

# CHAPITRE 4B - MÉTHODOLOGIE ET GESTION DE VERSIONS

## *Développement agile et SCRUM*

1	Développement agile .....	2
2	Scrum .....	2
2.1	Comment fonctionne Scrum .....	2
2.2	Outils Scrum .....	4
3	Programmation extrême .....	5
4	Exemple de gestion SCRUM avec Jira .....	5

# 1 Développement agile

Le **développement agile** est plus une philosophie, ou une façon de faire, qu'une méthode. Il s'agit de dégager du processus, tout ce qui est superflus et de l'adapter aux besoins du projet. Cette philosophie **prône les éléments suivants** :

- Utilisation d'une **méthode itérative**.
- Les différents aspects de la **méthode** choisie doivent demeurer le plus simple possible et **le plus flexible possible**.
- Faculté de **répondre rapidement aux changements** plutôt que de suivre aveuglement des spécifications rigides.
- La communication en priorité sur les méthodes et les outils.
- **Logiciel fonctionnel** en priorité sur la documentation.
- Collaboration avec le client plutôt que la négociation avec celui-ci.
- Travail en équipes semi-autonomes où tout le monde participe à toutes les phases du développement (analyse, conception, implémentation, tests, ...).
- .... ET SURTOUT GARDER LE TOUT LE PLUS SIMPLE POSSIBLE ...

Pour plus de détails : [Lien Wikipedia](#)

## 2 Scrum

### 2.1 Comment fonctionne Scrum

Scrum est une méthodologie permettant d'encadrer un processus de développement itératif. Il est basé sur les principes suivants :

- Suivre les principes du développement agile.
- Réaliser des sprints (ou itérations) de 2 à 4 semaines, ayant une version fonctionnelle du logiciel de plus en plus complète.
- Gérer le « backlog de produit ».
- Réaliser des mêlées quotidiennes.
- Gérer le « backlog de sprint ».
- Nommer un « scrum master »

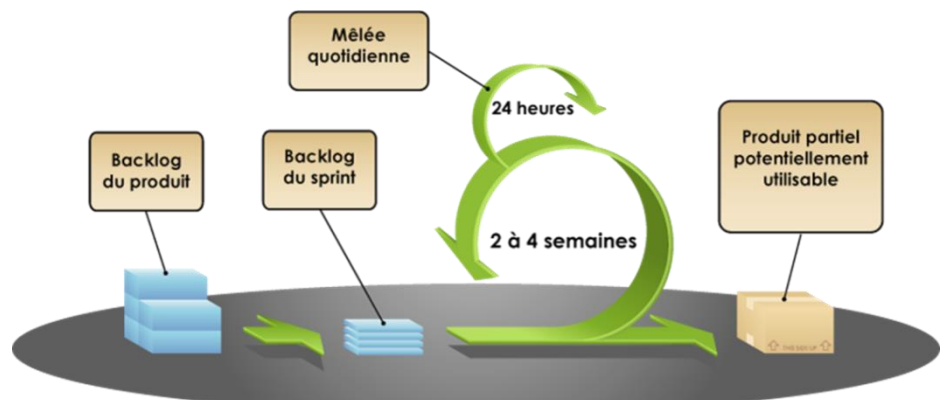


image tirée du site :  
[mountaingoatsoftware.com](http://mountaingoatsoftware.com)

**Au début du projet**, on établit et on note la liste des tâches à accomplir dans le projet, sans qu'elles ne soient dans un ordre précis, sans les affectées à qui que ce soit. Cette liste devient le **backlog du produit**. Bien entendu, tout au long du projet, si on pense à de nouvelles tâches, ou si on se rend compte que des tâches ne sont plus nécessaires, on ajuste le backlog du produit en conséquences.

Backlog 7 issues		
Créer la définition du projet	GHEX-1	1
Créer l'architecture du projet sur GIT	GHEX-7	-
Créer le diagramme entité-association pour la BD	GHEX-2	-
Écrire, tester et implémenter les scripts de la BD	GHEX-5	-
Créer le diagramme de classe du modèle	GHEX-6	-
Écrire la classe Equipe	GHEX-3	-
Écrire le test de la classe Equipe	GHEX-4	-

Pour chaque tâche, on peut définir les paramètres que l'on connaît : description, durée prévue, priorité, points (score à affecter selon la complexité de la tâche), ... Attention de ne pas affecter quelqu'un immédiatement à cette tâche, cela sera fait plus tard.

**Créer la définition du projet**

Description

Dans un fichier word, placez le nom du projet, le nom de tous les membres de l'équipe, une description générale du site, pour chaque type d'acteur (visiteur, utilisateur connecté, ...) une liste d'actions qu'il pourra accomplir (se connecter, changer son mot de passe, ...)

Save Cancel

**Au début de chaque sprint**, on se rencontre pour déterminer les tâches qui devront être réalisées pendant la durée du sprint, et on **attribut** chacune de ces **tâches aux** différent **membres** de l'équipe, de façon équitable (même pointage total pour chacun). **On transfère alors les tâches du backlog du produit vers celui du sprint.**

Projects / G16\_H2020\_Equipe\_X / GHEX board

Backlog

Search [PC] Only My Issues Recently Updated

**GHEX Sprint 1 3 issues** (circled in red)

GHEX Sprint 1 3 issues		
Créer la définition du projet	GHEX-1	1
Créer le diagramme entité-association pour la BD	GHEX-2	-
Créer le diagramme de classe du modèle	GHEX-6	-

+ Create issue

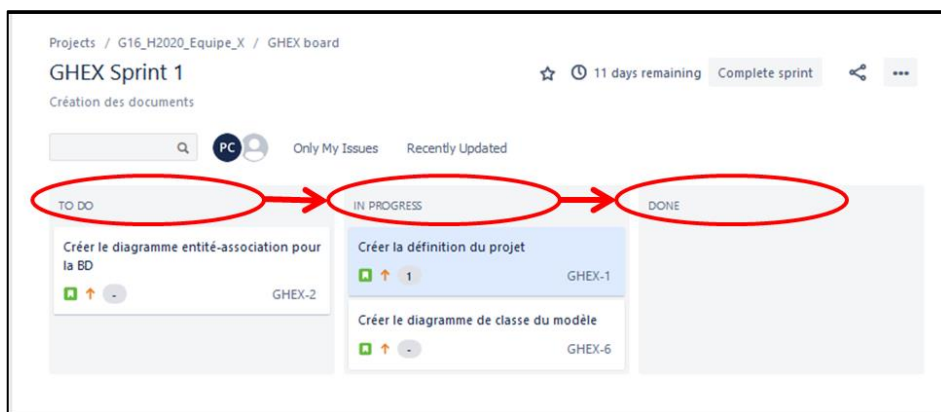
**Backlog 4 issues** (circled in red)

Backlog 4 issues		
Créer l'architecture du projet sur GIT	GHEX-7	-
Écrire, tester et implémenter les scripts de la BD	GHEX-5	-
Écrire la classe Equipe	GHEX-3	-
Écrire le test de la classe Equipe	GHEX-4	-

+ Create issue

**Pendant un sprint**, on **effectue, chaque 1 ou 2 jours**, une rencontre rapide d'une quinzaine de minutes maximum, permettant à chacun des membres de mentionner ce qu'ils ont accompli la veille, de déterminer ce qu'il feront dans la journée et de mentionner les nouveaux problèmes à être adressés. Ces rencontres sont nommées **mêlée quotidienne (scrum : terme tiré du rugby ...)**. On y maintient le statut de la liste des tâches à accomplir pendant le sprint courant. En général ce statut contient 3 à 4 colonnes (À faire, en progrès, à vérifier, complété)

En général, les rencontres se déroulent dans la salle de projet qui contient idéalement un tableau blanc sur lequel sont incluses toutes les informations pertinentes du projet : backlogs, diagrammes essentiels, le synopsis des commandes, ... A défaut d'un tableau blanc, un logiciel



spécialisé ou un éditeur de texte simple peut faire l'affaire. Chaque rencontre est modérée et dirigée par le **scrum master**, qui en général, est une personne différente de la personne en charge du projet.

**À la fin du sprint**, soit 2 à 4 semaines après le début, on doit avoir un **produit partiel, mais totalement fonctionnel**. On fait aussi une rencontre de rétroaction sur le sprint qui vient de se terminer et on en profite pour planifier le prochain.

## 2.2 Outils Scrum

Voici quelques outils pour faire la gestion Scrum avec leur site web :

**Jira** : <https://www.atlassian.com/software/jira> **(Ce que nous allons essayer d'utiliser dans ce cours)**

Trello <https://trello.com/> **(Pas spécifique à Scrum)**

MS Planner Votre compte microsoft office 365 du collège **(Pas spécifique à Scrum)**

Icescrum <https://www.icescrum.com/pricing/>

Taiga <https://taiga.io/>

Scrumpy <http://scrumpy.wikidot.com/>

Hansoft <https://hansoft.com/downloads/>

Scrumblr <http://scrumblr.ca/>

Yodiz <https://www.yodiz.com/pricing.html>

ScrumDo <https://www.scrumdo.com/>

### 3 Programmation extrême

La programmation extrême est simplement une façon de maintenir un haut standard de qualité de codage. Dans cette approche, on jumelle les programmeurs 2 à 2, et chacun est responsable de réviser le contenu et la forme du code de l'autre programmeur. Bien que ceci semble, en effet, un peu « extrême », il est quand même bon de temps à autre de faire des revues de code sur un bout de programme choisi au hasard.

### 4 Exemple de gestion SCRUM avec Jira

Voir l'annexe du chapitre 4B...