

MICROSERVICES & DEVOPS

B. Adanlessossi

AGENDA



MICROSERVICES RAPPEL



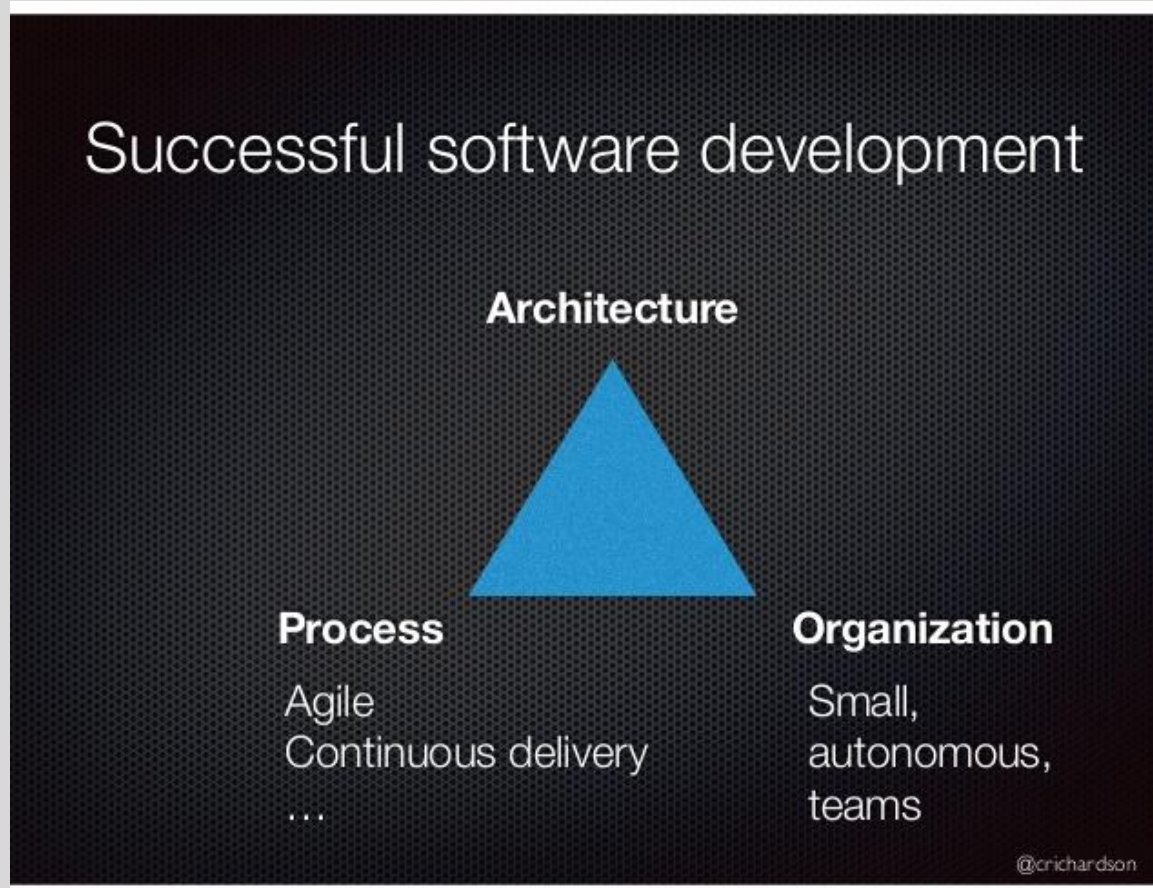
MICROSERVICES ET
DEVOPS



DEMO

Introduction

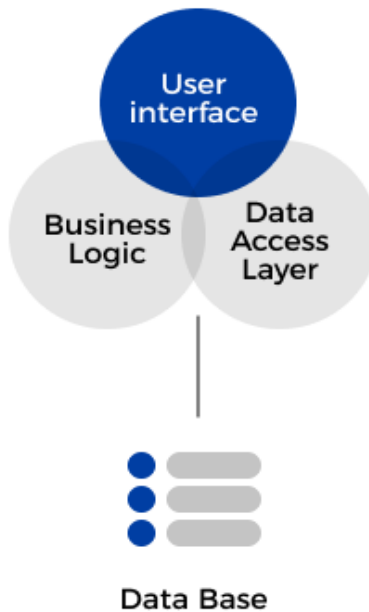
- Comment développer avec succès un logiciel? Le triangle du succès



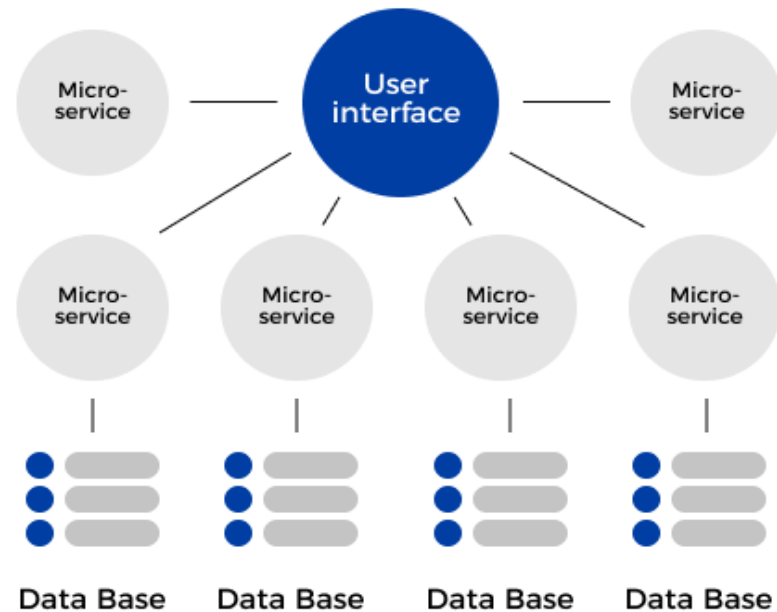
Monolith ou Microservices

- Nous avons le choix entre une architecture Monolithique ou une architecture Microservices.

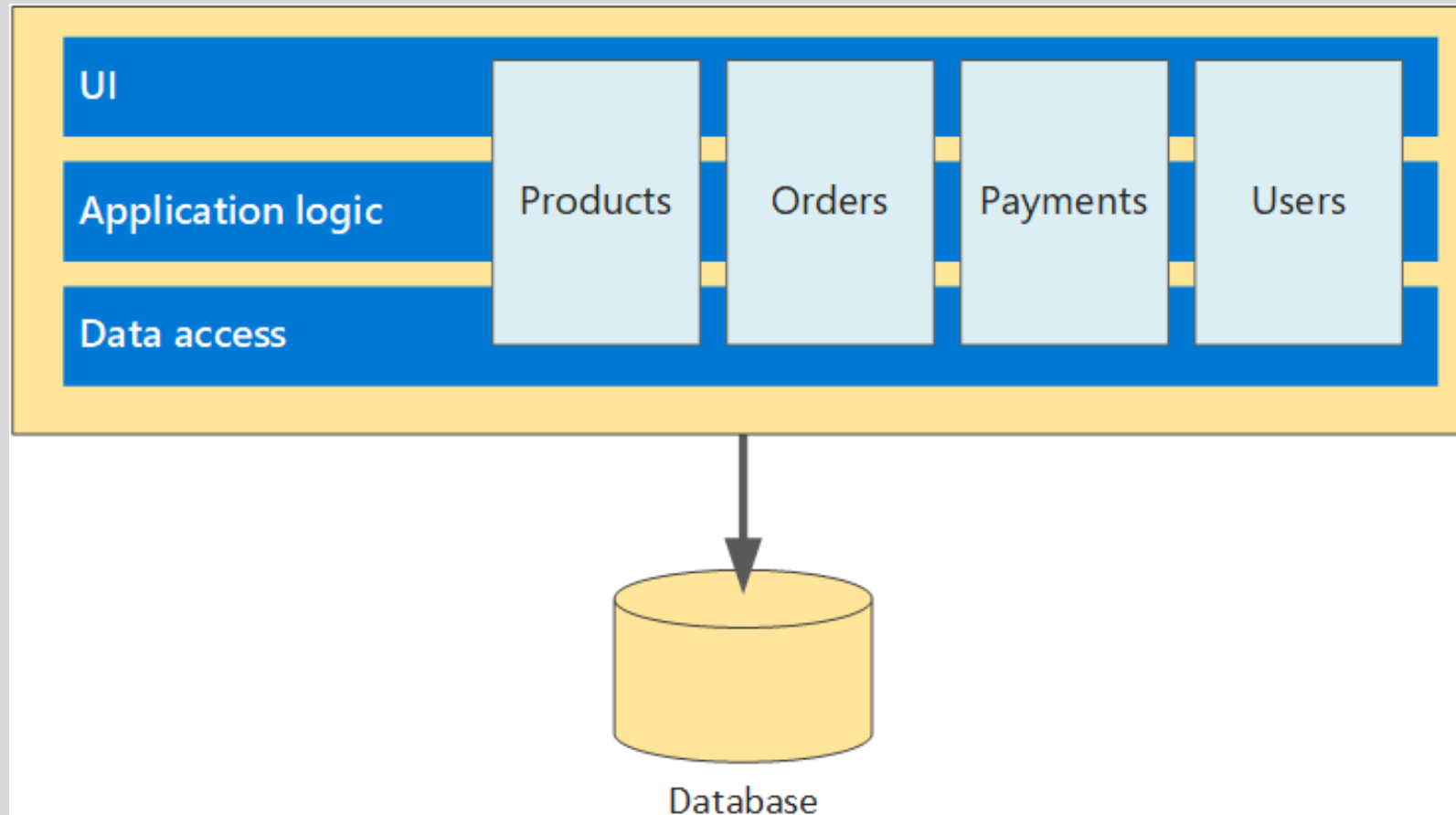
MONOLITHIC ARCHITECTURE



MICROSERVICE ARCHITECTURE



L'architecture Monolithique

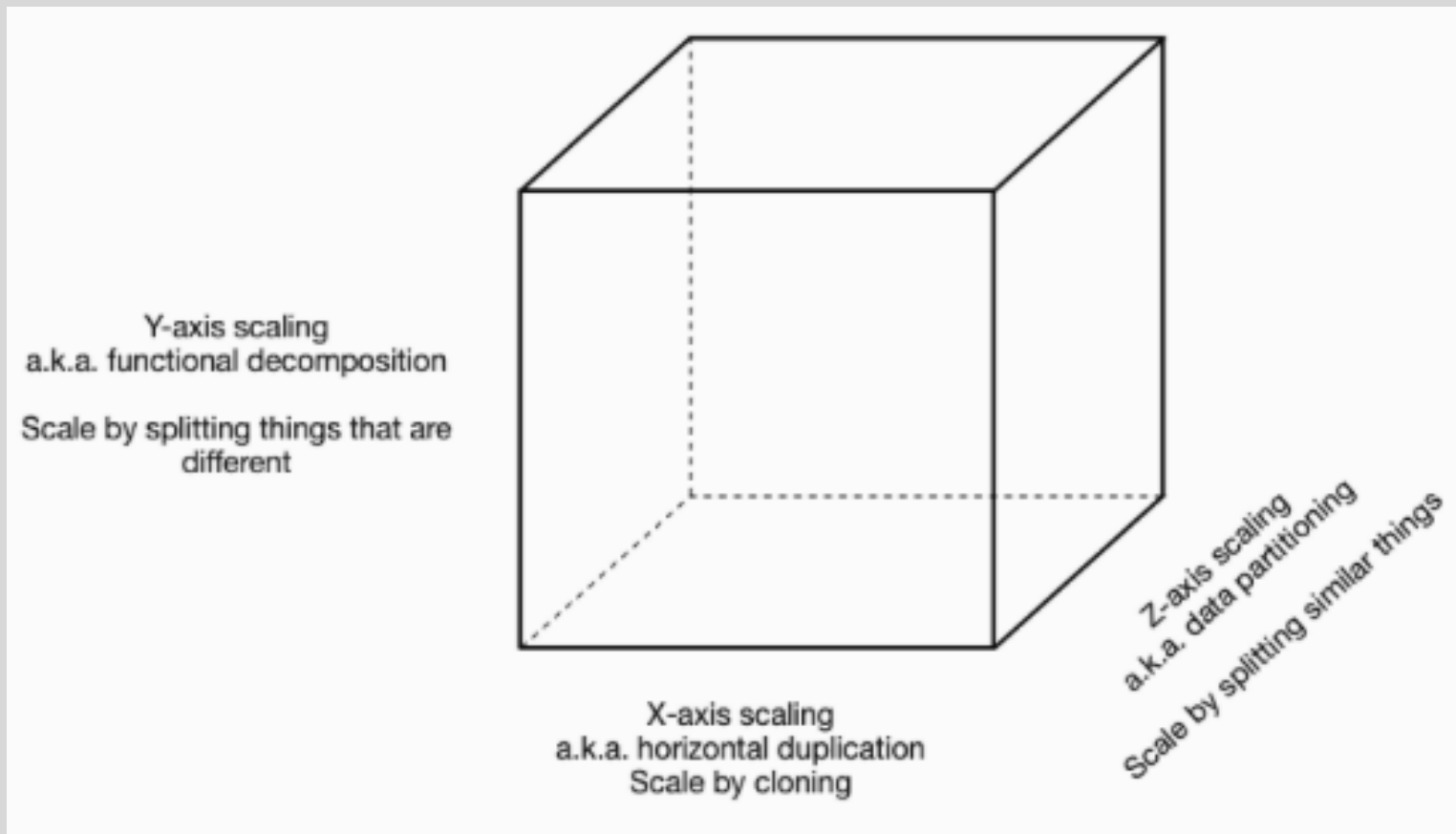


Problèmes avec le Monolith

- Au fur et à mesure du temps, l'application devient trop grosse
- Chaque changement nécessite le déploiement de toute l'application
- Les erreurs et les bugs sont difficiles à identifier.
- L'application n'est plus alignée à l'organization.
- La méthodologie agile n'est pas adoptable
- Il n'y a plus d'autonomie entre les équipes
- Bref,... ce qu'on appelle "l'enfer du monolith", monolithic hell!

La décomposition fonctionnelle

- La solution est d'appliquer la décomposition fonctionnelle.

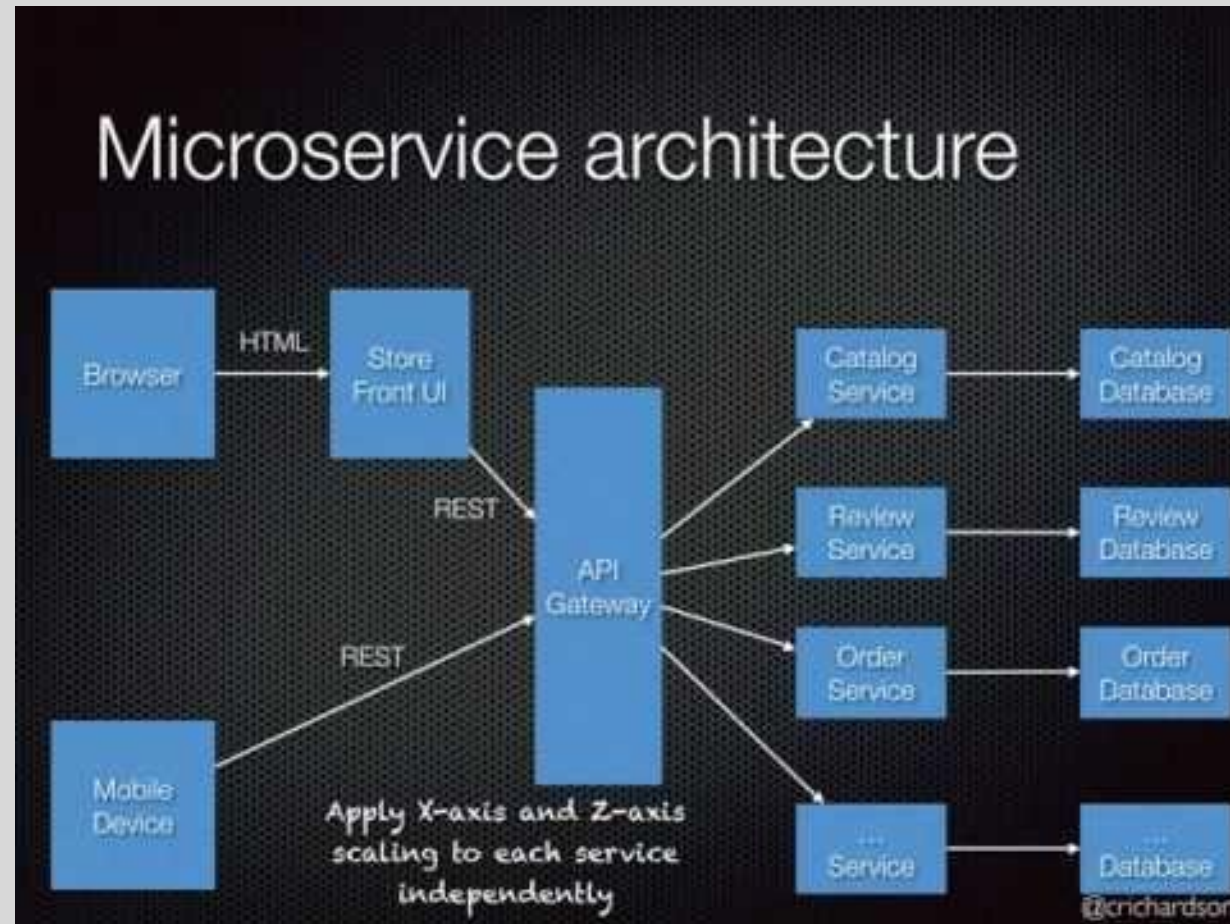


La décomposition fonctionnelle (2)

- L'axe **y** permet de décomposer une application par fonction(en séparant les fonctions différentes), c'est la décomposition verticale. C'est la partie la plus importante où vous décomposez votre application en petites fonctions ou services (microservices).
- L'axe **x** permet de décomposer une application par duplication (clonage), c'est ce qu'on appelle la décomposition horizontale où vous exécutez plusieurs copies de l'application derrière un Load Balancer.
- L'axe **z** permet de décomposer l'application par partitionage (en regroupant les choses similaires). Dans cette situation, vous exécutez plusieurs copies de l'application, mais au lieu du Load Balancer, vous avez un Router qui inspecte les requêtes et utilise certains attributs pour router cette requête vers un serveur particulier

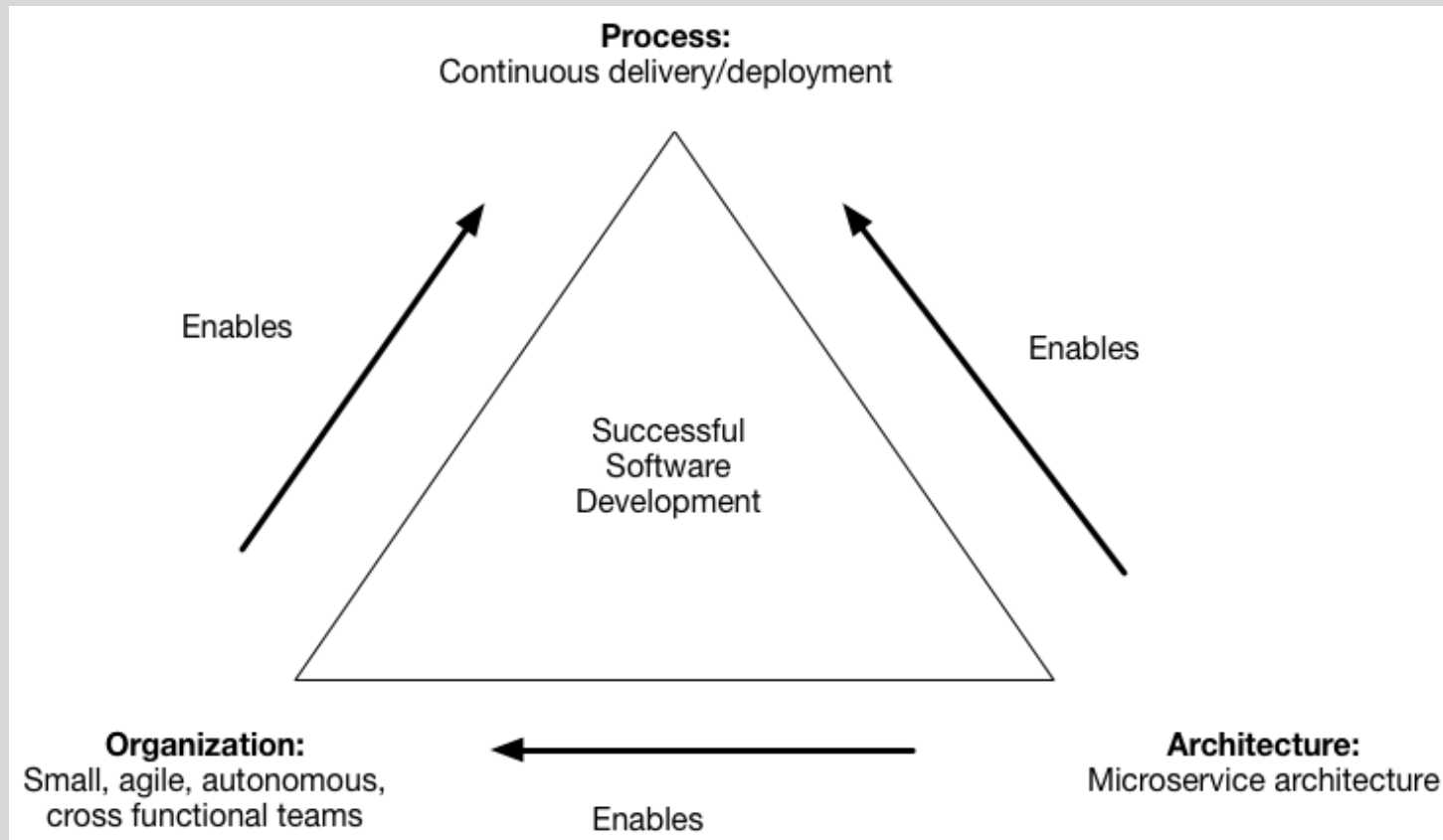
Microservices re-définition

- Construire une architecture Microservices est l'application de la décomposition fonctionnelle sur ces 3 axes.



Architecture Microservice

- Cette architecture permet de couvrir les processus et l'organization, tout en ouvrant la voie à Continuous Integration et Continuous Delivery / Deployment (**CI/CD**)



Désavantages des Microservices

- La complexité devient plus grande quand on développe un système distribué
 - Communications entre processus (inter-process communication)
 - Gestion des erreurs quand un serveur est indisponible ou trop lent
- Complexité d'implémenter des transactions métiers entre plusieurs bases de données (sans utiliser le 2-phase commit)
- Complexité de tester un système distribué
- Complexité de déployer et d'opérer un système distribué
- Gestion du développement et du déploiement des fonctionnalités couvrant plusieurs services
 - ***Heureusement, il existe une solution à la plupart de ces problèmes!***

De quoi faut-il tenir compte?

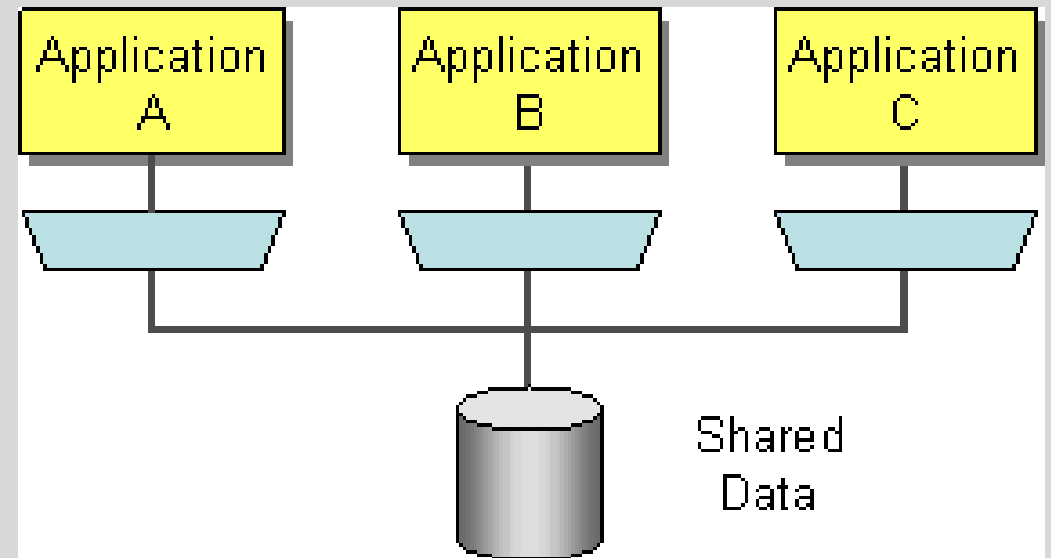
- Comment deployer les services?
- Comment doivent communiquer les services?
- Comment les clients de cette application commniquent avec les services?
- Comment decomposer le système en services?
- Quelle approche adopter avec les problems de gestion de données distribuées?
- Etc...

Base de données Microservices

- Quel pattern adopter pour la gestion des bases de données dans un système de services distribués?
- Soit une base de données partagée par tous les services, ou bien une base de données par service.

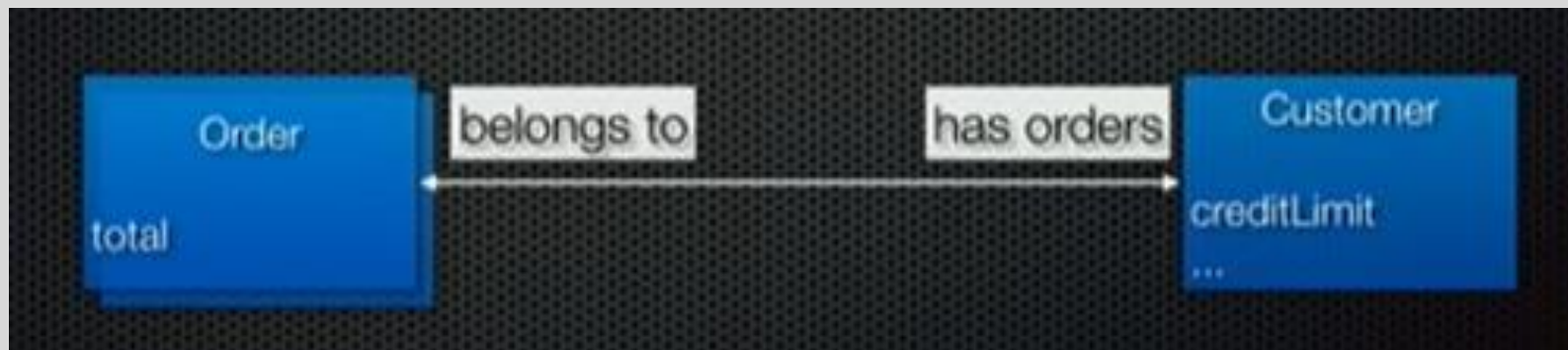
Base de données partagée

- Chaque service avec ses propres tables dans la base de données
- Chaque service peut accéder les données des tables d'un autre service
- Couplage fort entre les tables des services
- Tout changement d'une table nécessite de la coordination avec d'autres services
- Cela ralentit toute l'organisation et l'équipe



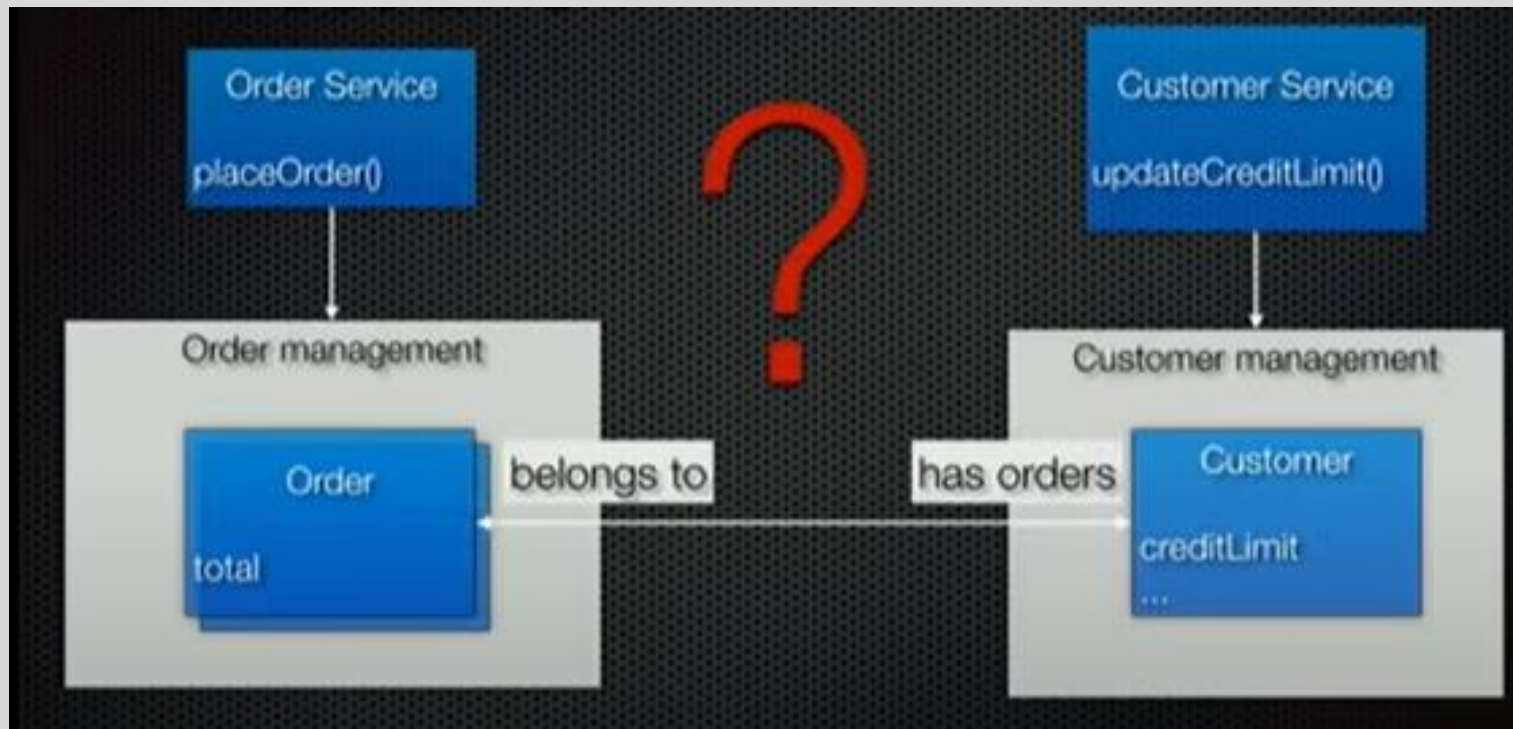
Base de données per service

- L'approche la plus recommandée est une base de données par service
- Ceci permet d'avoir un couplage faible entre les services
- Mais les choses deviennent plus compliquées:
 - Il faut gérer plusieurs bases de données (polyglot)
 - 2PC n'est plus une option viable
 - Comment maintenir la consistance des données entre plusieurs bases de données?



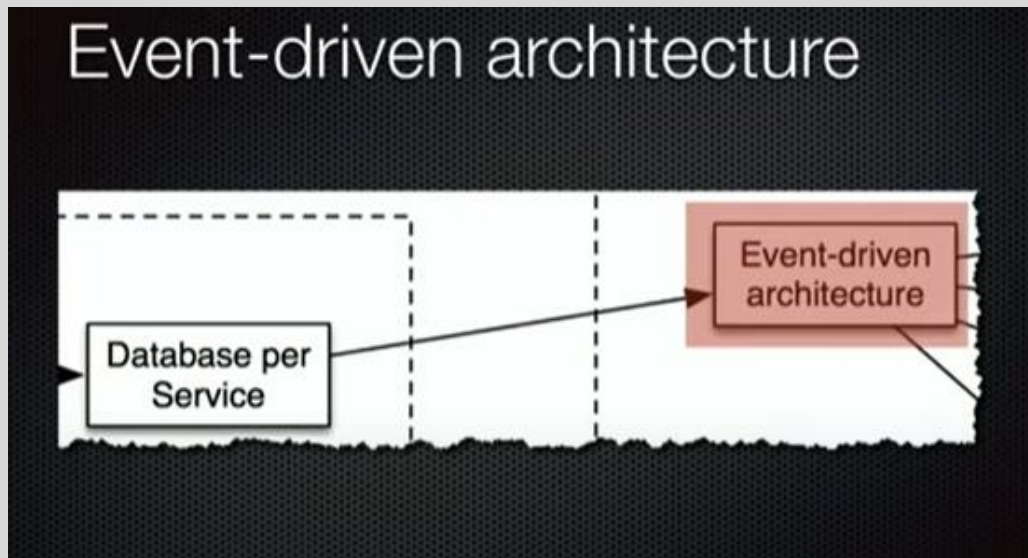
Consistence des données de la base

- Comment maintenir la consistance des données?



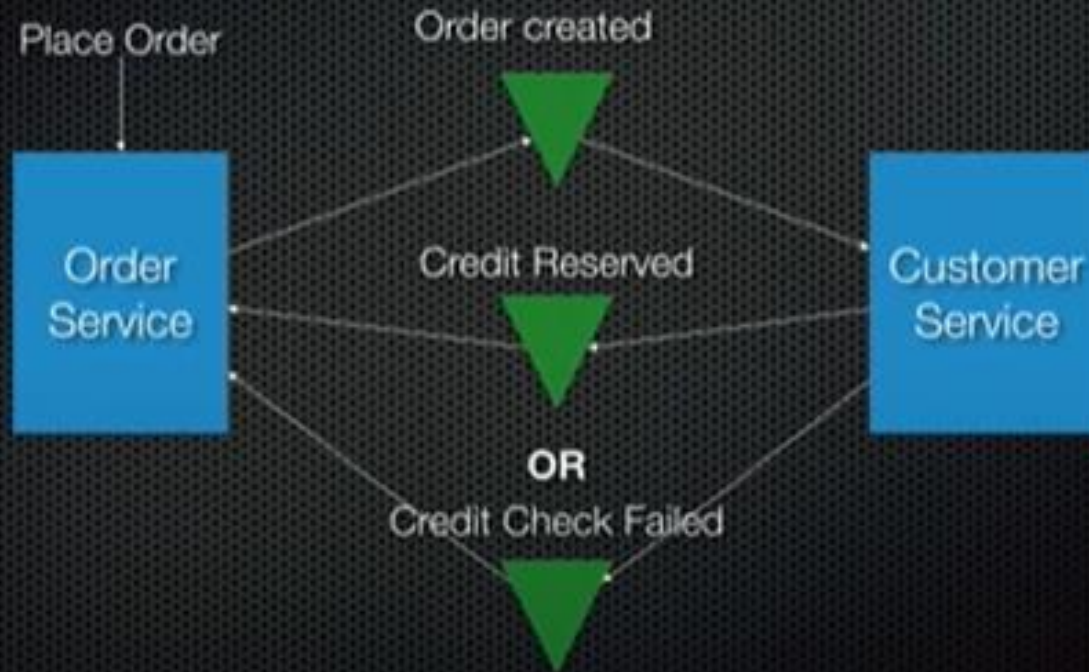
Consistence éventuelle des données

- La solution quand on opte pour une base de données par service est d'utiliser une architecture dite Event-driven
- L'idée est que chaque changement d'état de la base de donnée émet un événement auquel réagissent les autres services en émettant d'autres événements.
- Chaque transaction émet un événement qui déclenche un autre événement, et ainsi de suite, on atteint ce qu'on appelle, la consistance éventuelle! Eventually Consistency!



Example

Use event-driven, eventually consistent order processing



Event sourcing

- L'idée est d'enregistrer les sequences de tous les événements envoyés par les service dans une table des événements.
- Ceci permet également de re-construire l'histoire complète d'un service à un moment t , en rejouant la table des événements.
- Ceci résout le problème de consistance des données dans une architecture microservices
- Permet également de publier les événements dans des applications analytiques
- Ouvre la voie à l'audit et au monitoring.

Use event-sourcing

Event table

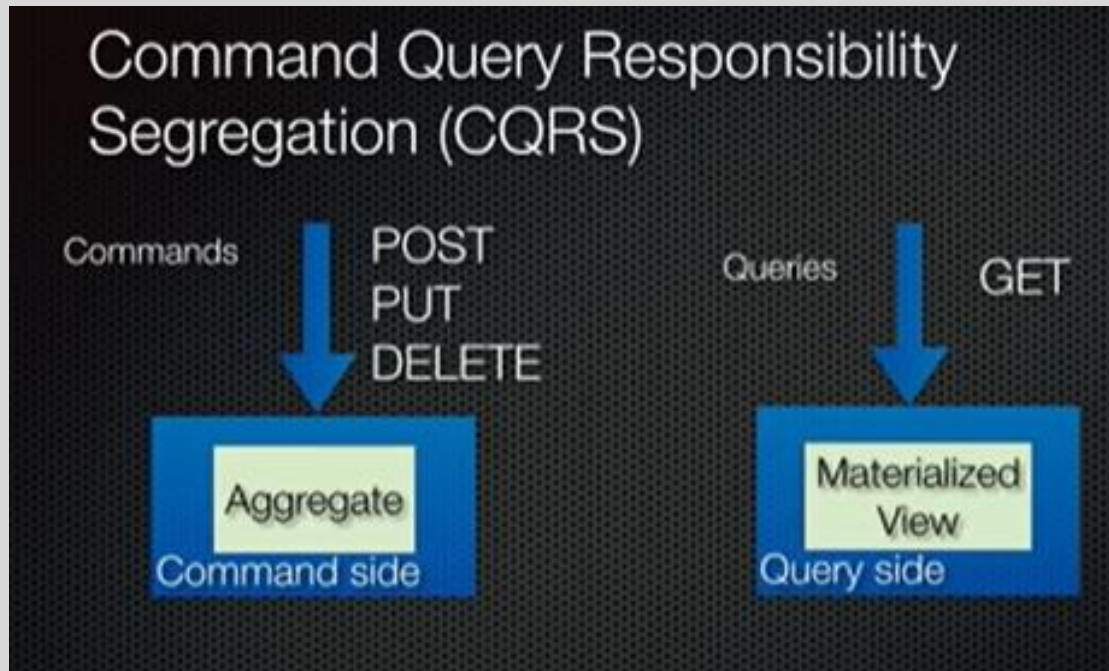
| Aggregate id | Aggregate type | Event id | Event type | Event data |
|--------------|----------------|----------|---------------|------------|
| 101 | Order | 901 | OrderCreated | ... |
| 101 | Order | 902 | OrderApproved | ... |
| 101 | Order | 903 | OrderShipped | ... |

Désavantages de Event sourcing

- Il faut réécrire toute l'application
- Peut entraîner à la confusion à cause de cette façon non familière de programmer
- Les événements enregistrent également toutes les mauvaises décisions prises lors du développement du service
- Gestion des duplicats (détection des duplicats et handlers idempotents)
- Faire des requêtes sur la table des événements peut être très difficile.

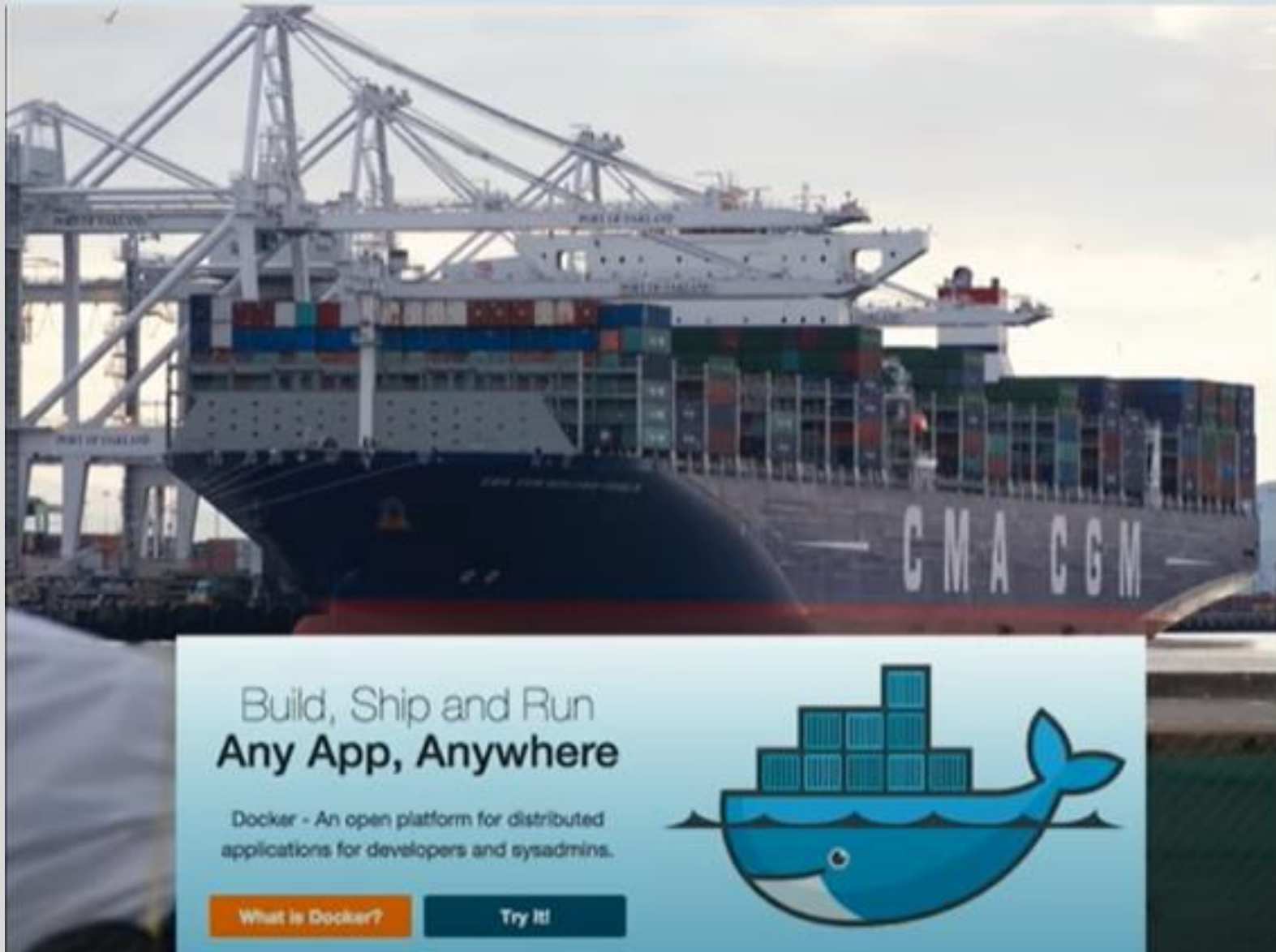
Command Query Responsibility Segregation (CQRS)

- L'idée est de séparer tout ce qui est changement de la base de données (creation, modification, suppression) de tout ce qui est lecture (sélectionner tout, par id, etc.)
- C'est ce qu'on appelle Command Query Responsibility Segregation.



Déploiement des services


- Comment déployer 10, 100 ou même 1000 microservices?
- Ce que nous voulons, c'est:
 - Déployer des services écrits dans des langages de programmation et frameworks différents
 - Chaque service est constitué de multiples instances
 - Le développement et le déploiement doivent être rapides
 - Les services doivent être déployés et distribués indépendamment
 - Les instances des services doivent être isolés
 - Le déploiement doit être fiable et d'un coût minimum
- Nous pouvons utiliser une seule machine pour déployer tous les services, non recommandé
- La solution est d'utiliser une machine par service, que ce soit une VM ou un conteneur docker



Build, Ship and Run
Any App, Anywhere

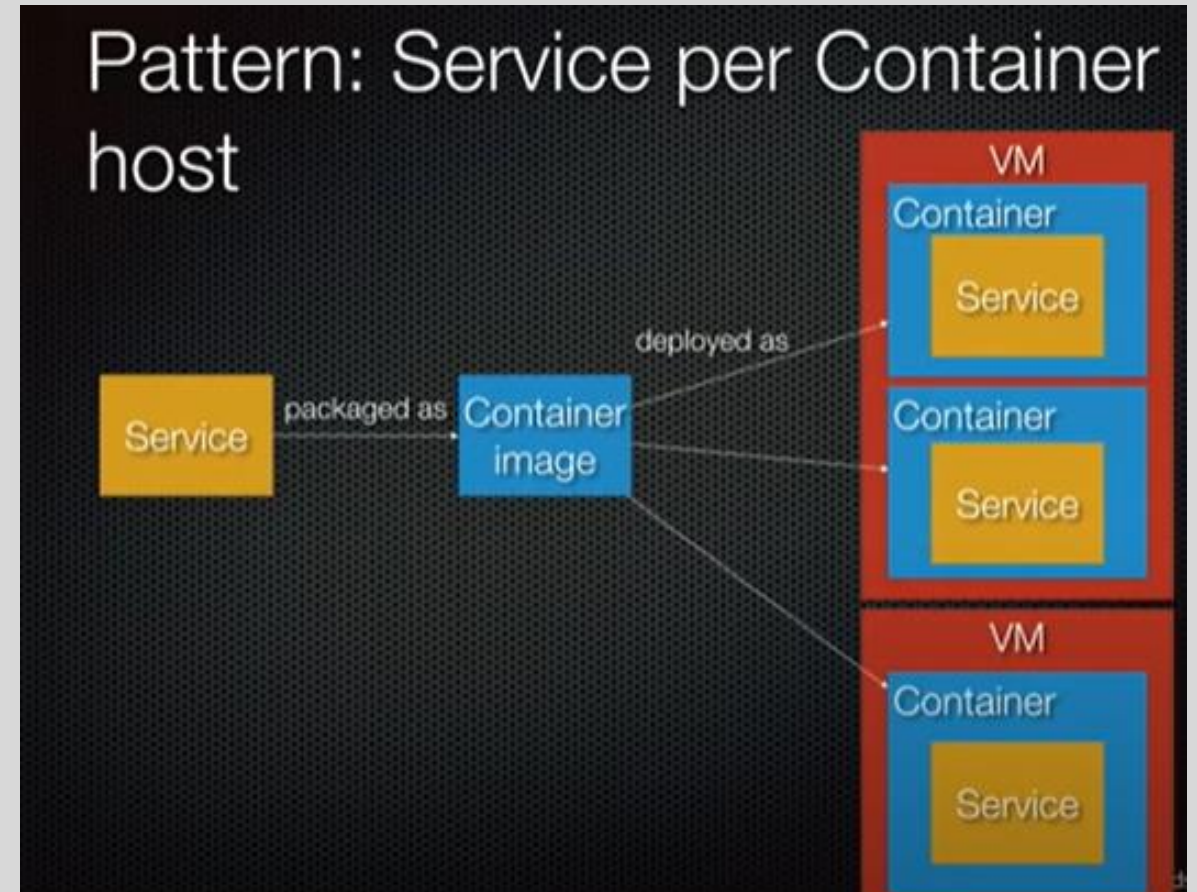
Docker - An open platform for distributed applications for developers and sysadmins.

[What is Docker?](#) [Try It!](#)



Utilisation de docker pour le déploiement

- Le service est donc installé dans une image de conteneur et déployé sur une plateforme supportant docker.
- Ceci permet:
 - Bonne isolation du service
 - Le service est “manageable”
 - Utilisation de technologies différentes
 - Utilisation efficiente des ressources machine
 - Déploiement rapide



Utilisation de docker en développement

- Docker permet de simplifier le développement des services.
- Très souvent en développement, nous avons besoin de services d'infrastructure (base de données, Message broker, etc.)

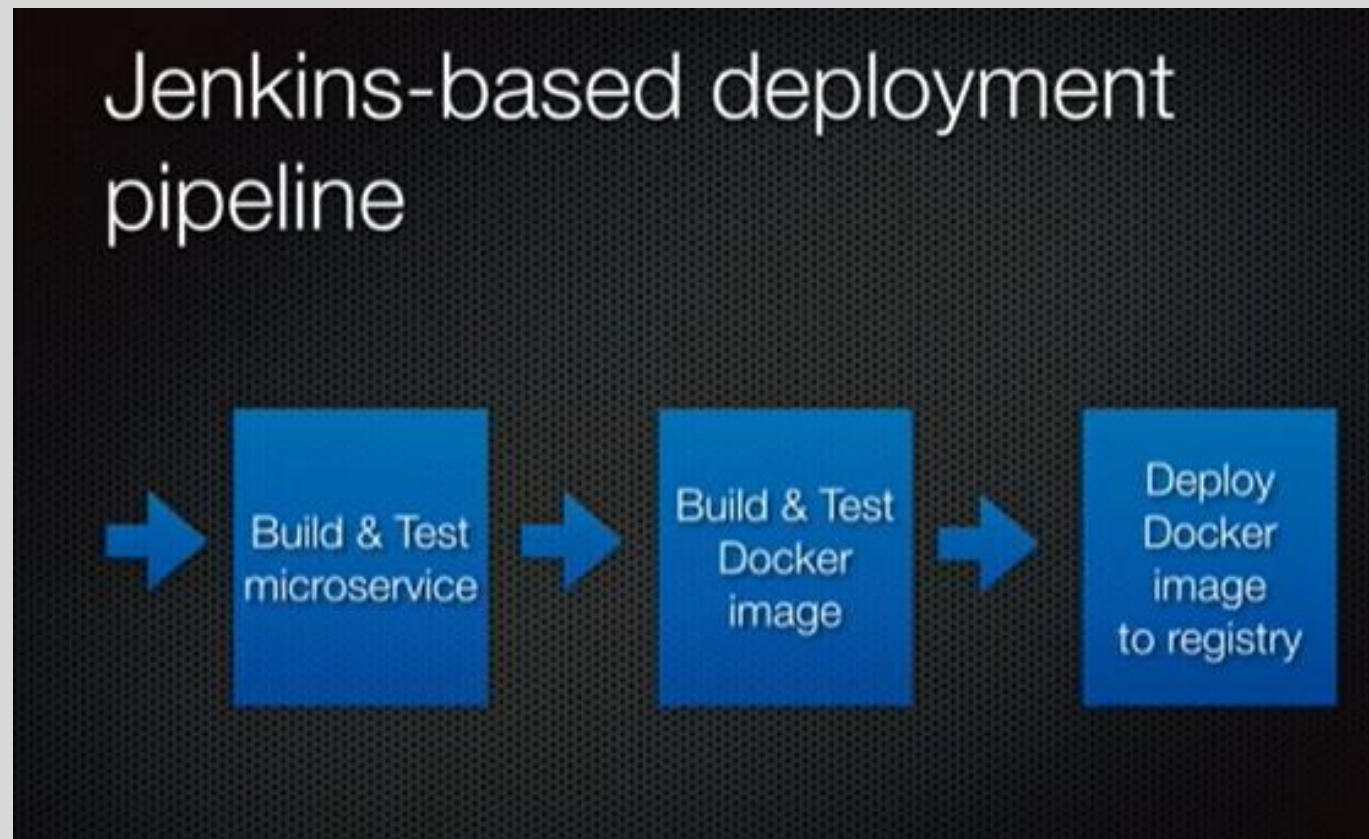
Docker
(Compose)
also simplifies
development



```
restfulservice:
  image: java:openjdk-8u45-jdk
  working_dir: /app
  volumes:
    - ./spring-boot-restful-service/build/libs:/app
  command: java -jar /app/spring-boot-restful-service.jar
  ports:
    - "8081:8080"
  links:
    - rabbitmq
    - mongodb
  environment:
    SPRING_DATA_MONGODB_URI: mongodb://mongodb/userregistration
    SPRING_RABBITMQ_HOST: rabbitmq
```

Déploiement docker avec Jenkins

- L'utilisation de docker permet également de configurer un pipeline de deployment des services par Jenkins.





Démo



CONTAINER ET IMAGE



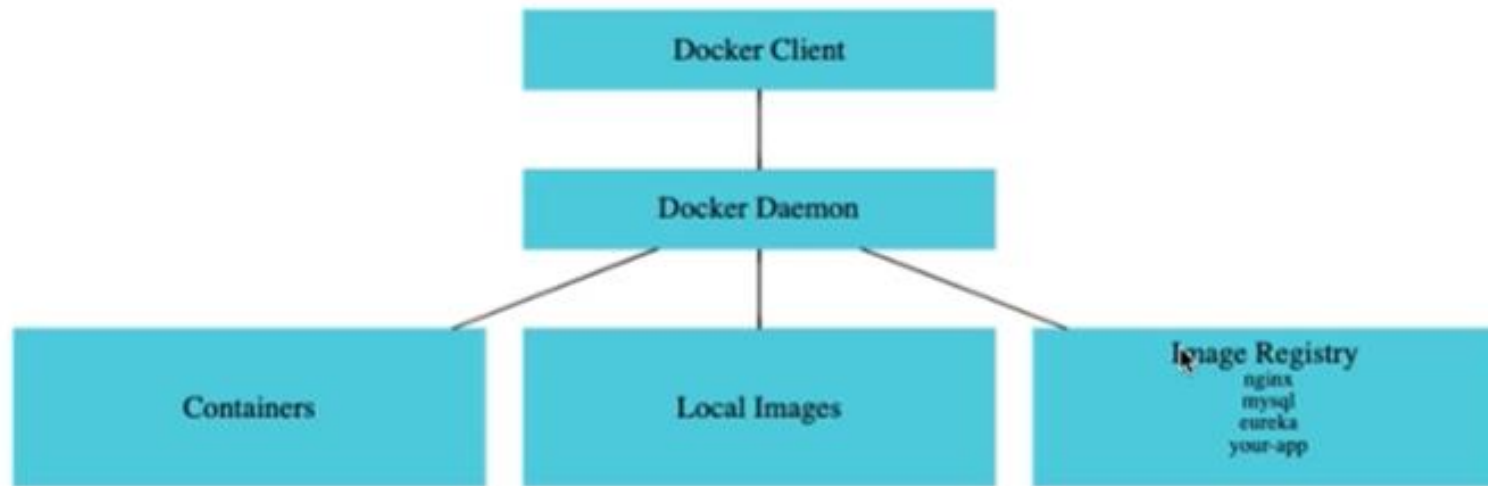
GITHUB ET
DOCKERHUB



CI/CD SUR JENKINS

L'architecture de docker

- Docker est une combinaison de Client, du Daemon, du Registre des images
- Les images peuvent être téléchargées localement pour s'exécuter dans des conteneurs



Docker Architecture

Projet – Conteneur – Image - Repository

- Nous allons utiliser des images pour executer une simple application
- Pour rappel, votre projet est copié dans une image que vous pouvez ou non mettre sur un repository.
- De cette image, plusieurs conteneurs docker peuvent être executes.



Containers

Hello World polyglot!

- Nous allons exécuter les commandes suivantes pour tester docker:

```
Bernard@digitalcloud-1 ~  
$ docker run -p 5000:5000 adanlessossi/hello-world-python-rest:0.0.1.RELEASE|
```

```
Bernard@digitalcloud-1 ~  
$ docker run -p 5000:5000 adanlessossi/hello-world-java-rest:0.0.1.RELEASE|
```

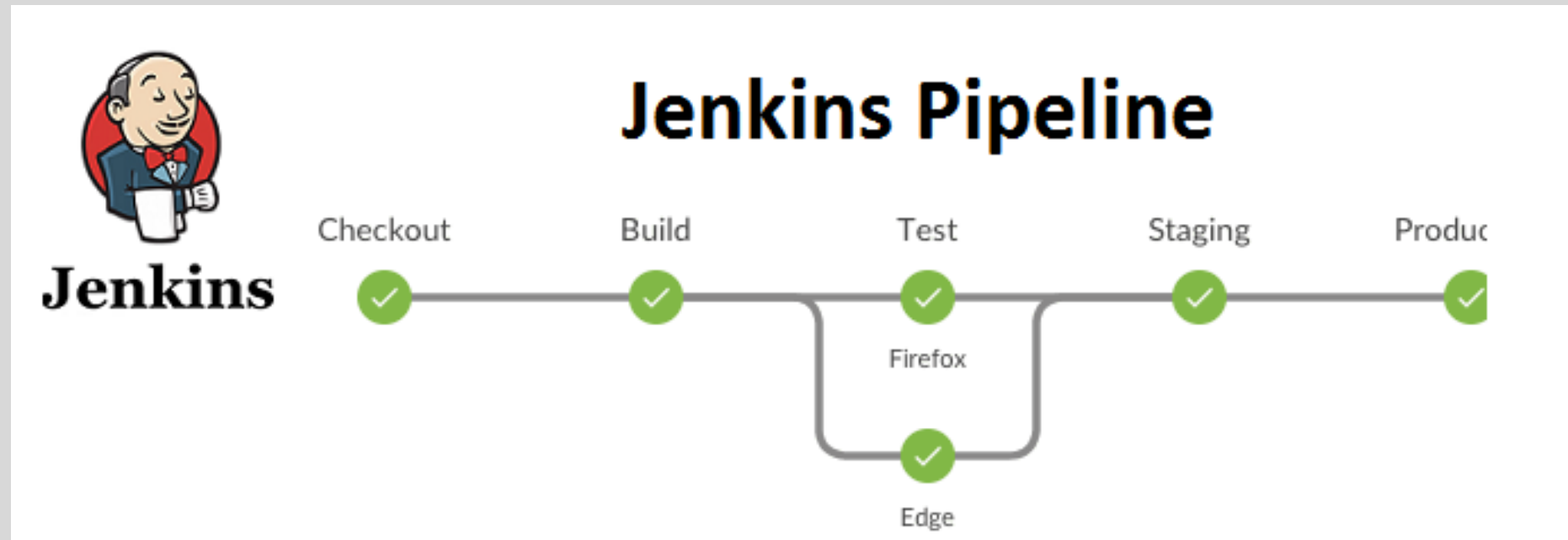
```
Bernard@digitalcloud-1 ~  
$ docker run -p 5000:5000 adanlessossi/hello-world-node-rest:0.0.1.RELEASE|
```

Hello World version2!

- Vous pouvez cloner les repositories suivants pour créer une version 2
- Toutes les instructions se trouvent dans le README
- <https://github.com/adanlessossi/hello-world-python-rest.git>
- <https://github.com/adanlessossi/hello-world-java-rest.git>
- <https://github.com/adanlessossi/hello-world-node-rest.git>
- Pour executer toutes les instructions, il vous faut un compte gratuit chez Dockerhub
- <https://hub.docker.com/>

Pipeline CI/CD

- Nous allons créer un pipeline de Deployment avec Jenkins
- <https://www.jenkins.io/>



Docker pour créer la Pipeline CI/CD

- Nous allons utiliser une image docker pour créer notre pipeline.
- <https://github.com/adanlessossi/jenkins.git>
- Avant de commencer il faudra s'assurer que docker est installé

```
Bernard@digitalcloud-1 /cygdrive/d/ateliers togojug/  
$ docker-compose --version  
docker-compose version 1.25.5, build 8a1c60f6
```

- Naviguez jusqu'au dossier Jenkins. Analysez le fichier compose et exécutez:

```
Bernard@digitalcloud-1 /cygdrive/d/ateliers togo  
$ docker-compose up|
```

Installation de Jenkins

- Après un bref moment dépendant de votre connection internet, l'image sera téléchargée et le conteneur démarré.
- Vous verrez également le mot de passe Administrateur dans les logs pour vous connecter pour la première fois.

```
jenkins | *****
jenkins | *****
jenkins | *****
jenkins |
jenkins | Jenkins initial setup is required. An admin user has been created and a pass
jenkins | Please use the following password to proceed to installation:
jenkins |
jenkins | c9d1725e27334bc7b7a8de88fba57e46
jenkins |
jenkins | This may also be found at: /var/jenkins_home/secrets/initialAdminPassword
jenkins |
jenkins | *****
jenkins | *****
jenkins | *****
```

Installation des plugins Jenkins

- Naviguez à l'URL : <http://localhost:8081> et insérez le mot de passe copié:

Getting Started

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log (**not sure where to find it?**) and this file on the server:

```
/var/jenkins_home/secrets/initialAdminPassword
```

Please copy the password from either location and paste it below.

Administrator password

Continue

Installation des plugins Jenkins

- Cliquez sur install suggested plugins et continuez:

Getting Started

Customize Jenkins

Plugins extend Jenkins with additional features to support many different

Install suggested plugins

Install plugins the Jenkins community finds most useful.

Select plugins to install

Select and install plugins most suitable for your needs.

Création de l'administrateur Jenkins

- Créez votre administrateur Jenkins et continuez.

Getting Started

Create First Admin User

Username:

Password:

Confirm password:

Full name:

E-mail address:

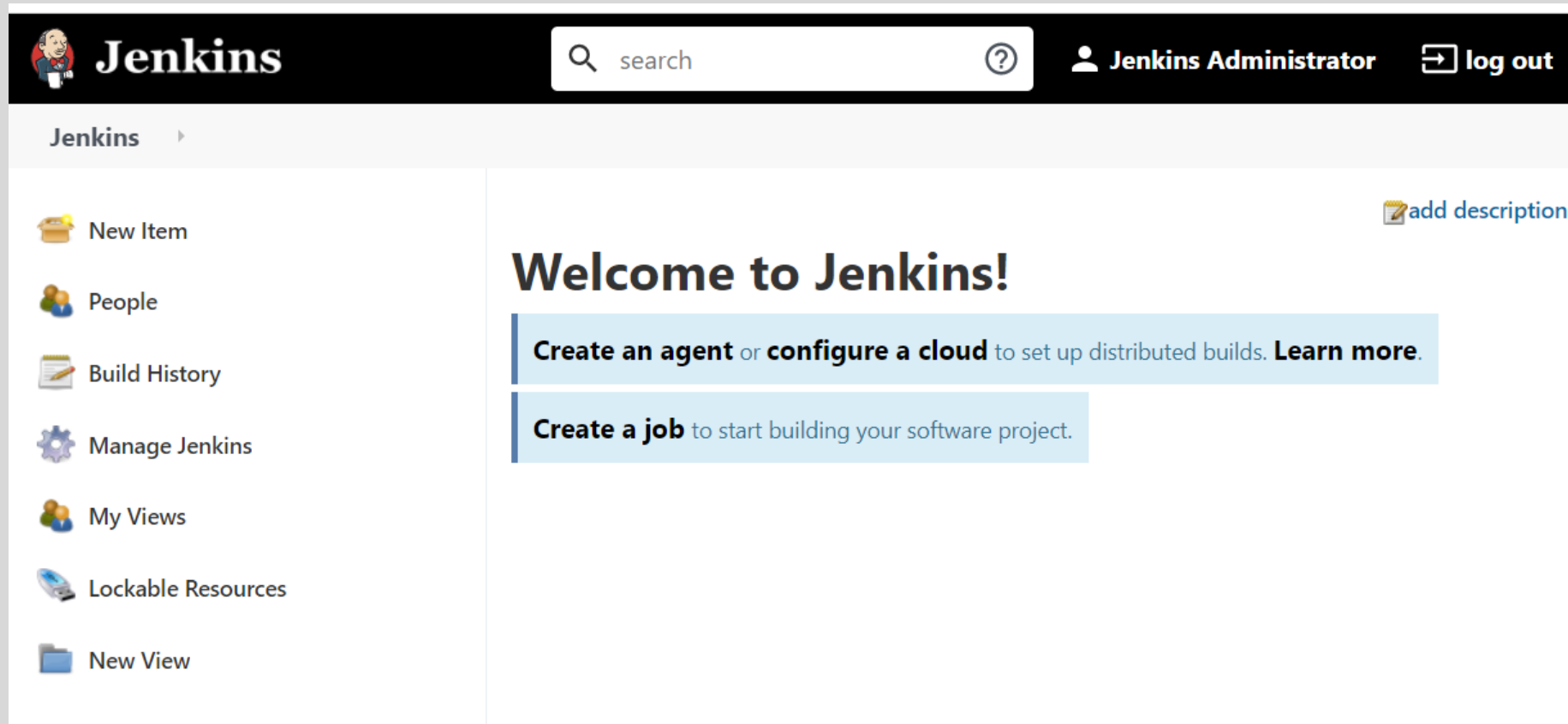
Jenkins 2.249.1

[Skip and continue as admin](#)

[Save and Continue](#)

Le Dashboard de Jenkins

- Après un moment, vous verrez le Dashboard où nous allons configurer les projets.



Configuration de maven et docker

- Pour créer un pipeline de projet java, nous avons besoin des plugins de maven et docker
- Dans “manage Jenkins”, sélectionnez “manage plugins”, installez les plugins de maven et de docker.
- Le menu qui nous intéresse vraiment, c’est “Global Tool Configuration” où nous pouvons configurer le pipeline.

Configuration de maven et docker

- Pour créer un pipeline de projet java, nous avons besoin des plugins de maven et docker
- Dans “manage Jenkins”, sélectionnez “manage plugins”, installez les plugins de maven et de docker.
- Le menu qui nous intéresse vraiment, c’est “Global Tool Configuration” où nous pouvons configurer le pipeline.

Maven

Maven installations

Add Maven

Maven

Namejenkins-maven

☒ Install automatically

Install from Apache

Version3.6.3

Docker

Docker installations

Add Docker

Docker

Namejenkins-docker

☒ Install automatically

Add Installer

Création du pipeline

- Au Dashboard de Jenkins, choisissez “Create new Job”

Enter an item name

» *Required field*



Freestyle project

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.



Maven project

Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.



Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Configuration du trigger

- Nous aimerions que ce job soit déclenché chaque 5 minutes

Build Triggers

☐ Build after other projects are built

☐ Build periodically

☐ GitHub hook trigger for GITScm polling

☒ Poll SCM

Schedule

Configuration du Pipeline

- Nous devons configurer notre git repository pour exécuter le pipeline

Pipeline

Definition Pipeline script from SCM

SCM Git

Repositories

Repository URL https://github.com/adanlessossi/hello-

Credentials - none - Add

Advanced...

Add Repository


Script Path

Jenkinsfile

Lightweight checkout ☒

déclenchement du Pipeline

- Cliquez sur “Build now”

 [Back to Dashboard](#)

 [Status](#)

 [Changes](#)

 [Build Now](#)

 [Configure](#)

Pipeline jenkins-devops-pipeline


Java Pipeline Job

 [Recent Changes](#)

Résultat du Pipeline


 Delete Pipeline

 Full Stage View

 Rename

 Pipeline Syntax

 Polling Log

 Build History trend ^

find x

 #1 [Oct 3, 2020 8:08 AM](#)

 [Atom feed for all](#)  [Atom feed for failures](#)

Stage View

Average stage times:
(Average full run time: ~11s)

#1
Oct 03 10:08
No Changes

Build

Test

67ms

12ms

67ms

12ms

Permalinks

CONCLUSION

- Ceci n'est qu'une introduction aux nouveaux pouvoirs qui sont donnés aux développeurs.
- Plus de renseignements:
- <https://docs.docker.com/>
- <https://github.com/>
- <https://hub.docker.com/>
- <https://www.jenkins.io/>

