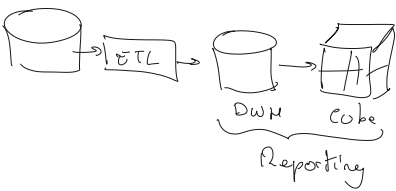
Bonjour tout le Monde

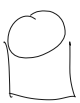
1) Présentation générale "moderne"
2) Conteneurisation → Docker, OpenShift.
3) Automatisation des dev (DevOps, CI/CD)
4) (API Rest)
5) µ Svc (Openshift Service Mesh)

Data...

BI

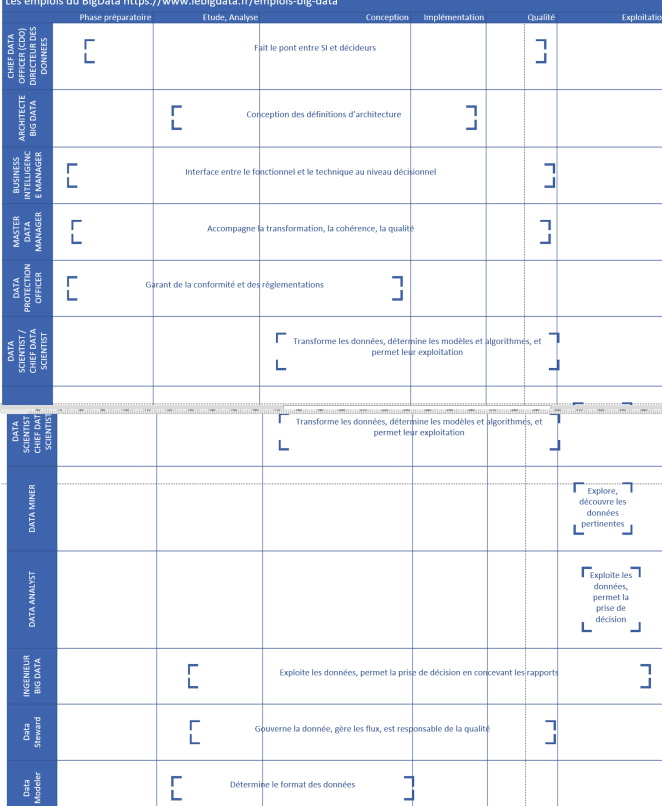


Big Data



↑ scale Up → Relationnel

→ Big Data (NoSql) :


Les emplois du BigData https://www.lebigdata.fr/emplois-big-data
Phase préparatoire
Etude, Analyse
Conception
Implémentation
Qualité
Exploitation
CHIEF DATA OFFICER (CDO) DIRECTEUR DES DONNEES
Fait le pont entre SI et décideurs
ARCHITECTE BIG DATA
Conception des définitions d'architecture
BUSINESS INTELLIGENC E MANAGER
Interface entre le fonctionnel et le technique au niveau décisionnel
MASTER DATA MANAGER
Accompagne la transformation, la cohérence, la qualité
DATA PROTECTION OFFICER
Garant de la conformité et des règlementations
DATA SCIENTIST / CHIEF DATA SCIENTIST
Transforme les données, détermine les modèles et algorithmes, et permet leur exploitation
DATA SCIENTIST/ CHIEF DATA SCIENTIST
Transforme les données, détermine les modèles et algorithmes, et permet leur exploitation
DATA MINER
Explore, découvre les données pertinentes
DATA ANALYST
Exploite les données, permet la prise de décision
INGENIEUR BIG DATA
Exploite les données, permet la prise de décision en concevant les rapports
Data Steward
Gouverne la donnée, gère les flux, est responsable de la qualité
Data Modeler
Détermine le format des données

Clusters NoSql > 10.000 machines.

1000 machines (Data)

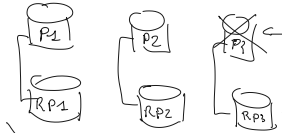
↓

Reporting (1 machine)

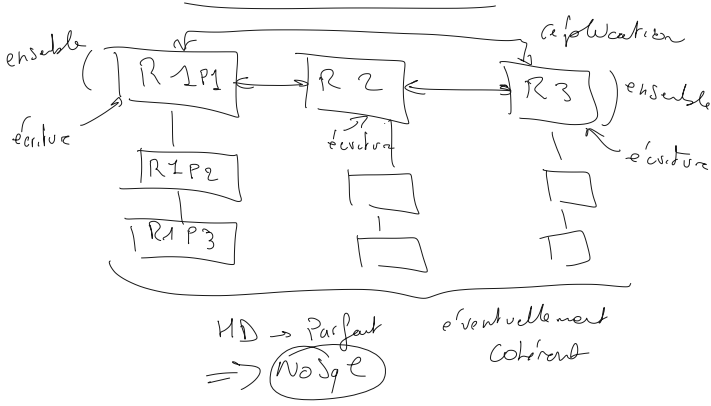
P1, P2, P3

ensemble

↓



"HB" → Cohérent



HD → Parfait

⇒ NoSql

éventuellement Cohérent

- Tradition : VM → Lourd, long
  + adapté : Conteneurisation → + Rapide, Légère
    - "conteneuriser" applis, limites, contraintes

VM :
Bases de données
Infrastructures (legacy)

Conteneurs :
- front Web } "Simple" à conteneuriser
- API

IaaS (VM, XaaS)
→ + Complexe à conteneuriser + critique
→ BDD → Traditionnel
→ front → Open Shift

IaaS ↑ connecter

PaaS (Open Shift)
Conteneurs

+ léger, + rapide à déployer
+ facilement conteneurisable

Futurs Projets intégralement dans le PaaS
- fronts et APIs
- BDD NoSql

Serveurs Web :

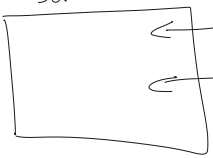
Traditionnel / Legacy :

VM
→ IIS (windows)
→ apache / nginx } statique
tomcat / wildfly / websphere } JEE dynamique

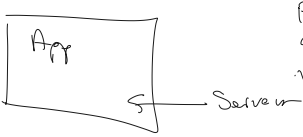
S. Web : .jar, .aspx etc... ← Appli (code, archives)

→ Solution DevOps → Serveurs embarqués

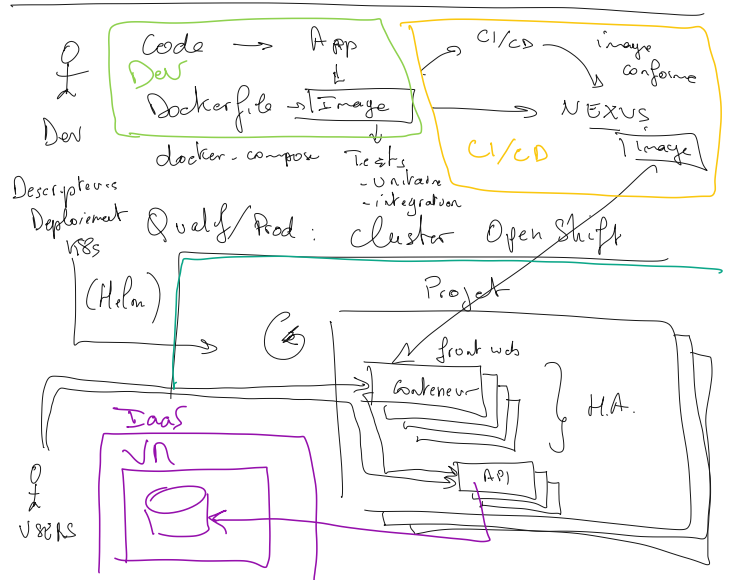
avant : Serveur



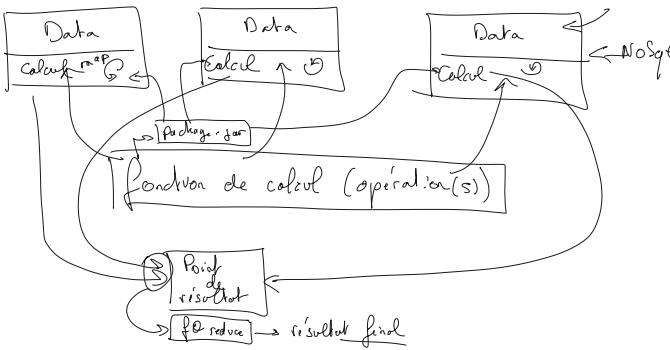
maintenant : ex : Spring boot (java) ..
php symphony ...
asp.net mvc 6 ..
python flask ...



App web = App Console



Big Compute



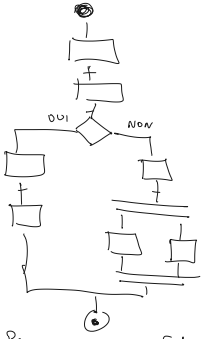
Code → map → jar → distribuer → exec → résultat
reduce → jar →

↓

Scala

Spark.

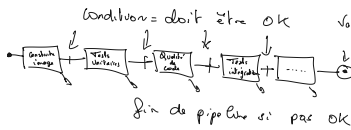
Workflow



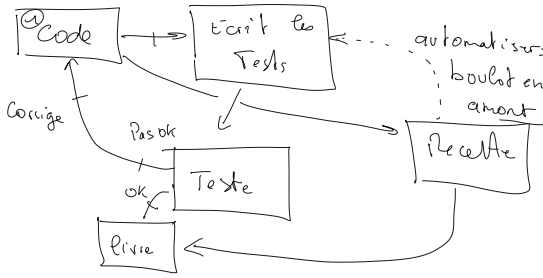
→ Process
→ Administration

→ Fabrication
etc...

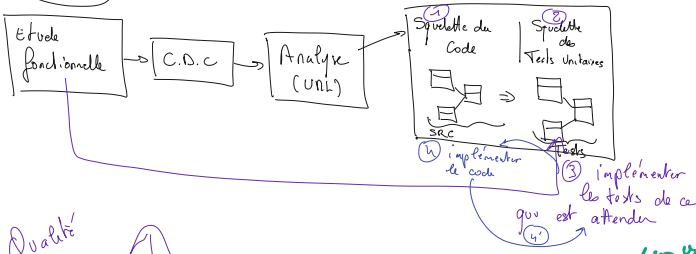
Pipeline



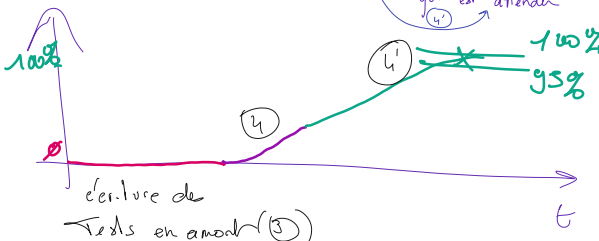
Agile



TDD — Test Driven Development



Qualité



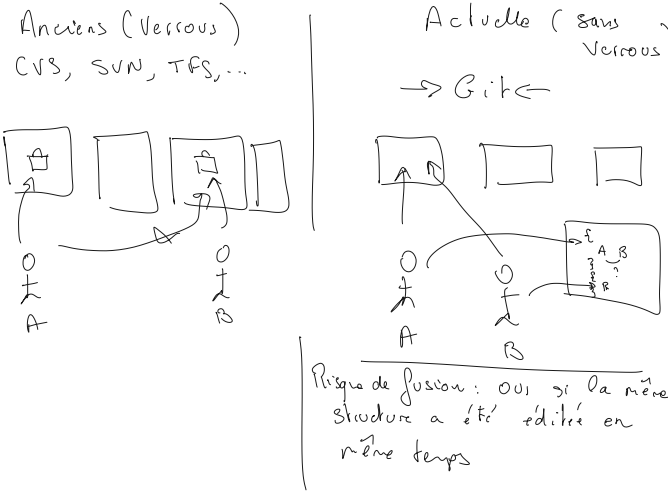
DevOps

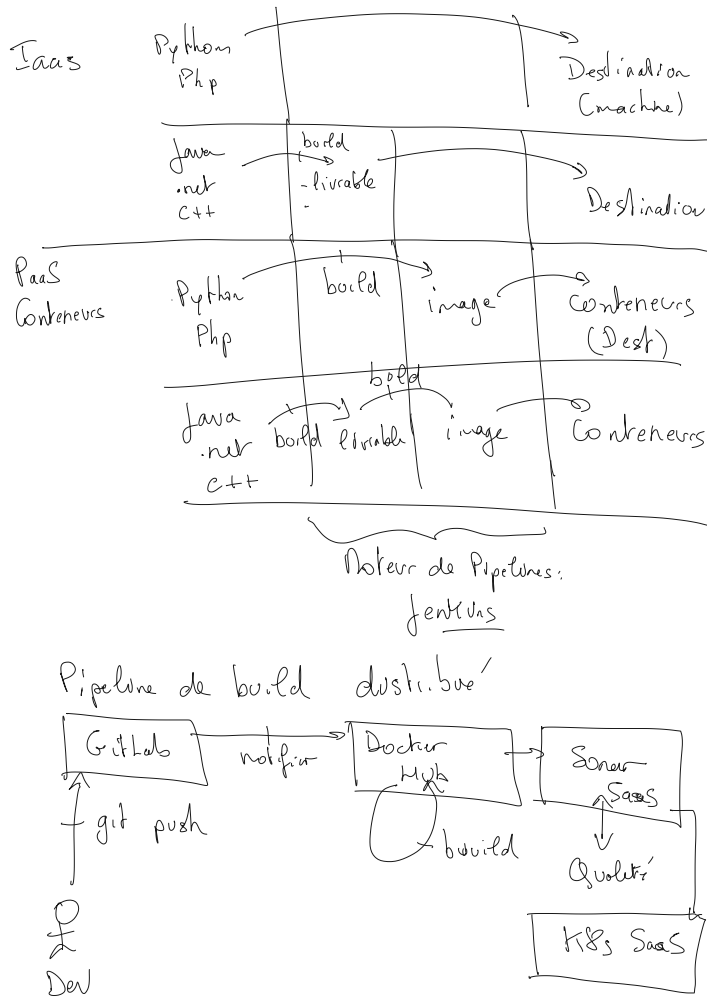
- Tout nouveau Projet de code inclut une partie sources et Tests

Production d'une image Docker → SRC
→ Tests

• Accessibility testing
• Acceptance testing
• Black box testing : tester un système inconnu.
• End to end testing : tester un workflow fonctionel
• Functional testing : s'assurer qu'il fait exactement ce qui était prévu.
• Interactive testing : eq tests manuels
• Interface tests
• Integration testing : test en environnement équivalent à la prod dans son ensemble.
• Load testing
• Non functional testing : catégorie de tests d'interface (conformité), accessibilité, performance, ...
• Performance testing : recherche des meilleurs métriques, benchmark.
• Regression testing
• Sanity testing : contrôle que les bugs ciblés sont bien corrigés
• Security testing : contrôle face aux menaces
• Single user performance testing
• Smoke testing : valider les fonctions critiques d'un système, envers la stabilité.
• Stress testing : test selon des conditions exceptionnelles de charge, au dela des specs.
• Unit testing

White-box testing : teste les entrées sorties d'un logiciel bien connu, concernant le design, l'utilisabilité, la sécurité, la stabilité.




Iaas
Python
Php
Destination
(machine)
Java
.net
C++
build
-livrable
-
Destination
PaaS
Conteneurs
Python
Php
build
image
Conteneurs
(Dest)
build
Java
.net
C++
build
livrable
image
Conteneurs
Moteur de Pipelines:
Jenkins
Pipeline de build distribué
GitLab
notifier
Docker
Myk
Sonar
Saas
git push
build
Qualité
K8s Saas
Dev

Versions

Versioning Sémantique

Majeur . **Mineur** . **Revision** . [build number]

- Majeur
  - Globalement, l'App est assez différente (conceptuel)
  - Pas de respect de la compatibilité ascendante avec la version précédente

ex

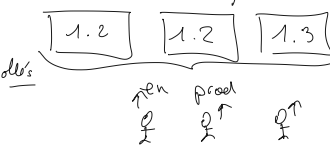
  - Peut éventuellement être une "Application distincte en même temps"

ex Office 2016 et Office 2018 installés

- Mineur
  - Reset sur nouveau majeur
  - nouvelles fonctions
  - correctifs
  - optimisations

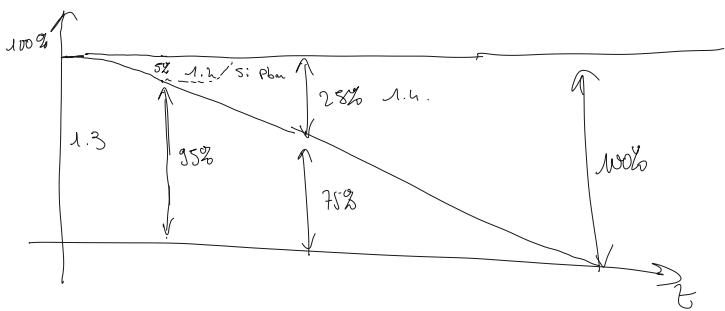
ET respect de la compatibilité ascendante

  - peut remplacer un mineur précédent
  - peut fonctionner éventuellement en même temps que le mineur précédent

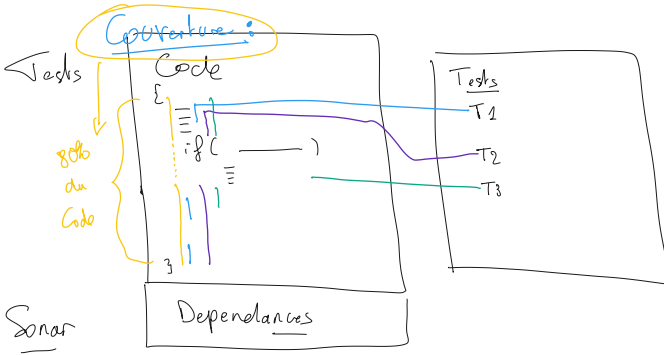


- Revision
  - Reset sur nouveau mineur
  - inc si correction de bugs

- [build number]
  - \+ sur recompilation



Apprendre Git : https://learngitbranching.js.org/?locale=fr_FR



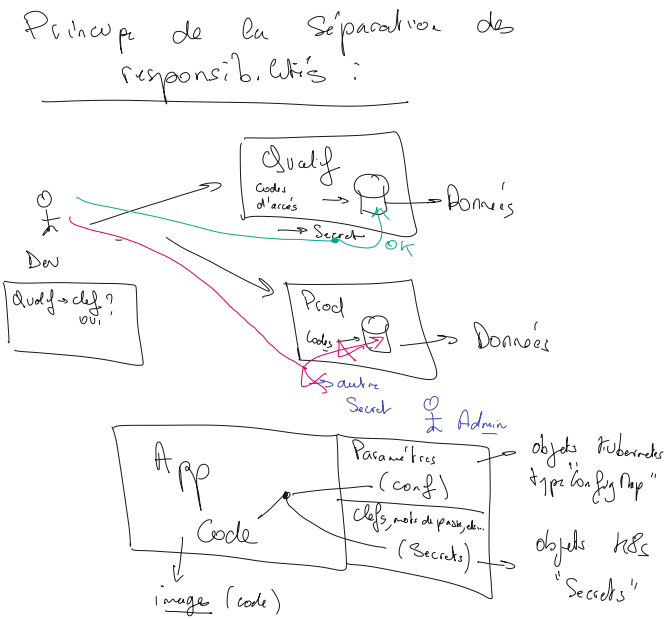
Sonar

- Evalue
  - bonnes pratiques de codage
  - éventuels bugs
  - Risques de sécurité

https://rules.sonarsource.com/

- Contrôler les dépendances
  - Failles (CVE)

http://cve.mitre.org/
https://nvd.nist.gov



Tutoriel OpenShift :
https://developers.redhat.com/courses/openshift/getting-started

A faire dans un navigateur basé sur chromium (Edge, Chrome)

Film Youtube présentant OpenShift Service Mesh :
https://youtu.be/GHYdOpGUJ3w