

GIT

Präsentation von Lara und Bendix

Mit freundlicher Unterstützung von Felix Ehm

Content

1. CERN's Controls Goup
2. What is Git
3. Using Git for modern software development
4. CI & CD

Controls System Architecture



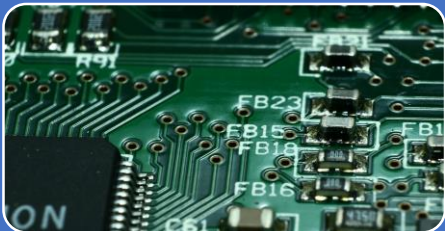
Presentation Layer

- Graphical interactive applications



Business Layer

- General purpose and specific applications server



Embedded Layer

- Real-time services providing access to hardware

What is GIT?

- Version Control System
 - Track changes on files
 - Allows to revert to any previous version
- Basic Concepts
 - Files are organised in a `Repository`
 - `commit` saves a state of a file
 - `checkout` reverts
- **Local** repository may be a copy of a **remote** one
- De-facto standard in modern software development
- Invented by Linus Torvalds -> inventor of Linux

Example

- Create Repository and add 2 files

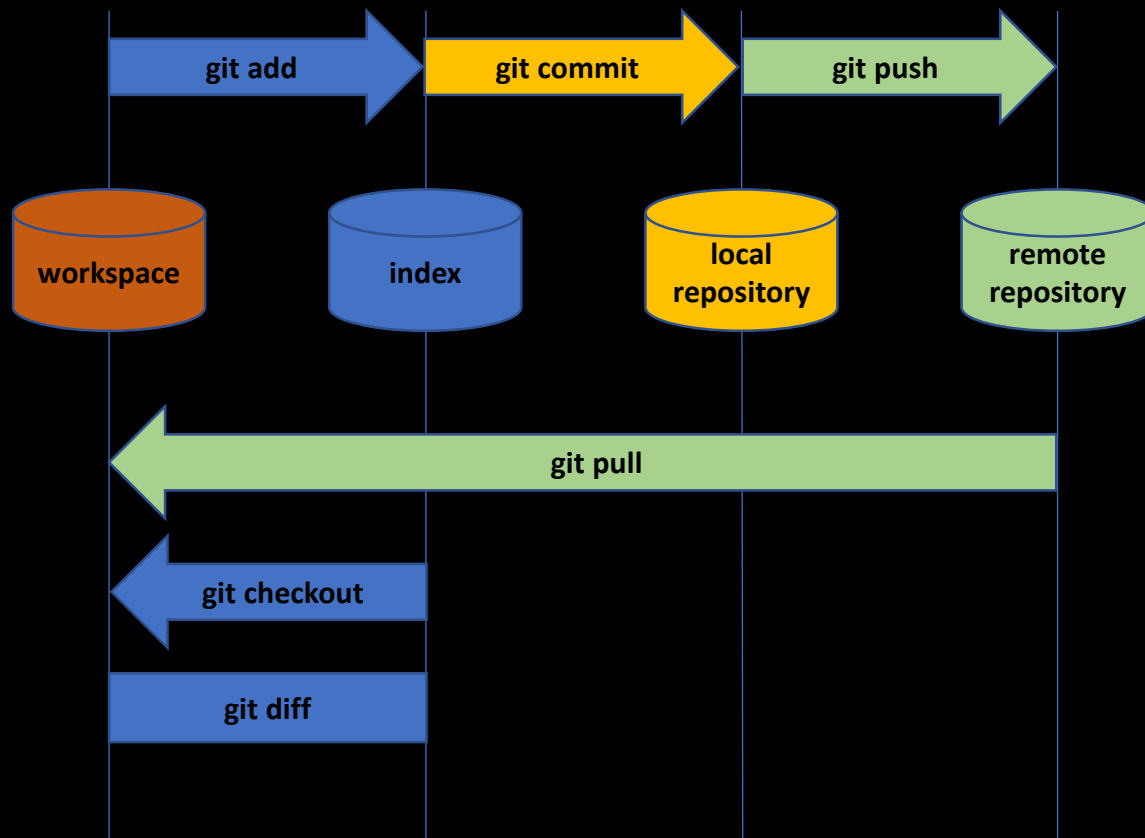
```
git init
git add myfile
git add myfile2
git commit -m 'initial files'
```

- Do changes and revert it to last state

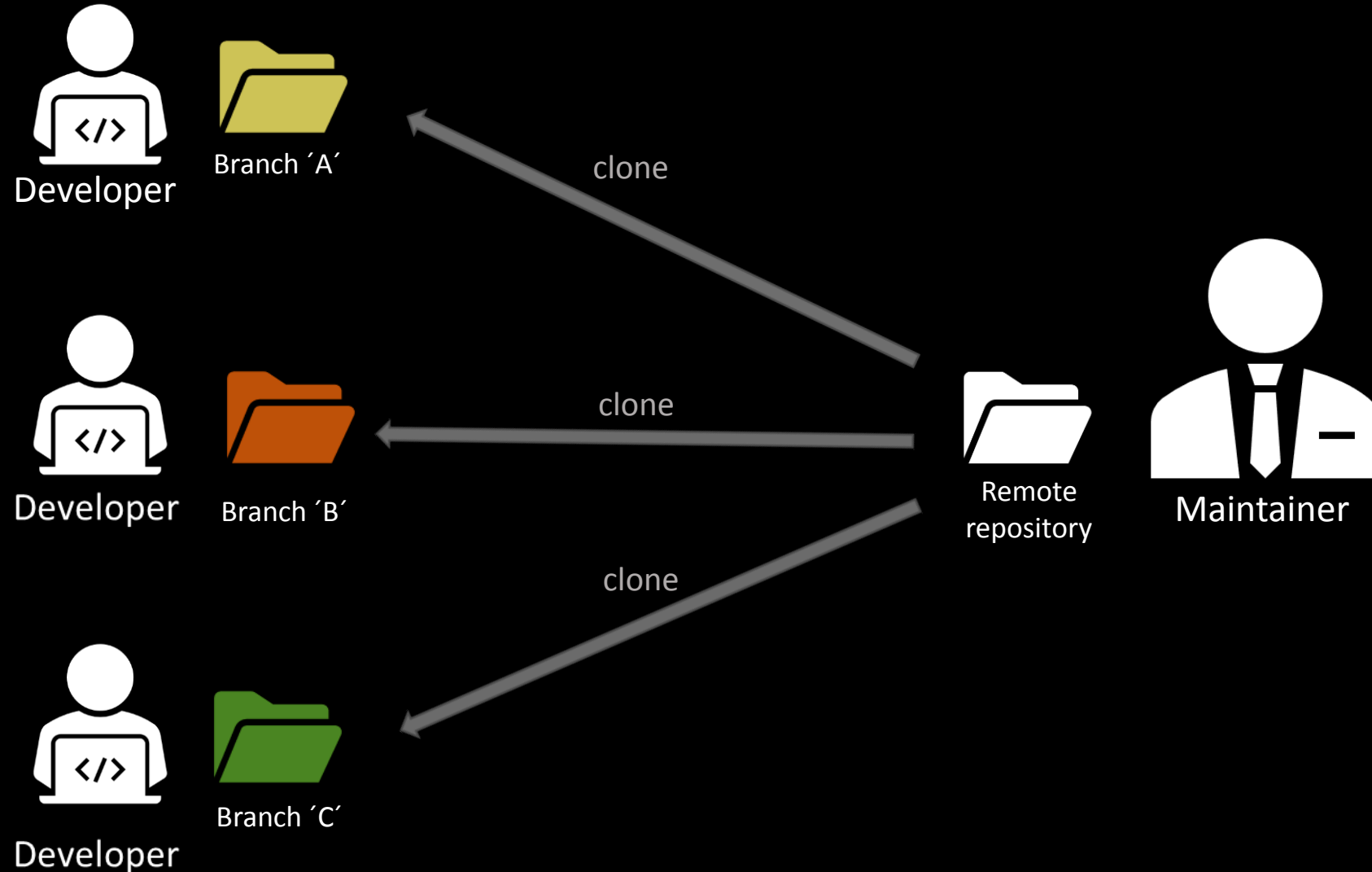
```
rm myfile          # delete file
git checkout myfile # restore file
```

Remote Repository Example

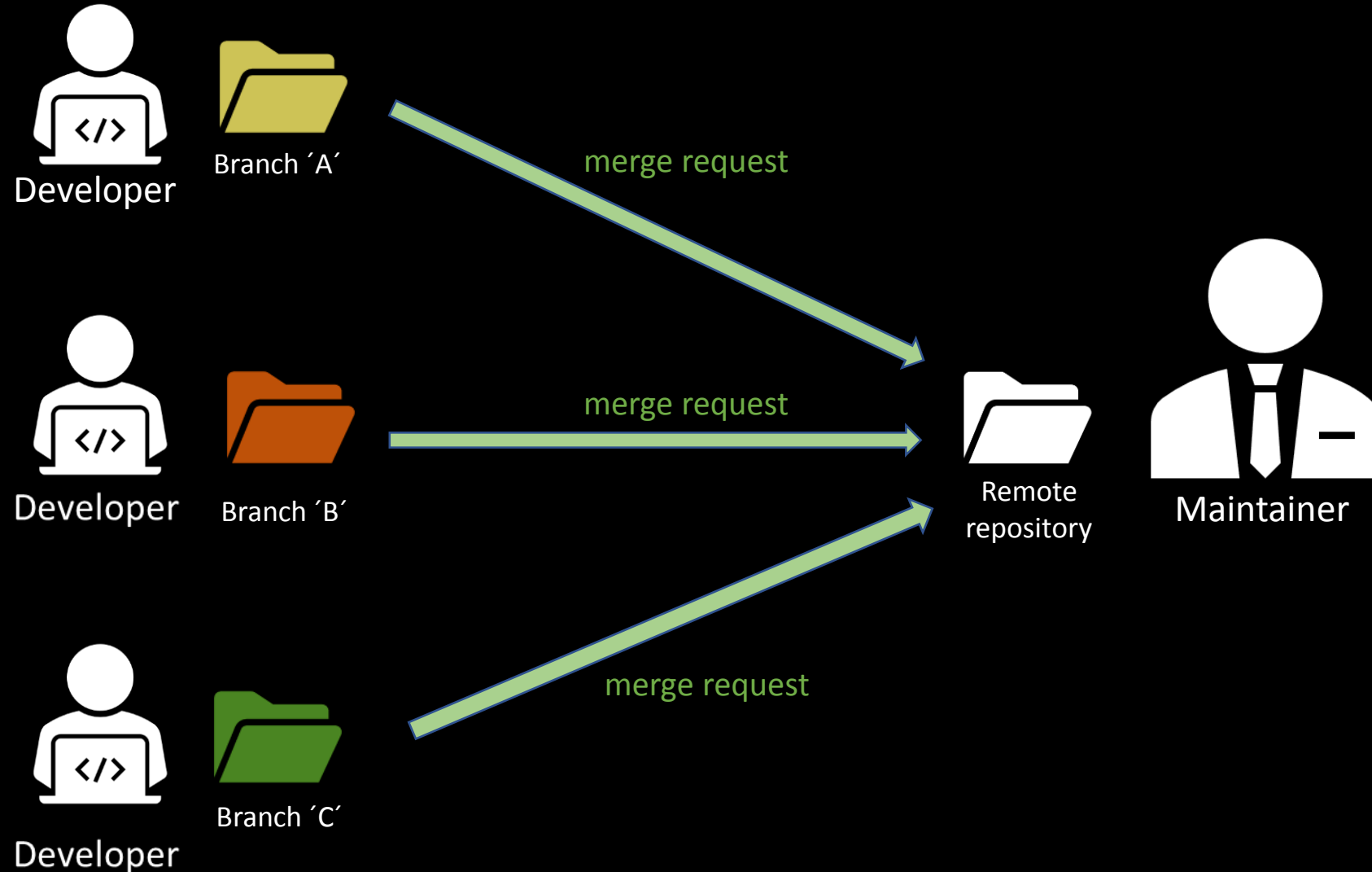
- We have a remote repository (e.g. gitlab.cern.ch/example-project)



Branching & Merge Request & Code Review



Branching & Merge Request & Code Review



Merge Conflicts

- Changes within a file come together
- Often differences are resolved automatically
- Sometimes conflicts need to be resolved manually

```
C:\Users\bxhar\firstproject>git merge testbranch
Auto-merging testfile.txt
CONFLICT (content): Merge conflict in testfile.txt
Automatic merge failed; fix conflicts and then commit the result.
```

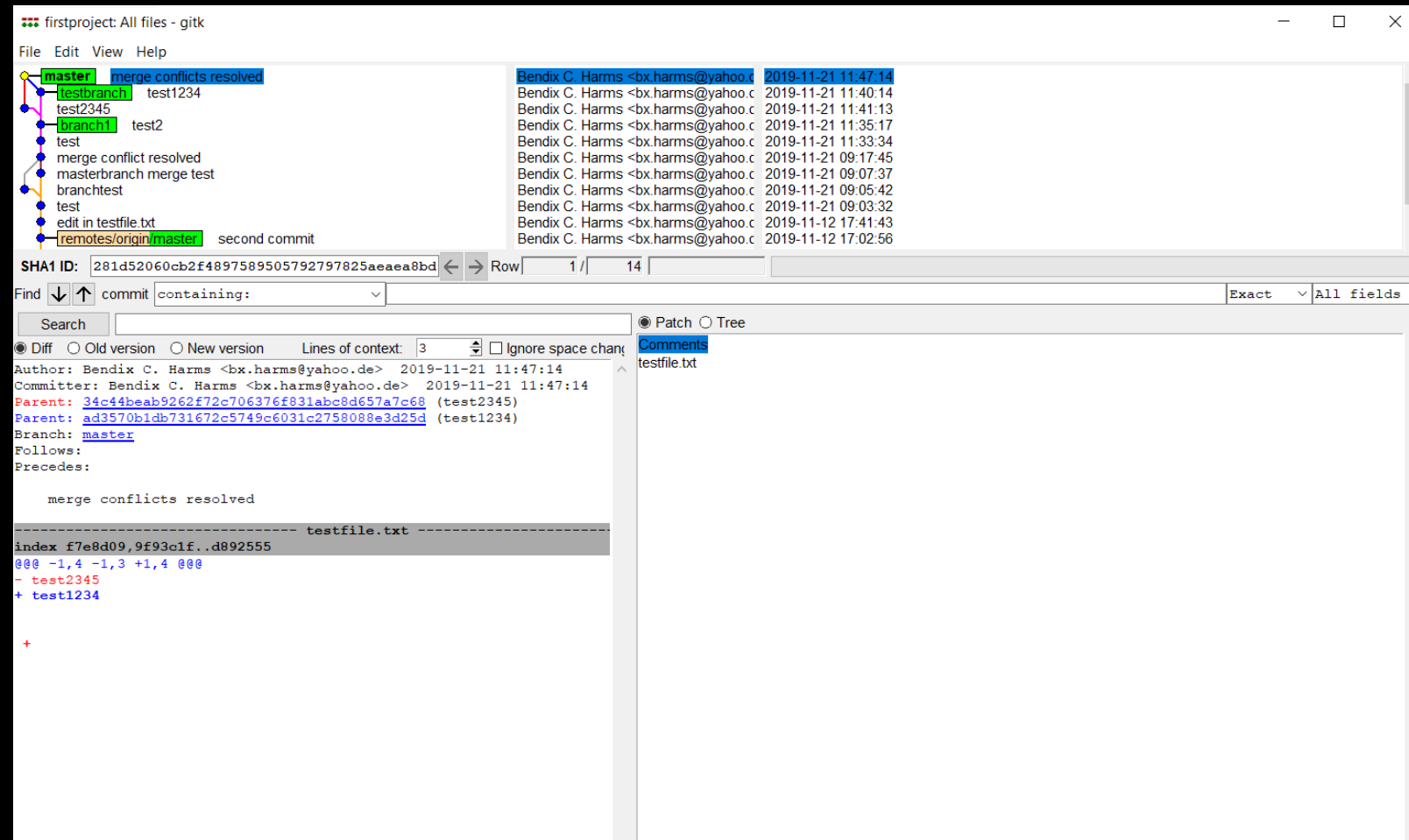
```
|<<<<<<< HEAD
|test2345
|
|=====
|test1234
|>>>>>> testbranch
```

Example: failed merge

- Delete the conflict markers and incorporate changes

Git GUI Example

- Open GUI with „gitk“
- There are plenty of others

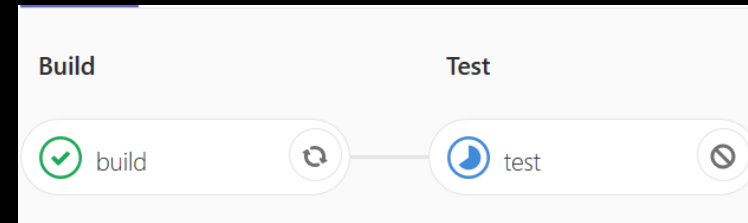


The screenshot displays the Git GUI (gitk) interface for a project named "firstproject: All files - gitk". The interface is divided into several sections:

- Graph View (Top Left):** A commit history graph showing branches (master, testbranch, branch1) and commits (test1234, test2345, test2, test, merge conflict resolved, masterbranch merge test, branchtest, test, edit in testfile.txt, remotes/origin/master, second commit).
- Commit Log (Top Right):** A list of commits with their SHA1 IDs, author names, and timestamps. The current commit is highlighted in blue.
- SHA1 ID (Middle Left):** The SHA1 ID of the selected commit: 281d52060cb2f4897589505792797825aeaea8bd.
- Find (Middle Right):** A search bar with the text "commit containing:" and a dropdown menu.
- Diff View (Bottom):** A diff view for the file "testfile.txt". It shows the commit message "merge conflicts resolved" and the diff content: "index f7e8d09,9f93c1f..d892555", "@@ -1,4 -1,3 +1,4 @@", "- test2345", "+ test1234", and a "+" sign at the bottom.

CI & CD Example

- Automatic building and testing of code:
(ERROR = mail)



- History of CI builds:

Status	Pipeline	Triggerer	Commit	Stages
passed	#1225538 latest		🔗 master -> 4002810c enabled manual triggering of pu...	✓ ✓ »
passed	#1224597		🔗 master -> b644fdbc Update .gitlab-ci.yml	✓ ✓ ✓
failed	#1224461		🔗 master -> 42c5c6a8 Merge branch 'bharms-master-...	✓ ✓ ✗

How is Git used ?

- Track changes on text files + revert to previous version
- Improve software quality
 - encourages **review** of code via merge request
 - Changes are cleanly separated in branches
 - Git CI can run code tests for you on commit
- Developers can push their changes to central as **merge requests**
- CERN's Controls System is critical => is developed using Git