

# Génie logiciel avancé - 2016

## TD n°6

### Système de planification des vols

---

Les objectifs de ce TD sont :

- De poursuivre l'implémentation du projet et de réaliser un backlog
- De mettre en place un environnement d'intégration continue.

#### **Question 1 – Backlog**

Un backlog est un listing de toutes les améliorations qui peuvent être apportées à un logiciel. Chaque amélioration doit être suffisamment détaillée pour donner lieu à un développement futur.

A partir des remarques et commentaires que vous avez eu durant les soutenances des séances précédentes ainsi que de vos propres remarques, vous devez réaliser un backlog sous la forme d'un fichier Excel contenant les éléments suivants pour chaque amélioration :

- Texte explicatif le plus clair possible
- Priorité sous la forme d'une note de 1 pour le moins prioritaire à 5 pour le plus urgent
- Charge de travail associée de 1 pour le plus difficile à réaliser à 5 pour le plus facile

Une fois réalisé, et en triant les éléments en fonction de la priorité puis de la charge de travail associée, vous serez capable de définir les prochaines « releases » de votre projet.

Ce backlog devra être fourni sur la plateforme du cours à la fin de la semaine (dimanche 20 mars).

## Question 2 – Installation de Jenkins

L'intégration continue consiste à mettre en place des outils vérifiant de manière la plus automatique possible la qualité et la consistance du code.

La plateforme que nous allons utiliser dans le cadre de ce projet est Jenkins. Ce logiciel a vocation à être déployée sur un serveur, mais pour faciliter son installation, vous devez le mettre en place sur votre propre machine.

Pour cela, téléchargez depuis le site internet le package d'installation adapté à votre système d'exploitation et suivant les instructions correspondantes :

<http://jenkins-ci.org/>

Une fois installée, vous accédez à Jenkins via une interface web comme celle-ci et qui devrait être disponible à l'url suivante :

<http://localhost:8080>



Pour terminer l'installation, vous devez installer les plugins « Git plugin » et « GitHub plugin » via le menu « Administrer Jenkins » puis « Gestion des plugins ».

### Question 3 – Configuration du « job »


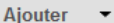
L'objectif du TD est de permettre à Jenkins de :

- Télécharger le code source de votre application via Git
- Compiler le logiciel via Maven
- Lancer les tests JUnit

Pour cela, vous devez configurer un nouveau « Job » à partir d'un projet Maven.

A titre d'exemple, vous trouverez dans les deux pages suivantes la configuration associée au projet « datanucleus » correspondant à la logique suivante :

- Accès au code source via le dépôt Github :
  - URL du projet : <https://github.com/ktekkal/datanucleus/>
  - URL du dépôt : <https://github.com/ktekkal/datanucleus.git>
  - Branche à utiliser : \*/master
- Construction déclenchée si un nouveau commit est disponible. Une vérification est faite toutes les 5 minutes :
  - Scrutation de l'outil de gestion de version : « H/5 \* \* \* \* »
- Utilisation via Maven :
  - Lancement des tests via la commande « test » qui englobe aussi les commandes « clean », « build » et « enhance ».
  - Attention, le fichier pom.xml a évolué pour inclure la fonction « Enhance » de Datanucleus !

Nom du Maven project	DataNucleus	
Description	<div></div>	
	[Plain text] <a href="#">Prévisualisation</a>	
<input type="checkbox"/> Ce build a des paramètres		
<input checked="" type="checkbox"/> GitHub project		
Project url	https://github.com/ktekkal/datanucleus/	
<input type="checkbox"/> Supprimer les anciens builds		
<input type="checkbox"/> Désactiver le projet (Aucun nouveau build ne sera exécuté jusqu'à ce que le projet soit réactivé.)		
<input type="checkbox"/> Exécuter des builds simultanément si nécessaire		
<b>Options avancées du projet</b>		
<b>Gestion de code source</b>		
<input type="radio"/> Aucune		
<input type="radio"/> CVS		
<input type="radio"/> CVS Projectset		
<input checked="" type="radio"/> Git		
Repositories	Repository URL	https://github.com/ktekkal/datanucleus.git
	Credentials	- none - 
Branches to build	Branch Specifier (blank for 'any')	*/master
Navigateur de la base de code	(Auto)	
Additional Behaviours		

#### Ce qui déclenche le build

- ☐ Lance un build à chaque fois qu'une dépendance SNAPSHOT est construite
- ☐ Build when a change is pushed to GitHub
- ☐ Construire après le build sur d'autres projets
- ☐ Construire périodiquement
- ☒ Scrutation de l'outil de gestion de version

Planning

H/5 \* \* \* \*

Aurait été lancé à samedi 12 mars 2016 22 h 30 CET, prochaine exécution à samedi 12 mars 2016 22 h 35 CET.

Ignore post-commit hooks

☐

#### Pre Steps

Ajouter une étape pré-build ▾

#### Build

POM Racine

pom.xml

Goals et options

test

#### Post Steps

☐ Run only if build succeeds ☐ Run only if build succeeds or is unstable ☒ Run regardless of build result

Should the post-build steps run only for successful builds, etc.

Ajouter une étape post-build ▾

#### Configuration du build

- ☐ Notification par email

#### Actions à la suite du build

Ajouter une action après le build ▾

Sauver

Appliquer

## Question 4 – Utilisation

Une fois correctement configuré, Jenkins vous permettra de surveiller plusieurs aspects de votre gestion de projet :

- Bonne exécution des tests :

The screenshot displays the Jenkins web interface for a Maven project named 'DataNucleus'. On the left, a sidebar contains links: 'Espace de travail' (workspace), 'Changements récents' (recent changes), and 'Derniers résultats des tests (aucune erreur)' (last test results, no error). The main content area is titled 'Résultats des tests' (Test Results) and shows '0 échecs' (0 failures) with a blue progress bar. It indicates '1 tests' (1 test) and 'A pris 2,8 s.' (Took 2.8 s.). A link 'Ajouter une description' (Add description) is visible. Below this, a section titled 'Tous les tests' (All tests) contains a table with the following data:

Package	Durée	Échec	(diff)	Sauté	(diff)	Pass	(diff)	Total	(diff)
<a href="#">datanucleus</a>	2,8 s	0		0		1	+1	1	+1

- Couverture de code :
  - Via le logiciel Emma<sup>1</sup>, vous pouvez lancer la commande « emma:emma » dans Maven qui calculera la couverture de code effectuée par vos tests.
  - Le plugin dédié vous permettra ensuite de visualiser les résultats.
- Qualité de code :
  - Vous pouvez installer le logiciel SonarQube et lancer automatiquement une analyse via le plugin Jenkins dédié.

<sup>1</sup> <http://emma.sourceforge.net/>