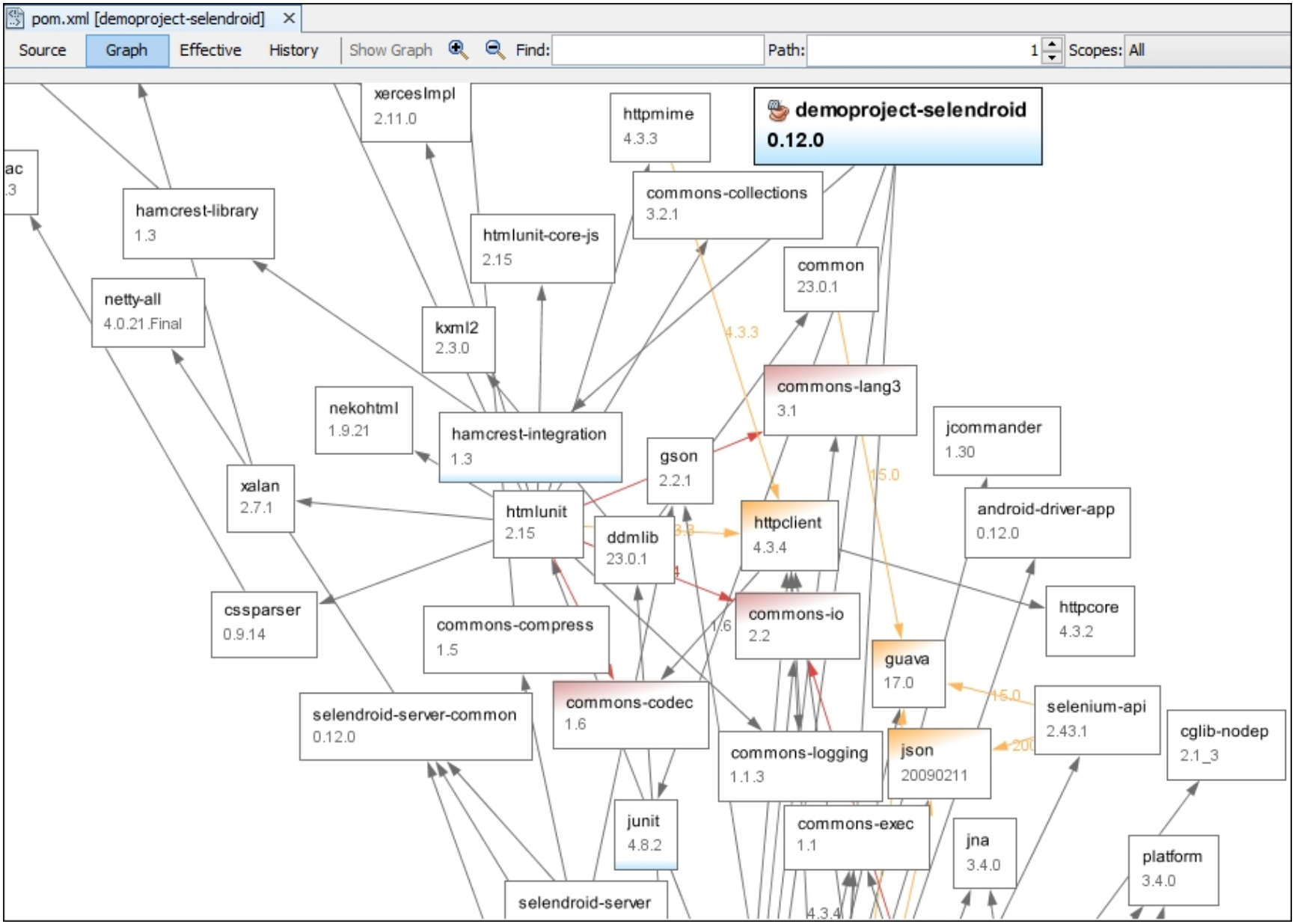
GIT & GitLab
Version Control Management
How to use it for
Agile Project Management

International Research Project Management

January 2020

Olivier.Boissier@emse.fr


pom.xml [demoproject-selendroid]
Source
Graph
Effective
History
Show Graph
Find:
Path:
1
Scopes:
All
demoproject-selendroid
0.12.0
xercesImpl
2.11.0
httpmime
4.3.3
commons-collections
3.2.1
hamcrest-library
1.3
htmlunit-core-js
2.15
common
23.0.1
netty-all
4.0.21.Final
kxml2
2.3.0
commons-lang3
3.1
nekohtml
1.9.21
hamcrest-integration
1.3
gson
2.2.1
jcommander
1.30
xalan
2.7.1
htmlunit
2.15
ddmlib
23.0.1
httpclient
4.3.4
android-driver-app
0.12.0
cssparser
0.9.14
commons-compress
1.5
commons-io
2.2
httpcore
4.3.2
guava
17.0
selendroid-server-common
0.12.0
commons-codec
1.6
selenium-api
2.43.1
cglib-nodep
2.1_3
commons-logging
1.1.3
json
20090211
junit
4.8.2
commons-exec
1.1
jna
3.4.0
platform
3.4.0
selendroid-server

Versioning

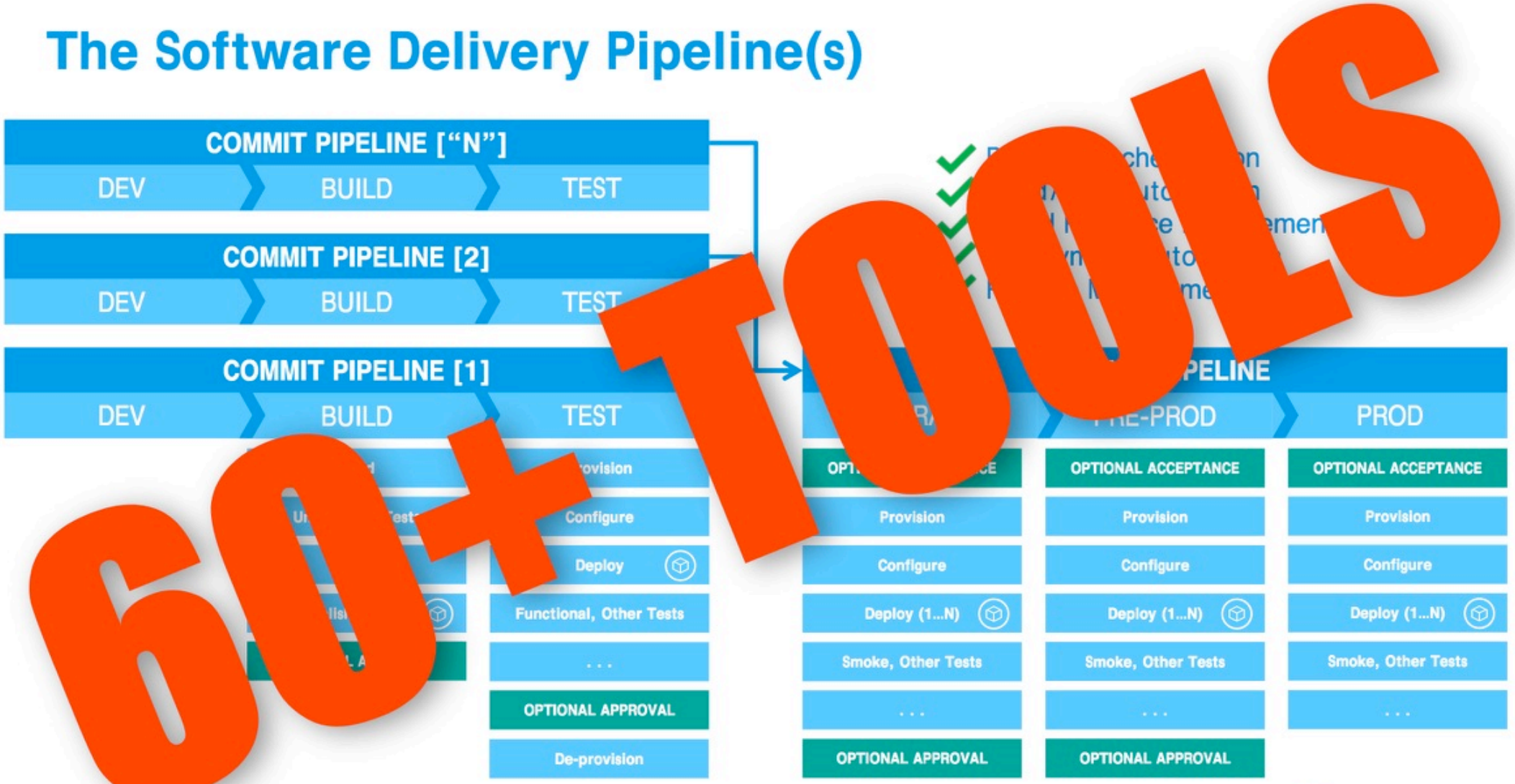
- Semantic Versioning 2.0.0 https://semver.org/

Automating

- Automate ALL the Things: Taking Advantage of Free Tools to Automate Your End-to-End Release Pipeline, Avantika Mathur https://saturn2017.sched.com/event/9bQZ/automate-all-the-things-taking-advantage-of-free-tools-to-automate-your-end-to-end-release-pipelines


API Technologies
API Infrastructure
LAYER 7
MuleSoft
MASHERY
RESTLET
apigee
3SCALE
API Middleware
OAuth.io
kloudless
singly
API Services
Runscope
import io
apiary
API Markets / Directories
mashape
programmableweb
Analytics and Testing
App Testing / Automation Technologies
Koality
CRITTERCISM
appurify
SAUCE LABS
APPLAUSE
Mobile App Analytics
FLURRY
Kontagent
swrve
Web App Analytics Technologies
mixpanel
App Performance Management
APPDYNAMICS
AppNeta
Coding Tools
Version Control
CVS
mercurial
SUBVERSION
git
Code Collaboration
Koding
github
Sourcegraph
runnable
VAGRANT
Open Source Code Frameworks
zend
RAILS
prototype
Meteor
django
Coding Environment
Atlassian
Dw
eclipse
BRAINS
NetBeans
Bowery
Open Source Coding Languages
Ruby
php
python
node
Continuous Integration
Jenkins
drone.io
circleci
Operating System Dev
Commercial Desktop OS
CANONICAL
redhat
Open Source Desktop OS
ubuntu
Linux
Mobile Device Dev
Windows Phone
Dev
BlackBerry Developer
iOS Dev Center
Developers
Data Dev Technologies
Big Data Dev Platforms
CONTINUUITY
hp
VERTICA
Cloudant
splunk>
Firebase
infochimps
NoSQL Technologies
Couchbase
mongoDB
cassandra
riak
Hadoop Dev Technologies
wibidata
Hortonworks
cloudera
NewSQL Technologies
FOUNDATIONDB
memsql
nuodb
Clustrix
splice MACHINE
Graph Database Technologies
neotechnology
Objectivity
SQL Technologies
MySQL
PERCONA
ORACLE
Cloud Tools
Cloud Hosting
rackspace
DreamHost
S3
greenqloud
Processing-as-a-Service
EC2
PubNub
xRTML
Iron.io
User Cloud Services
Stormpath
DevOps Technologies
Server Management
CoreOS
linode.com
CFEngine
Ops Management
CHEF
catchpoint
radware
Apica
DATADOG
pingdom
Dyn
Content Delivery
fastly
CLOUDFLARE
edgecast
Log management
loggly
logentries
sumologic
Configuration Management
SALTSTACK
puppet labs
CHEF
Dev Platforms
Web App Platforms
heroku
Engine Yard
docker
elasticbox
Mobile App Platforms
built.io
Kii
famous
appcelerator
Parse
Sencha
kinvey
PhoneGap
Commercial Language Platforms / Libraries
REVOLUTION ANALYTICS
Joyent
Typesafe
OPENSHIFT ONLINE
DevNetwork.com, 2014

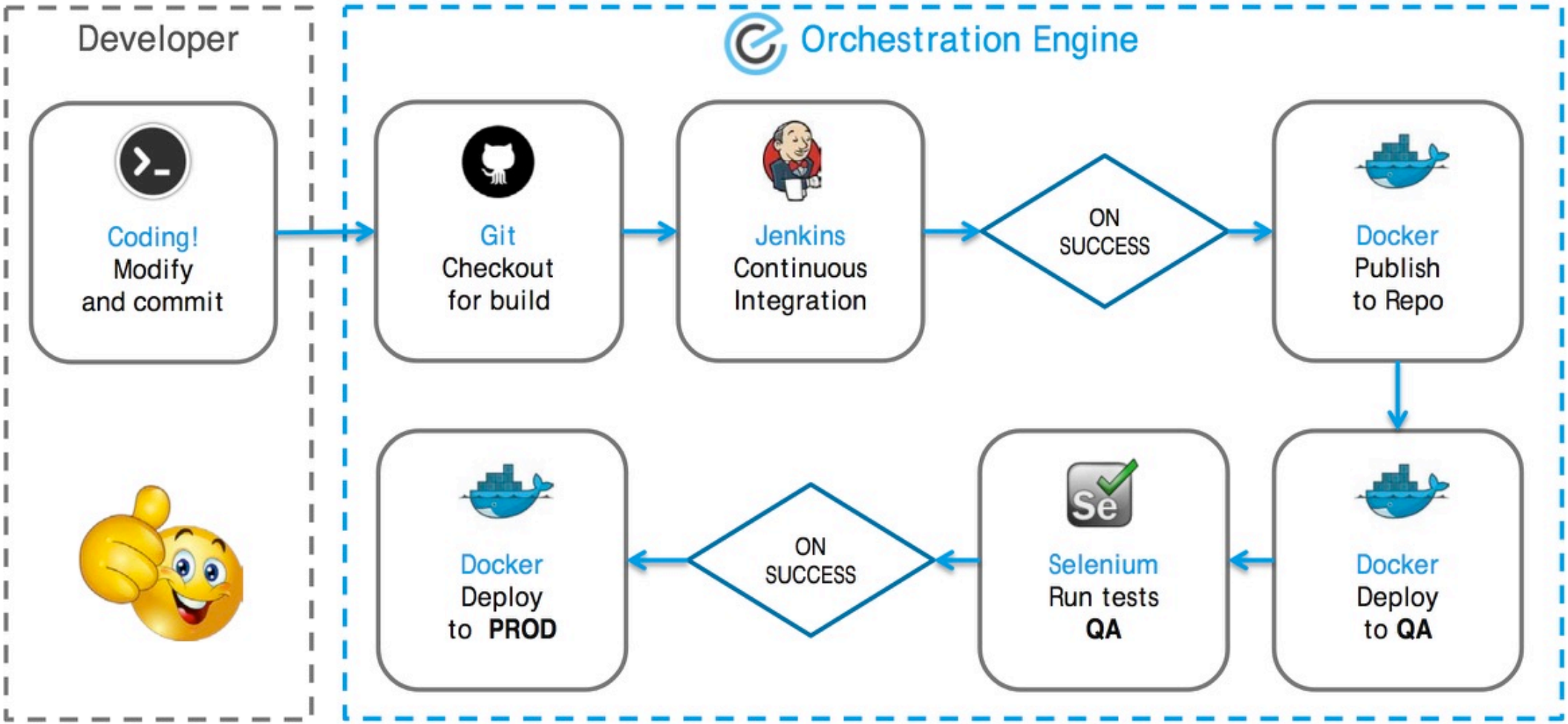
The Software Delivery Pipeline(s)



Typical number of tools * environments we run into for most legacy SDLC processes

Electric Cloud

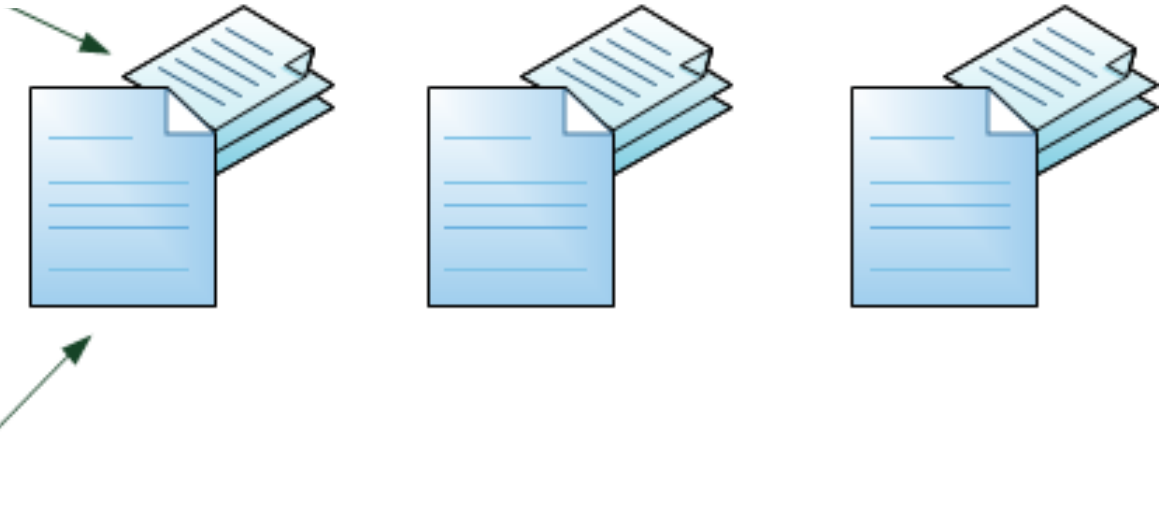
Example pipeline



ElectricCloud

Versioning

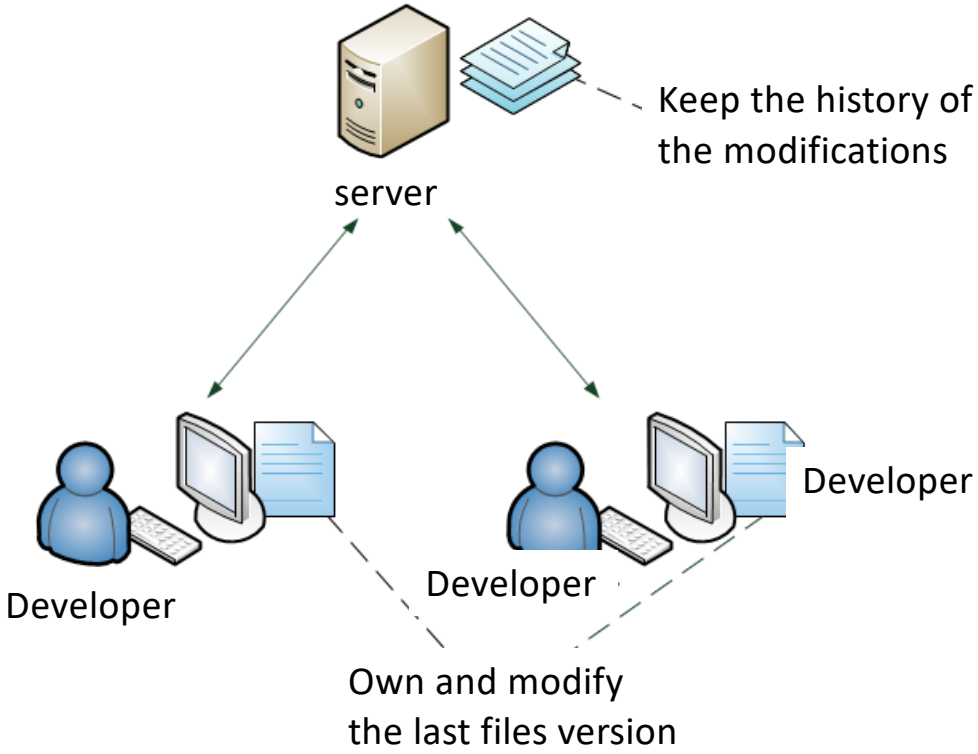
- Follow the evolution of a source code, to keep track of the modifications made on each file and thus be able to go back in case of a problem;



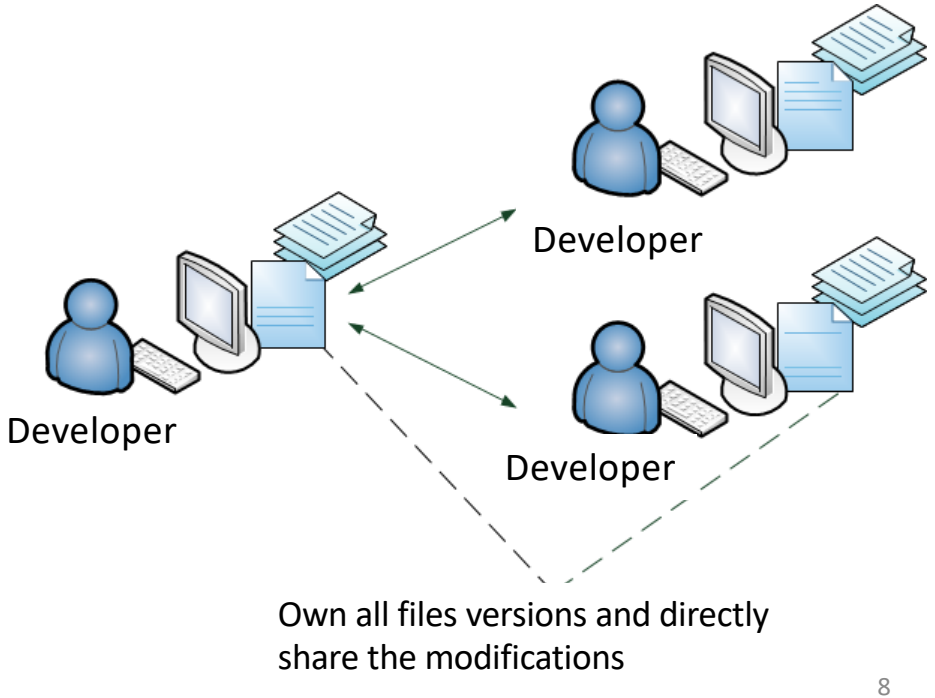
- Work in groups, without the risk of stepping on each other's toes.
  If two people modify the same file at the same time, their modifications must be able to be merged without loss of information.

Version Control Management

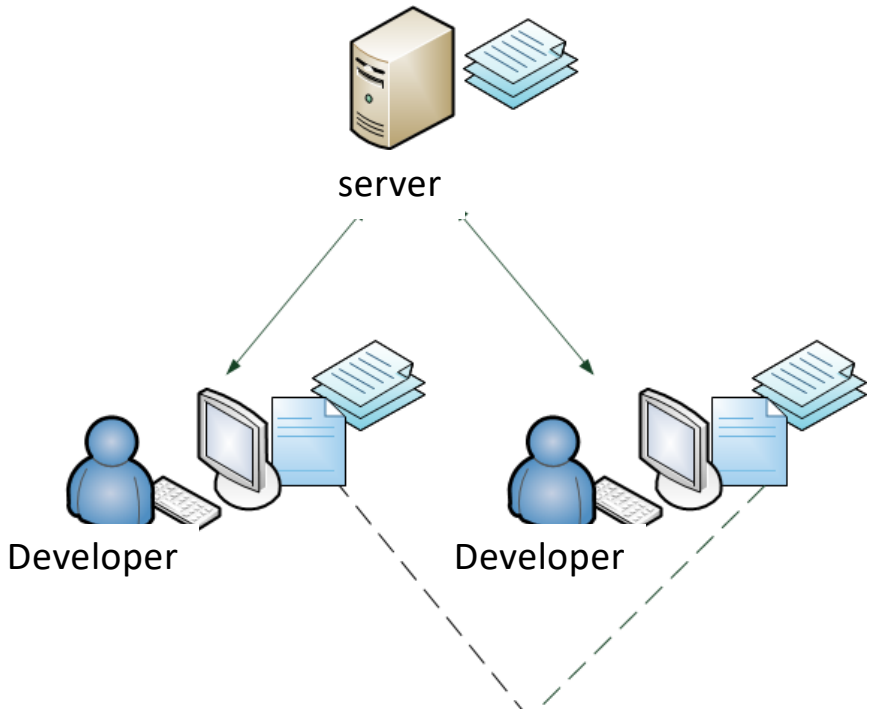
Centralized Version Control Management



Distributed Version Control Management



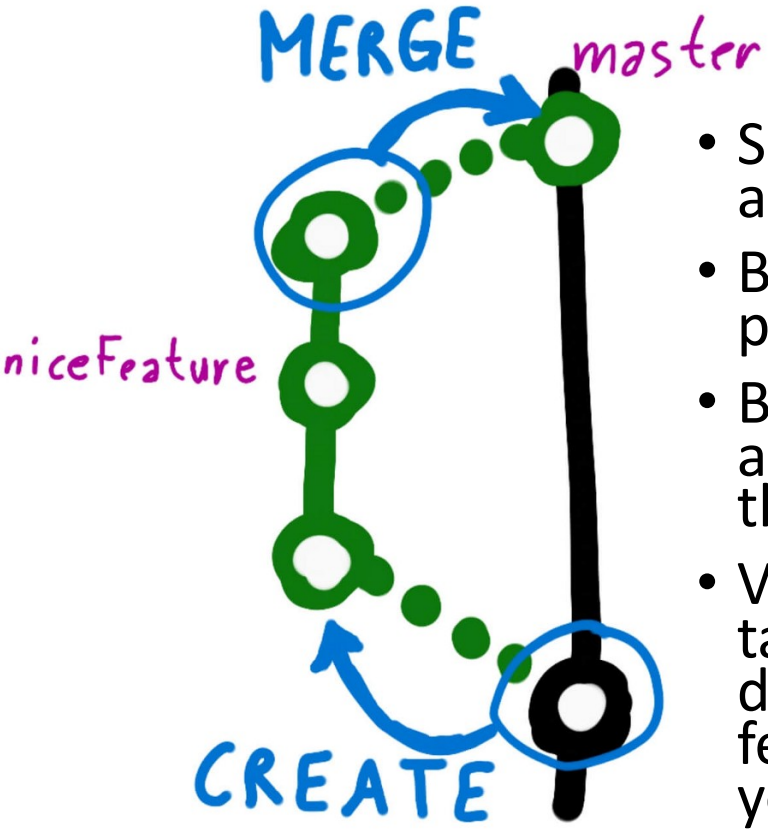
Distributed Version Control Management with Server



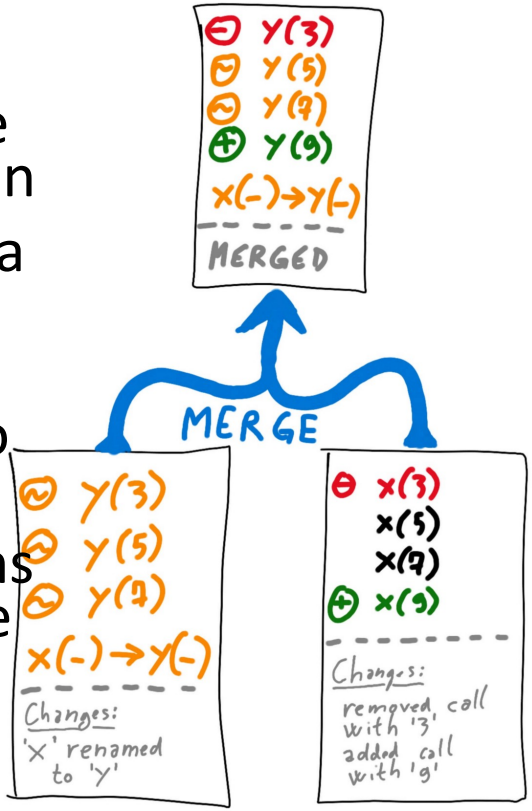
Version management Tools

CVS	Centralized	It is one of the oldest version management software. Although it works and is still used for some projects, it is preferable to use SVN (often presented as its successor) which fixes a number of its flaws, such as its inability to track renamed files for example.	OpenBSD, …
SVN (subversion)	Centralized	Probably the most used tool at the moment. It is fairly simple to use, although like all tools of the same type it requires some time to adapt. It has the advantage that it is well integrated into Windows with the Tortoise SVN program, where many other programs are mainly used on the command line in the console.	Apache, Redmine, Struts, …
Mercurial	Distributed	More recent, it is complete and powerful. It appeared a few days after the beginning of Git's development and is comparable to it in many aspects.	Mozilla, Python, OpenOffice.org
Bazaar	Distributed	Another tool, complete and recent, like Mercurial. It is sponsored by Canonical, the company that publishes Ubuntu. It focuses on ease of use and flexibility.	Ubuntu, MySQL, Inkscape
Git	Distributed	Very powerful and recent, it was created by Linus Torvalds, who is among others the man behind Linux. It is distinguished by its speed and its branch management, which allows new features to be developed in parallel.	Linux Kernel, Debian, VLC, Android, Gnome, Qt, …

Branching, Branches and Merging



- Several versions of the software are simultaneously in production
- Bug-fixes must be delivered on a previous version
- Big features take a lot of time and cannot be added directly to the last, current version
- Validating new software versions takes time and cannot block the development of new software features but cannot be added yet to the version in production
- And so on…



GIT: Practical Introduction

- Git Presentation at Hubert Curien Lab by Rémi Emonet https://github.com/twitwi/Presentation-2016-09-01-git-gitlab-hcurien
- In Unix systems:
  - man gittutorial
  - man gittutorial2
  - man giteveryday
  - man gitworkflows
  - man gitglossary

References

- About Git: https://git-scm.com/about
- References: https://git-scm.com/docs
- Tutorial: https://git-scm.com/docs/gittutorial
- Book: https://git-scm.com/book
- Interactive cheat sheet: http://ndpsoftware.com/git-cheatsheet.html
- Supporting Git Platform:
  - GitHub: https://guides.github.com/
  - GitLab: https://about.gitlab.com/

GIT Further references

- https://guides.github.com/introduction/flow/
- https://guides.github.com/activities/hello-world/
- https://guides.github.com/introduction/git-handbook/
- https://guides.github.com/activities/forking/
- https://guides.github.com/features/issues/
- https://guides.github.com/features/mastering-markdown/
- https://git-scm.com/docs/gittutorial
- https://git-scm.com/docs/user-manual.html

Steps to use Git

- Git Installation
- Git Configuration (.gitconfig)
- Create a new repository

```
[color]
        diff = auto
        status = auto
        branch = auto
[user]
        name = votre_pseudo
        email = moi@email.com
[alias]
        ci = commit
        co = checkout
        st = status
        br = branch
```

```
cd /home/mateo21
mkdir plusoumoins
cd plusoumoins
```

```
git init
```

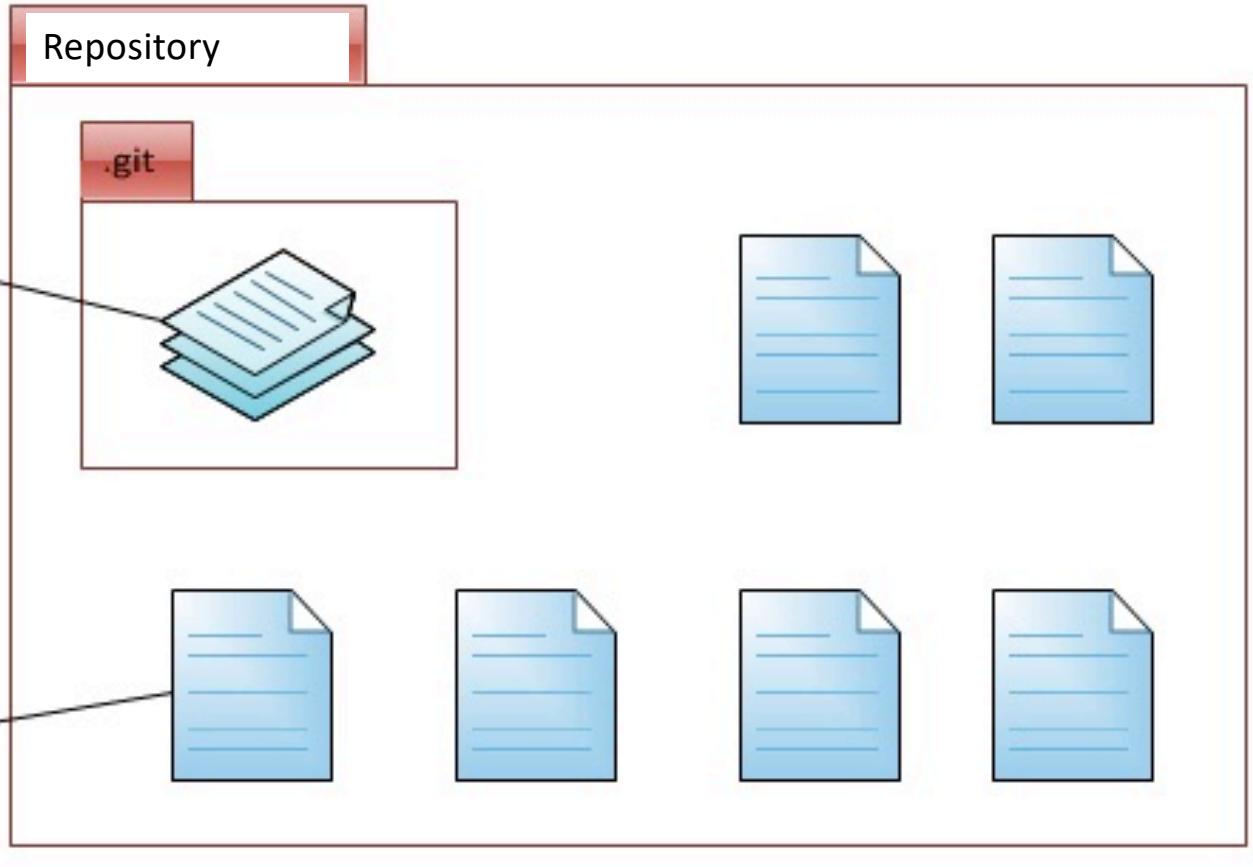
- Clone a repository

```
git clone http://github.com/symfony/symfony.git
```

How does your repository look like?

Hidden folder in which Git stores the history of the files and the configuration of the repository. It contains:
- the Index which acts as a staging area
- and the HEAD which points to the last commit you've made

Last version of the files of the project (called working directory)



Steps to use Git

- Propose changes (add them to the Index) using:
  - git add <filename>

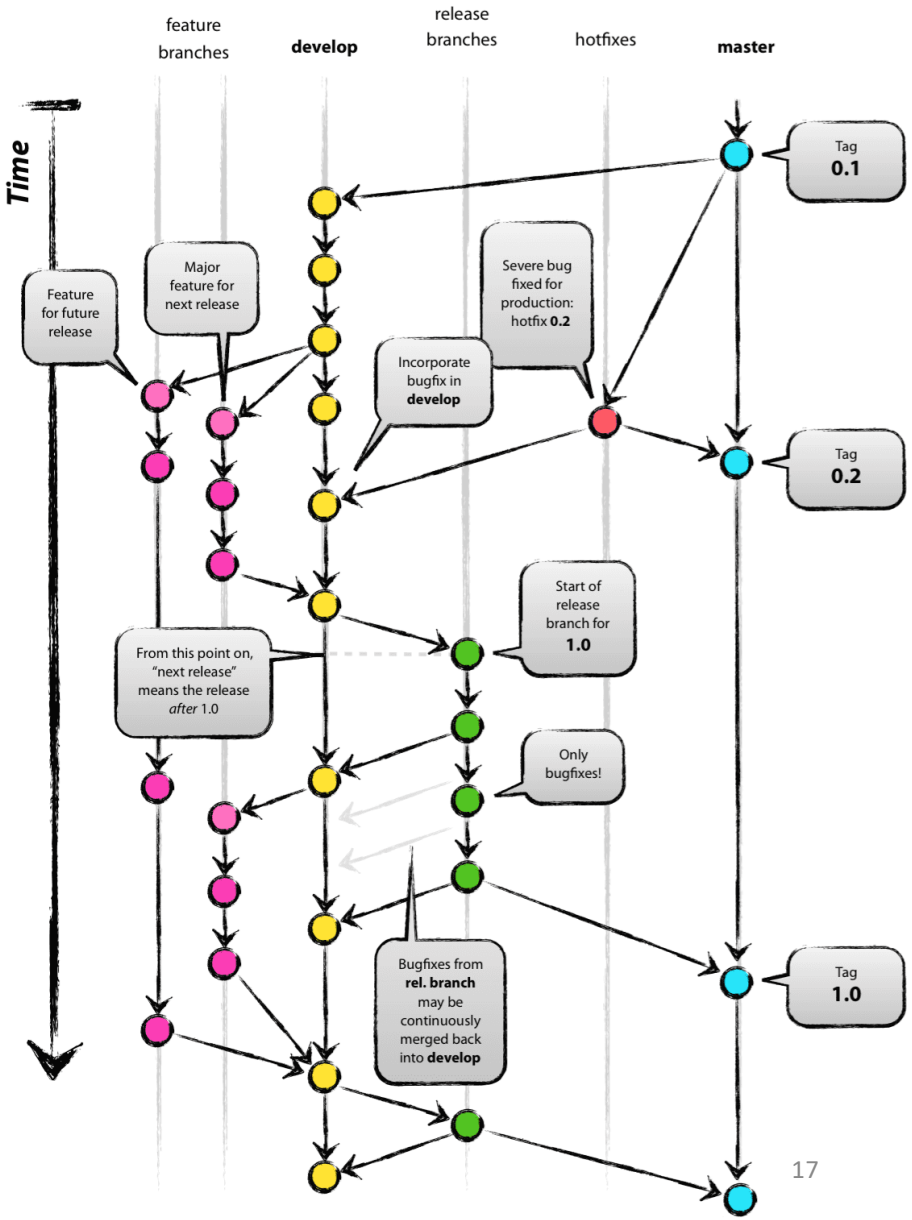
  Or git add *
- Commit these changes to the HEAD (not in the remote repository yet)
  - git commit –a –m ’’commit message’’
- Pushing changes on the remote repository
  - git push origin master master has to be changed by whatever you want

A lot more commands …. Do the tutorials

https://book.git-scm.com/, http://think-like-a-git.net/, http://marklodato.github.io/visual-git-guide/index-en.html

GitFlow Branching Model

- *master* represents what's in production
- Features and fixes are worked on their own branch
- The features and fixes branches are not directly merged into *master* when the work is done; it is merged into a branch called *develop*
- *develop* is thus some buffer between "dev done" and "in production"; release branches are made from *develop* and merged into *master*
- Release branches can thus be tested and validated at length in staging environment



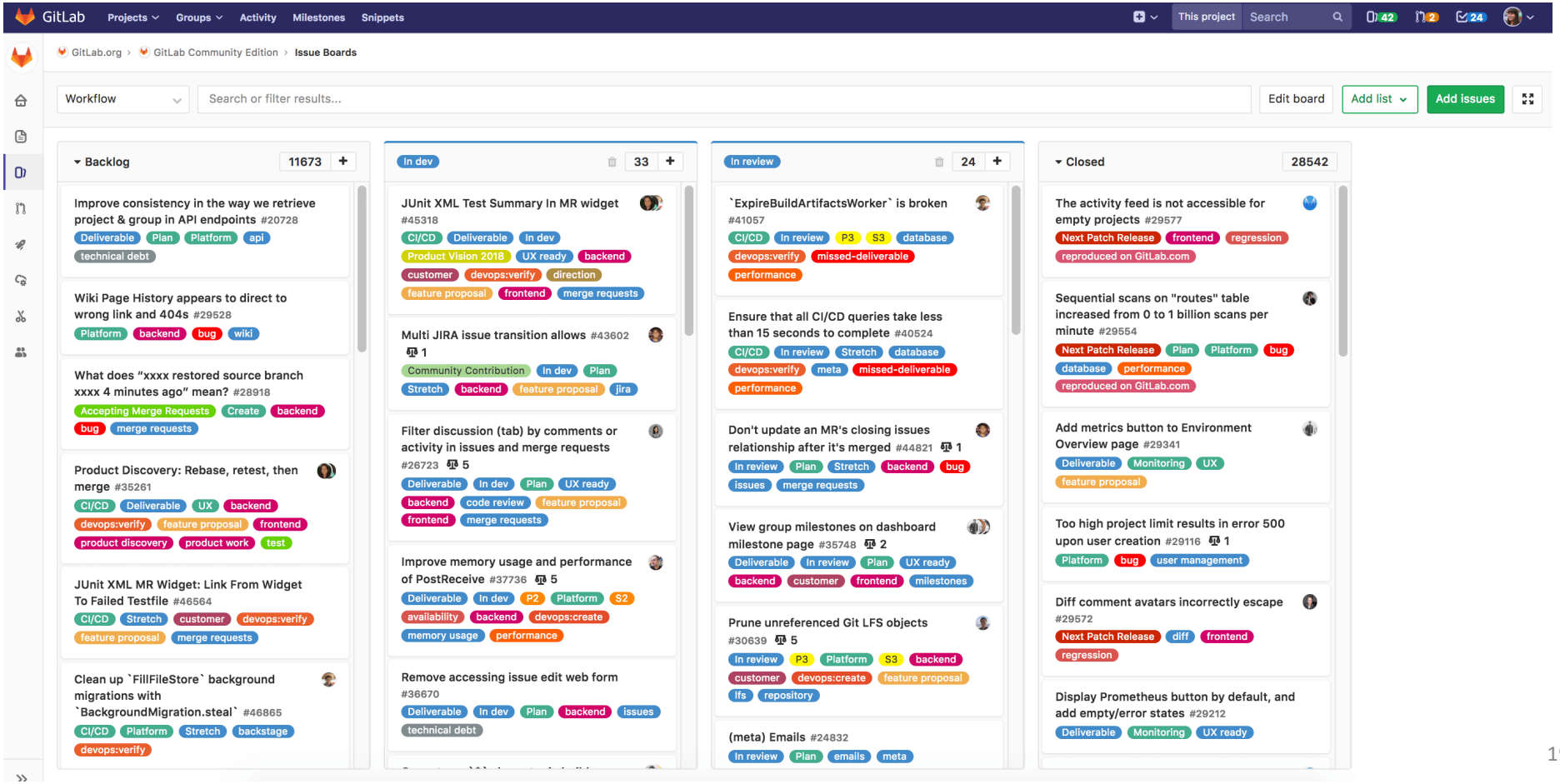
Integration, publication

- Continuous Integration with Travis: https://travis-ci.org/
- Travis CI documentation: https://docs.travis-ci.com/

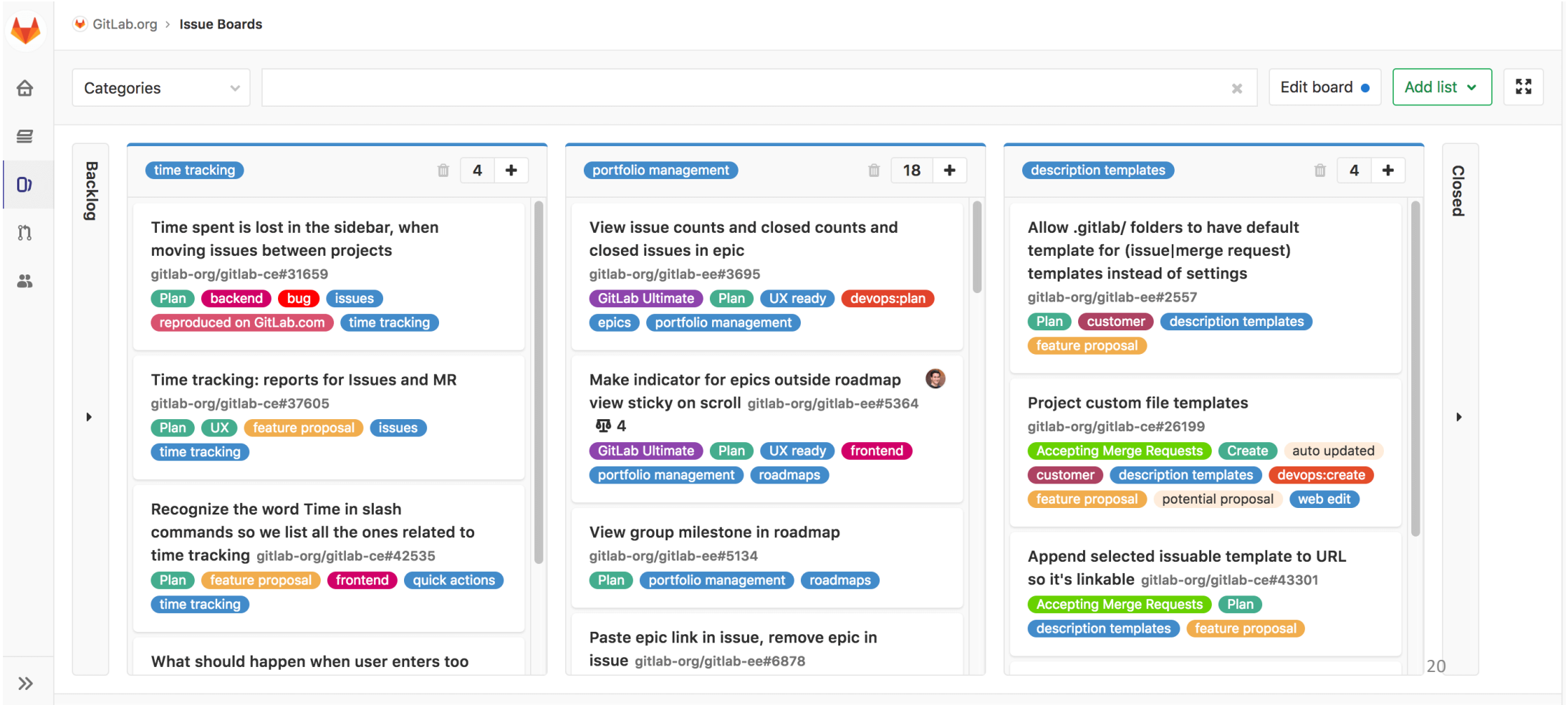
- Heroku is a Cloud Platform you can use for free to publish dynamic apps using git. https://devcenter.heroku.com/

- Docker enables to decouple application and infrastructure concerns. Using Docker, you can ensure your testing and deployment environment is exactly the same. https://docs.docker.com/get-started/

Gitlab & Agile Project Management



Cross-functional Planning



Gitlab, Agile Project Management

- See https://about.gitlab.com/product/issueboard/
- Create your project in Gitlab
- Go into the Issues menu in order to add lists, boards, labels
- Labels are used to structure and categorize the issues
  - E.g. : Epic, User Story, Bug, Ready, Rejected, High, Medium, Low, In Review
- Boards are used to structure and organize the issues in relation to their labels. They correspond to the backlogs. Use the labels to label the list of issues that appear in a Board
  - E.g. Development backlog with To Do, Doing, In Review …
    Product Backlog with Low, Medium, High, Rejected, …
- Create Sprints by using Milestones and by assigning issues to milestones.

Some references

- Version Control with Git https://swcarpentry.github.io/git-novice/
- Gérer vos codes avec Git https://openclassrooms.com/fr/courses/1233741-gerez-vos-codes-source-avec-git (in French)
- Git Presentation at Hubert Curien Lab by Rémi Emonet https://github.com/twitwi/Presentation-2016-09-01-git-gitlab-hcurien
- GIT, GitFlow and Continuous Integration for Dummies https://jp-lambert.me/git-gitflow-and-continuous-integration-for-dummies-5e4300148fbf
- GIT et GITLAB au quotidien https://bayol.pages.math.unistra.fr/tp-gitlab/git.html#workflow-et-gitlab