

Pratiques de développement

David THIBAU – 2024

david.thibau@gmail.com



Agenda

- **Environnement de dév**
- **Gestion des sources**
- **Interactions entre services**
 - Introduction
 - RPC
 - Messaging
- **Cohérence des données et transactions**
 - Introduction
 - Saga Pattern
- **Logique métier**
 - Introduction
 - *Transactional Script Pattern*
 - Patterns orienté objet
 - *Event Sourcing Pattern*
- **Requêtage**
 - *API Composition Pattern*
 - *CQRS Pattern*
- **API Externe**
 - *Gateway Pattern*
- **Tests**
 - Introduction
 - Tests unitaires
 - Tests d'intégration
 - Tests de composants
 - Tests End To End
- **Déploiement**
 - Préparation
 - Infrastructure
 - Kubernetes et Istio

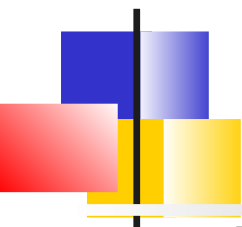


IDEs

Intellij IDEA (Communautaire / payant)
Dédié à Java et Kotlin

VSCode
Multi-langages

Eclipse : Le crépuscule ?
Lourdeur, Assistance moyenne



Copilot

Payant (licence corporate ou individuelle)

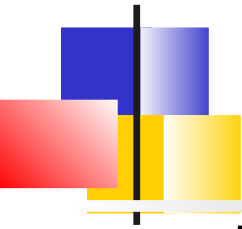
Pair-programming avec AI

Nécessite un compte GitHub

Compatible IntelliJ, VScode

Démo :

<https://docs.github.com/fr/copilot/using-github-copilot/getting-started-with-github-copilot>



SonarLint

Free

Detect coding issue in Real-Time

Peut être connecté à SonarQube

=> Apprendre les bonnes règles de codage



IDEs pour SpringBoot

Plugins maintenu par l'équipe SB pour :

- 1) VSCode
- 2) Eclipse
- 3) Theia

Principaux Apports du plugin :

- Ajout de starter pour modifier son *pom.xml* / *build.gradle*
- Assistance pour l'édition des propriétés de configuration SB et applicative si starter *configuration-processor*
- Boot Dashboard (Démarrage / redémarrage des services), Intéressant si l'on développe plusieurs micro-services
- Run Configurations (Profil, Arguments de CLI)
- Redémarrage automatique si starter *DevTools*

Intellij ?



Outils de build

Maven, l'ancêtre :

- Très bien supporté dans les IDE
- Beaucoup de plugins disponibles
- Bien maîtrisé ?
- *pom.xml* verbeux et peu lisible
- Pas très performant et verbeux lors de l'exécution
- Limité à Java

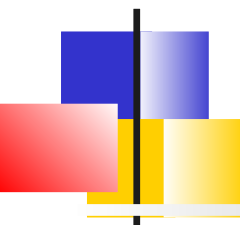
Gradle, le challenger

- Performant et rapide
- Beaucoup de plugins disponibles :
 - Plugin officiel => utilisable ad-hoc, exemple SB
 - Non officiel => Confiance relative, nécessite l'accès au source
- Très customisable mais courbe d'apprentissage pas simple
- Moins bien supporté dans les IDE
- Bon support pour C++

TypeScript ? : ng ?

build.gradle x

```
1 plugins {  
2     id 'java'  
3     id 'org.springframework.boot' version '3.2.2'  
4     id 'io.spring.dependency-management' version '1.1.4'  
5 }  
6  
7 group = 'org.formation'  
8 version = '0.0.1-SNAPSHOT'  
9  
10 java {  
11     sourceCompatibility = '17'  
12 }  
13  
14 configurations {  
15     compileOnly {  
16         extendsFrom annotationProcessor  
17     }  
18 }  
19  
20 repositories {  
21     mavenCentral()  
22 }  
23  
24 dependencies {  
25     implementation 'org.springframework.boot:spring-boot-starter-actuator'  
26     implementation 'org.springframework.boot:spring-boot-starter-data-jpa'  
27     implementation 'org.springframework.boot:spring-boot-starter-web'  
28     compileOnly 'org.projectlombok:lombok'  
29     developmentOnly 'org.springframework.boot:spring-boot-devtools'  
30     runtimeOnly 'org.postgresql:postgresql'  
31     annotationProcessor 'org.springframework.boot:spring-boot-configuration-processor'  
32     annotationProcessor 'org.projectlombok:lombok'  
33     testImplementation 'org.springframework.boot:spring-boot-starter-test'  
34 }  
35  
36 tasks.named('test') {  
37     useJUnitPlatform()  
38 }  
39
```

```
build.gradle  DemoBuild-2/pom.xml x
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema"
3   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/m
4   <modelVersion>4.0.0</modelVersion>
5   <parent>
6     <groupId>org.springframework.boot</groupId>
7     <artifactId>spring-boot-starter-parent</artifactId>
8     <version>3.2.2</version>
9     <relativePath/> <!-- lookup parent from repository -->
10  </parent>
11  <groupId>org.formation</groupId>
12  <artifactId>DemoBuild-2</artifactId>
13  <version>0.0.1-SNAPSHOT</version>
14  <name>DemoBuild-2</name>
15  <description>Demo project for Spring Boot</description>
16  <properties>
17    <java.version>17</java.version>
18  </properties>
19  <dependencies>
20    <dependency>
21      <groupId>org.springframework.boot</groupId>
22      <artifactId>spring-boot-starter-actuator</artifactId>
23    </dependency>
24    <dependency>
25      <groupId>org.springframework.boot</groupId>
26      <artifactId>spring-boot-starter-data-jpa</artifactId>
27    </dependency>
28    <dependency>
29      <groupId>org.springframework.boot</groupId>
30      <artifactId>spring-boot-starter-web</artifactId>
31    </dependency>
32
33    <dependency>
34      <groupId>org.springframework.boot</groupId>
35      <artifactId>spring-boot-devtools</artifactId>
36      <scope>runtime</scope>
37      <optional>true</optional>
38    </dependency>
39    <dependency>
40      <groupId>org.springframework.boot</groupId>
```



Stack Technologique

Persistance :

SQL, NOSQL, autres, Broker, Kafka

Langages :

Java (version?), Kotlin, Javascript,
TypeScript

Serveurs : Serveur JEE, Tomcat

Embarqué, Netty embarqué, Node



Frameworks

Framework maison : Retours ?

Spring Boot : Versions

Quarkus : ?



Tests Développeur

Opinion sur les tests unitaires ?

- Utilité (Dév, recette?)
- Granularité. TU dans un contexte de framework

Pratiques ? TDD

Calcul de couverture ?

Démo Tests en continus Quarkus



Tests SpringBoot

Support JUnit5, Hamcrest, AssertJ,
Mockito, JsonPath

Tests *système* : *@SpringBootTest*

Tests d'intégration : *@DataJpaTest*,
@DatMongoTest, *@JsonTest*,
@WebFluxTest

Sécurité : *@WithMockUser*, ...



Code, Deploy, Test

L'utopie : « Live Coding »

Les frameworks essaient d'y arriver :

- ng
- DevTools SpringBoot
- quarkus dev

La TDD peut raccourcir le cycle. Plus rapide de lancer le test que de démarrer un serveur et effectuer des clicks



Services de support

Exemple BD :

- BD de test partagée.
 - => Les développeurs se marchent dessus
- Installation locale.
 - => Les développeurs ont ils vraiment la même installation
 - => Que faire si l 'on a besoin de 2 versions différentes pour 2 projets différents
- Base de données embarquée.
 - => Très pratique car toujours dans un état connu
- Docker-compose.
 - => Très pratique car toujours dans un état connu
 - => Plus proche des conditions réelles



Gestion des sources



Dépôt

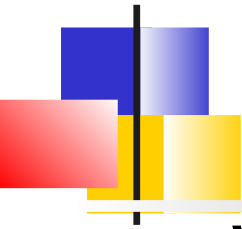
Git ou autre ?

Pratiques :

- IDE, Ligne de commande ?
- Utilisation de branches locales
- Fréquences des commits, des push
- Commandes avancées
 - git rebase -i*
 - git cherry-pick*
 -

Référentiel projet

- Gitlab
- Github
- Custom ?

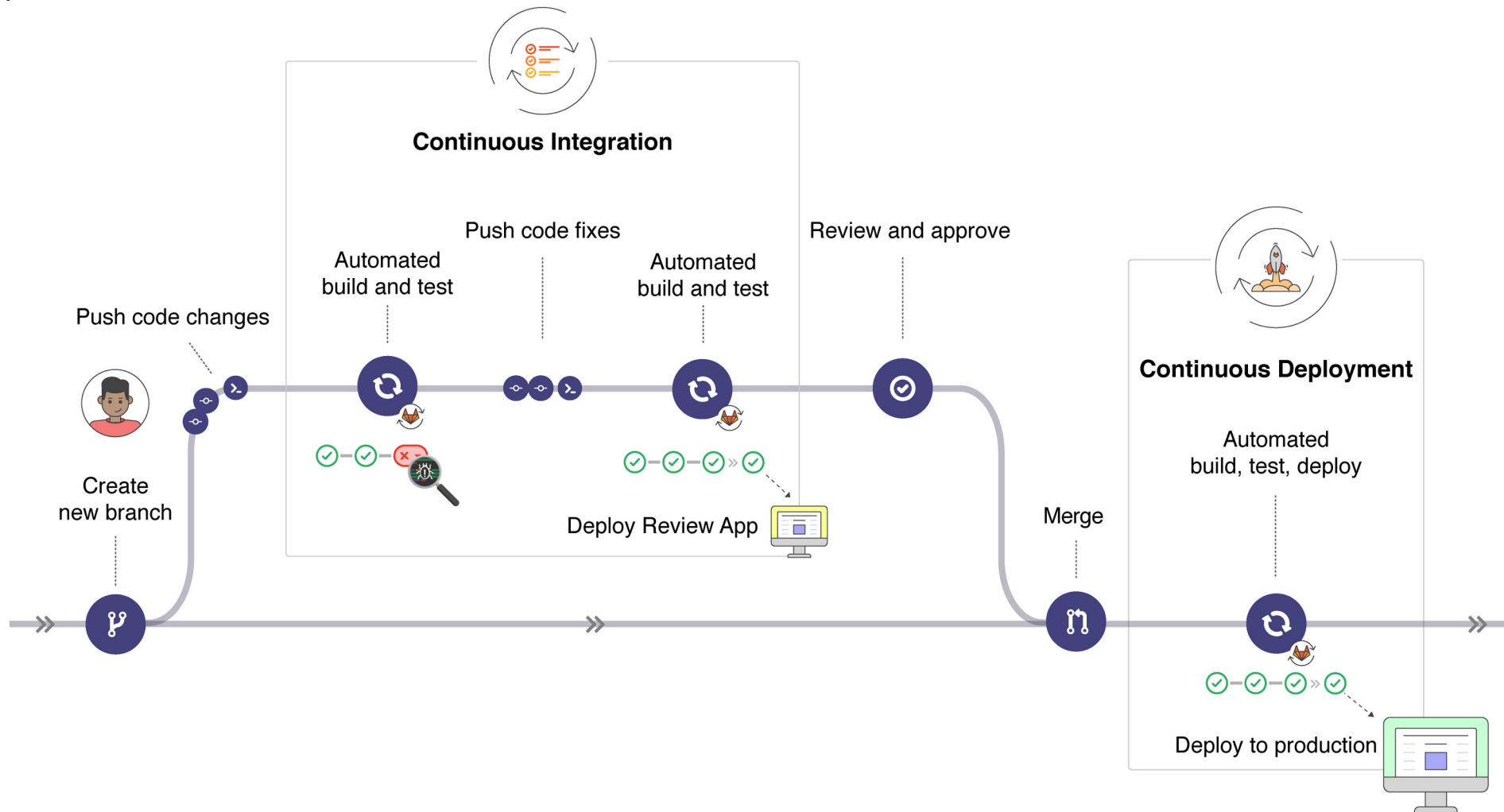


Workflow

Workflow de collaboration :

- Rôles dans un projet : Développeur, Mainteneur, PO ?
- PR, MR ?
- Branches stables : main, master, pré-prod, prod ?
- Revue de code avant intégration dans branches stables

Exemple Gitlab CI





Pilotage projet



Pratiques

Outils ?

Dashboard KanBan ?

Lien entre issue et commits

Implication du métier

Démo : MR Gitlab



CI/CD



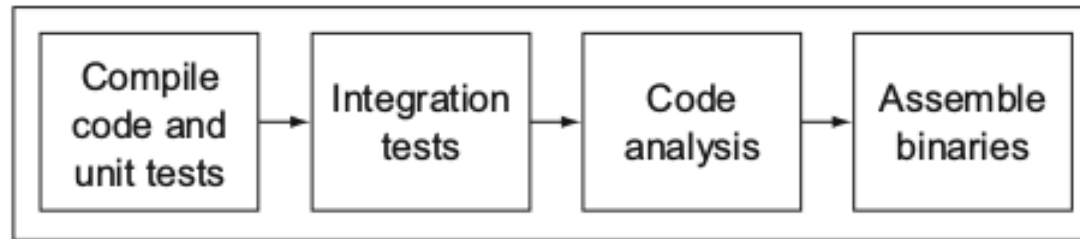
Distinction CI/CD



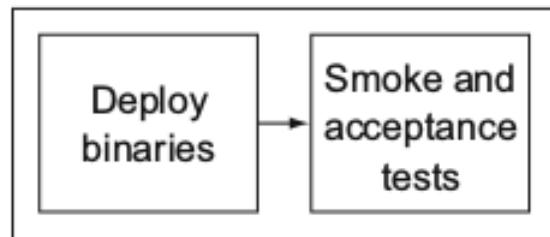


Example CD

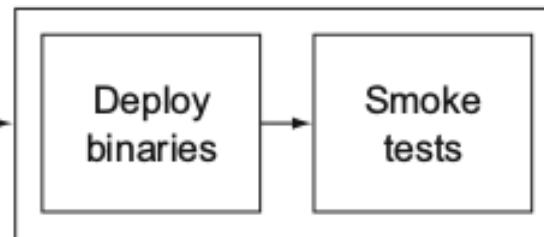
Commit stage



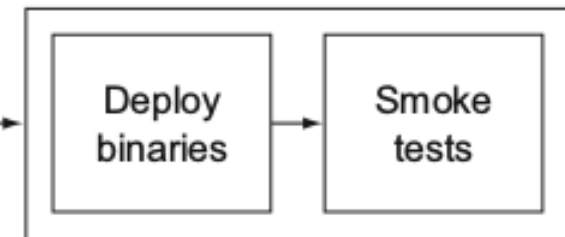
Acceptance stage



UAT



Production





Pratiques

Outils :

- Jenkins pipeline en groovy
=> Souplesse
- Gitlab-ci déclaratif lié à Gitlab et aux containers

Publication de métriques :

- Rapport des tests unitaires, intégration performance
- Rapport des analyses statiques



Qualité



Analyse statique

Outils :

- Sonarqube : Détection de bugs, code smells, Vulnérabilité
- Gitlab / Code climate



Analyse statique de vulnérabilité (SAST)

Code source:

- Free :
 - FindSecurityBug (Appli Java)
 - Gitlab CI
- Commerciaux :
 - CheckMarx
 - Snyk

Dépendances :

- Dependency-check Maven plugin
- Gitlab CI + Container Scanning



Analyse dynamique (DAST)

Gitlab (version payante?)

- DAST, API Fuzzing

OWASP zap

Burp (commercial)



Releasing



Dépôts d'artefacts

Outils :

- Nexus
- Artifactory
- Gitlab Package

Packaging :

- War
- Jar
- Docker



Release

Processus de release automatisé ?

- Set n° de version
- Packaging et test
- Commit
- Déploiement dépôt d'artefact
- Incrément version
- Commit



Tests d'acceptation



Pratiques

Environnements et type de test :

- Intégration
- QA



Déploiement



Pratiques

DevOps ou dépendant de l'équipe Infra ?

Fréquence des déploiements ?

Procédure de déploiement automatisé ?

A partir de dépôt ?

Déploiement immuable ?

Roll-back possible ?

Outil de migration de schéma



Retours sur la production



Pratiques

Endpoint de surveillance

- Métriques disponibles

Statistiques d'utilisation ?

Mécanisme de feed-back utilisateurs
finaux

Gestion des issues en prod