ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

02
Chapter

Distributed Version Control
The GitLab example

2I1AC3 : Génie logiciel et Patrons de conception

Régis Clouard, ENSICAEN - GREYC

"Il est facile se tirer une balle dans le pied avec Git, mais il est également facile de revenir à un pied précédent et de le fusionner avec sa jambe actuelle."

Jack William Bell

Recall: Git objectives

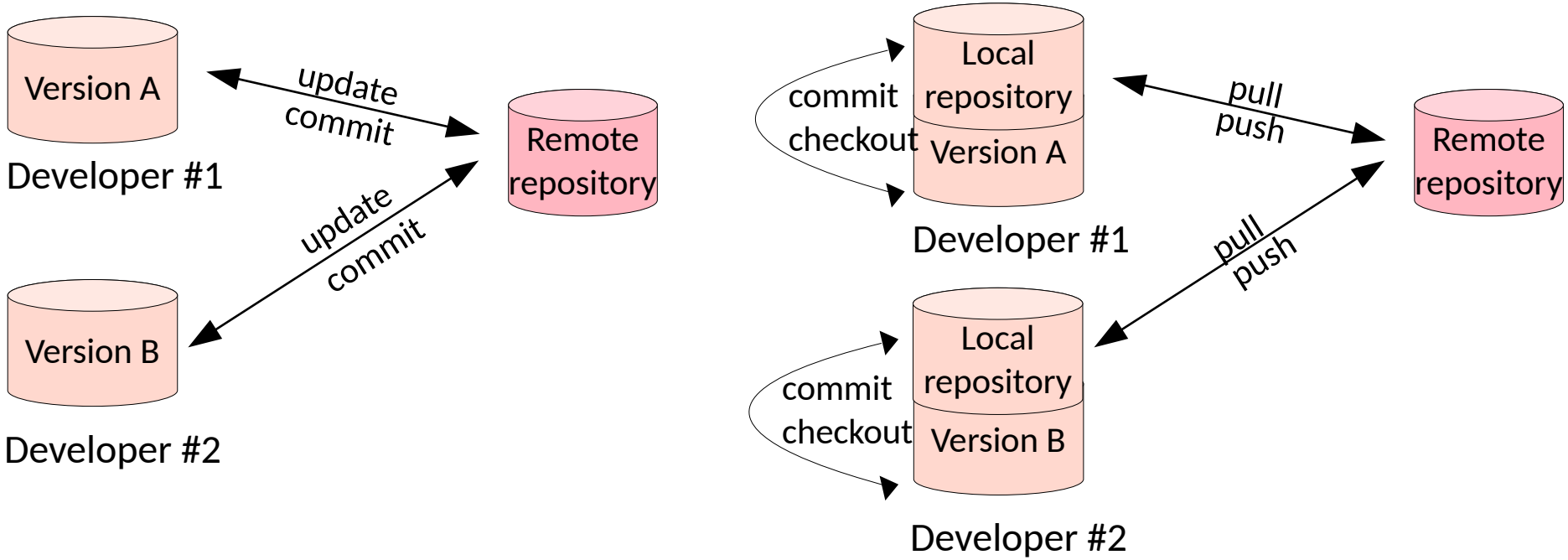
1) For tracking and managing changes to files (lecture 1)
   - `add` + `commit` + `checkout`
2) For managing multiple versions of the same project (lecture 1)
   - `branch+checkout+merge`
3) For sharing files (lecture 2)
4) For organizing collaborative work (lecture 2)

Outline

1
Centralized
Versus
Distributed
Version Control

Distributed vs Centralized Version Control

- Git is a distributed Version Control System
  - Each contributor has a complete and distinct copy of the repository
  - Can be used offline



Setting Up the Communication Protocol

- **https protocol**
  - Needs to enter the password for each writing transaction
  - **Hint**: cache the password (in memory)

```
$ git config –-global credential.helper ‘cache –-timeout=7200’ # 2 hours
```

- **Git protocol**
  - Needs SSH keys

Outline

1
Centralized
Versus
Distributed
Version Control

2
Remote Repository
for sharing files (3)

Create Remote Repository

- On a server accessible by way of URL

  (e.g. *https://www.ecole.ensicaen.fr/~rclouard/*)

  - Create a remote repository (*bare*: without working directory)

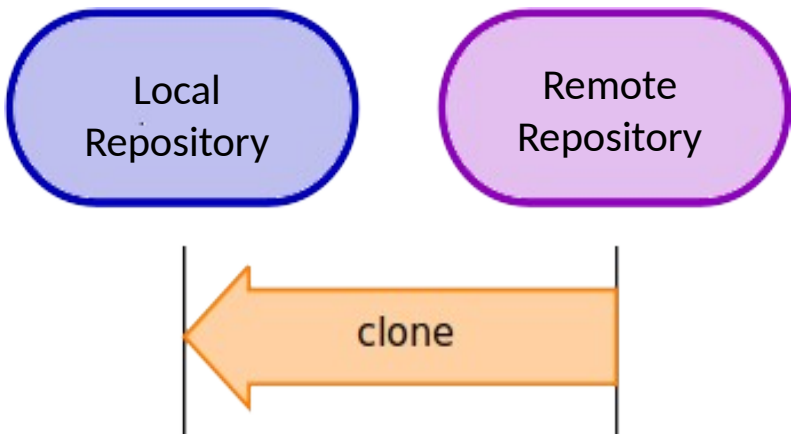
```
$ git init --bare my_repo.git
$ cd my_repo.git
$ git update-server-info
$ cd hooks
$ mv post-update.sample post-update
```

Local Repository

- Get a copy of the remote repository

```
$ git clone https://www.ecole.ensicaen.fr/~rclouard/my_repo.git
$ cd my_repo
```

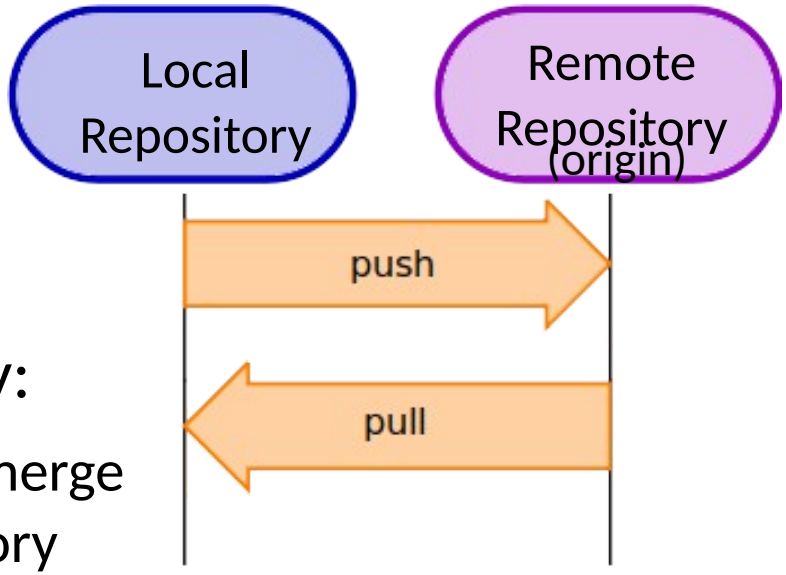
- Remote repository is named "*origin*" by default



Local Repository ↔ Remote Repository

- **Fetch** the repository:
  - update local repository from remote repository (all branches) but not working directory

```
$ git fetch
```



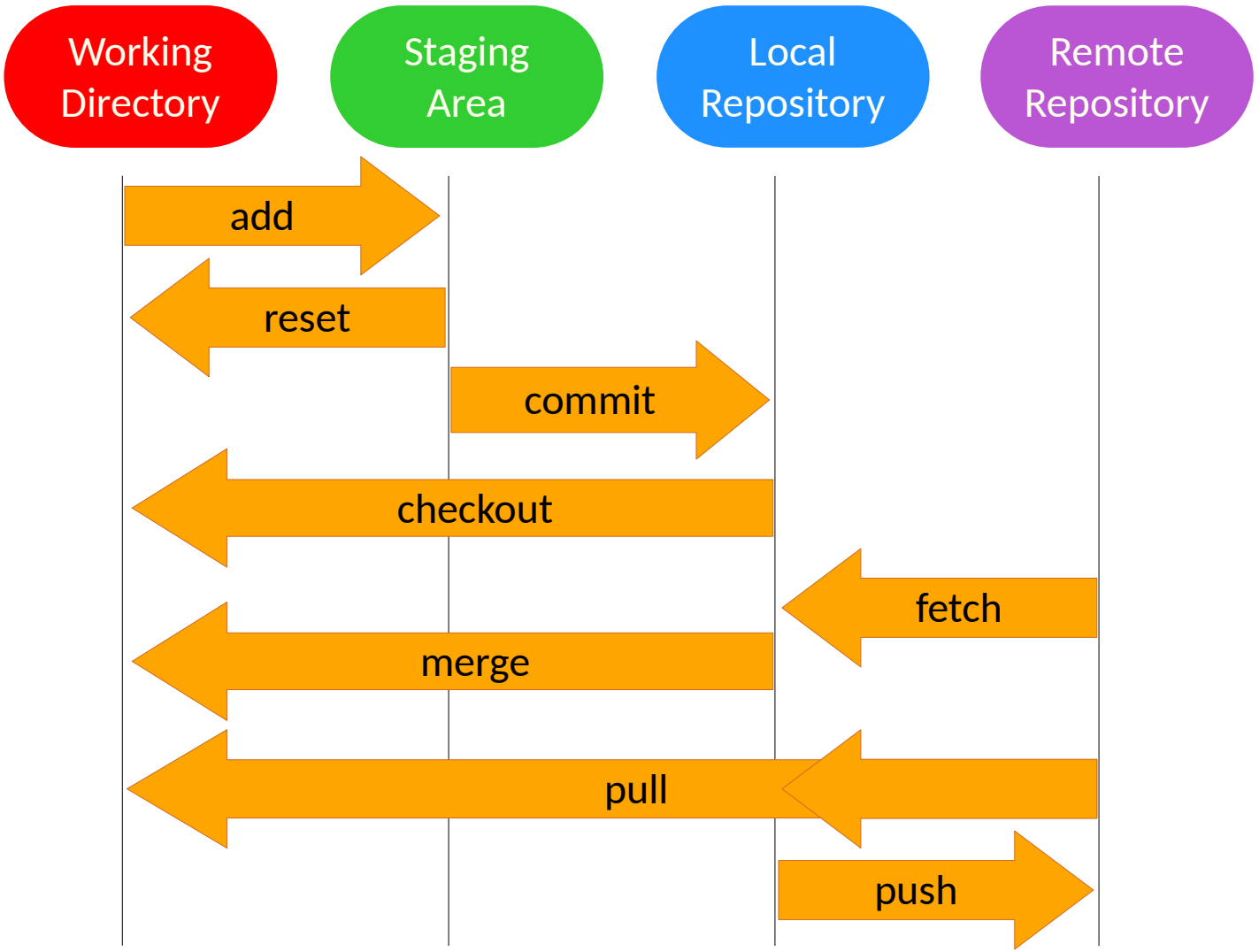
- **Pull** the version from the remote repository:
  - Update the local repository (all branches) and merge the main branch into the current branch directory

```
$ git pull origin main
```

  - *pull is equivalent to fetch + merge main branch into the current branch*
- Push the current branch of the local repository to the remote repository

```
$ git push
```

Decentralized Version Control with Git



Outline

1
Centralized
Versus
Distributed
Version Control

2
Remote
Repository
for sharing files (3)

3
The GitLab
Example

Remote Repository with GitLab

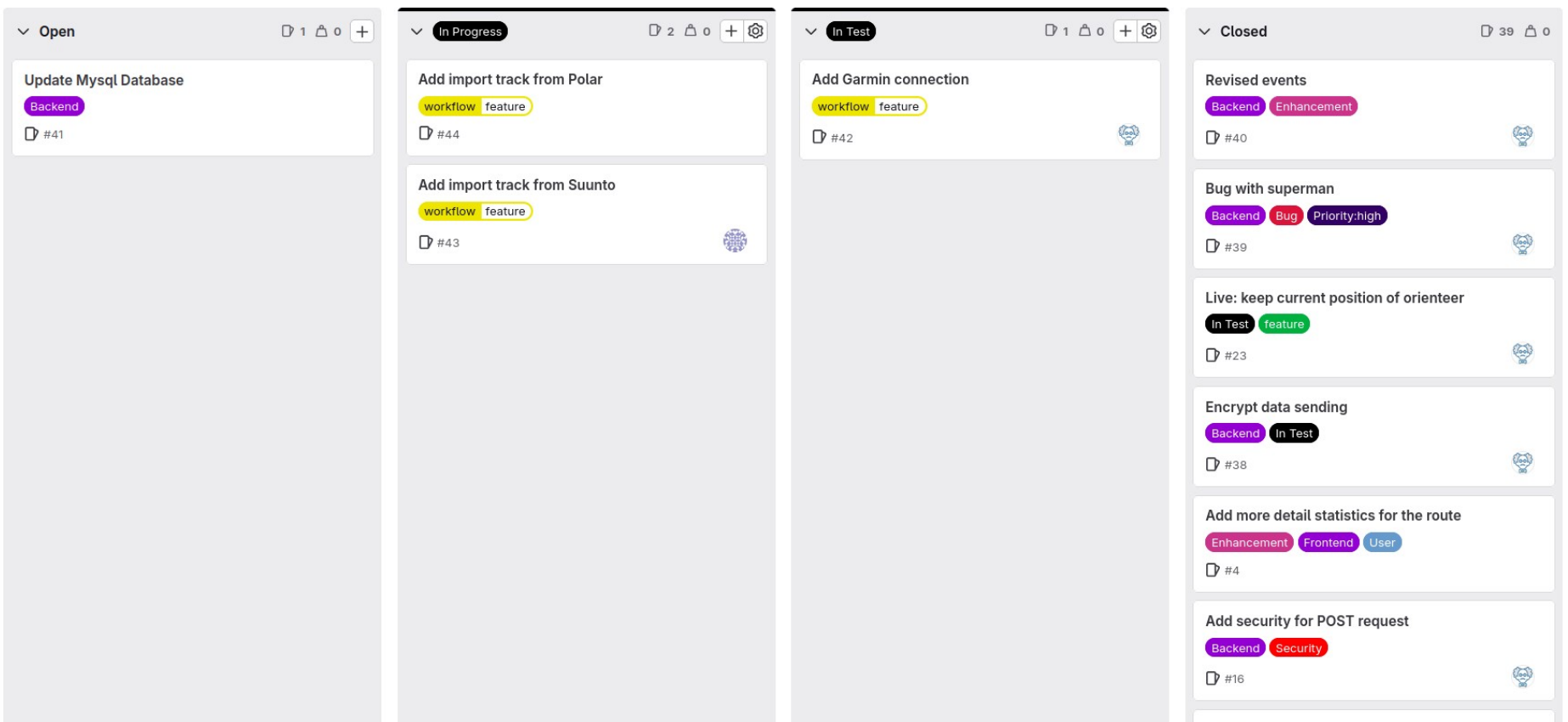
- GitLab is a free web-based collaborative software platform (*aka forge*) built on top of Git
  - ENSICAEN hosts a local version
    - *`https://gitlab.ecole.ensicaen.fr`*

GitLab Features: (1) Repository

- Git Repository
  - Managed by “maintainer” (*aka “version manager”*)
    - modify the “main” branch
    - add / remove developers from the project

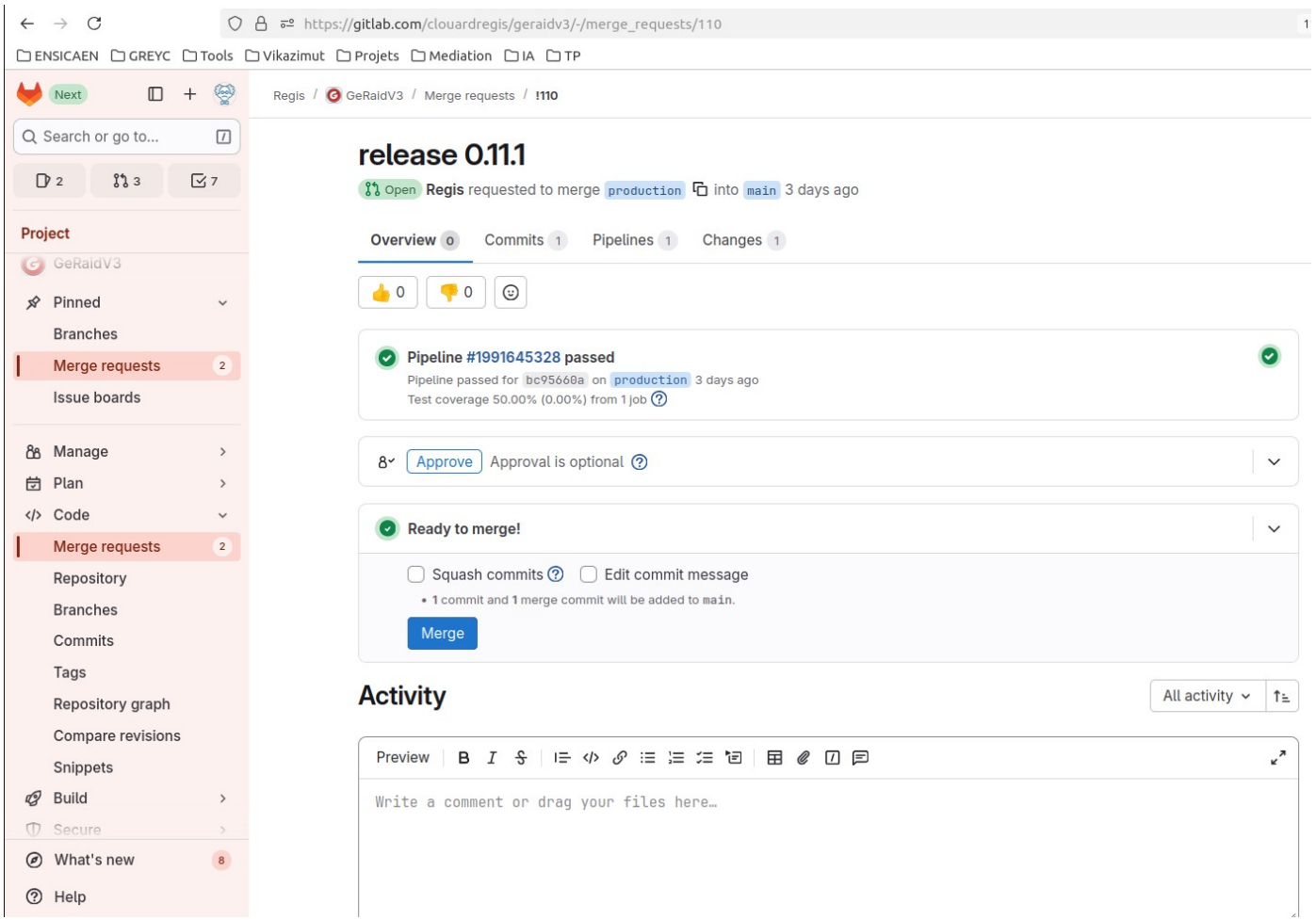
GitLab Features: (2) Kanban Board

- Kanban board based on issues
  - Issue: number + name + labels + assignment + comments



Merge GitLab Features: (3) Request

- Merge branches
- Run CI/CD



GitLab Features: (4) CI/CD tool

- CI/CD: continuously release changes to end environment
  - CI: **Continuous Integration**
    - Build + Test verification
  - CD:
    - **Continuous Deployment**: deploy automatically (deployment)
    - **Continuous Delivery**: deploy manually: someone decides when to release
- Pipelines
  - **Pipelines** are made up of jobs
  - **Jobs** are code executed with **runner**. For example, a job can compile or test code
  - **Runners** are docker containers
    - Use dedicated image (Docker container)

GitLab Features: (4) CI/CD tool

- Script file: *.gitlab-ci.yml*

```
image: gradle:jdk16

variables:
  # Using a fresh runtime for each build to isolate each runtime from any previous builds.
  GRADLE_OPTS: "-Dorg.gradle.daemon=false"
  # Variable for the runner at ENSICAEN
  http_proxy: http://193.49.200.22:3128

before_script:
  - chmod +x gradlew
  - mv gradle.properties.ci gradle.properties
  - apt-get install findutils

unit_test:
  tags :
    - gitlab-runner-ensicaen
  stage: test
  coverage: '/Coverage Total: ([0-9]{1,3})%/'
  only:
    - main
  script:
    - ./gradlew -Pci --console=plain test jacocoTestReport
    - cat build/jacocoHtml/index.html | grep -o 'Total[^%]*%' | sed 's/<.*>/ /; s/Total/Coverage Total:/'

Build: …

Deploy: …
```