

# Intégration continue

## Table of Contents

Intégration .....	2
Intégration continue .....	17
Jenkins .....	23

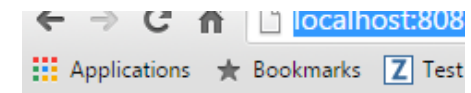
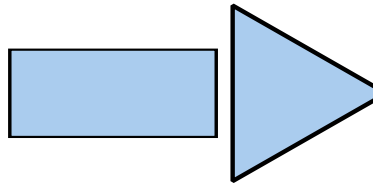
Intégration continue S. Fauvel – Univ. Nantes 2021

# Intégration

## Entre le développement et la production ?

```
public StoreServer(PriceStorage priceStorage) {
    this.priceStorage = priceStorage;
    Basket basket = new Basket();
    this.shopping = new Shopping();
}

public void start(int port) {
    HttpServer server = HttpServer.create(port, 1024);
    server.createContext("/");
}
```



### Grocery Store

Cliquer sur le fruit que vous voulez :

- [apple](#) (2 euros)
- [banana](#) (3 euros)

---

Montant du panier: 10 euros

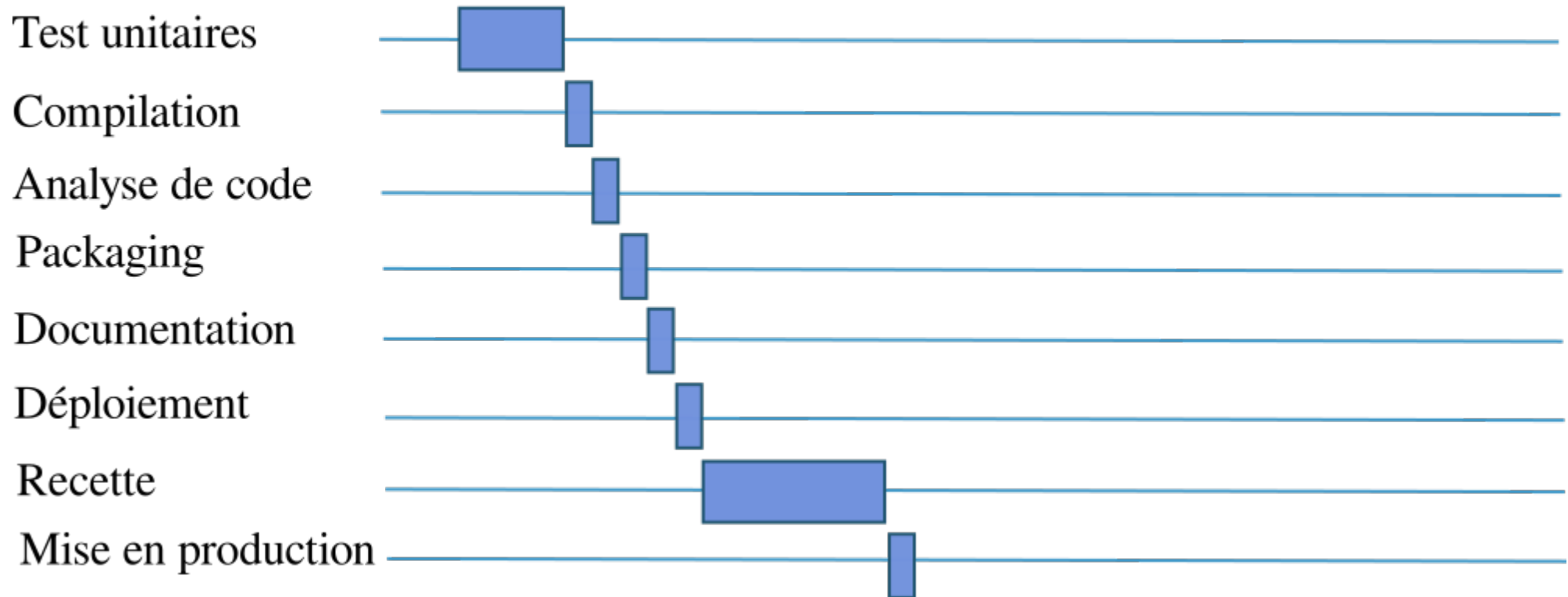
Contenu du panier:

- apple
- banana

On appelle "intégration" tout ce qu'il reste à faire à une équipe projet, quand le travail de développement à proprement parler est terminé, pour obtenir un produit exploitable, "prêt à l'emploi".

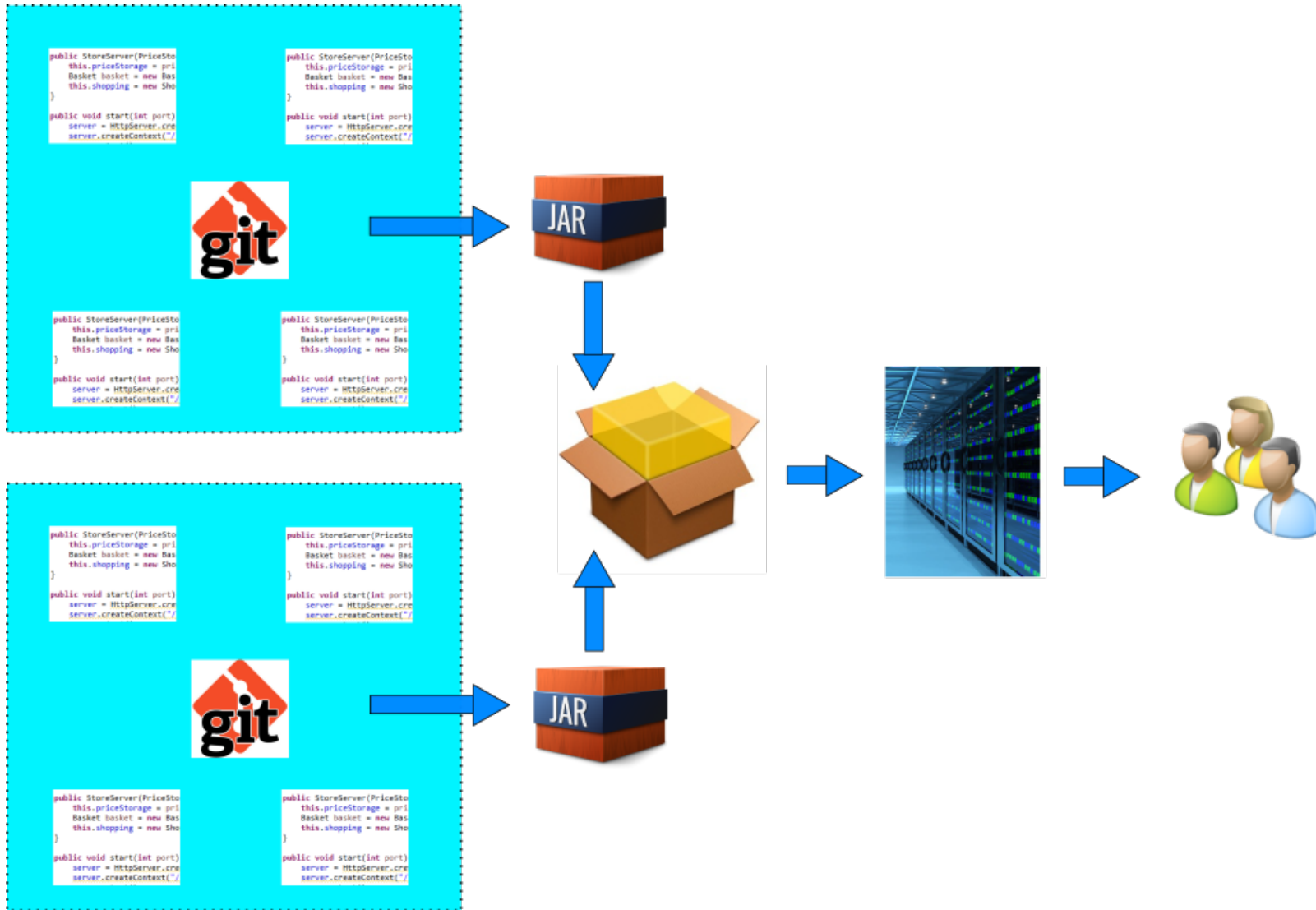
— Institut Agile

# Planning



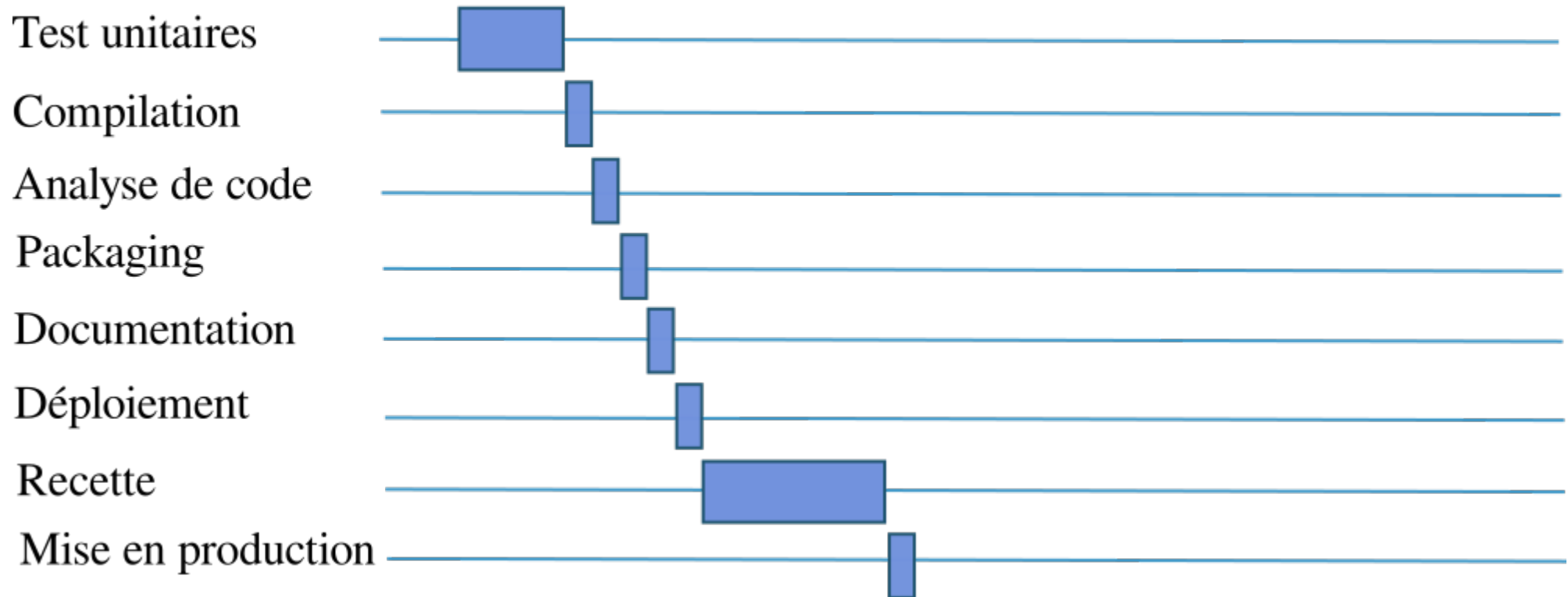


# Intégration



# Planning

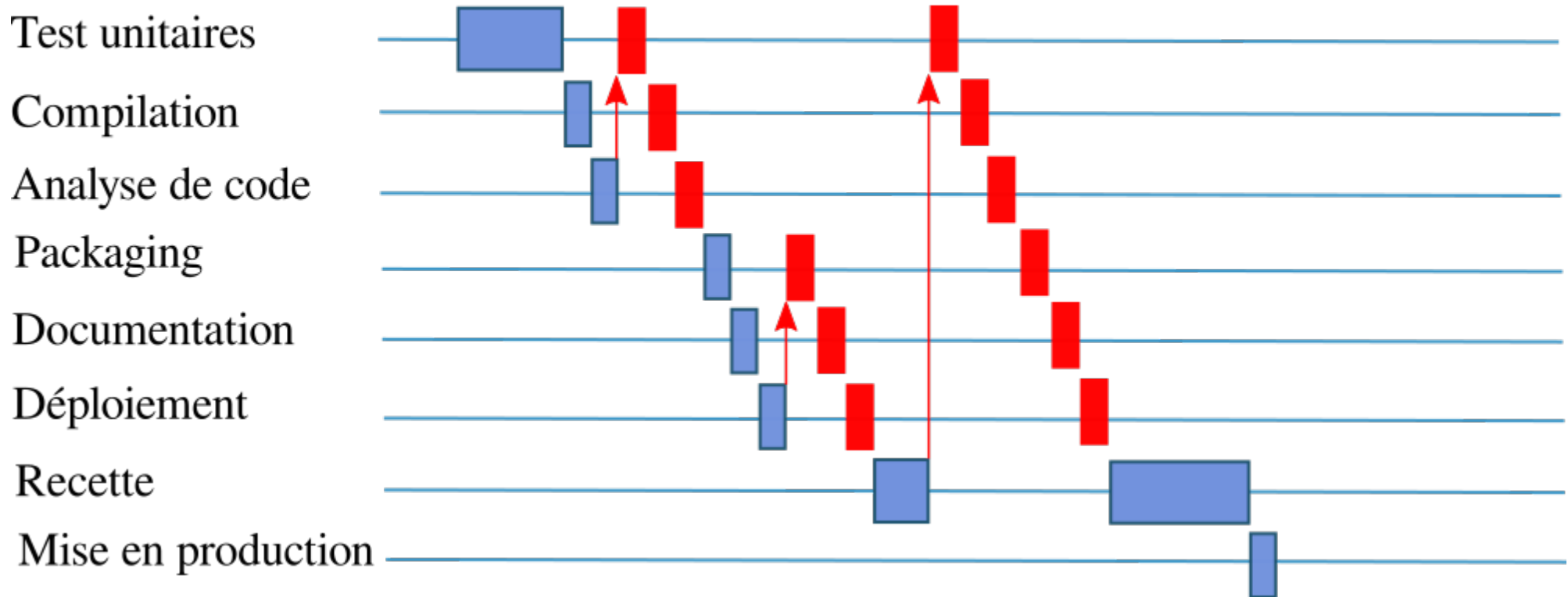
# Planning





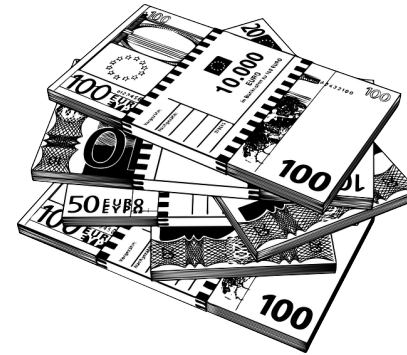
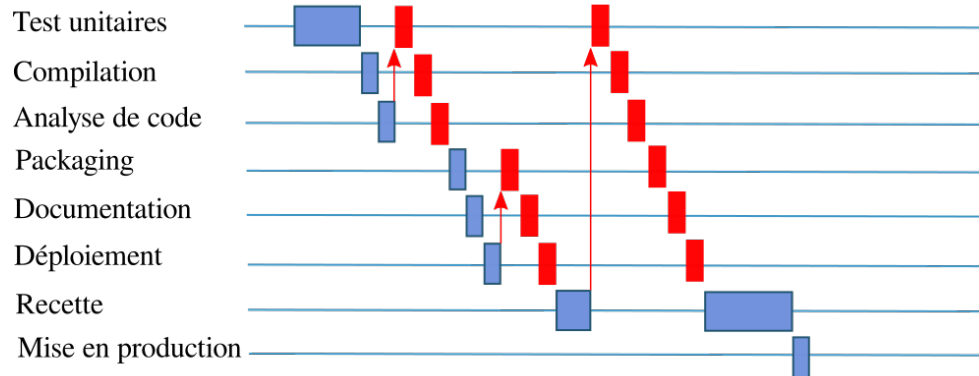


# En pratique



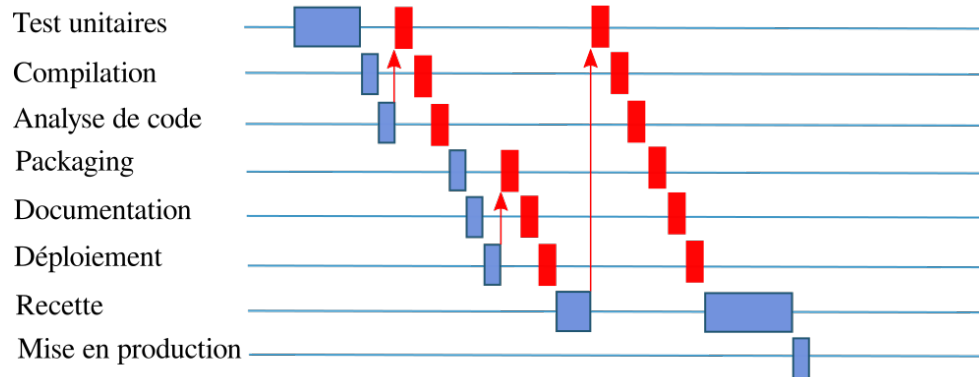


# Charge



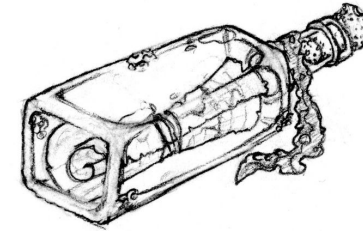
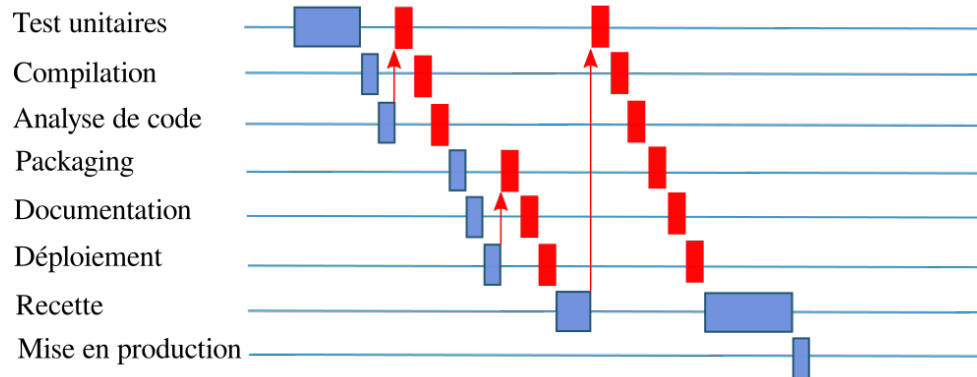
Les étapes à repasser sont une **charge de travail** supplémentaire

# Temps



Chaque étape à refaire allonge le **délai** de livraison

# Feedback



Le temps d'intégration éloigne d'autant le **délai entre le développement et sa validation**

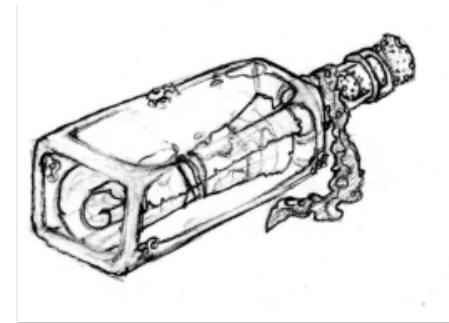
# Incertitude



Charges



Temps



Feedback



# 1 constat

- L'intégration est une phase compliquée
- C'est une étape incertaine
- La charge de travail est importante

# 2 visions

## Cycle en V

---

Le **moins** souvent possible

1 seule grande intégration

Estimer et planifier la phase  
d'intégration

Prévoir pour limiter les surprises

## Agilité

---

Le **plus** souvent possible

De nombreuses petites intégrations

Incorporer cette phase au  
développement

Avoir une feedback au plus tôt



# Intégration continue

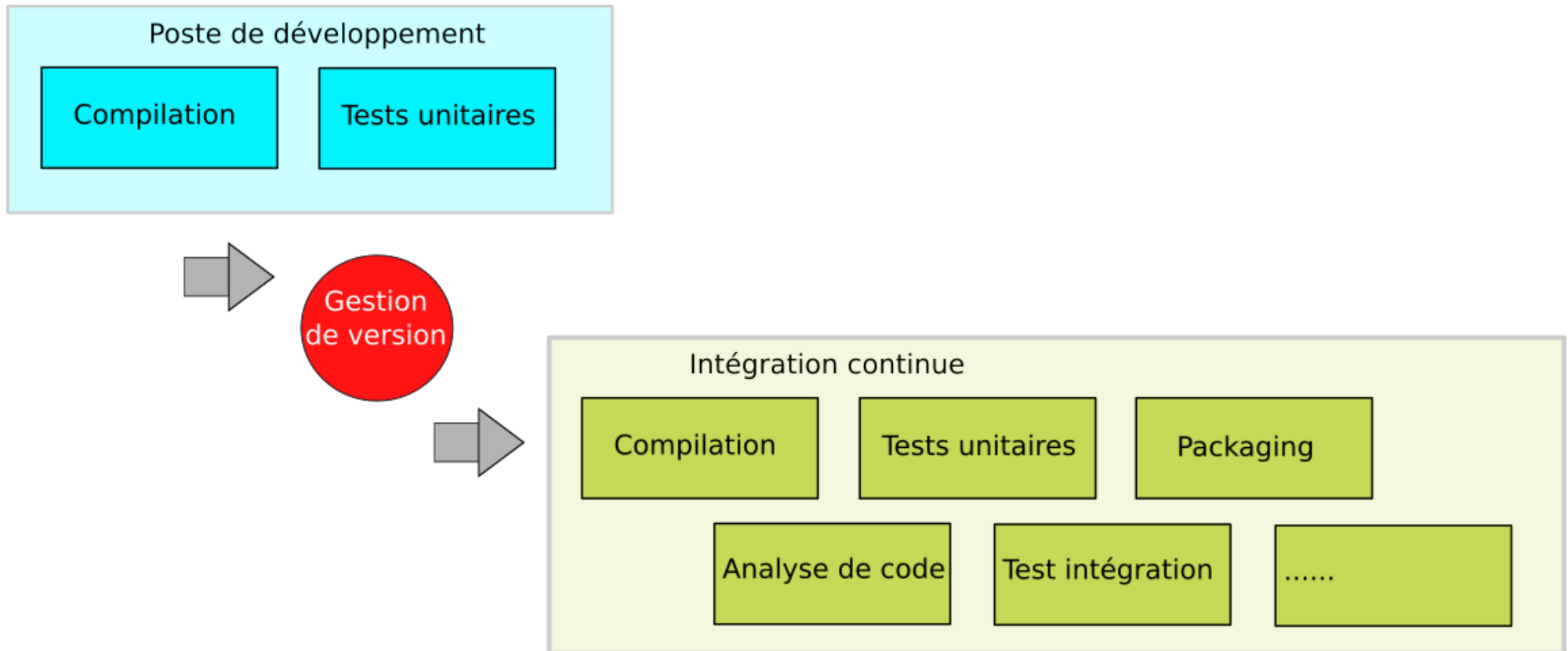
# Origine

- Grady Booch - **Object Oriented Design: With Applications**, 1991
- Kent Beck - **Extreme Programming Explained**, 1999
  - Diminuer les risques de conflit
  - Détecter les défauts au plus tôt
  - Avoir un logiciel opérationnel en permanence
- Martin Fowler - **Continuous Integration**, 2000

<http://www.martinfowler.com/articles/originalContinuousIntegration.html>

# Intégration continue

Automatiser la chaîne de build pour faire l'intégration de manière transparente



# Bonnes pratiques

- Commiter régulièrement du code fonctionnel
  - L'intégration continue s'appuie sur le code commité
  - L'application doit compiler et les tests passer
- Avoir des tests automatiser pour détecter les anomalies
- Corriger les builds en échec
- Optimiser les temps de builds pour améliorer le feedback

# Bénéfices

- Réduction des risques lié à l'incertitude
  - Etat du projet connu en permanence
  - Dernière version stable identifiée
- Réduction des tâches répétitives
  - Gain de temps pour les équipes
  - Réduction des délais pour l'intégration
- Impose des bonnes pratiques
  - Oblige une rigueur de travail
  - Assure la reproductibilité

# Définitions

- **Continuous Integration:** Vérifie l'intégration
- **Continuous Delivery:** Prêt à être mis en production
- **Continuous Deployment:** Mis en production

<https://medium.com/jorgeacetozi/continuous-integration-vs-continuous-delivery-vs-continuous-deployment-d5839a85a959>

# Jenkins



# Jenkins

Serveur d'intégration continue open source

Gère et contrôle les tâches du cycle de vie d'un logiciel:

- Compilation
- Documentation
- Tests
- Packaging
- analyse de code
- ...

Extensible en ajoutant des plugins



# Jenkins - installation

Serveur Web, un simple war

Lancement :

```
java -jar jenkins.war
```

Accès:

```
http://localhost:8080
```

# Jenkins - jobs

- Un **job** est **une configuration** pour l'intégration d'un projet
- Un projet peut avoir plusieurs jobs
- Le lancement d'un job exécute les instructions décrites dans sa configuration

# Jenkins – Tableau de bord

Création d'un job

Job

rechercher

Rafraîchissement automatique

Ajouter une description

Tous +

S	M	Nom du projet ↓	Dernier succès	Dernier échec	Dernière durée	
		<a href="#">SimpleProject</a>	10 j - <a href="#">#40</a>	10 j - <a href="#">#46</a>	19 s	
		<a href="#">ProjetSousGit</a>	9 j 23 h - <a href="#">#2</a>	9 j 23 h - <a href="#">#3</a>	9,1 s	
		<a href="#">Once</a>	2 an. 2 mo. - <a href="#">#1</a>	s. o.	1 mn 45 s	
		<a href="#">Epicerie</a>	1 j 23 h - <a href="#">#1</a>	s. o.	29 s	
		<a href="#">BasicProj</a>	22 h - <a href="#">#129</a>	s. o.	0,51 s	

État du lanceur de compilations

- 1 Au repos
- 2 Au repos

File d'attente des constructions

File d'attente des constructions vide

Administrer Jenkins

Credentials

Utilisateurs

Historique des constructions

Relations entre les builds

Vérifier les empreintes numériques

Nouveau Item

Jenkins

Légende

RSS pour tout

RSS de tous les échecs

RSS juste pour les dernières compilations

Statut du job

Nom

Lancement manuel du job

# Jenkins – Statut du job



Succès

Tout fonctionne



En échec

La construction n'a pas pu se faire (compilation, ...)



Instable

La build a abouti mais des tests échouent



Désactivé

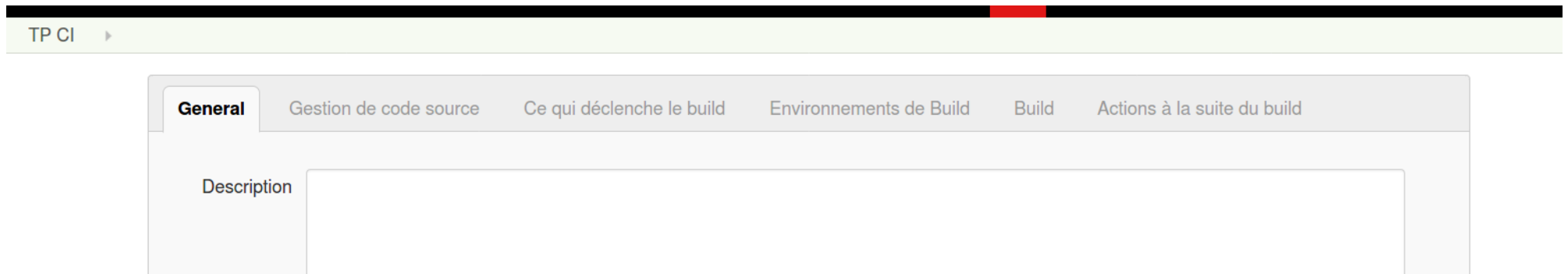
Le job est désactivé

# Jenkins - jobs

Plusieurs types de jobs

- **Free-style** : accès à toutes les options disponibles
- **Maven** : basé sur une configuration Maven
- **Multi-configuration** : projets complexes avec plusieurs builds
- **Job externe** : monitoring de processus externe
- **Pipeline** : définition du processus d'intégration/déploiement sous forme de script

# Jenkins - jobs








- Général
- Gestion de code source
- Ce qui déclenche le build
- Environnement de Build
- Build
- Actions à la suite du build

# Général

Description

[Plain text] [Prévisualisation](#)

- ☐ Ce build a des paramètres 
- ☐ GitHub project
- ☐ This build requires lockable resources
- ☐ Supprimer les anciens builds 
- ☐ Throttle builds 
- ☐ Désactiver le projet 
- ☐ Exécuter des builds simultanément si nécessaire 

[Avancé...](#)

- Paramètres du build
- Supprimer les anciens build
- ...

# Gestion de code source

**Gestion de code source**

☐ Aucune

☒ Git

Repositories

Repository URL

/home/project/cours\_ic

Credentials

- aucun -

Ajouter

Avancé...

Add Repository

Branches to build

Branch Specifier (blank for 'any')

\*/master

Add Branch

Navigateur de la base de code

(Auto)

Additional Behaviours

Ajouter

☐ Mercurial



# Gestion de code source

- Récupération dans un espace dédié
- Construction de l'application dans cet espace

The screenshot shows the Jenkins web interface for the 'GildedRose' project. The breadcrumb navigation at the top indicates the path 'Jenkins > GildedRose'. On the left sidebar, there are several action links: 'Retour au tableau de bord' (Home), 'État' (Status), 'Modifications' (Changes), 'Répertoire de travail' (Workspace), 'Effacer l'espace de travail' (Clean workspace), 'Lancer un build' (Build now), 'Supprimer Maven project' (Delete Maven project), 'Configurer' (Configure), and 'Modules'. The main content area is titled 'Workspace of GildedRose on ma'. It features a file explorer view showing a directory structure with folders '.git', 'src', and 'target'. Below these are files '.gitignore' (43 B) and 'pom.xml' (1.65 KB), both last modified on 5 nov. 2018 at 12:16:15. Each file has a 'vue' (view) link. At the bottom of the file list, there is a zip icon and a link '(Tous les fichiers dans un zip)'.

Jenkins > GildedRose >

[Retour au tableau de bord](#)

[État](#)

[Modifications](#)

**Répertoire de travail**

[Effacer l'espace de travail](#)


[Lancer un build](#)






[Supprimer Maven project](#)


[Configurer](#)

[Modules](#)

## Workspace of GildedRose on ma



-  [.git](#)
-  [src](#)
-  [target](#)
-  [.gitignore](#) 5 nov. 2018 12:16:15 43 B [vue](#)
-  [pom.xml](#) 5 nov. 2018 12:16:15 1.65 KB [vue](#)

 [\(Tous les fichiers dans un zip\)](#)

# Ce qui déclenche le build

## Ce qui déclenche le build

☐ Déclencher les builds à distance (Par exemple, à partir de scripts)  
☐ Construire après le build sur d'autres projets  
☐ Construire périodiquement  
☐ GitHub hook trigger for GITScm polling  
☒ Scrutation de l'outil de gestion de version

Planning

H/15 \* \* \* \*

Aurait été lancé à Sunday, November 4, 2018 8:44:21 PM UTC; prochaine exécution à Sunday, November 4, 2018 8:59:21 PM UTC.

Ignore post-commit hooks ☐





# Jenkins - déclenchement

- Planification:
  - Construire périodiquement
  - Scruter la gestion de configuration
- La syntaxe est celle du Cron
  - [MINUTES] [HEURES] [JOURMOIS] [MOIS] [JOURSEMAINE]
  - '\*' représente l'ensemble des valeurs possibles
  - 'N' déclenche lorsque la valeur vaut N
  - '\*/X' déclenche tous les X
- Exemple:
  - toutes les 5 minutes: \*/5 \* \* \* \*

- A 23h30 le vendredi : 30 23 \* \* 5


# Jenkins – gestionnaire de configuration

Déclenchement toutes les 2 minutes

 <b>#4</b>	25 oct. 2018 21:23
 <b>#3</b>	25 oct. 2018 21:21
 <b>#2</b>	25 oct. 2018 21:19
 <b>#1</b>	25 oct. 2018 21:17

Dernier accès au gestionnaire de configuration

 Javadoc

 Log du dernier accès à Git

 Open Blue Ocean

## Dernier Log du dernier accès à Git

```
Started on Nov 5, 2018 12:28:00 PM
Using strategy: Default
[poll] Last Built Revision: Revision fb3eb3a8c455c03bd4fcd2917db495781282b508 (refs/remotes/origin/master)
> git --version # timeout=10
> git ls-remote -h /home/project/cours_ic # timeout=10
Found 1 remote heads on /home/project/cours_ic
[poll] Latest remote head revision on refs/heads/master is: fb3eb3a8c455c03bd4fcd2917db495781282b508 - already built
by 26
Done. Took 14 ms
No changes
```

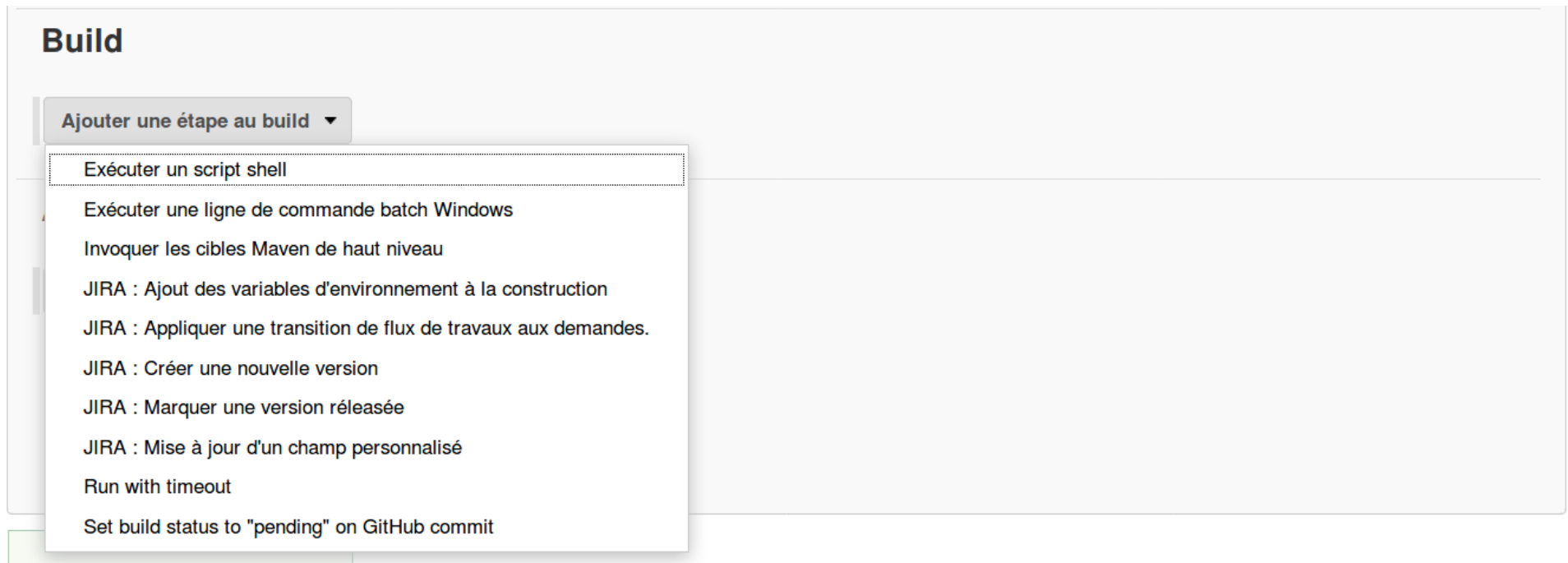
# Environnement de Build

## Environnements de Build

- ☐ Delete workspace before build starts
- ☐ Use secret text(s) or file(s)
- ☐ Abort the build if it's stuck
- ☐ Add timestamps to the Console Output
- ☐ Generate Release Notes



# Build



# Build

## Script shell

**Build**

Exécuter un script shell

Commande `echo "Version 1.0" > version.txt`

Voir [la liste des variables d'environnement disponibles](#)

Avancé...

Ajouter une étape au build ▾

## Build Maven

**Build**

Invoquer les cibles Maven de haut niveau

Version de Maven

Cibles Maven  ▾

Avancé...

Ajouter une étape au build ▾



# Jenkins – informations sur l'exécution

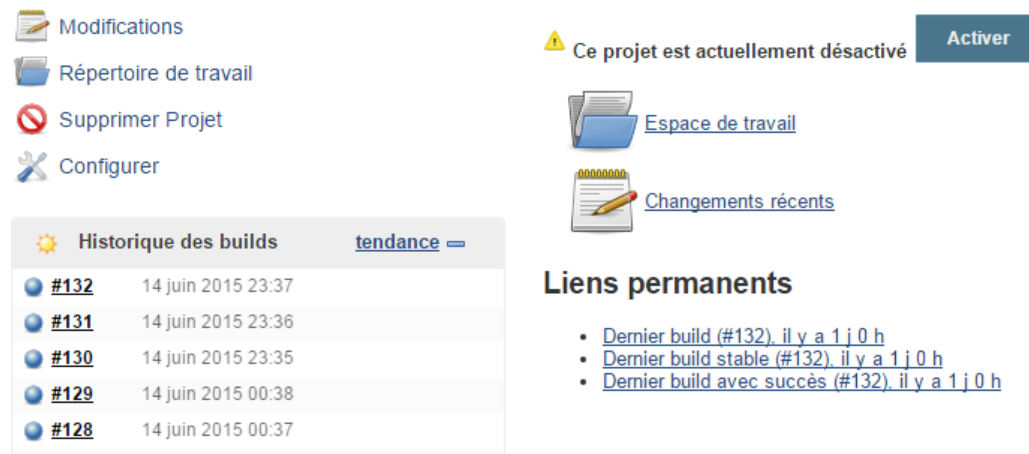
Sur le tableau de  
bord

MIAGE

Tous

+

Sur le job



Modifications  
Répertoire de travail  
Supprimer Projet  
Configurer

Ce projet est actuellement désactivé [Activer](#)

[Espace de travail](#)  
[Changements récents](#)

**Historique des builds** [tendance](#)

<a href="#">#132</a>	14 juin 2015 23:37
<a href="#">#131</a>	14 juin 2015 23:36
<a href="#">#130</a>	14 juin 2015 23:35
<a href="#">#129</a>	14 juin 2015 00:38
<a href="#">#128</a>	14 juin 2015 00:37

**Liens permanents**

- [Dernier build \(#132\), il y a 1 j 0 h](#)
- [Dernier build stable \(#132\), il y a 1 j 0 h](#)
- [Dernier build avec succès \(#132\), il y a 1 j 0 h](#)

Console



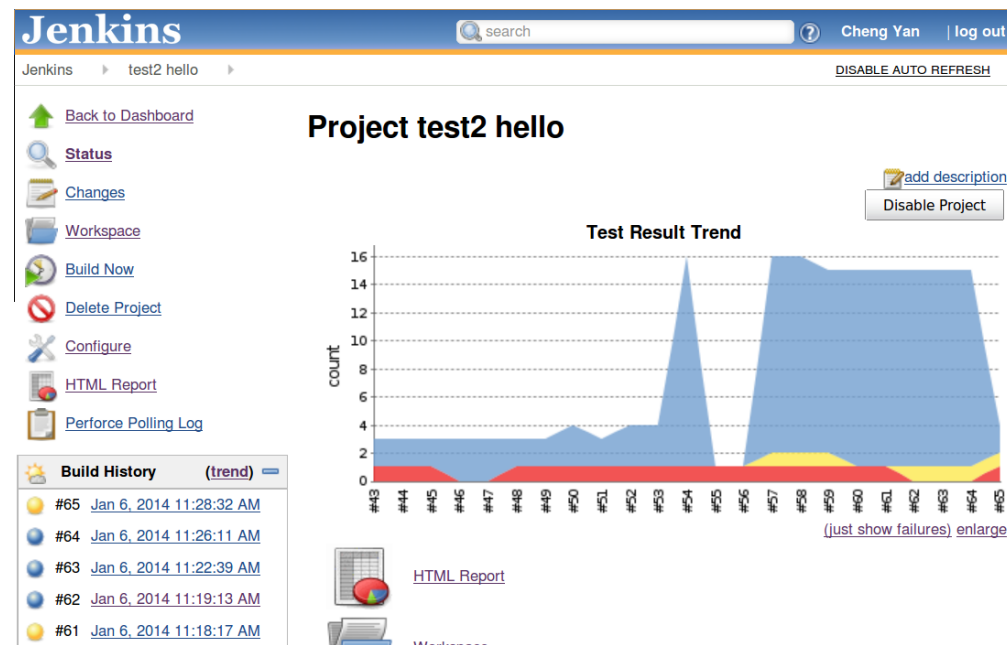
Retour au projet  
État  
Modifications  
**Console Output**  
View as plain text  
Informations de la construction  
Supprimer le build  
Build précédent

**Sortie de la console**

Lancé par une alarme périodique  
Building in workspace C:\Users\jrgf2315\.jenkins\workspace  
[BasicProj] \$ cmd /c call C:\Users\jrgf2315\AppData\Loca  
C:\Users\jrgf2315\.jenkins\workspace\BasicProj>C:\Users\  
C:\Users\jrgf2315\.jenkins\workspace\BasicProj>set texte  
C:\Users\jrgf2315\.jenkins\workspace\BasicProj>echo 14/0  
Finished: SUCCESS

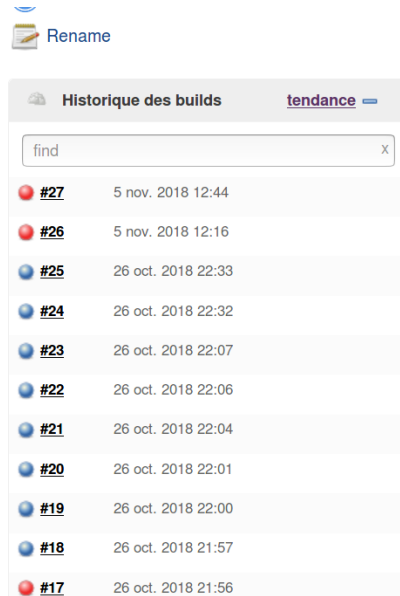
# Jenkins – Tests

- En cas d'échec des tests, l'état du build devient instable
- Le résultat des tests est affiché
- On peut accéder au rapport détaillé d'exécution des tests



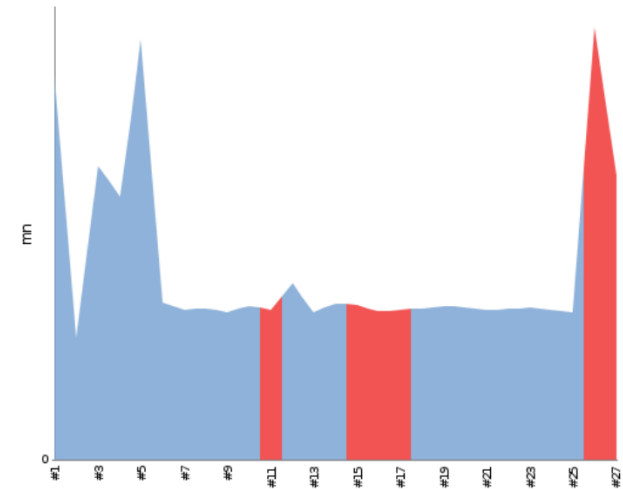
# Jenkins – Tendance

- Visualisation du temps de construction du build

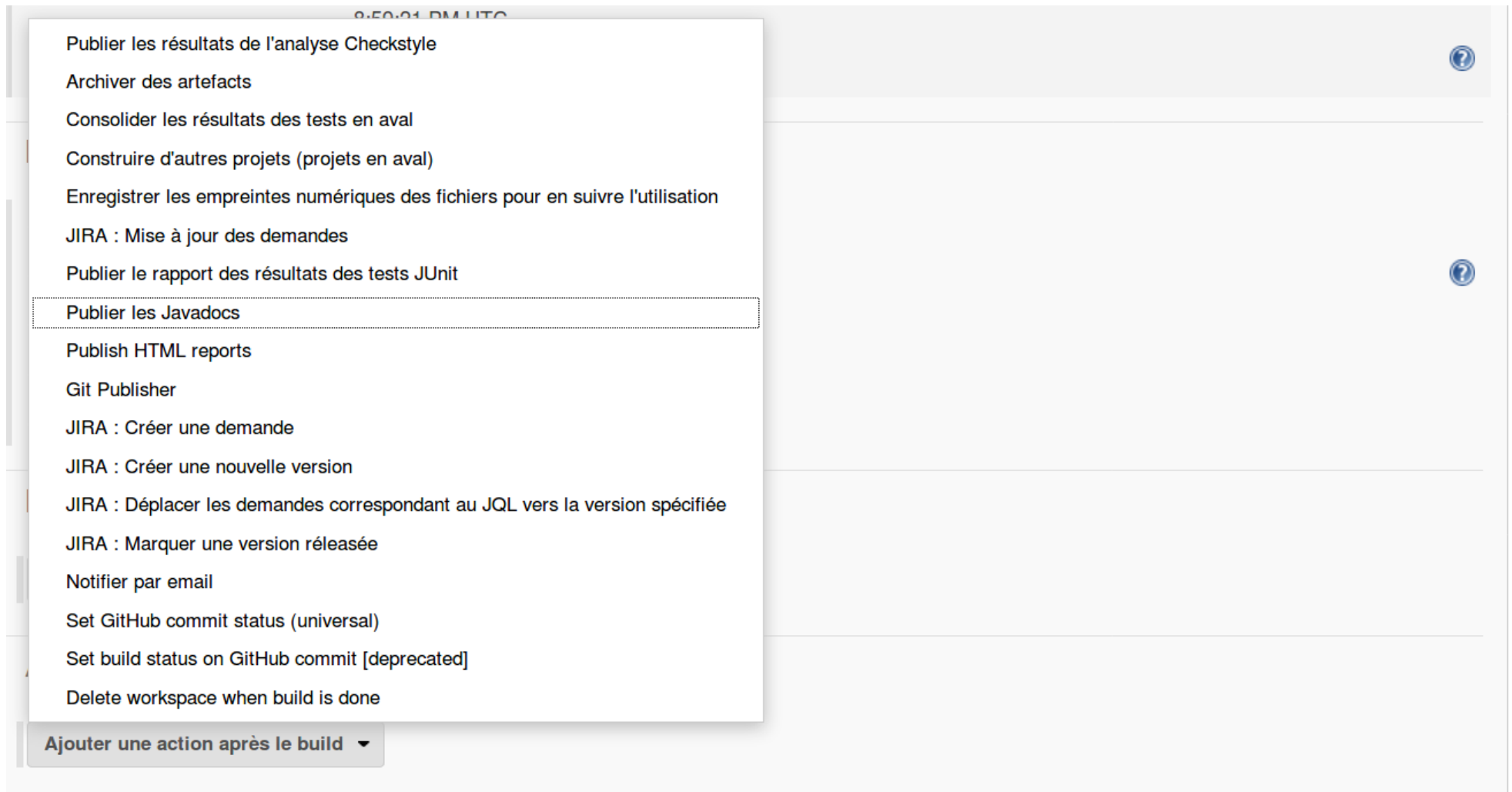


Tendance des temps de construction

Build	Durée
#27	19 s
#26	29 s
#25	10 s
#24	10 s
#23	10 s
#22	10 s
#21	10 s
#20	10 s
#19	10 s
#18	10 s
#17	10 s
#16	10 s
#15	10 s
#14	10 s
#13	10 s
#12	12 s
#11	10 s
#10	10 s



# Actions à la suite du build



The screenshot shows the GitHub Actions interface. A dropdown menu is open, displaying a list of actions to be executed after the build. The actions are as follows:

- Publier les résultats de l'analyse Checkstyle
- Archiver des artefacts
- Consolider les résultats des tests en aval
- Construire d'autres projets (projets en aval)
- Enregistrer les empreintes numériques des fichiers pour en suivre l'utilisation
- JIRA : Mise à jour des demandes
- Publier le rapport des résultats des tests JUnit
- Publier les Javadocs** (highlighted with a dashed border)
- Publish HTML reports
- Git Publisher
- JIRA : Créer une demande
- JIRA : Créer une nouvelle version
- JIRA : Déplacer les demandes correspondant au JQL vers la version spécifiée
- JIRA : Marquer une version réléasée
- Notifier par email
- Set GitHub commit status (universal)
- Set build status on GitHub commit [deprecated]
- Delete workspace when build is done

At the bottom of the dropdown, there is a button labeled "Ajouter une action après le build" with a downward arrow.

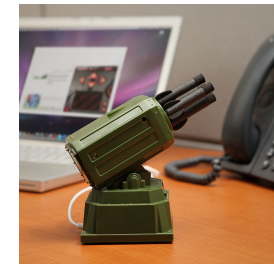
# Jenkins – Notification

Il est possible (recommandé) d'envoyer une notification en cas d'échec du build

**L'échec d'un build indique un problème qu'il faut régler au plus vite.**

La notification ne doit pas être considérée comme une simple information mais comme une alerte.

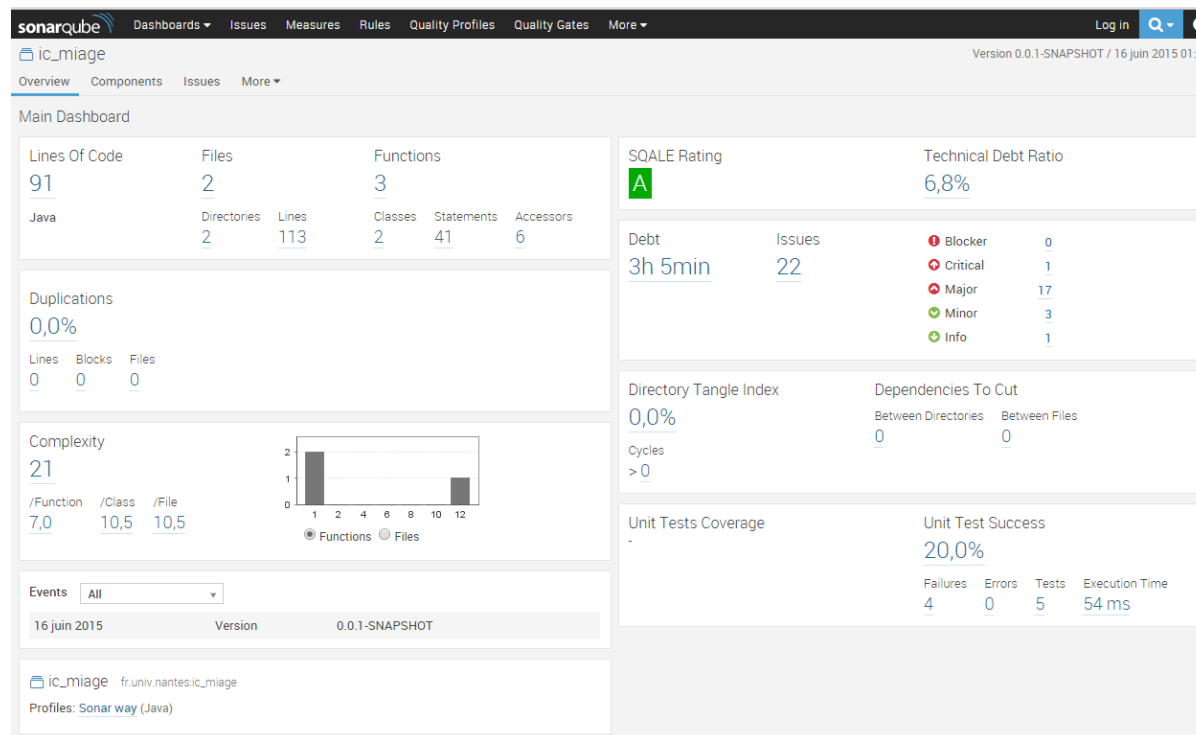
D'autres moyens peuvent être utilisés pour signaler le problème



La notification peut être 'ciblée' sur celui qui a cassé le build (nécessite d'ajouter les utilisateurs)

# Jenkins – Exécution post build: Analyse de code

- D'autres actions peuvent être réalisées en plus de la construction
- On peut déclencher une analyse de code (Sonar, Checkstyle, ...)
- Le rapport d'analyse est généré et accessible via des plugins



# Jenkins – Organisation des builds

- La multiplication des analyses augmente le temps de traitement et la charge du serveur d'intégration
- Certaines analyses ont moins d'intérêt à être réalisées en continue
- Créer plusieurs jobs avec des objectifs et des fréquences différents
- Exemple:
  - En continue : compilation, tests unitaires
  - Quotidien : analyse de code, javadoc, test bout en bout
  - Hebdomadaire: test de charge, test multi-navigateur

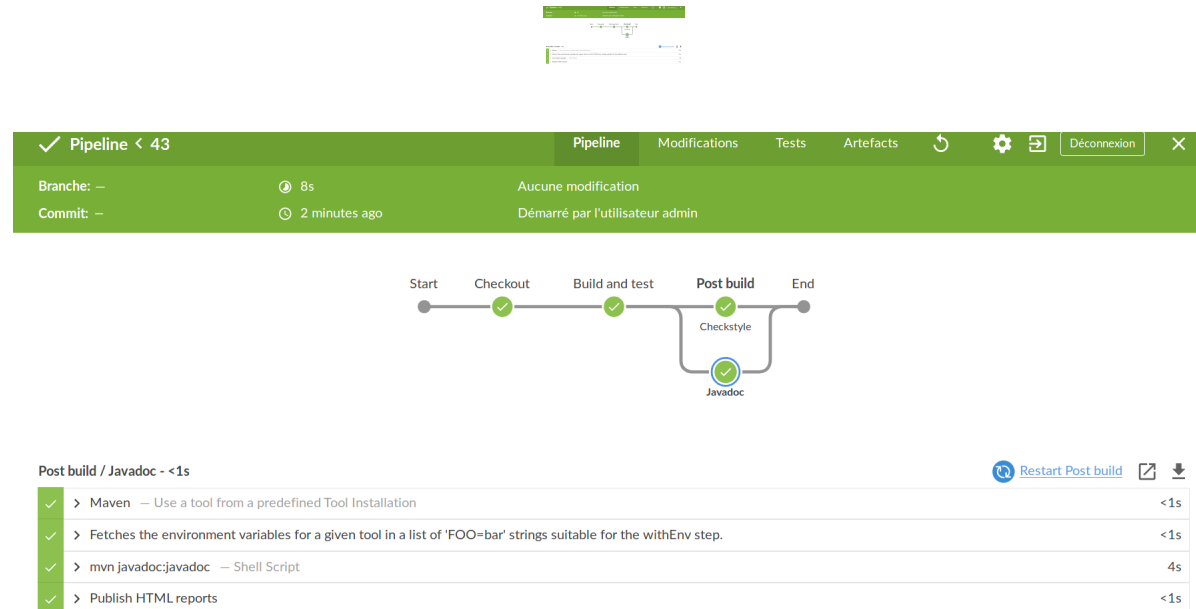
# Jenkins – Mode Maître / Esclave

- Le maître
  - défini les esclaves
  - agrège les données
- Les esclaves
  - répartir la charge
  - rendre scalable
  - permet d'avoir des environnements spécifiques (OS, version java, ...)



# Jenkins – Pipeline

- Description du job sous forme de code
- Permet la réutilisation
- Historisation du job dans le gestionnaire de source
- Exécution d'étapes en parallèle



# Jenkins – Pipeline

```
pipeline {
  agent any
  tools {
    maven 'Maven'
  }
  stages{
    stage('Checkout') {
      steps {
        checkout scm: [
          $class: 'GitSCM',
          branches: [[name: '*/master']],
          userRemoteConfigs: [[url: 'file:///home/project/cours_ic']]
        ]
      }
    }
    stage('Build and test') {
      steps {
        sh "mvn install"
      }
    }
    stage('Post build') {
      parallel {
        stage('Javadoc') {
          steps {
            sh "mvn javadoc:javadoc"
            // publish html
            publishHTML target: [
              reportDir: 'target/site/apidocs',
              reportFiles: 'index.html',
              reportName: 'Javadoc'
            ]
          }
        }
        stage('Checkstyle') {
          steps {
            sh "mvn checkstyle:checkstyle"
            publishHTML target: [
              reportDir: 'target/site',
              reportFiles: 'checkstyle.html',
              reportName: 'Checkstyle'
            ]
          }
        }
      }
    }
  }
}
```