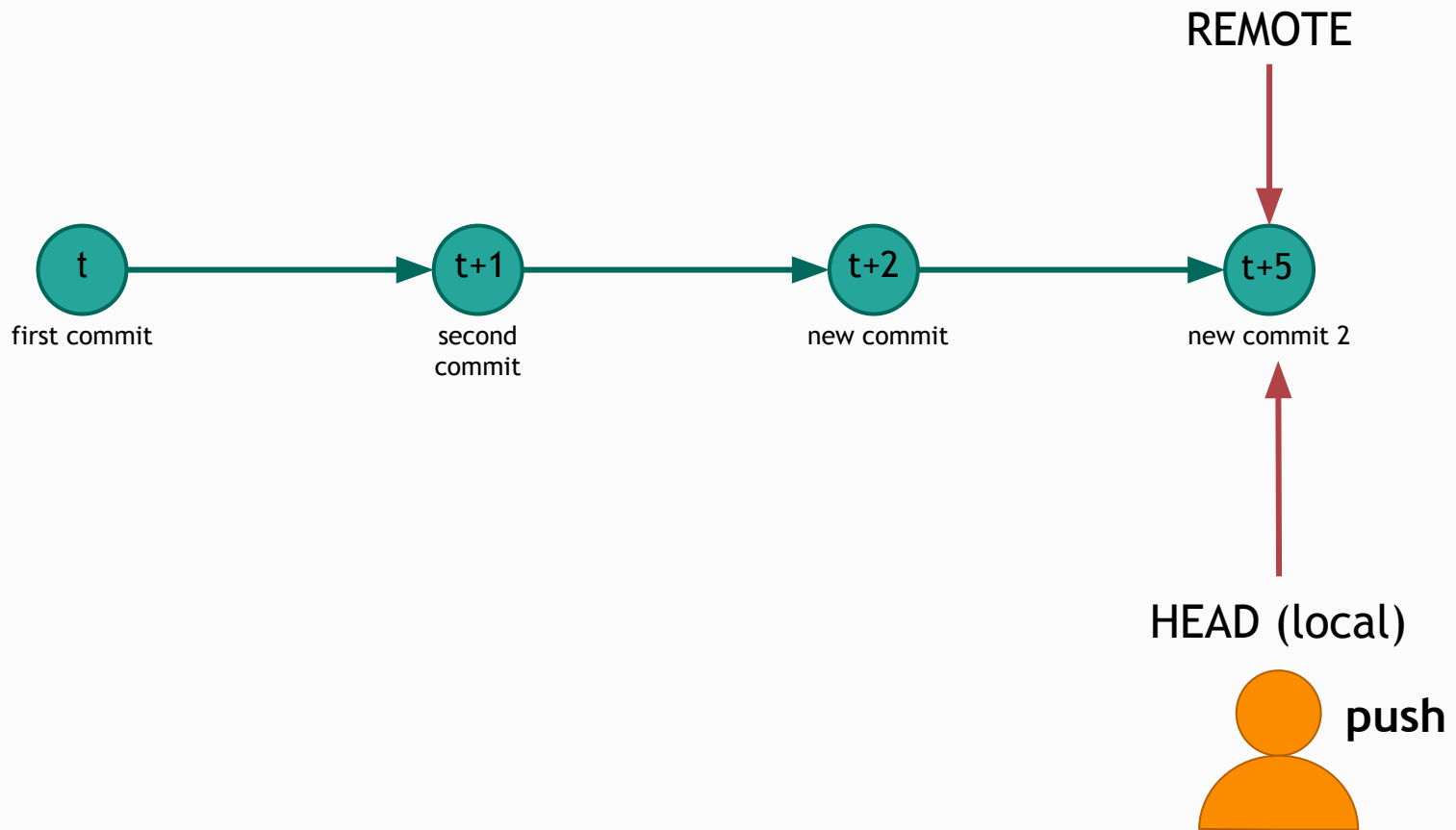
History



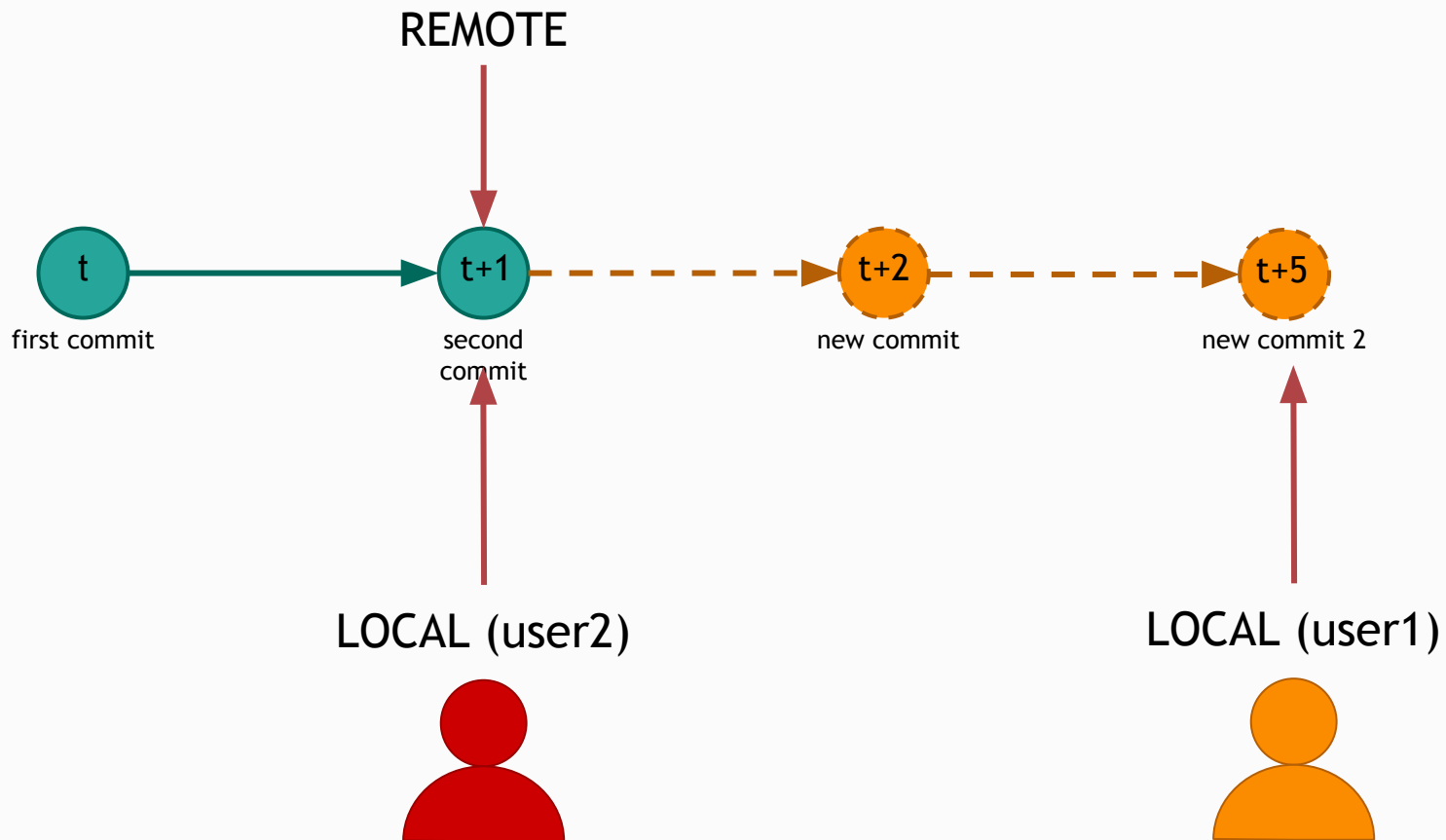
History



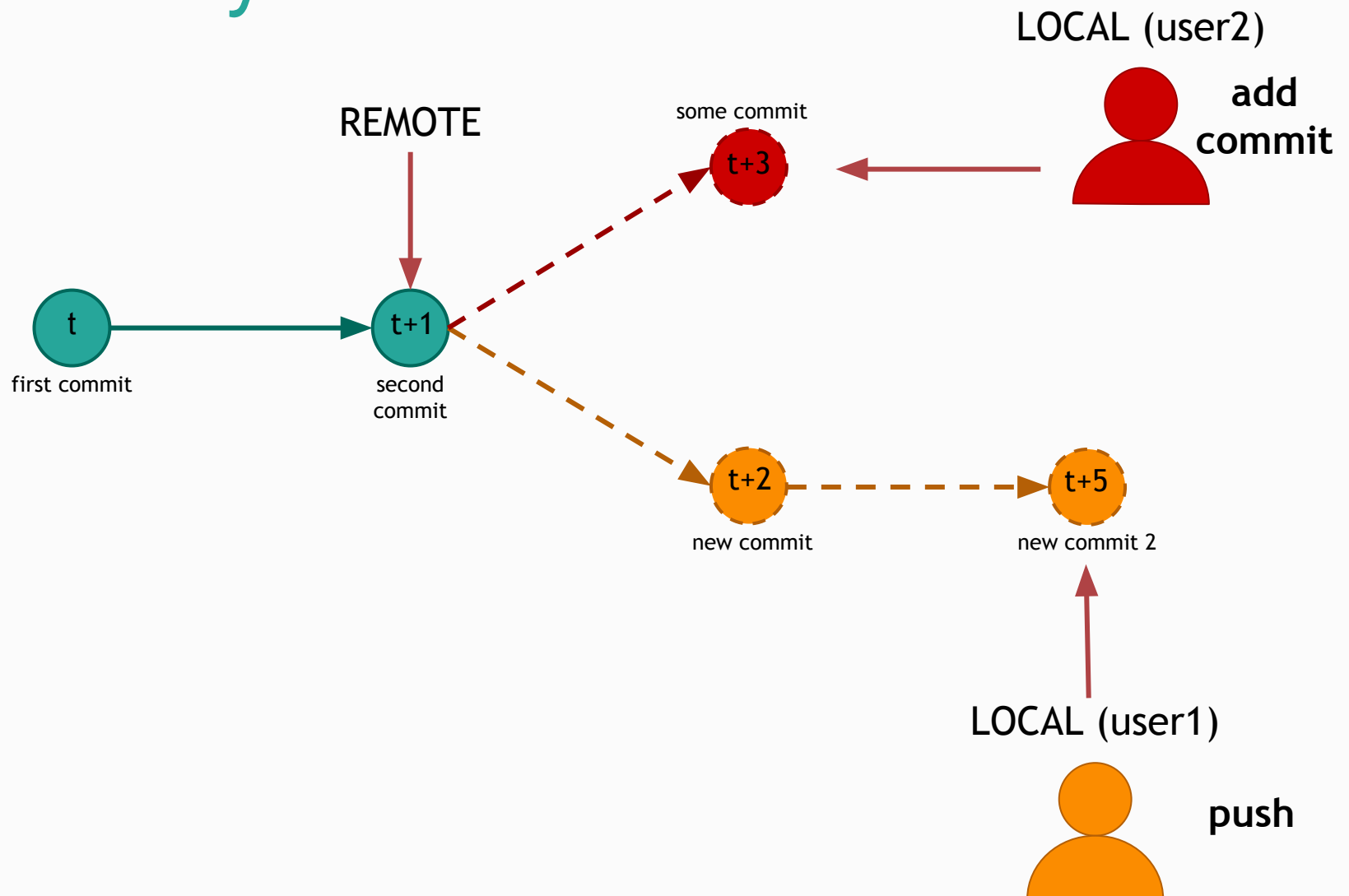
History



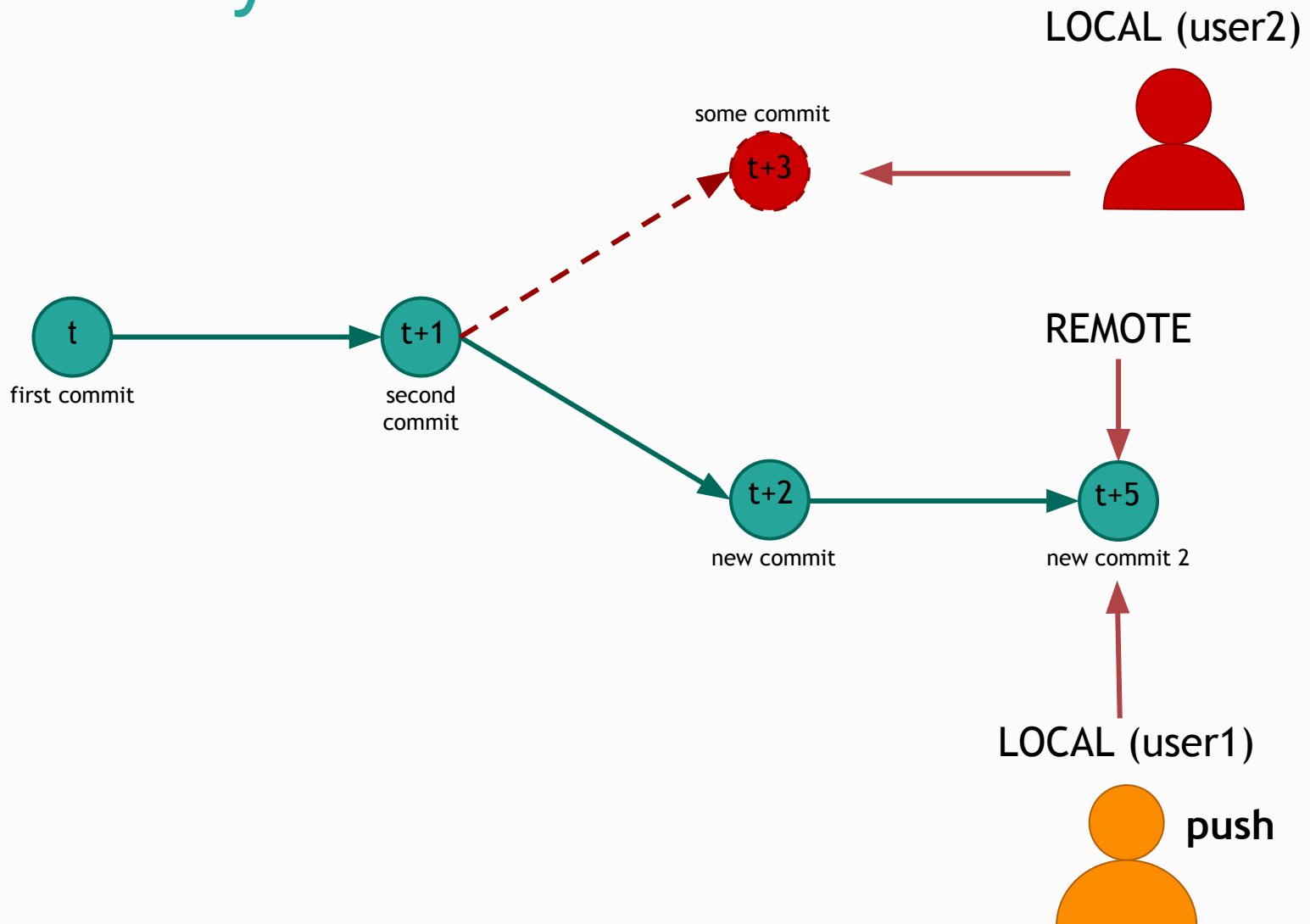
History



History

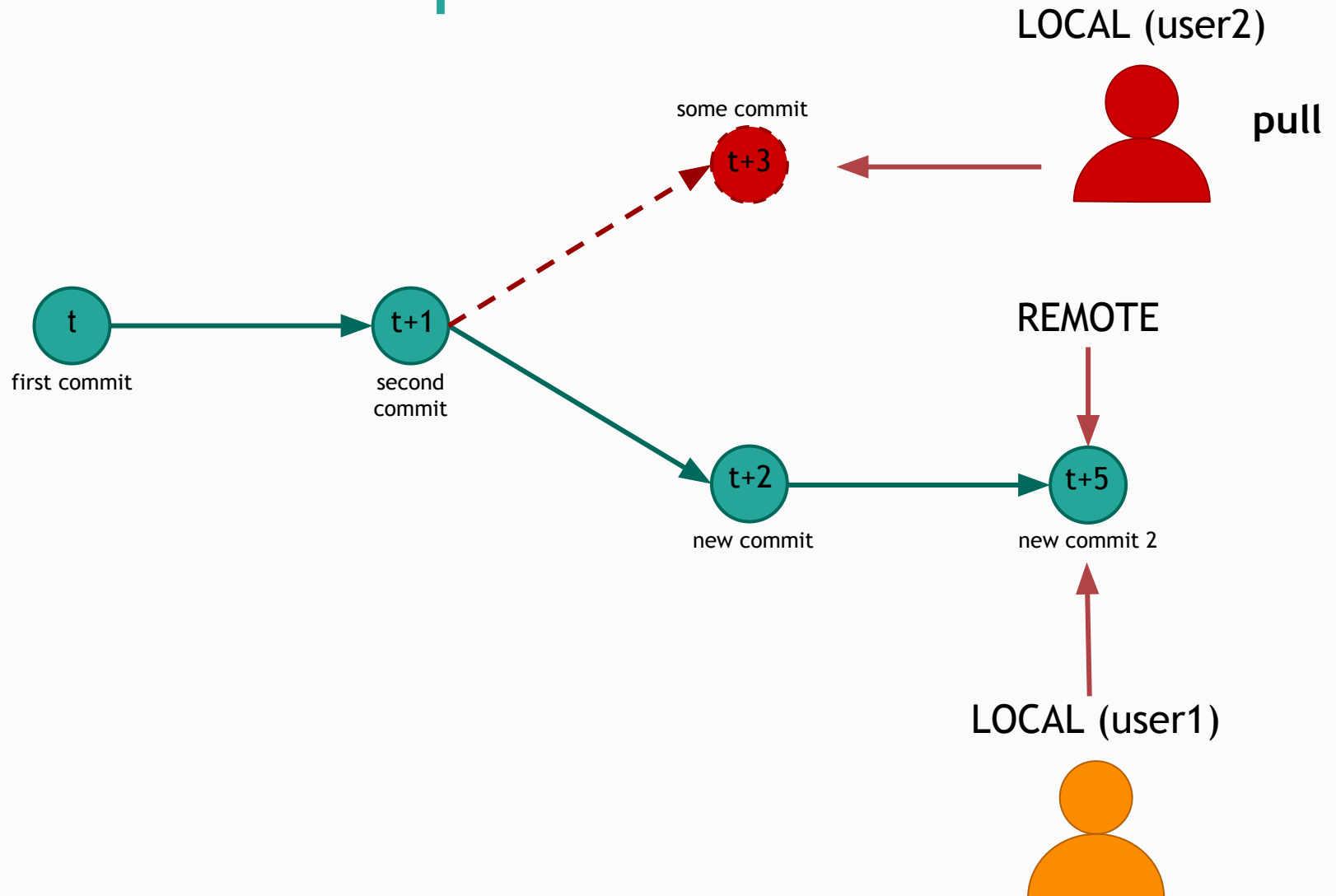


History



Handling conflicts

Pull before push



Auto merging

Different files OR
Different part of same file



Automatic Merge



Merge commit is created
automatically

```
glc@Stagiaire-PC MINGW64 ~/
$ git pull
remote: Counting objects: 116, done.
remote: Compressing objects: 100% (14/14), done.
remote: Total 116 (delta 18), reused 15 (delta 15), pack-reused 86
Receiving objects: 100% (116/116), 91.36 MiB | 2.43 MiB/s, done.
Resolving deltas: 100% (37/37), completed with 7 local objects.
From https://github.com/[redacted]
 * branch          master      -> FETCH_HEAD
   8d1af7b..d33cf6f  master      -> origin/master
Auto-merging models16-me-workshop/sig-alternate.cls
Auto-merging models16-me-workshop/paper.tex
Auto-merging models16-me-workshop/biblio.bib
Removing WorkshopModels/figures/archi-bp.png
Merge made by the 'recursive' strategy.
```

```
glc@Stagiaire-PC MINGW64 ~/
$ git log
commit 404eb88a7b2caad3fb214495e2a5d92f9d6432f6
Merge: 37a0bb7 d33cf6f
Author: Cecile Camillieri <cecile.camillieri@gmail.com>
Date:   Mon Sep 26 16:09:35 2016 +0200

    Merge branch 'master' of https://github.com/[redacted]

commit 37a0bb7ddd008df52254e843d76adfe1cc3fb28e
Author: Cecile Camillieri <[redacted]>
Date:   Mon Sep 26 16:07:59 2016 +0200

    test commit 2

commit 2d3b7cb24041a5eb449d87c9cc427ba4155f8991
Author: Cecile Camillieri <[redacted]>
Date:   Mon Sep 26 16:07:05 2016 +0200

    test commit

commit d33cf6f258921ff266068660c5cb2ca01142247e
```

Merge conflicts

Same part of same file



Automatic Merge fail



Need to Resolve conflicts manually



Add and create merging commit

```
glc@Stagiaire-PC MINGW64 ~/Desktop/git101/newrepo
$ git pull
Auto-merging somefile.txt
CONFLICT (content): Merge conflict in somefile.txt
Automatic merge failed; fix conflicts and then commit the result.
```

No conflicts



```
glc@Stagiaire-PC MINGW64 ~/Desktop/git101/newrepo
$ git status
On branch newbranch
You have unmerged paths.
  (fix conflicts and run "git commit")

Changes to be committed:
  new file:   newfile.txt

Unmerged paths:
  (use "git add <file>..." to mark resolution)
  both modified: somefile.txt
```

Conflicts



Resolving conflicts

```
1 sometext
2 <<<<<<< HEAD
3 sometext from user1
4 =====
5 sometext from user2
6 >>>>>>> 9bae6a3104ac4c8b7269171c28b3de3a24217946
7
```

Code coming from the local repository (HEAD)

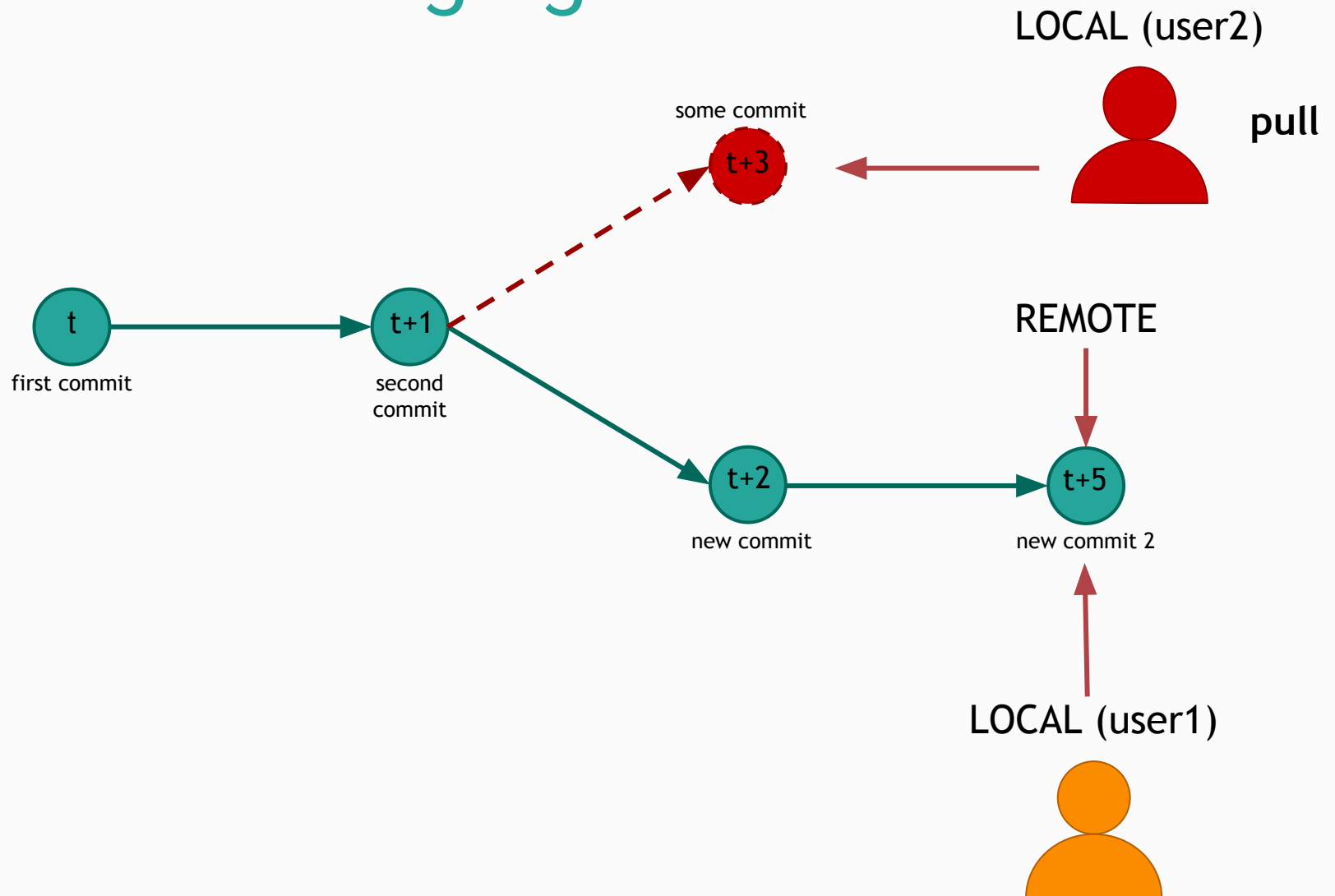
Code coming from the remote repository

choose what to keep

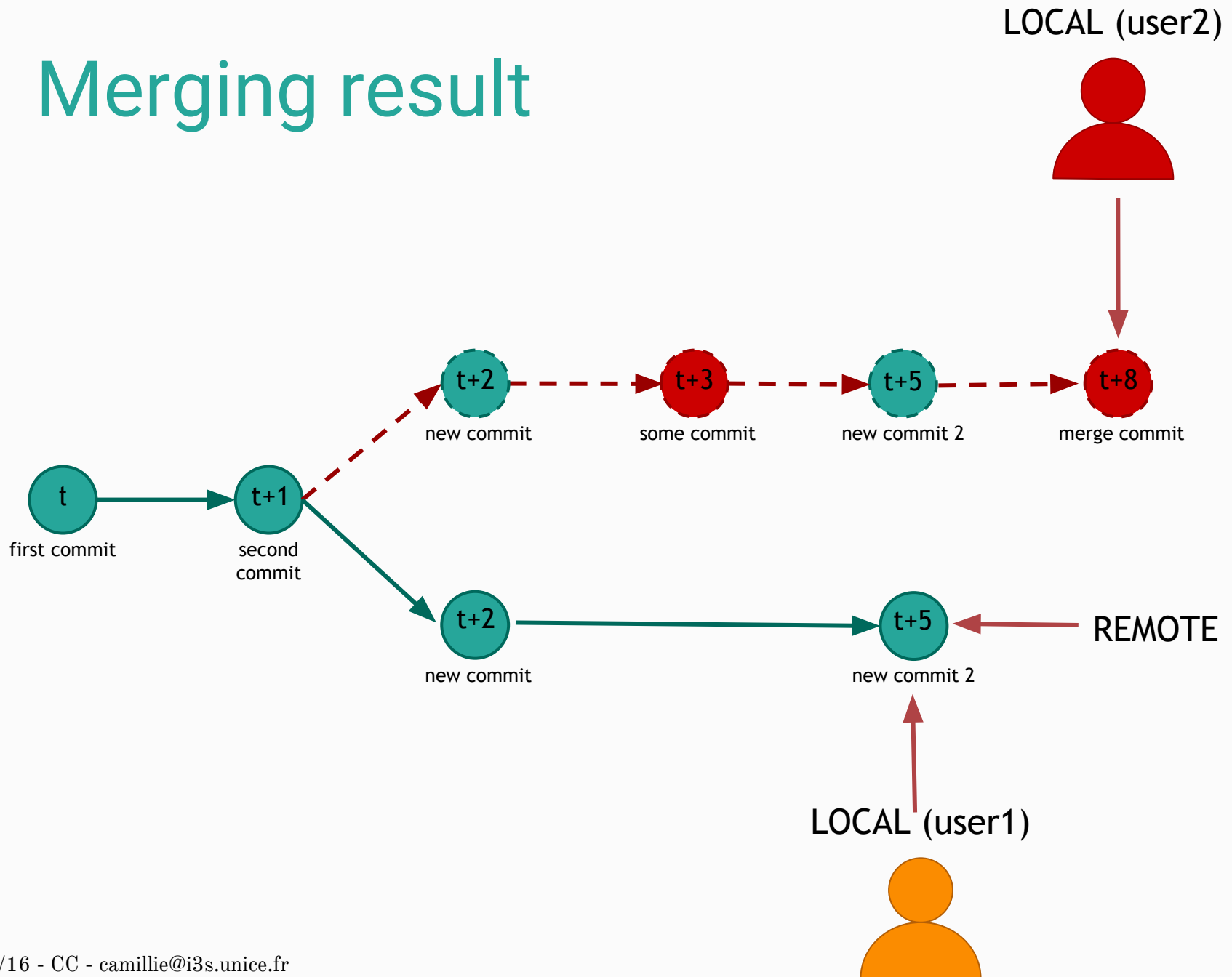
```
1 sometext
2 sometext from user2
3 sometext from user1
4
```

→ **add!** → commit → push

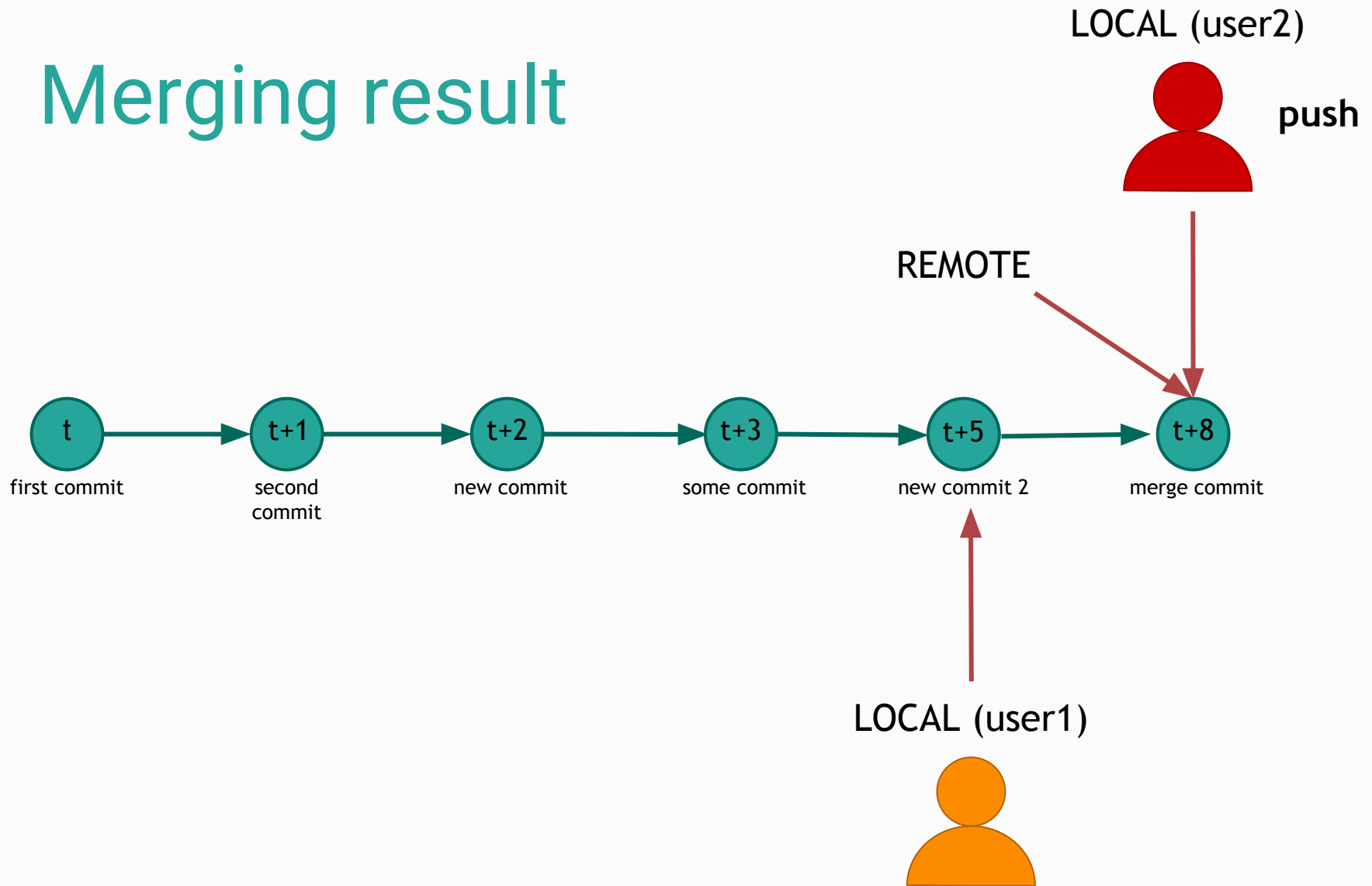
Before merging



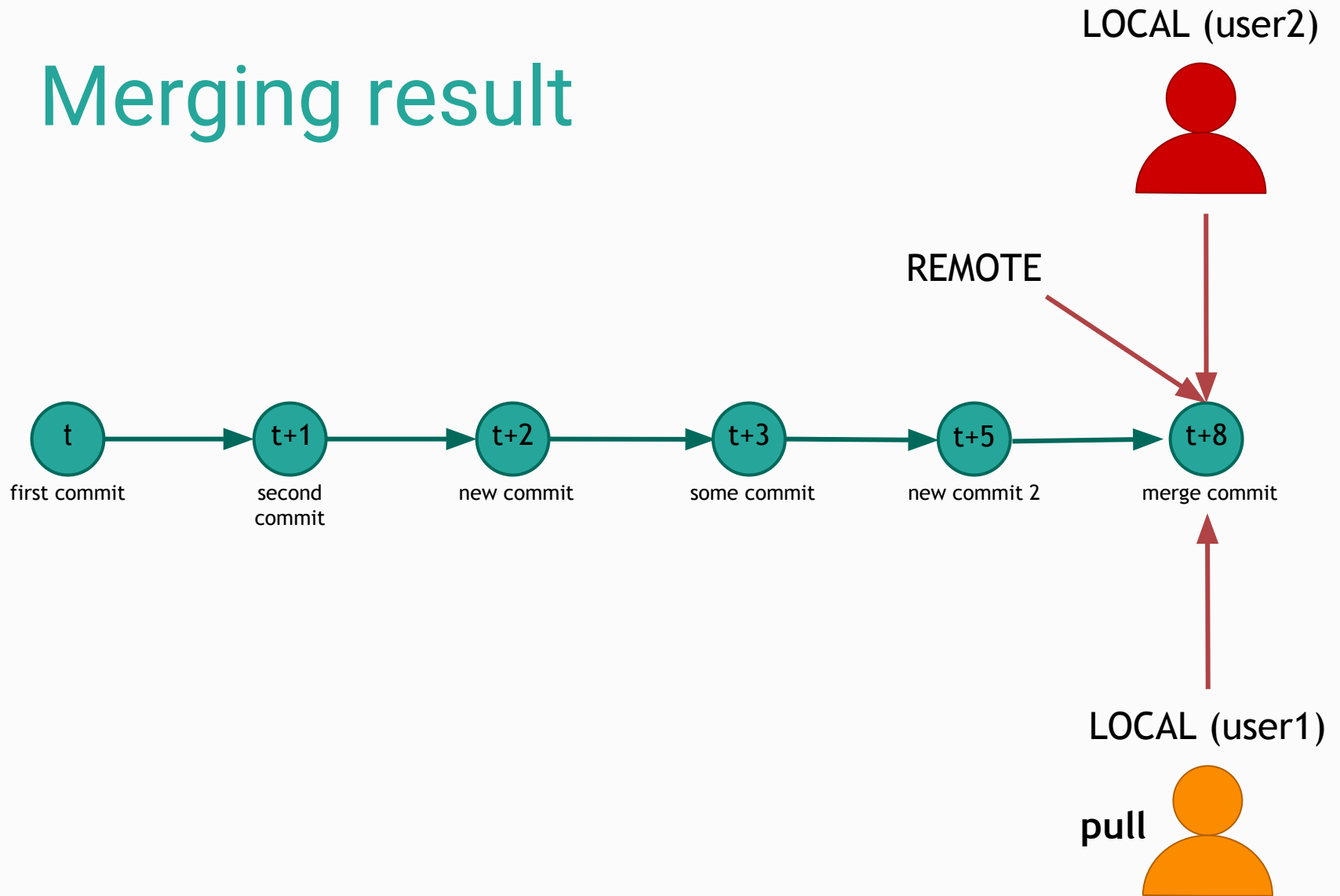
Merging result



Merging result

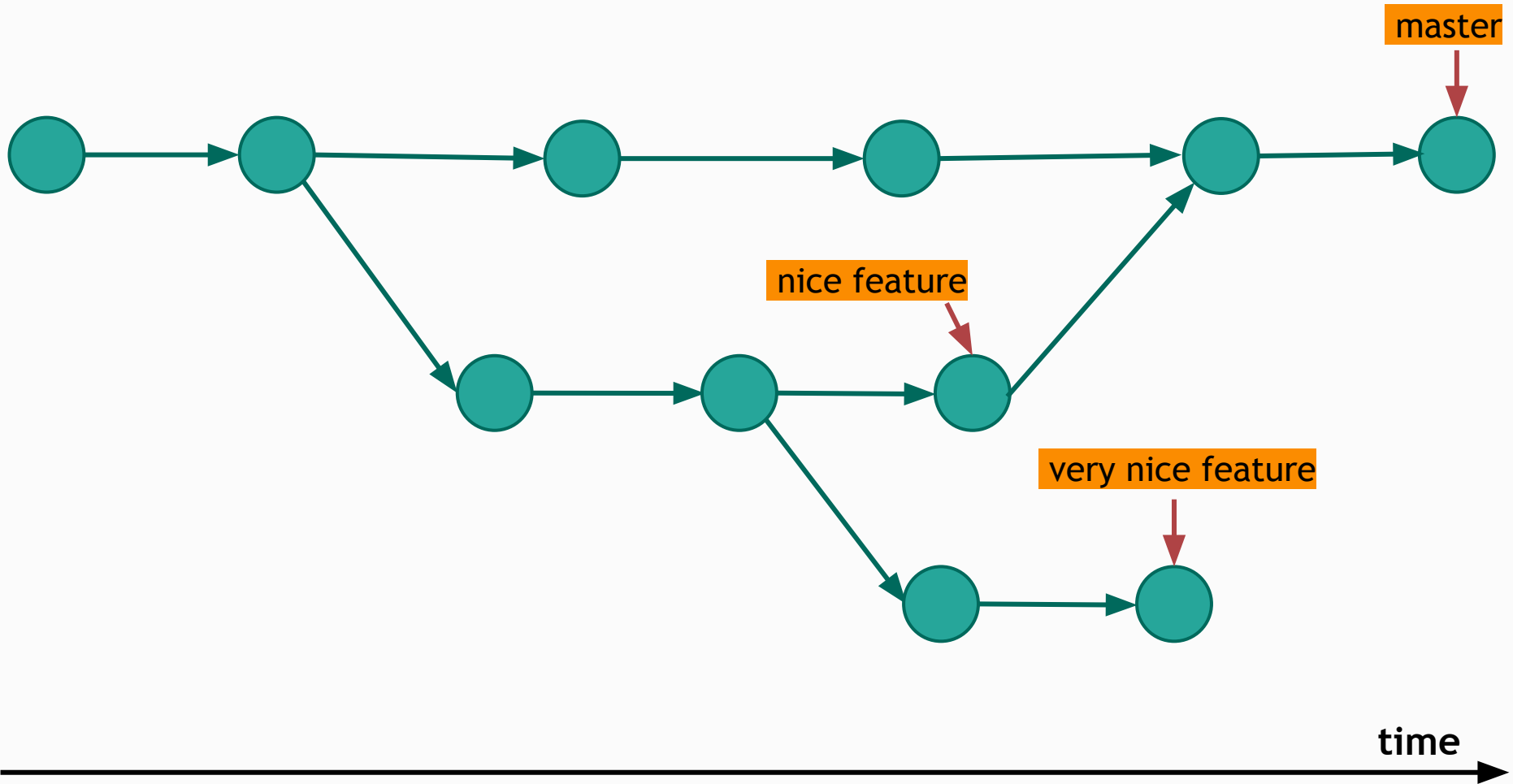


Merging result



Branches & Tags

Branches



Branches

- Work on different features at the same time
- Switch between features/versions/releases
- Merge all changes on the same branch in the end
- Always have a stable version

Git basic commands (continued)

- **git checkout** -> Switch to a different branch/version
- **git branch** -> Manage branches

Create and switch to new branch

Go back to master

See all branches

On branch master

```
g1c@Stagiaire-PC MINGW64 ~/Desktop/git101/newrepo (master)
$ git checkout -b new
Switched to a new branch 'new'

g1c@Stagiaire-PC MINGW64 ~/Desktop/git101/newrepo (new)
$ git checkout master
Switched to branch 'master'

g1c@Stagiaire-PC MINGW64 ~/Desktop/git101/newrepo (master)
$ git branch
* master
  new
  newbranch
```

Git basic commands (continued)

- `git merge {branch}` -> Merge the given branch into the current one

Create commit on 'new' →

Go back to master →

Merge 'new' into 'master' →

Updated history →

```
g1c@Stagiaire-PC MINGW64 ~/Desktop/git101/newrepo (new)
$ git commit -m "added new file"
[new 8e14aee] added new file
1 file changed, 1 insertion(+), 1 deletion(-)

g1c@Stagiaire-PC MINGW64 ~/Desktop/git101/newrepo (new)
$ git checkout master
Switched to branch 'master'

g1c@Stagiaire-PC MINGW64 ~/Desktop/git101/newrepo (master)
$ git merge new
Updating 28ca35f..8e14aee
Fast-forward
 newfile.txt | 2 +-
1 file changed, 1 insertion(+), 1 deletion(-)

g1c@Stagiaire-PC MINGW64 ~/Desktop/git101/newrepo (master)
$ git log
commit 8e14aee3a0947bf0044f43a95ac8bcad72849ddf
Author: Cecile Camillieri
Date: Mon Sep 26 17:23:36 2016 +0200

    added new file
```

Git basic commands (updated)

pull = merge remote branch into current local branch

git push origin master

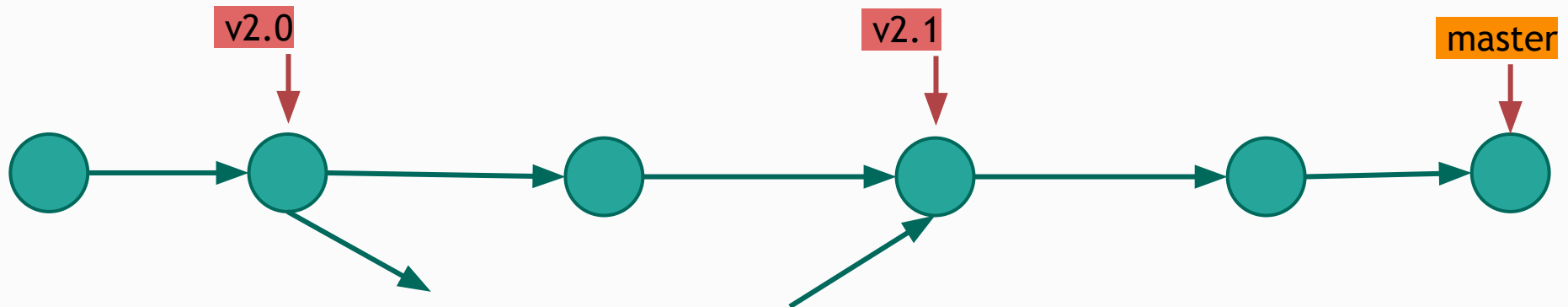
= merge **local branch** into the **remote** (origin) **master** branch

git pull origin master

= merge the **remote** (origin) **master** branch into **local branch**

Tagging & releases

- `git tag -a v2.1` -> Create tag at current commit
- `git push origin v2.1` -> push tag v2.1
- `git push origin --tags` -> push all tags
- `git checkout -b version2-1 v2.1`
-> switch to a new branch at tag v2.1



Some
branching
solutions

Basic branching

- **master** branch should always be **stable:** releases
- Development on a **develop** branch
- **Hotfixes** can be made on master

More advanced branches

- **master** branch for **releases**
- Development on a **develop** branch
- A branch for the **most important/risky features**
- **Hotfixes** can be made on master

Git flow

- **master** branch for **releases**
- Development on a **develop** branch
- A branch for important **features** (from develop)
- Branches to **prepare for releases** (from develop)
- Branches for **hotfixes** (from master)

Git flow

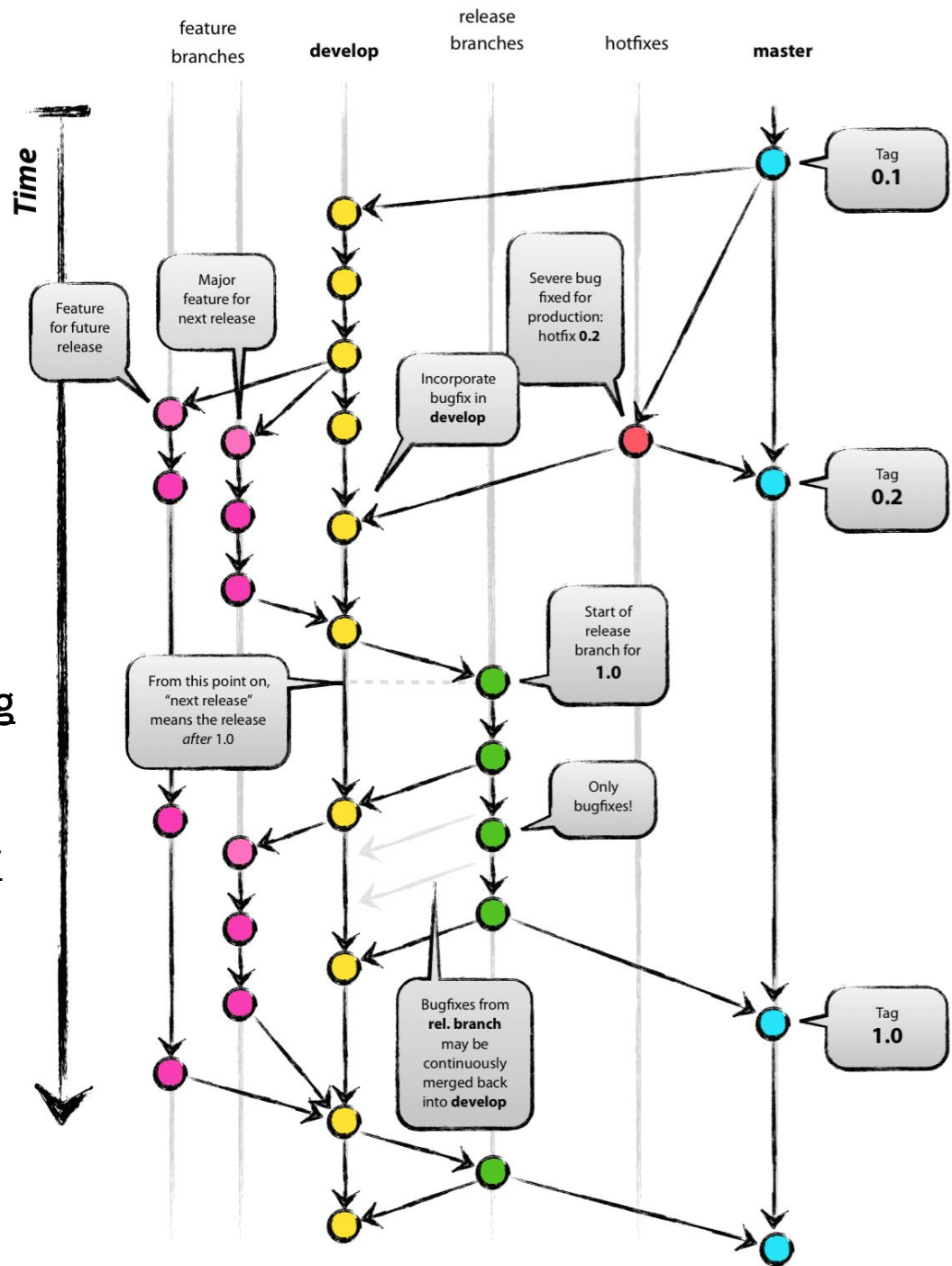
- A set of commands to create, merge, push, etc... branches

```
git flow feature start MYFEATURE
git flow feature finish MYFEATURE
git flow feature publish MYFEATURE
```

- Syntactic sugar

<http://danielkummer.github.io/git-flow-cheatsheet/>

<https://github.com/nvie/gitflow>





Still no trick that solves everything

Sum-up

Source Code Versioning

Share changes

Trace changes

Rollback changes

Git basic commands

- `git clone {url}` -> Get an existing git repository
- `git init` -> Create a new local repository

- `git add {file}` -> Stage a file for the next commit
- `git commit -m {msg}` -> Create a new (local) commit
- `git push origin {branch}` -> Send commits to the remote repository
- `git pull origin {branch}` -> Get changes from the remote repository

- `git checkout` -> Switch to a different branch/version
- `git branch` -> Manage branches
- `git merge {branch}` -> Merge the given branch into the current one

- `git status` -> Status of the local repository
- `git log` -> See commits history of the local repository

Best practices

- **Commit** often, small.
- Always add **commit messages**.
- **.gitignore**: don't commit unnecessary files.
- Branch your IDE directly on the local repository folder.
- Don't forget to **add** files when resolving merge conflicts.
- Use **branches** (at least the master/develop model).

