

Task Estimation

Estimation 2

Create a small house

X Point(s)



| | | | |
|----|---------------|-----|----------|
| 0 | $\frac{1}{2}$ | 1 | 2 |
| 3 | 5 | 8 | 13 |
| 20 | 40 | 100 | ∞ |
| ? | ☕ | | |

Task Estimation

Estimation 3

Create a programmable Lego robot

X Point(s)

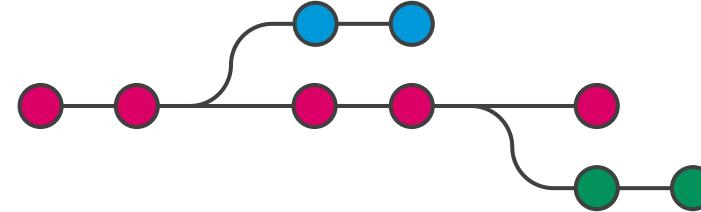


| | | | |
|----|---------------|-----|----------|
| 0 | $\frac{1}{2}$ | 1 | 2 |
| 3 | 5 | 8 | 13 |
| 20 | 40 | 100 | ∞ |
| ? | ☕ | | |



System Design Version Control

Systems Engineering program



Silvan Zahno silvan.zahno@hevs.ch



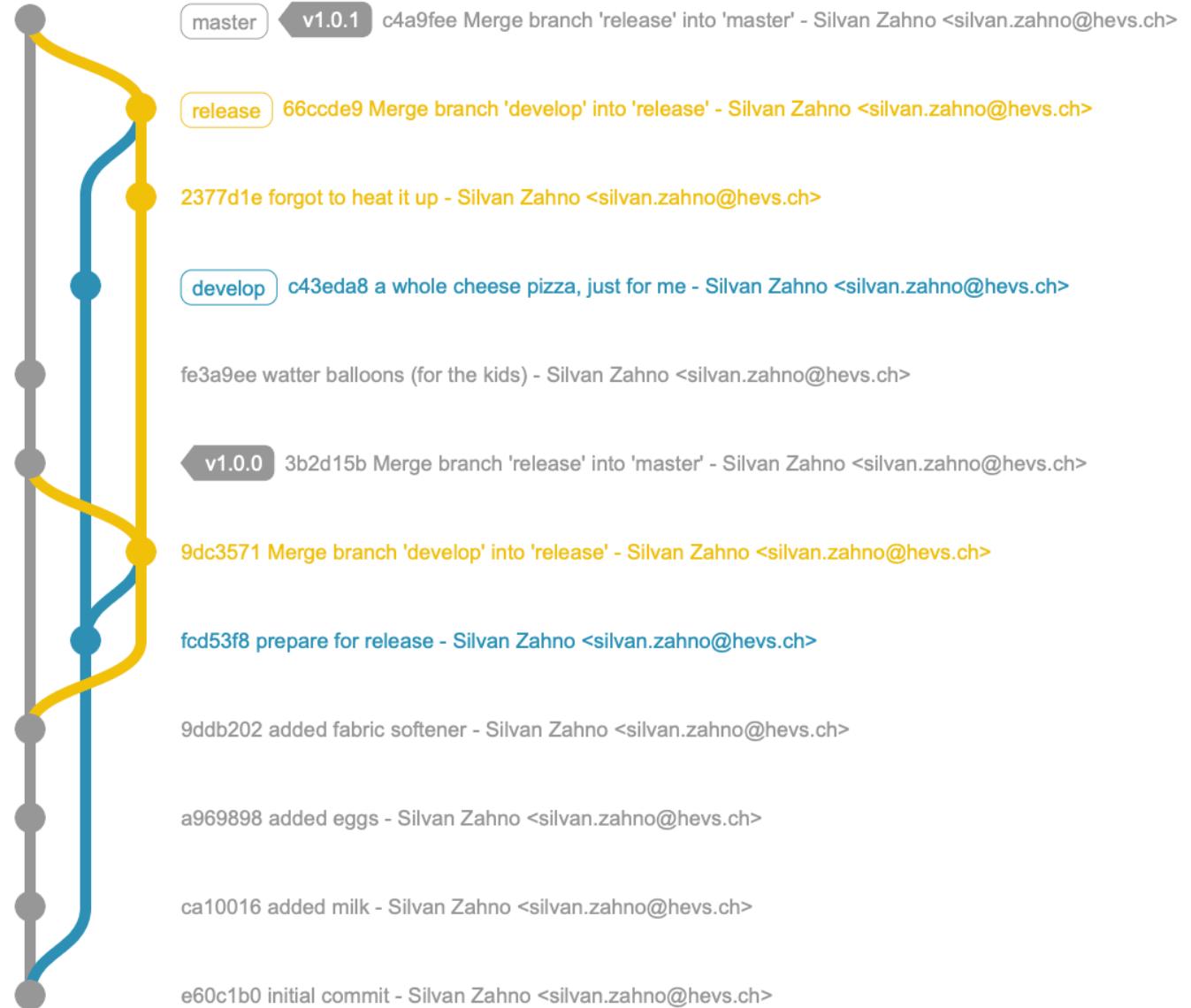
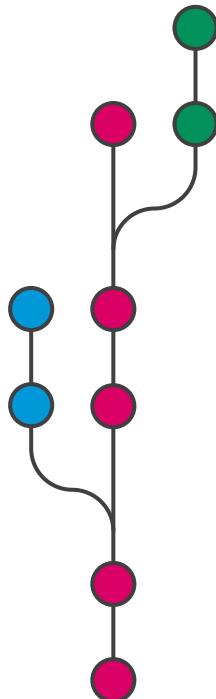


Your current system

```
• └── important-file copy.txt
    ├── important-file v1.txt
    ├── important-file v2.1 backup.txt
    ├── important-file v2.1 backup.txt.old
    ├── important-file v2.1.txt
    ├── important-file v2.txt
    └── important-file.txt
```

1 directory, 7 files

Version control



Possible Tools

Git (git)



Subversion (svn)



Mercurial (hg)

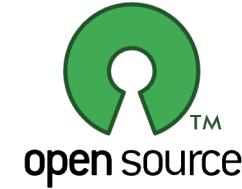
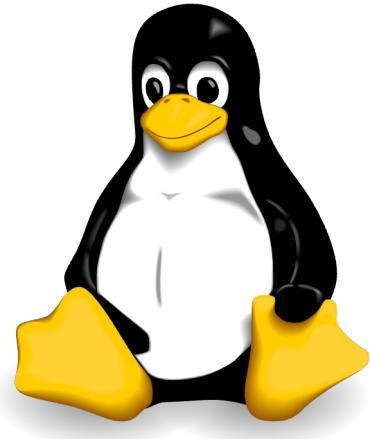


mercurial

SyD Version Control

Linux Torvalds

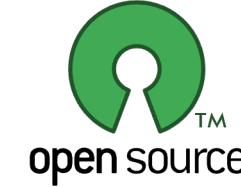
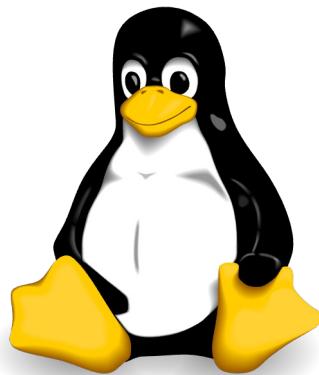
Linux (1991)



Git (2005)

Why Linux needs git

Linux has become the largest collaborative development project in the history of computing over the last 30 years.



- 600 active Linux distribution
- 85% of all Smartphones
- 500 top supercomputers
- >27.8 million lines of code
- > 12'000 contributors
- > 1 million commits

<https://truelist.co/blog/linux-statistics/>

Git Platforms



Gitlab

<https://gitlab.com>

<https://gitlab.hevs.ch>



Github

<https://github.com>



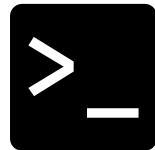
Bitbucket

<https://bitbucket.com>

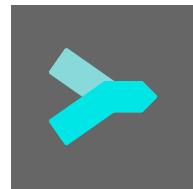


Git Tools

Commandline



Sublime Merge



Git Cola



Git Kraken



Fork

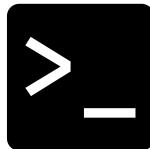


Tower

SmartGit



Git Commandline



```
git status  
git diff  
git add <file>  
git diff --staged  
git reset <file>  
git commit -m "<commit message>"  
git fetch <remote>  
git merge <remote> <branch>  
git push <remote> <branch>  
git pull
```

```
Last login: Tue Mar  8 09:26:26 on ttys004
[zas@zac ~] (base)
$ git --help
usage: git [--version] [--help] [-C <path>] [-c <name>=<value>]
           [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
           [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
           [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
           [--super-prefix=<path>] [--config-env=<name>=<envvar>]
           <command> [<args>]

These are common Git commands used in various situations:

start a working area (see also: git help tutorial)
  clone   Clone a repository into a new directory
  init    Create an empty Git repository or reinitialize an existing one

work on the current change (see also: git help everyday)
  add     Add file contents to the index
  mv      Move or rename a file, a directory, or a symlink
  restore Restore working tree files
  rm      Remove files from the working tree and from the index

examine the history and state (see also: git help revisions)
  bisect  Use binary search to find the commit that introduced a bug
  diff    Show changes between commits, commit and working tree, etc
  grep   Print lines matching a pattern
  log    Show commit logs
  show   Show various types of objects
  status  Show the working tree status

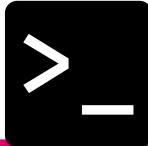
grow, mark and tweak your common history
  branch List, create, or delete branches
  commit Record changes to the repository
  merge  Join two or more development histories together
  rebase Reapply commits on top of another base tip
  reset  Reset current HEAD to the specified state
  switch Switch branches
  tag    Create, list, delete or verify a tag object signed with GPG

collaborate (see also: git help workflows)
  fetch  Download objects and refs from another repository
  pull   Fetch from and integrate with another repository or a local branch
  push   Update remote refs along with associated objects

'git help -a' and 'git help -g' list available subcommands and some
concept guides. See 'git help <command>' or 'git help <concept>'
to read about a specific subcommand or concept.
See 'git help git' for an overview of the system.

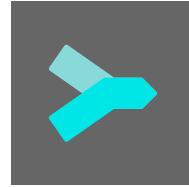
[zas@zac ~] (base)
$
```

git Commands



| Command | Description | Command | Description |
|--------------------------------------|--|---|--|
| Start a working area | | Grow, mark and tweak your common history | |
| clone | Clone a repository into a new directory | branch | List, create, or delete branches |
| init | Create an empty Git repository or reinitialize an existing one | checkout | Switch branches or restore working tree files |
| Work on the current change | | commit | Record changes to the repo |
| add | Add file contents to the index | diff | Show changes between commits, commit and working tree, etc |
| mv | Move or rename a file, a directory, or a symlink | merge | Join two or more development histories together |
| reset | Reset current HEAD to the specified state | rebase | Reapply commits on top of another base tip |
| rm | Remove files from the working tree and from the index | tag | Create, list, delete or verify a tag object |
| Examine the history and state | | Collaborate | |
| log | Show commit logs | fetch | Download objects and refs from another repo |
| show | Show various types of objects | pull | Fetch from and integrate with another repo or a local branch |
| status | Show the working tree status | push | Update remote refs along with associated objects |

Sublime Merge



Sublime Merge interface showing a git repository and code editor.

Locations:

- BRANCHES (1)
 - master
- REMOTES (1)
 - origin
 - master
- TAGS (0)
- STASHES (0)
- SUBMODULES (0)

Commits:

- 1 untagged file Commit Changes
- Merge remote-tracking branch 'origin/master' (16 commits)
 - CHG: updated planning (4 commits)
 - ADD: ALU and ImmSrc doc (4 commits)
 - ADD: EBS2/EBS3 specs (4 commits)
 - FIX: memory stack images (5 commits)
 - FIX: errors in immediate and type images (44 commits)
 - ADD: files in arc exercises (33 commits)
 - ADD: note on Ripes memory management (11 commits)
 - CHG: Planning (4 commits)
 - FIX: reverse engineering solution (8 commits)
 - FIX: ISA syntax errors (3 commits)
 - Merge remote-tracking branch 'origin/master' (6 commits)
 - FIX: errors in ISA (22 commits)
 - ADD: windows Geekbench window (5 commits)
 - FIX: add scripts folder (Axel Amand, 1 commit)
 - REM: car-herv and car-labs doc deployment (Axel Amand, 1 commit)
 - CHG: BEM labo from geekbench 5 to 6 (14 commits)
 - UPD: all PDFs (Axam, 30 commits)
 - FIX: errors in ARC, ISA, FUN and PER slides (26 commits)
- Thu 08:11
- Thu, 13 Apr 15:19
- Tue, 11 Apr 15:25
- Tue, 4 April 11:00
- Tue, 4 April 07:46
- Mon, 3 April 13:30
- Fri, 31 Mar 15:38
- Fri, 31 Mar 08:45
- Thu, 30 Mar 17:25
- Tue, 28 Mar 07:59
- Thu, 28 Mar 07:29
- Thu, 16 Mar 10:14
- Thu, 16 Mar 09:31
- Tue, 14 Mar 14:25
- Wed, 8 Mar 19:10
- Thu, 7 Mar 09:09

Summary:

- Commit Hash: 702c8f1738addb639884c48b724ab68361d3c
- Tree: f163ce5b55f992b7c78549993694b86780d0db8
- Author: zas <silvan.zahn@hevs.ch>
- Date: Thu, 20 April 2023 08:12
- Parents: 6e927fe, 68810079
- Branches: master origin/master
- Stats: 16 files changed: -15 +110

Files:

- 00-solution.tex
- 03-controlunit.tex
- 04-simulation.tex
- 05-deployment.tex
- 05-deployment_ebs3.tex
- 08-CAR-Labor-SCR-d.pdf
- 08-CAR-Labor-SCR-f.pdf
- CAR-Labor-SCR-s.pdf
- CAR-Labor-SCR-t.pdf

Code Editor:

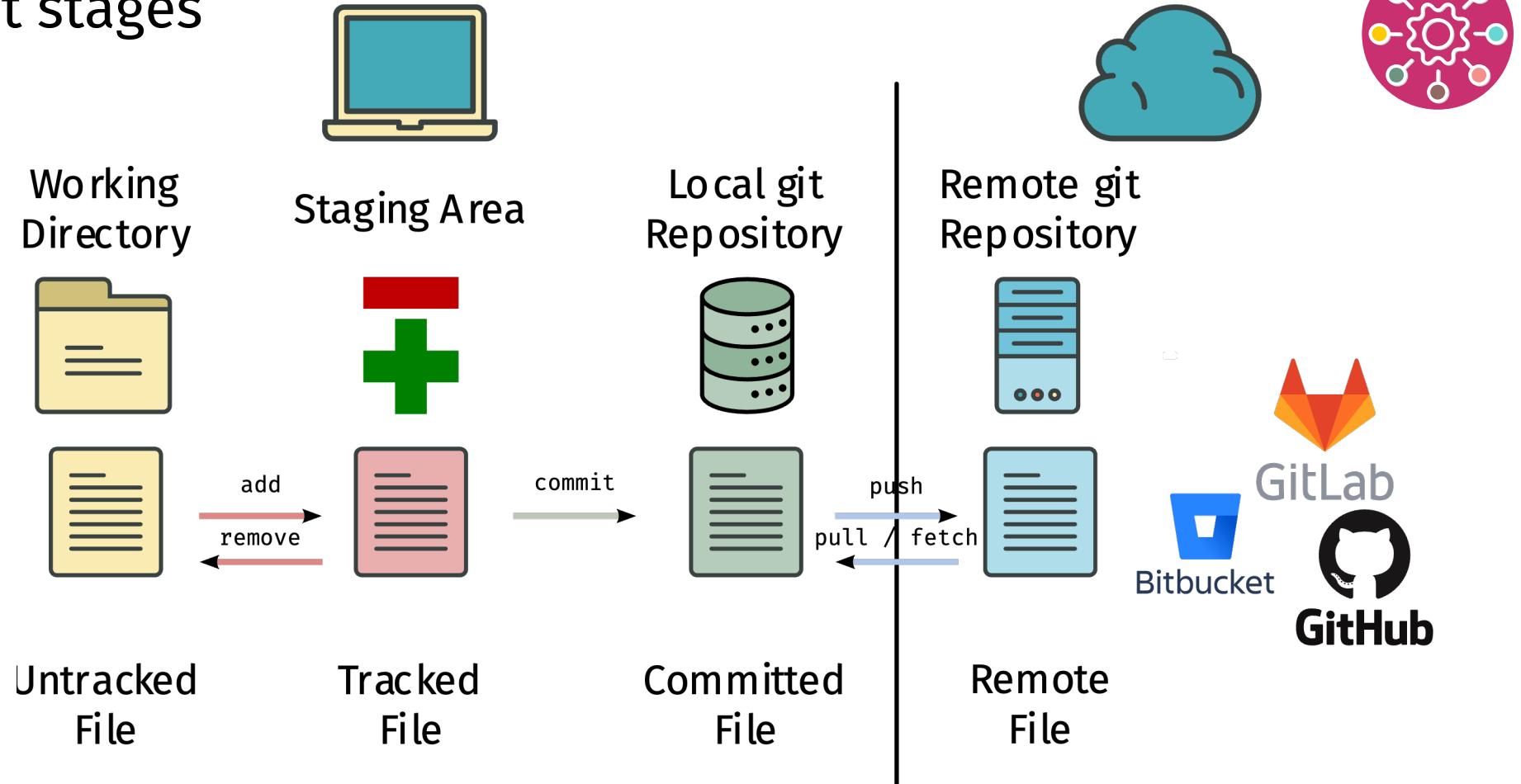
```
00-solution.tex
03-controlunit.tex
04-simulation.tex
05-deployment.tex
05-deployment_ebs3.tex
08-CAR-Labor-SCR-d.pdf
08-CAR-Labor-SCR-f.pdf
CAR-Labor-SCR-s.pdf
CAR-Labor-SCR-t.pdf
```

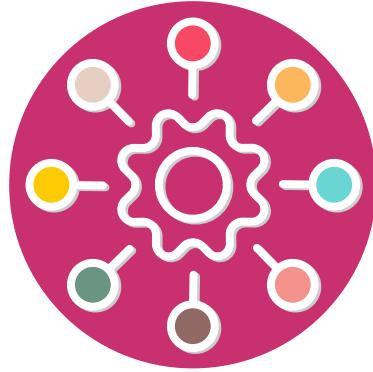
Code Snippets:

- labo/latex/02-content/scr/00-solution.tex
- labo/latex/02-content/scr/03-controlunit.tex

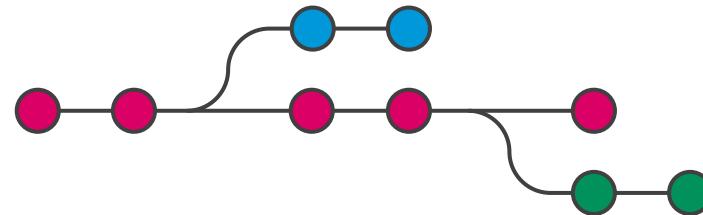
License Upgrade Required:

Git stages





Git Branch and Merge Example

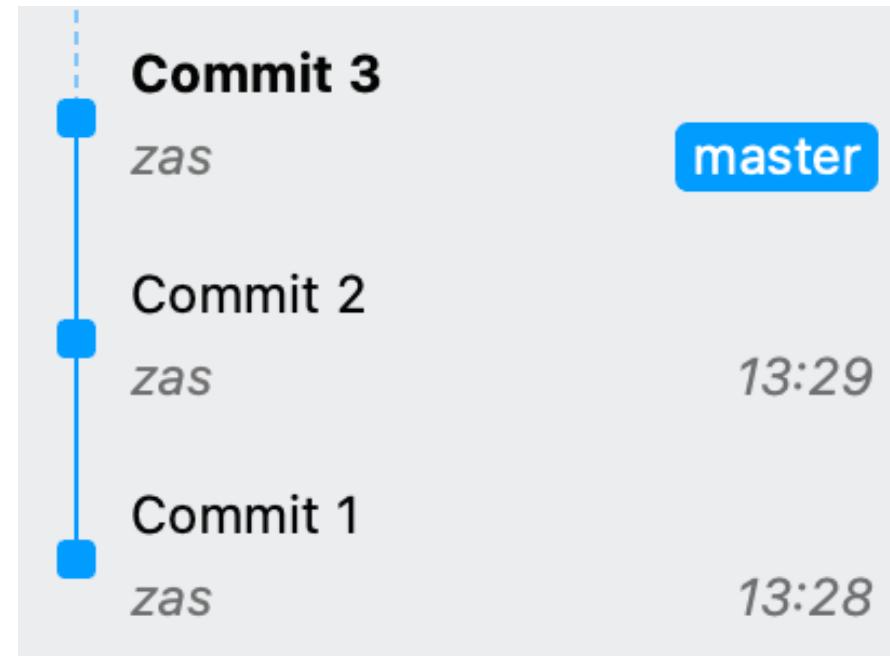




Branch and Merge

Initial repo state

Every repo has either a **main** or **master** branch as default branch



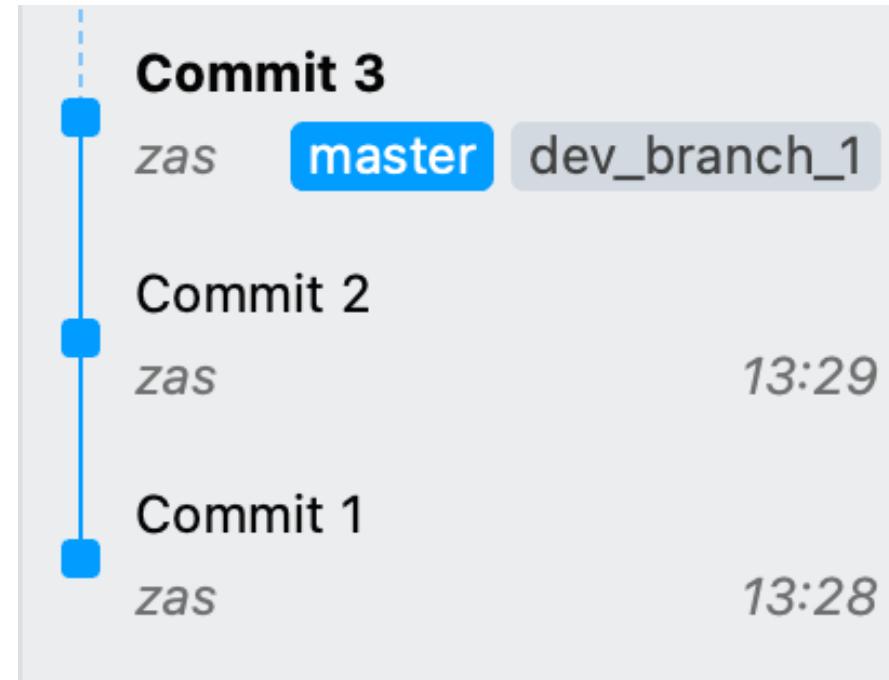
Branch and Merge

Create branch dev_branch_1



Create branch dev_branch_1

```
$ git branch dev_branch_1
```





Branch and Merge

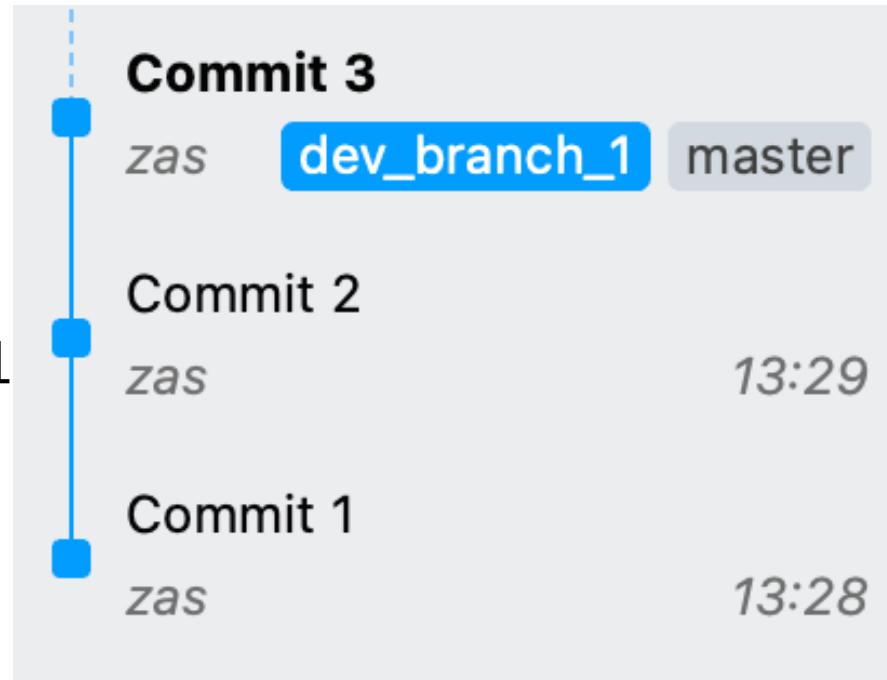
Checkout branch dev_branch_1

Check on which branch we are on

```
$ git branch
```

Checkout branch dev_branch_1

```
$ git checkout dev_branch_1
```





Branch and Merge

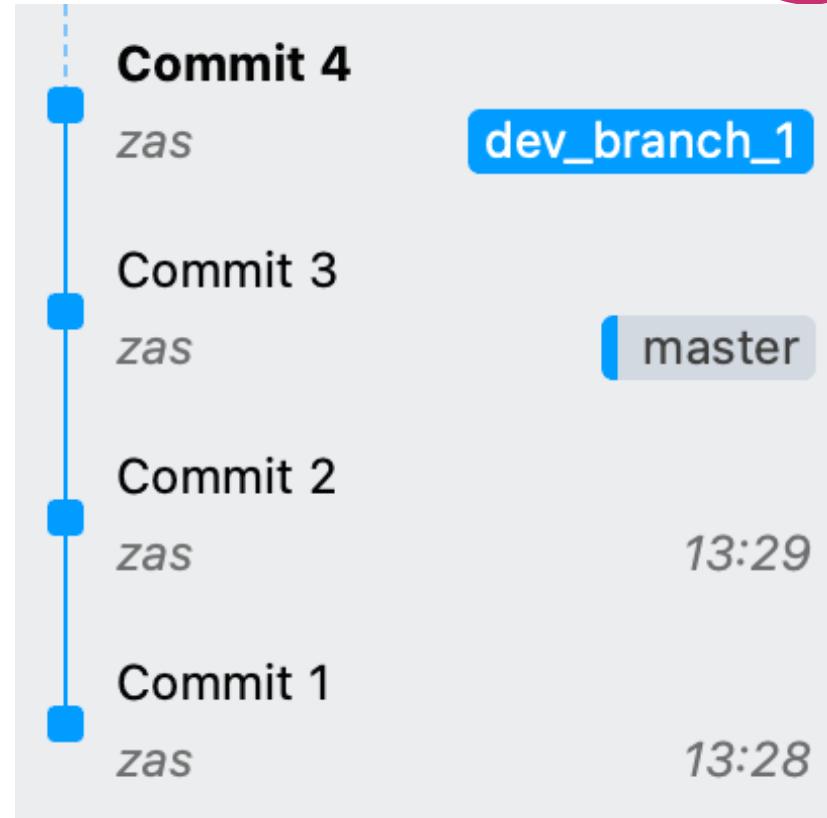
Commit on dev_branch_1

Stage new file

```
$ git add file.md
```

Commit stages files

```
$ git commit -m "Commit 4"
```





Branch and Merge

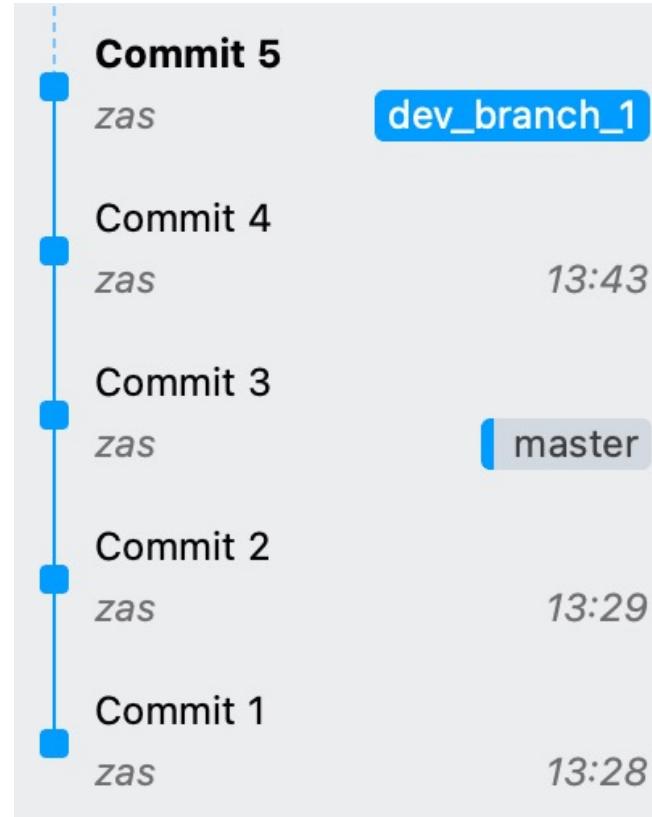
Commit on dev_branch_1

Stage new file

```
$ git add file.md
```

Commit stages files

```
$ git commit -m "Commit 5"
```





Branch and Merge

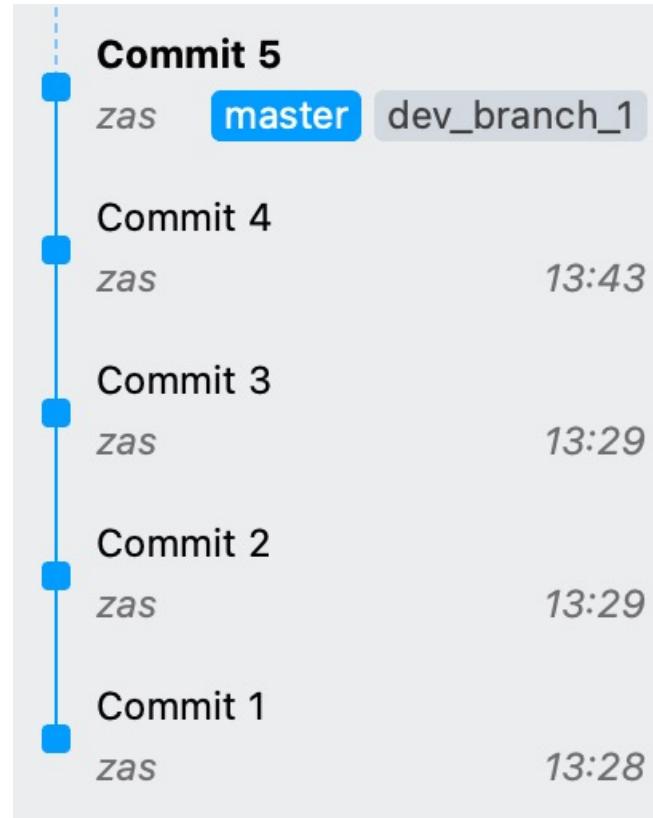
Merge master and dev_branch_1

Checkout master branch

```
$ git checkout master
```

Merge dev_branch_1 into master

```
$ git merge dev_branch_1
```





Branch and Merge

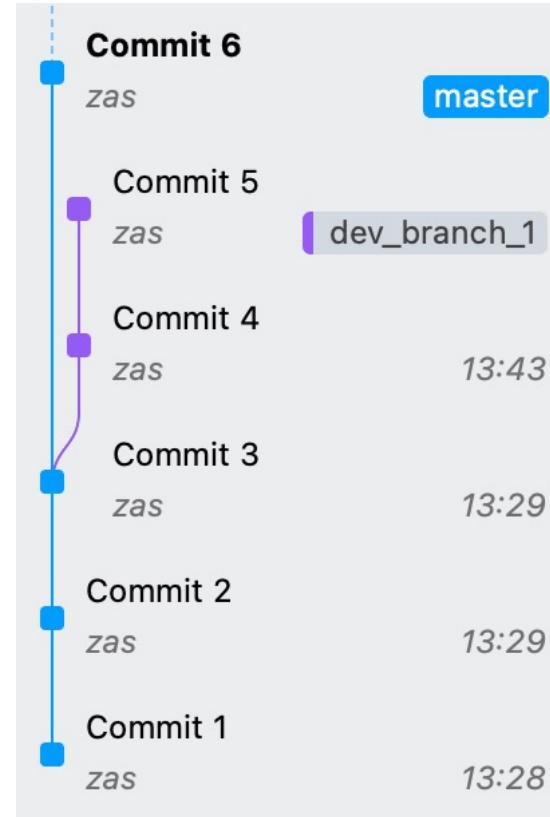
Commit on master branch

Stage new file

```
$ git add file.md
```

Commit stages files

```
$ git commit -m "Commit 6"
```



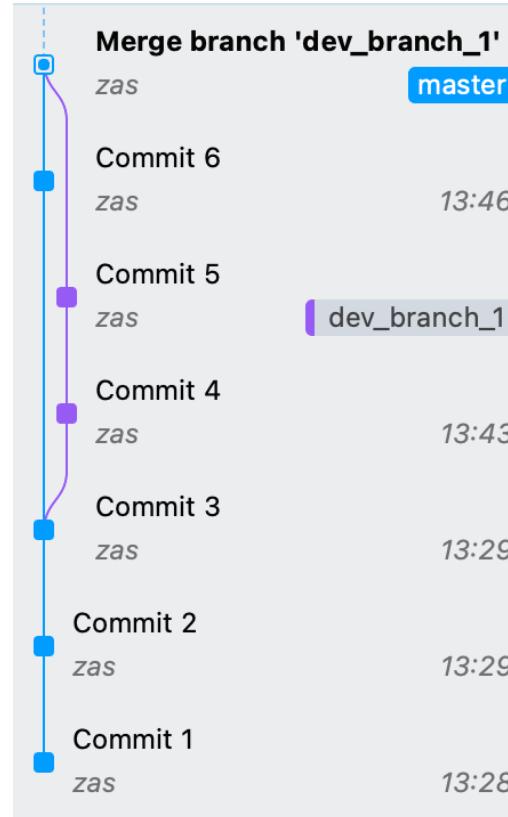
Branch and Merge

Three way merge master and dev_branch_1

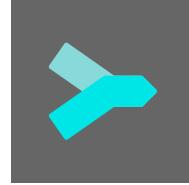


Merge dev_branch_1 into master

```
$ git merge dev_branch_1
```



Demo



■ -/work/repo/edu/car/car-course

LICENSE UPGRADE REQUIRED

did-course car-course car-exams znotes syd-course

Commits Files Summary

BRANCHES (1) master

REMOTES (1) origin

TAGS (0)

STASHES (0)

SUBMODULES (0)

Commit Hash: 702c8f1738addb639884c48b724ab86361d3c
Tree: 702c8f1738addb639884c48b724ab86361d3c
Author: zas <silvan.zahn@hevs.ch>
Date: Thu, 20 Apr 2023 08:12
Parents: 6e927fe, 68810079
Branches: master, origin/master
Stats: 16 files changed: 15 +110

Merge remote-tracking branch 'origin/master' (16)

CHG: updated planning (4)

ADD: ALU and ImmSrc doc (4)

ADD: EBS2/EBS3 specs (4)

FIX: memory stack images (5)

FIX: errors in immediate and type images (44)

ADD: files in arc exercises (33)

ADD: note on Ripes memory management (11)

CHG: Planning (4)

FIX: reverse engineering solution (8)

FIX: ISA syntax errors (3)

Merge remote-tracking branch 'origin/master' (16)

CHG: BEM labo from geekbench 5 to 6 (14)

UPD: all PDFs (30)

FIX: errors in ARC, ISA, FUN and PER slides (26)

Thu 08:11

Thu, 13 Apr 15:19

Tue, 11 Apr 15:25

Tue, 4 April 11:00

Tue, 4 April 07:46

Mon, 3 April 13:30

Fri, 31 Mar 15:38

Fri, 31 Mar 08:45

Thu, 30 Mar 17:25

Tue, 28 Mar 07:59

Thu, 28 Mar 07:29

Thu, 16 Mar 10:14

Thu, 16 Mar 09:31

Tue, 14 Mar 14:25

Wed, 8 Mar 19:10

Thu, 7 Mar 09:09

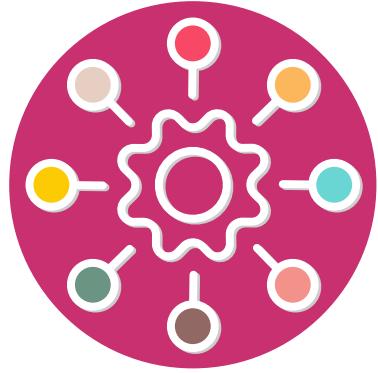
Summary: 00-solution.tex 03-controlunit.tex 04-simulation.tex 05-deployment.tex 05-deployment_ebs3.tex 08-CAR-Labor-SCR-d.pdf 08-CAR-Labor-SCR-f.pdf 08-CAR-Labor-SCR-s.pdf 08-CAR-Labor-SCR-t.pdf

Subsection: Simulation:

```
164 done;
165 beq x2, x2, main      # infinite loop
166 end(minted)
167
168 Each instruction takes one clock cycle => 19 are executed (addi x5, x0, 0 not
169 executed because of previous beq, and addi x2, x0, 1 not executed because of jal).
170 => 19/65M = 287.9 ns.
171 On the EBS2 board @ 60MHz \rightarrow t_{exec} = \frac{nb\_cycles}{f_{sys}} = \frac{19}{60M} = 287.9 ns.
172 On the EBS3 board @ 50MHz \rightarrow t_{exec} = \frac{nb\_cycles}{f_{sys}} = \frac{19}{50M} = 380 ns.
173
174 begin(center)
175   centerLine{\includegraphics[width=0.9\paperwidth]{scr/sol/simulation.pdf}}
176
177 Subsection: Umsetzung:
```

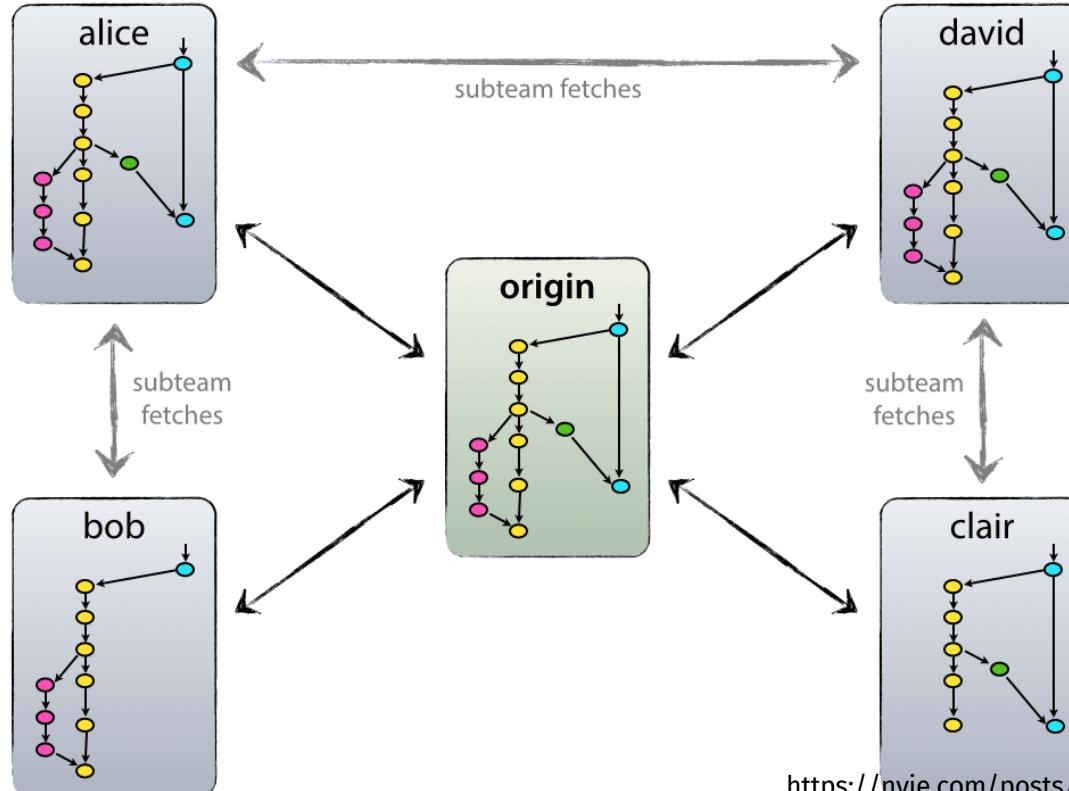
Subsection: Umsetzung:

```
63 Le \textbf{mainDecoder} peut être écrit en VHDL. Pour cela, vous pouvez analyser
64 l'exemple de code \ref{fig:riscv-mainDecoder-code} ci-dessous et l'adapter en
65 conséquence.
66
67 \begin{center}
68 \begin{tikzpicture}[font=\scriptsize]
69   \node [block] (mainDecoder) {\textbf{mainDecoder}};
70   \node [block, below=of mainDecoder] (alu) {ALU};
71   \node [block, below=of alu] (aluDecoder) {ALU Decoder};
72   \node [block, below=of aluDecoder] (ctrl) {Control Logic};
73   \node [block, below=of ctrl] (mem) {Memory};
74   \node [block, below=of mem] (bus) {Bus};
75
76   \draw [arrow] (mainDecoder) --> alu;
77   \draw [arrow] (alu) --> aluDecoder;
78   \draw [arrow] (aluDecoder) --> ctrl;
79   \draw [arrow] (ctrl) --> mem;
80   \draw [arrow] (mem) --> bus;
81   \draw [arrow] (bus) --> mainDecoder;
82
83   \draw [arrow] (mainDecoder) --> aluDecoder;
84   \draw [arrow] (aluDecoder) --> alu;
85   \draw [arrow] (alu) --> aluDecoder;
86   \draw [arrow] (ctrl) --> aluDecoder;
87   \draw [arrow] (ctrl) --> alu;
88   \draw [arrow] (ctrl) --> mem;
89   \draw [arrow] (mem) --> bus;
90   \draw [arrow] (bus) --> mainDecoder;
91
92   \draw [arrow] (ctrl) --> mainDecoder;
93   \draw [arrow] (ctrl) --> aluDecoder;
94   \draw [arrow] (ctrl) --> alu;
95   \draw [arrow] (ctrl) --> mem;
96   \draw [arrow] (mem) --> bus;
97   \draw [arrow] (bus) --> mainDecoder;
98
99   \draw [arrow] (ctrl) --> aluDecoder;
100  \draw [arrow] (ctrl) --> alu;
101  \draw [arrow] (ctrl) --> mem;
102  \draw [arrow] (mem) --> bus;
103  \draw [arrow] (bus) --> mainDecoder;
104
105  \draw [arrow] (ctrl) --> mainDecoder;
106  \draw [arrow] (ctrl) --> aluDecoder;
107  \draw [arrow] (ctrl) --> alu;
108  \draw [arrow] (ctrl) --> mem;
109  \draw [arrow] (mem) --> bus;
110  \draw [arrow] (bus) --> mainDecoder;
111
112  \label{fig:riscv-mainDecoder-code}
113
114 \end{center}
115
116 L'ALU réalise les fonctions arithmétiques et logiques selon la table suivante:
117
118
119 Die ALU realisiert die arithmetischen und logischen Funktionen gemäß der folgenden
Tabelle:
120
121
122 \begin{table}[h]
```



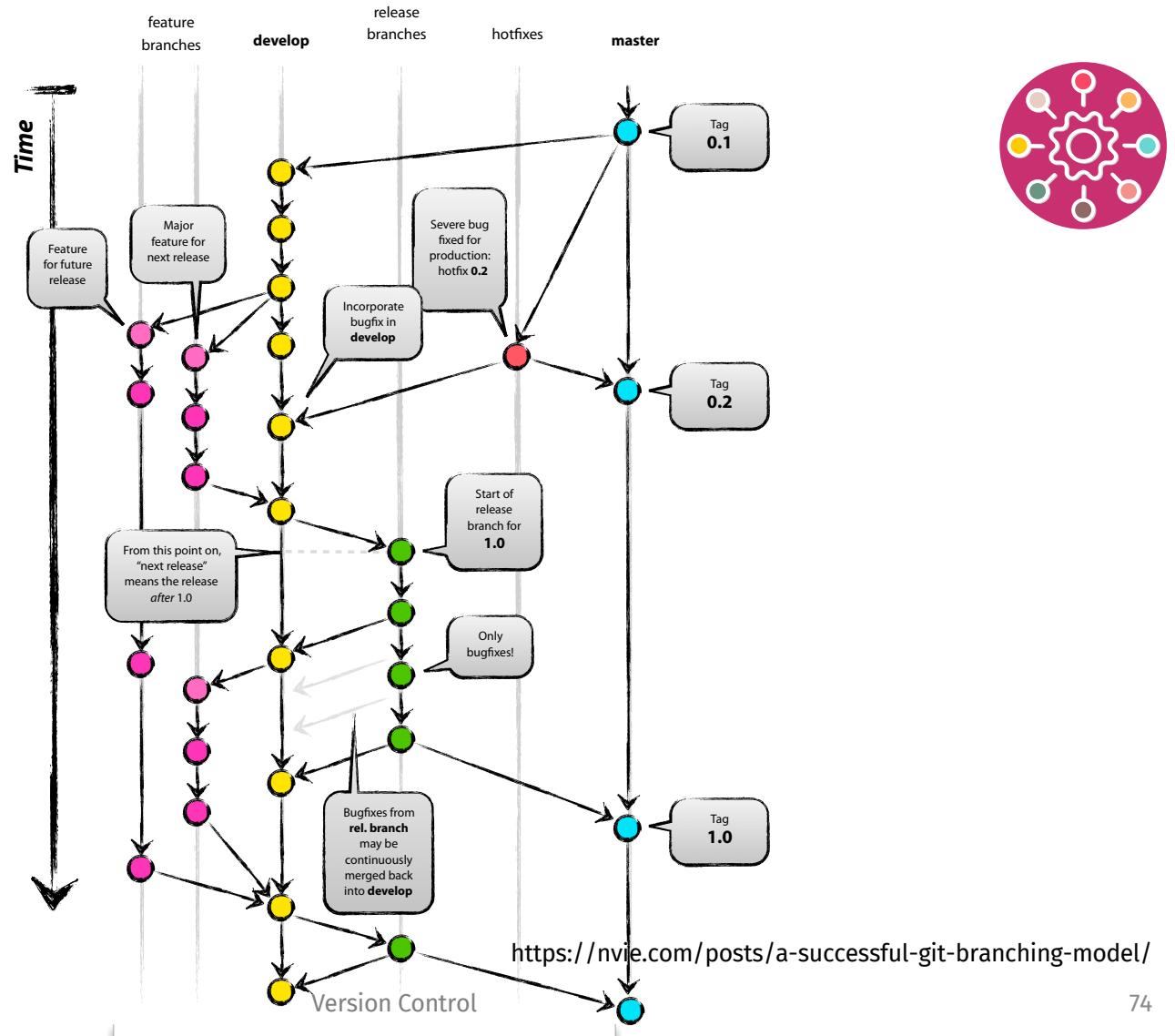
Gitflow

Gitflow Collaboration

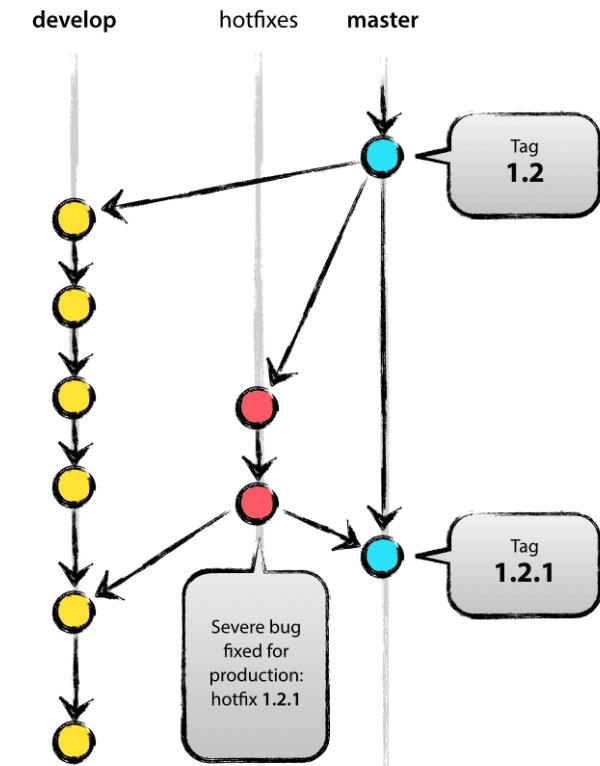
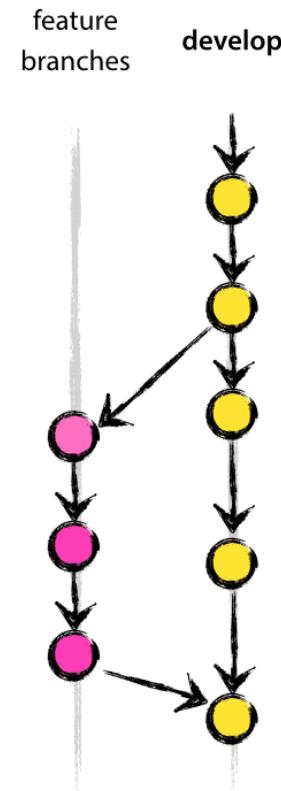
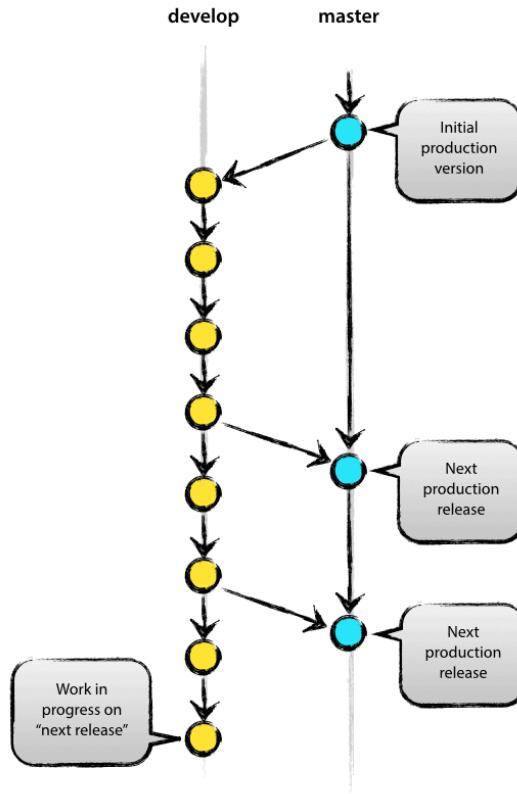


<https://nvie.com/posts/a-successful-git-branching-model/>

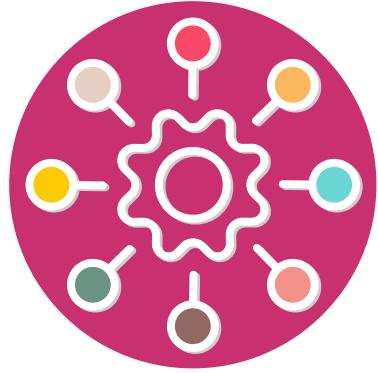
Gitflow



Gitflow Branching



<https://nvie.com/posts/a-successful-git-branching-model/>



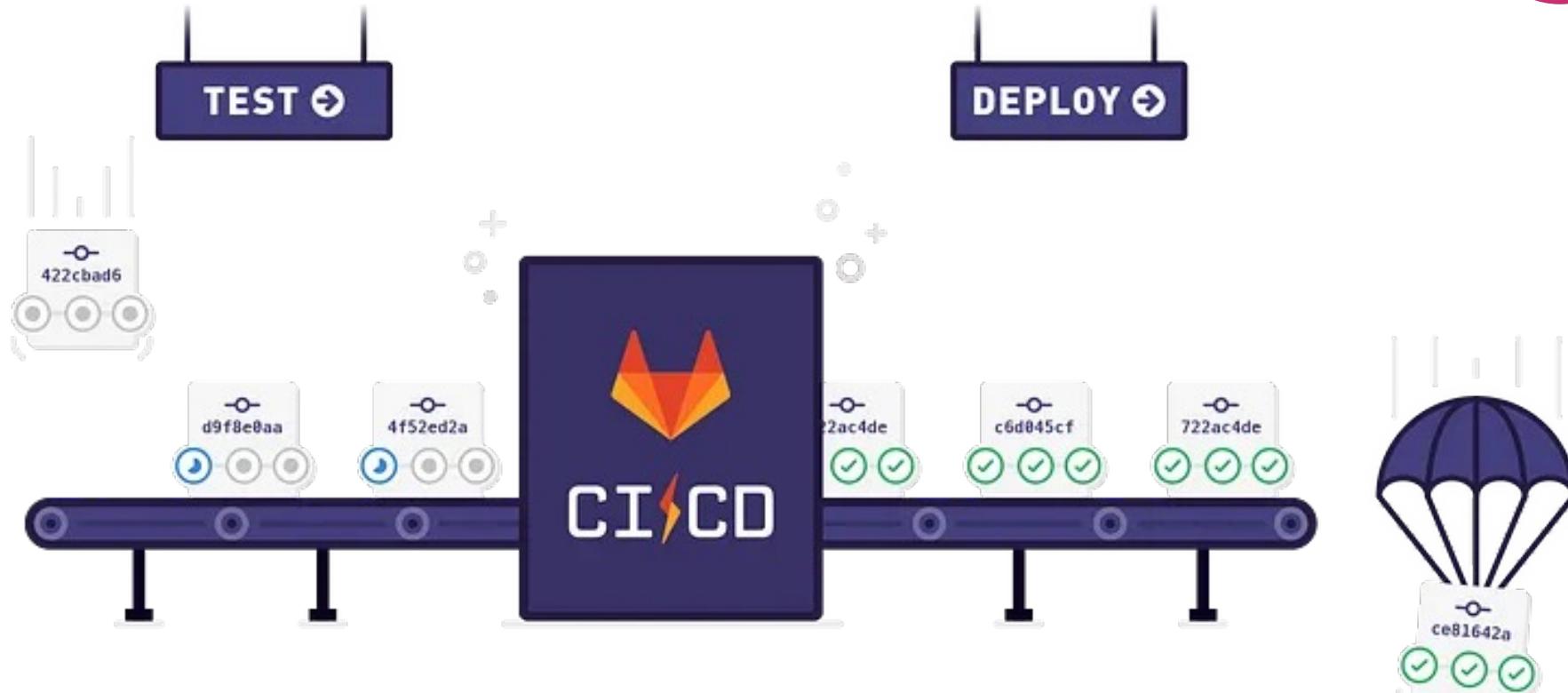
Git CI/CD

What is CI/CD?



- Continuous Integration (CI) is the practice of frequently integrating code changes into a shared repository, which is then automatically built and tested.
- Continuous Delivery (CD) takes CI a step further by automatically deploying code changes to production-like environments for further testing and validation.
- Automated testing is a critical component of CI/CD, as it helps catch bugs and other issues early in the development process.
- Popular tools are GitLab CI/CD, Github Actions, Jenkins, CircleCI, and Travis CI.

What is CI/CD?

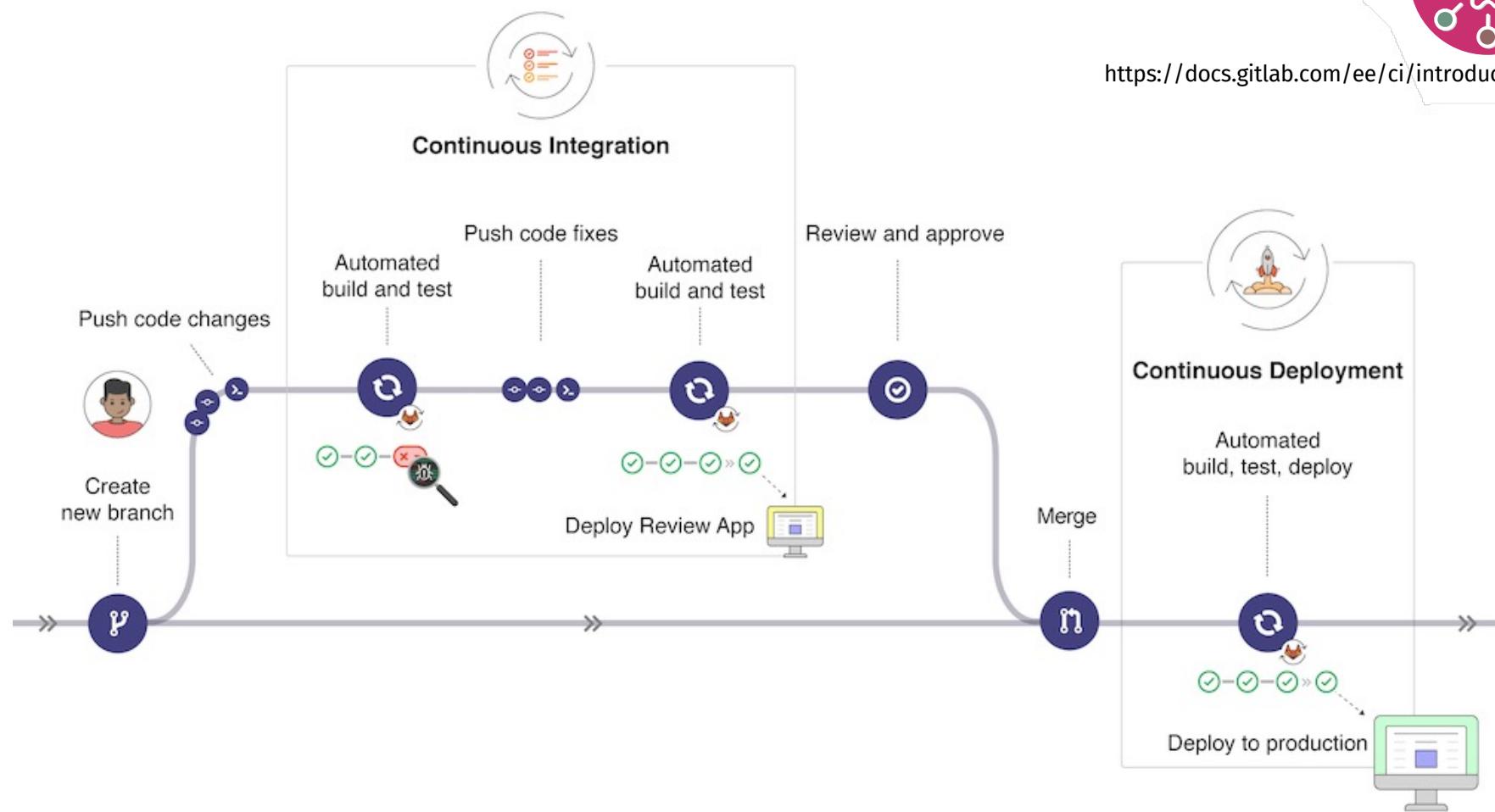


<https://about.gitlab.com/features/continuous-integration/>

Gitlab Workflow



<https://docs.gitlab.com/ee/ci/introduction/>

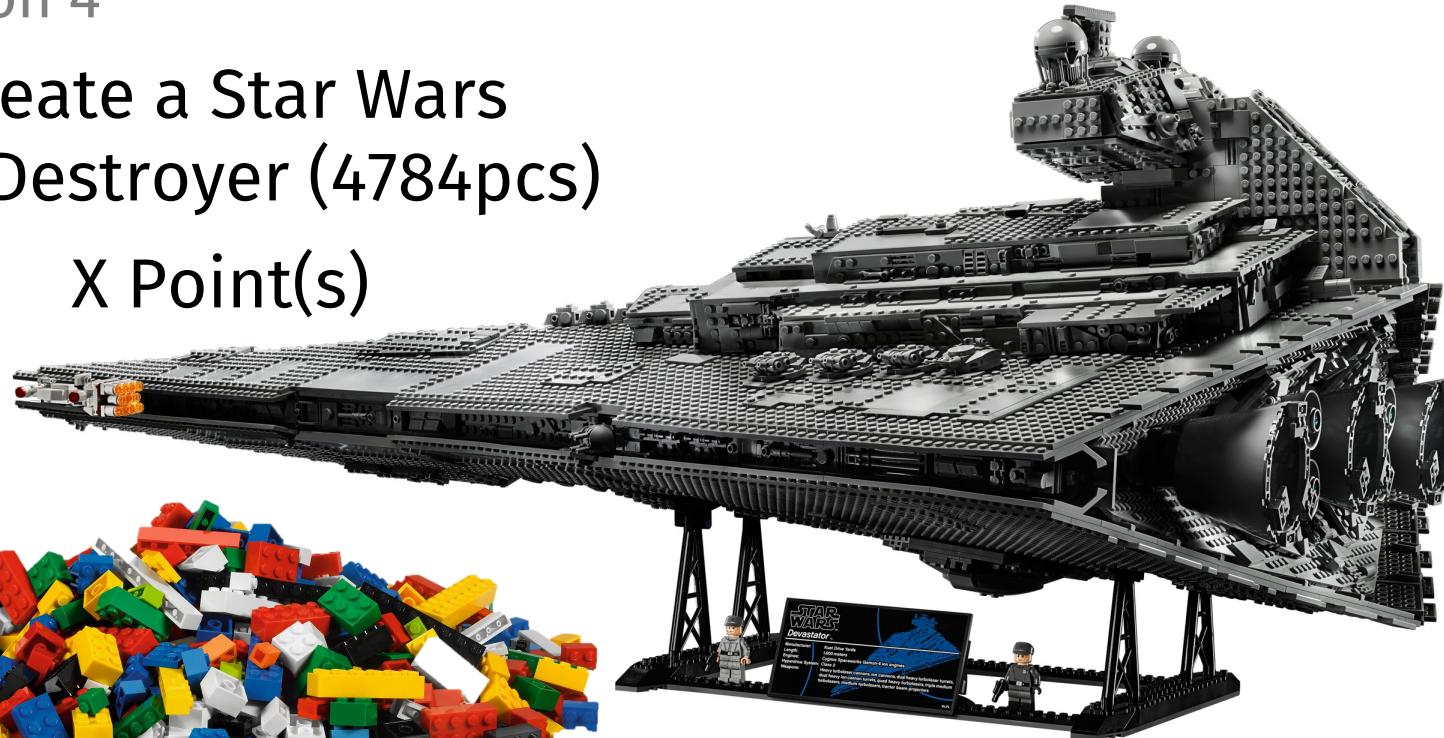


Task Estimation

Estimation 4



Create a Star Wars
Star Destroyer (4784pcs)
X Point(s)



| | | | |
|----|---------------|-----|----------|
| 0 | $\frac{1}{2}$ | 1 | 2 |
| 3 | 5 | 8 | 13 |
| 20 | 40 | 100 | ∞ |
| ? | | | |

Task Estimation

Estimation 5

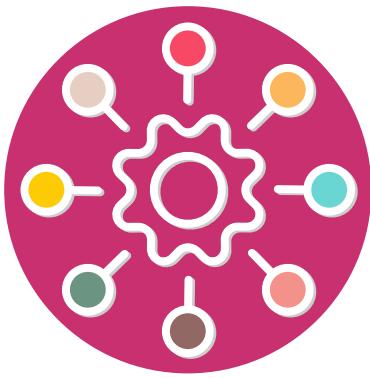


Create a Star Wars
Star Destroyer (37pcs)

X Point(s)



| | | | |
|----|---------------|-----|----------|
| 0 | $\frac{1}{2}$ | 1 | 2 |
| 3 | 5 | 8 | 13 |
| 20 | 40 | 100 | ∞ |
| ? | ☕ | | |



INTRODUCTION TO GIT FILE VERSIONING



Introduction to git file versioning



Contents

| | |
|--------------------------|----|
| 1 Goal | 1 |
| 2 Installation | 2 |
| 3 Basic operations | 5 |
| 4 Branch and Merge | 9 |
| 5 Gitflow | 10 |
| 6 GIT commands | 12 |
| Bibliography | 14 |