Projet dirigé

SCRUM, XP, KANBAN

Saliha Yacoub

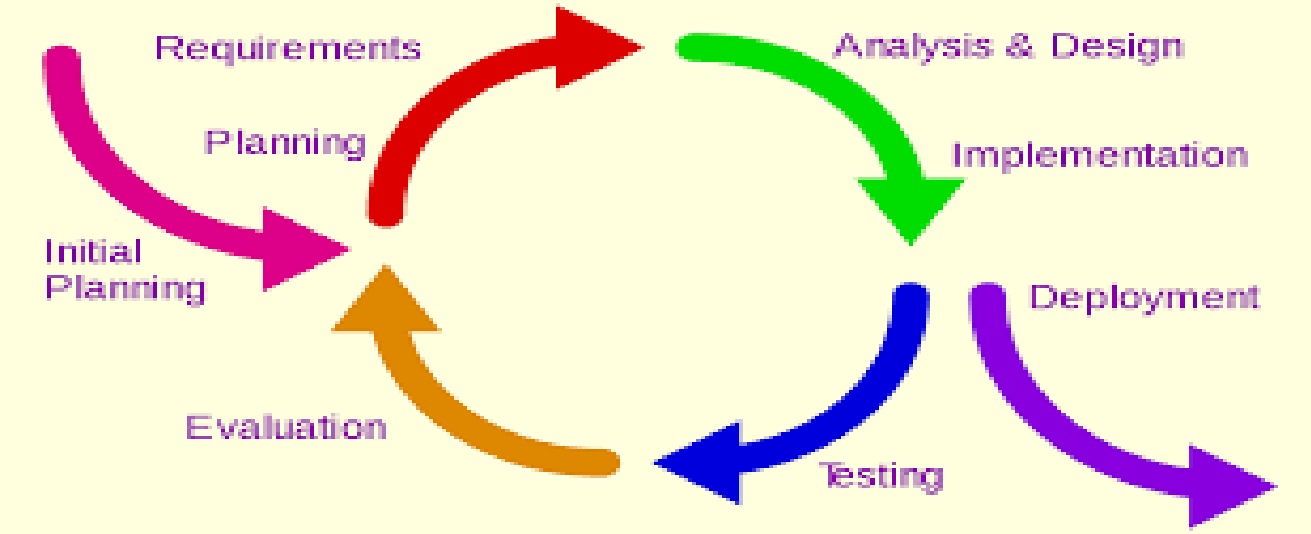
Introduction

Plan de séance

- Retour sur les user-stories
- Rappels
  - Approches agiles
  - Valeurs agiles
  - Manifeste agile.
- L'approche SCRUM
- Cycle de développement selon SCRUM
- L'équipe SCRUM
- Le backlog du produit.

Rappels

L'approche agile



Rappels

Avantages:

- Itératif et incrémental pour vrai pas uniquement dans le principe.
- Livraison tôt et souvent
- Implication du client: les besoins évoluent, feedback tôt.
- Les itérations (sprint) sont courtes.
- La documentation n'est pas lourde.
- Le rythme de travail est constant

Rappels

Manifeste agile: 4 valeurs et 12 principes

Valeurs agiles

1. Les personnes et leurs interactions sont plus importantes que le processus et les outils
2. Un logiciel qui fonctionne prime sur la documentation
3. La collaboration avec les clients est préférable à la négociation contractuelle
4. La réponse au changement passe avant le suivi d'un plan.

Rappels

Les principes :

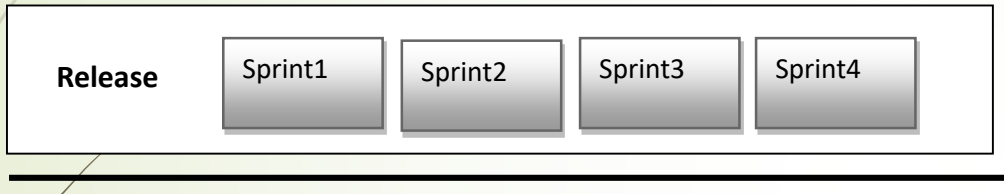
Le Manifeste annonce douze (12) principes, qui ne définissent pas une méthode agile.

1. Satisfaire le client en livrant tôt et régulièrement des logiciels utiles qui offre une véritable valeur ajoutée.
2. Accepter les changements même tard dans le développement.
3. Livrer fréquemment une application qui fonctionne.
4. Collaborer quotidiennement entre clients et développeurs.
5. Bâtir le projet autour de personnes motivées en leur fournissant environnement et support, et en leur faisant confiance.
6. Communiquer par des conversations face à face.
7. Mesurer la progression avec le logiciel qui fonctionne.
8. Garder un rythme de travail durable.
9. Rechercher l'excellence technique et la qualité de conception
10. Laisser l'équipe s'auto-organiser.
11. Rechercher la simplicité.
12. À intervalle régulier, réfléchir aux moyens de devenir plus efficace.

SCRUM

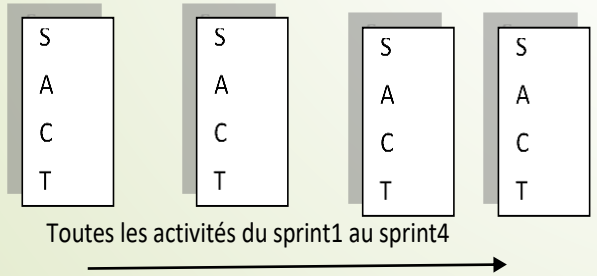
- SCRUM signifie mêlée en rugby. Scrum utilise les valeurs et l'esprit du rugby et les adapte aux projets de développement
- SCRUM est l'approche agile la plus populaire
- Définitions:
  - Un **sprint** est le terme utilisé dans SCRUM pour désigner une itération. Dans le langage SCRUM un sprint est un bloc de temps fixée aboutissant à un incrément de produit potentiellement livrable.
  - Une **version (**release**)** est produite par une série d'itérations d'un mois, parfois même de 15 jours, appelés **sprint**. Le contenu d'un *sprint* est défini par l'équipe avec le Product Owner, en tenant compte des priorités et de la capacité de l'équipe.
  - Une release : (selon le dictionnaire du jargon informatique) : Version d'un système, par exemple un logiciel, effectivement publiée, donc lâchée dans la nature.
  - **Backlog** du produit: liste unique des user-stories **classées par priorité.**

SCRUM



- Spécification fonctionnelle (Requierements ou étude des besoins fonctionnels)
- Architecture (conception)
- Codage (et test unitaire)
- Test (test d'intégration).

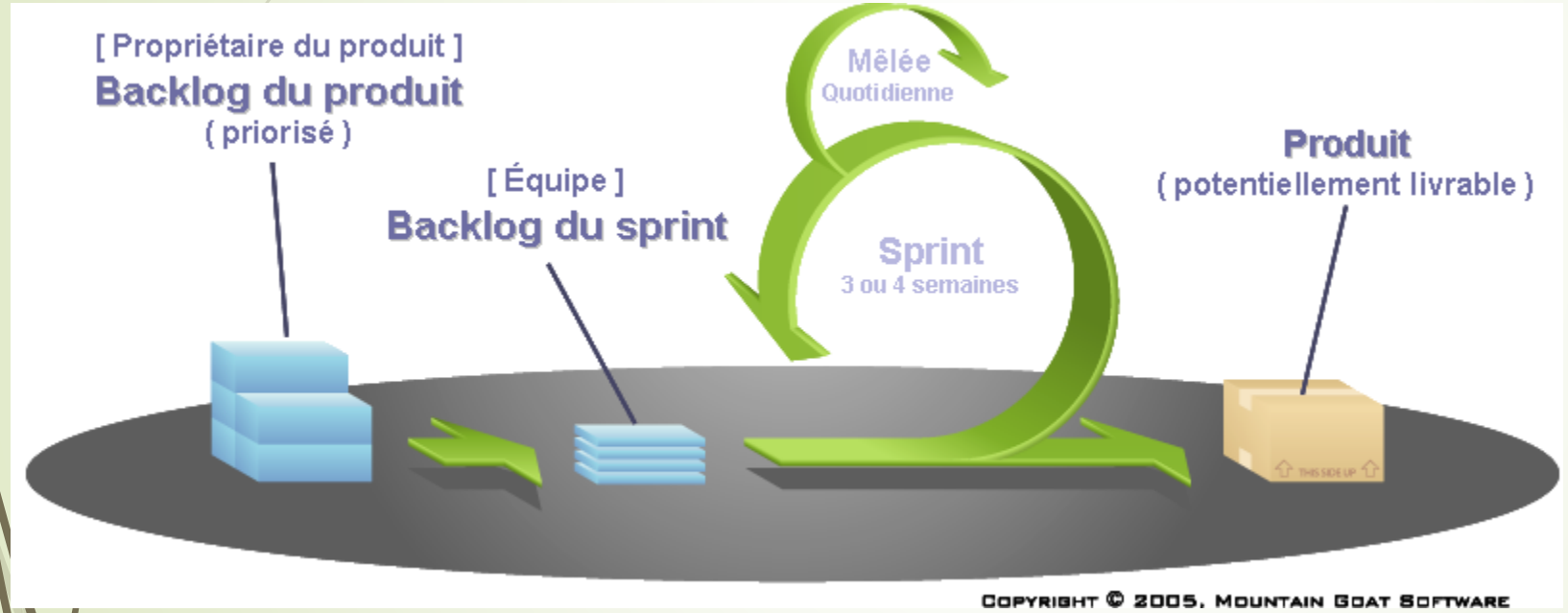
Cycle SCRUM : les sprints et leurs phases se déroulent en parallèle.



SCRUM

- Il n'y a pas de chevauchement. Les *sprints* s'enchaînent sans délais. Un *sprint* n'est commencé que si le précédent est terminé. Le nouveau *sprint* démarre immédiatement après le précédent
- À la fin de chaque *sprint,* l'équipe obtient un **produit partiel**, (qui s'enrichit d'un nouveau incrément à chaque *sprint*) **qui fonctionne**. Il est potentiellement livrable. L' évaluation et le *feedback* récolté permettent d'ajuster le *backlog* pour le *sprint* suivant.
- La date fine d'un *sprint* est fixée au début de celui-ci. Elle ne change pas même si l'équipe ne réalise pas tout ce qu'elle pensait faire
- Arrêter un sprint plutôt que de l'étendre.

SCRUM


[Propriétaire du produit]
Backlog du produit
(priorisé)
[Équipe]
Backlog du sprint
Mêlée
Quotidienne
Sprint
3 ou 4 semaines
Produit
(potentiellement livrable)
Copyright © 2005, Mountain Goat Software

SCRUM

- Pour une équipe, une release dure environs 3 mois avec des sprints de deux à trois semaines. Ce qui permet d'avoir de quatre à 6 sprints dans une release.
- Il n'y a pas de chevauchement entre les sprints. Ils s'enchaînent sans délais.
- La fin d'un sprint peu être un produit potentiellement livrable.
- Le résultat d'une *release* est le produit livrable fourni à ses utilisateurs. La façon dont il est fourni dépend de son déploiement.
- Souvent, le jalon majeur que représente la *release* correspond à une annonce marketing.

SCRUM

SCRUM…. Développement léger

- **Création du *backlog* (un to do list) de toutes les fonctionnalités d'un projet. Les éléments du backlog doivent être estimé (temps de réalisation) et priorisé.**
- **Création d'un sprint backlog : fonctionnalités à compléter durant la durée du sprint (15 jours ou un mois)**
- **Effectuer des rencontres quotidiennes durant le sprint : des mêlées quotidiennes (*scrums*).**
- **Finalisation du sprint avec démonstration et évaluation**

SCRUM

Rôles et responsabilités:

Le Product Owner : (PO) : Le PO est l'expert du domaine (niveau métier). En tant que représentant des clients et utilisateurs, il est responsable de définir les caractéristiques du produit développé par l'équipe, en termes de :

- Fonctionnalités offertes. Plus précisément, il identifie chaque exigence que doit satisfaire le produit et la collecte comme élément du backlog de produit. Il est souhaitable d'inclure les tests d'acceptation.
- Priorité. C'est lui qui définit l'ordre dans lequel ces éléments seront développés en fonction de la valeur qu'ils apportent aux clients et utilisateurs. Cela permet d'alimenter l'équipe avec un backlog de produit prêt pour la planification des sprints
- But. C'est lui qui définit l'objectif d'une release et qui prend les décisions concernant le planning de la release.
- Son implication dans le projet est capitale pour la réussite de celui-ci.

SCRUM

Rôles et responsabilités:

Le SCRUM Master :C'est le coach de l'équipe (ancien chef de projet). Il a pour rôle:

- Dans le cadre du développement d'un produit, d'aider l'équipe à travailler de façon autonome et à s'améliorer constamment. Il est le garant de l'application du processus, Scrum en l'occurrence.
- S'assurer que l'équipe bénéficie des meilleures conditions pour accomplir les tâches
- Éliminer les obstacles : prendre en compte les problèmes qui surviennent à tout moment sur un projet pour les éliminer au plus vite, en évitant qu'ils ralentissent l'équipe. Il protège l'équipe des interférences extérieures.
- Faire en sorte que l'équipe reste concentrée sur le véritable objectif du projet, qui est de réaliser les éléments du Backlog en collaboration étroite avec le Product Owner , et soit productive. Il s'assure que chacun participe pleinement aux travaux de l'équipe.
- Organise et anime les réunions qui constituent le cérémonial.

SCRUM

Le backlog … un peu plus:

- Le cœur de SCRUM est le *Product Backlog ou* ***backlog du produit*** qui représente la liste des requis **priorisés. C'est un tableau de user-stories**
- SCRUM débute avec un produit backlog priorisé**.**
- Chaque entrée du backlog représente une user-story décrite dans le langage et la terminologie du client du client. Chaque entrée sert à finaliser ce que le client désire obtenir.
- Le Backlog du doit être un document partagé, détenu par le PO.
- Le backlog du produit est vivant.
- Garder le **produi**t **backlog** niveau métier. Il doit focaliser sur les buts métier et non les technologies.

SCRUM

- **Ce qu'il faut faire :**
  - Cultiver le backlog : la backlog évolue dans le temps, il faudra le mettre à jour.
  - Partager le backlog avec toute l'équipe
  - Surveiller la taille du backlog : ne pas avoir plus de 150 éléments à faire dans le backlog
- **À Éviter :**
  - D'avoir plusieurs backlog pour le même de produit
  - De ne pas avoir de backlog
  - De confondre le backlog de produit avec le backlog de sprint

SCRUM

- **Prioriser les éléments du backlog de produit → Questions à se poser:**
  - **Quel est le bénéfice financier ou la valeur ajoutée à développer cette fonctionnalité ?**
  - **Quel est le coût de développement ? De maintenance …**
  - **L'implémentation de la fonctionnalité nous permet-elle d'apprendre ?de développer de nouvelles compétences ?**
  - **La fonctionnalité va-t-elle améliorer la productivité de mon personnel ?**
  - **Quel est le préjudice si cette fonctionnalité n'est pas implémentée ?**

- **→tenir compte de:**
  - **Le bénéfice financier attendu ou valeur ajoutée, c'est l'élément le plus important à considérer.**
  - **Le coût de développement**
  - **L'opportunité d'apprentissage pour l'équipe**
  - **Le risque de développement, le développement de cette fonctionnalité nous expose t-elle à d'avantage de risques.**

SCRUM

Prioriser les éléments du backlog de produit: priority Poker

- Chaque participant reçoit un lot de neuf cartes numérotées de 1 à 9
- Chaque story est étudiée successivement
- Le premier vote porte sur l'intérêt d'avoir le feature. Chaque participant vote avec une carte. On fait le total des points.
- Le deuxième vote, porte sur la pénalité de ne pas avoir le feature dans le produit. On vote également de 1 à 9.
- On définit l'importance des deux votes. On peut donner 4 au premier et 1 au deuxième.
- En faisant la somme des deux votes pondérés on obtient l'utilité de l'élément. Les stories les plus utiles sont les plus prioritaires.

SCRUM

Prioriser les éléments du backlog de produit: MoSCoW

La technique utilisée pour prioriser les besoins dans un contexte itératif est celle de MoSCoW. L'avantage de la méthode *MoSCoW* réside dans la signification de l'acronyme, qui est plus compréhensible que d'autres techniques de priorisation comme élevé/moyen/faible

- **M** pour **Must Have** : ***DOIT*** être fait. L'exigence est essentielle. Si elle n'est pas faîte le projet échoue. On peut dire également priorité haute.
- **S** pour Should **Have** : Il s'agit d'une exigence essentielle, qu'il faut faire dans la mesure du possible (***DEVRAIT***). Mais si elle n'est pas faîte, on peut la contourner et la livrer plus tard.
- **C** pour **Could Have**: Il s'agit d'une exigence souhaitable. Elle ***POURRAIT être*** faîte dans la mesure où elle n'a pas d'impact sur les autres tâches
- **W** pour **Won't Have** Il s'agit d'une exigence «Luxe». ***NE SERA PAS*** faîte cette fois mais plus tard, mais intéressante et à garder pour la prochaine version

SCRUM

Comment estimer une story ?

- **Demander à un expert:**

c'est bien mais le problème c'est qu'il va se basé sur son expérience, sa capacité. Dans les approches agiles, le développement se fait en équipe, il est difficile parfois d'avoir un expert dans l'équipe.

Demander à Usain Bolt le temps à mettre pour courir le 100 m ?

- **Analogie:**

Par analogie avec ce que vous connaissez : c'est un peu ce que nous faisons d'habitudes. On dira qu'une story est un peu plus longue que la story précédente. Ou qu'une story A prendra deux fois plus de temps qu'une story B..

- **Découper et détailler.**

C'est plus simple de dire qu'une fonctionnalité ou une petite story sera implémentée dans 3 jours que de dire que la story sera implémentée dans 102 jours ??

SCRUM

Comment estimer une story ?

Estimer les stories: le planning Poker

Pour commencer la séance de planification, il suffit de choisir une story connue de TOUS , (baseline) pour laquelle l'équipe décide en commun de lui fixer une valeur. Et, il est préférable e choisir une story de taille moyenne (3 ou 5) pour laisser une marge vers le bas et vers le haut

1. Le PO présente la story.
2. Les membre de l'équipe posent des questions pour clarifier la story.
3. Tous les participants présentent en même temps la carte choisie pour l'estimation
4. L'équipe discute des différences éventuelles entre les estimations.
5. On recommence jusqu'à une convergence des estimations.
6. On passe à la prochaine story.

Attention: complexité ne veut pas dire plus de points. (plus long)

SCRUM

Mike Cohen(un des contributeur à SCRUM) à propos de l'estimation des user-stories :

- Prenez une équipe composée d'un chirurgien du cerveau et d'un enfant. Le backlog de produit contient les stories suivantes :
  - En tant qu'enfant, je dois coller 1000 timbres sur 1000 enveloppes
  - En tant que chirurgien, je dois faire une opération simple au cerveau.
- Quelle est la story qui nécessite le plus de points ?

Il est probable que les deux stories se terminent en même temps. Que les deux stories aient besoin d'un même nombre de points pour se terminer.

Mais, si la question est : Quelle est la story la plus complexe? La réponse et claire : celle du chirurgien

SCRUM

SCRUM

Application:

- Prioriser, puis estimer le backlog de la bibliothèque
- Prioriser puis estimer le backlog de Trivial Pursuit

SCRUM

Plan de séance

- Planifier son projet avec SCRUM
  - Déterminer la date de fin de Release
  - Estimer les stories
  - Déterminer la durée d'un sprint
  - La vélocité de l'équipe
  - Le plan de Release.
- Application

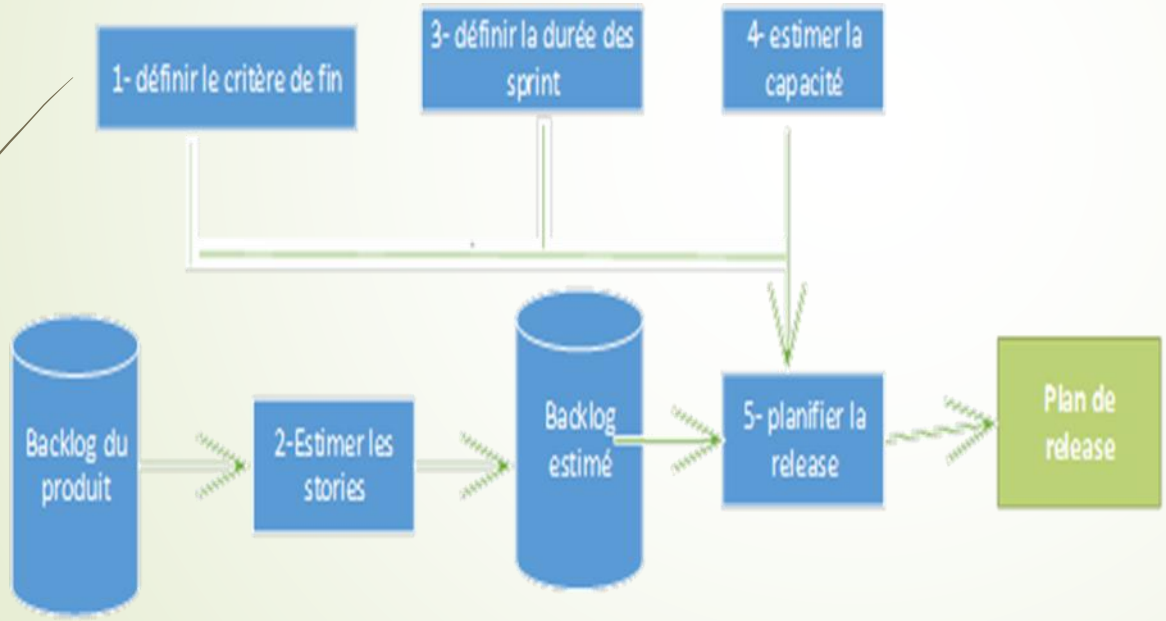
SCRUM

Planifier son projet avec SCRUM

- Une release est une séquence de sprints,
  - mais quand finit-elle?
  - Comment la planifier pour livrer un produit fini et à temps ?
  - Combien de sprints la release contiendra-t-elle ?
  - Quelle est la durée d'un sprint ?

SCRUM

On procède par étapes.



SCRUM

Étape 1: définir la fin de la release :

- Quand le backlog est vide. Mais le backlog est vivant
- Fixer les éléments à livrer à l'intérieur du backlog pour la Release courante. Les autres éléments seront livrés dans une nouvelle version: C'est ce qu'on appelle une Release à paramètres fixés. (le backlog étant priorisé→on livre les éléments prioritaires en premier
- Fixer la date fin à l'avance: cette façon de faire permet de fixer les éléments à livrer compte tenu de la date de livraison.

SCRUM

- Avantage d'une Release à date fixée:
  - Elle donne un objectif précis pas lointain ---> motivation de l'équipe
  - Elle impose au Product Owner d'avoir une réflexion poussée sur les priorités des éléments du backlog
  - Des éléments du backlog ayant peu d'intérêt ne seront pas développés.
  - On passe moins de temps à planifier puisque la date de livraison est connue.

- Une release à date fixée et à durée uniforme (exemple une durée de 3 mois) est la formule la plus facile à mettre en œuvre.

SCRUM

Étape 2:Estimer les stories

- L'estimation se fait en équipe: exemple du planning Poker

SCRUM

Étape 3:Définir la durée de sprint

- Lorsqu'on lance utilise une approche agile pour le développement logiciel, la question fondamentale est de déterminer la durée d'un sprint. Chaque projet est différent. Il n'y a donc pas de réponse universelle.
- Pour SCRUM, la pratique est de faire des sprints de maximum UN MOIS. Dans la plupart des cas, des sprints de 2 à 3 semaines sont recommandés.
- Un sprint trop long risque de démotiver l'équipe
- Un sprint trop court… risque de mettre plus de stress su l'équipe.

SCRUM

Pour définir la durée d'un sprint, on tient compte:

- L'implication des clients et des utilisateurs : Il faut tenir compte de leur disponibilité à utiliser les versions partielles produites à la fin de chaque sprint
- Le coût supplémentaire engendré par la préparation du sprint : Un sprint ajoute du travail supplémentaire pour préparer le produit partiel. Faire les tests de non régression, préparer la démonstration pour la revue de sprint
- La taille de l'équipe : plus il y a du monde dans l'équipe, plus il faudra du temps pour se synchroniser.
- La date de fin de la release : idéalement une release comporte au moins quatre sprints pour profiter des bénéfices des avantages de l'itératif
- La stabilité de l'architecture: il est plus facile d'obtenir un produit qui fonctionne si l'architecture est stable. (conception globale)

SCRUM

Étape 4:Estimer la capacité de l'équipe

- Définitions : La vélocité.
- La Vélocité: la vélocité de l'équipe **mesure** la partie du backlog **réalisée** par l'équipe à l'intérieur d'un sprint. À la fin d'un sprint, on mesure ce que l'équipe a été capable de réaliser.
  - La vélocité se calcule à la fin d'un sprint après la démonstration et lors de lors de la revue de sprint
  - La vélocité est une mesure de l'équipe et non d'une personne individuelle.
- La capacité: La capacité est une prévision de ce que l'équipe est capable de faire en tenant compte de sa vélocité

SCRUM

lorsque l'équipe n'a jamais travaillé ensemble, alors on pourrait simuler un sprint fictif pour déterminer la vélocité de l'équipe. On pourrait également calculer sa capacité

Exemple de prévision: Trois stories sont étudiées qui avaient été estimées à 3, 2 et 5 points. Les tâches identifiées pour ces stories sont estimées à 60 heures.

- L'équipe dispose de 300 heures pour le sprint
- En 60 heures, l'équipe pourrait réaliser 10 points. (3 +2 +5)
- En 300 heures (sprint) l'équipe pourrait réaliser (300/ 60) * 10 points.
- La capacité de l'équipe serait 50 points (estimation)

SCRUM

Étape 5:Produire un plan de Release

- Après avoir fait les étapes précédentes produire un plan de release devient facile, un jeu d'enfant. On procède comme suit :
  - On prend le backlog du produit priorisé et estimé.
  - On commence par le premier sprint de la release. On y associe les stories en commençant par les prioritaires
  - On continue dans ce sprint en additionnant la taille en points de stories jusqu'à atteindre la capacité de l'équipe
  - Quand on y arrive, on passe au sprint suivant.
- Et Si on ne tombe pas exactement sur la capacité de l'équipe à l'intérieur d'un sprint ?
  - On va légèrement au dessus , ou en dessous ou une story un peu moins prioritaire et qui rentre dans le sprint.
- **Garder du « lousse »**

SCRUM

Le sprint Zéro

Le développement agile a besoin d'un sprint de départ, qui ne se termine pas nécessairement par une livraison. D'une durée variable sert à mettre le projet sur de bons rails et d'apprendre à l'équipe de travailler ensemble. Concrètement, ce que l'on doit faire durant le sprint Zéro:

- Partager une vision claire du projet
- Préparer l'environnement de développement
- Produire un backlog du produit estimé et priorisé
- Roder l'équipe sur le backlog initial
- Définir la posture ergonomique de l'interface.
- Déterminer un plan de Release.
- Selon le contexte, travailler sur l'architecture, travailler sur la BD (Conception globale)
- S'offrir une belle rétrospective.

SCRUM

XP et Kanban

L’approche XP

- Adaptée aux équipes réduites avec des besoins changeants. Elle pousse à l'extrême des principes simples. Son but principal est de réduire les coûts du changement.

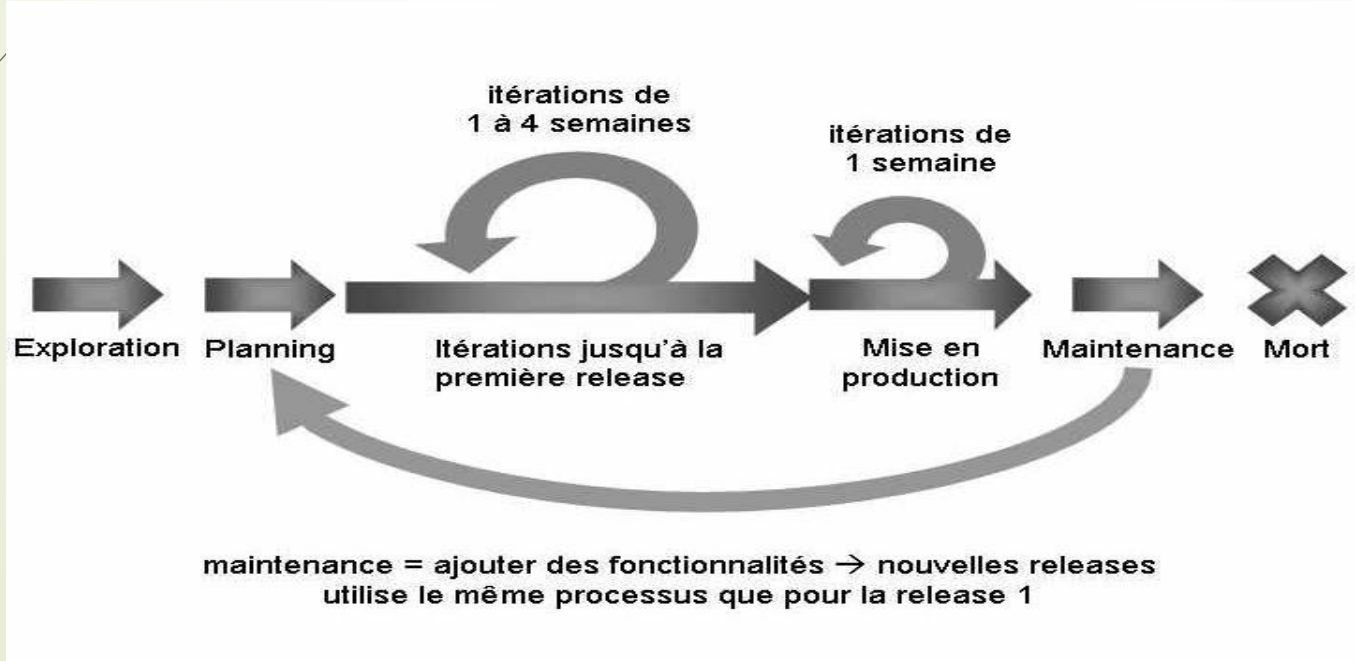
- Les principes de cette méthode ne sont pas nouveaux : ils existent dans l'industrie du logiciel depuis des dizaines d'années et dans les méthodes de management depuis encore plus longtemps. L'originalité de la méthode est de les pousser à l'extrême :

XP et Kanban

- Planning Game
  - phase d'exploration : écriture des besoins sous forme de "user stories" et estimation de leur durée
  - phase d'engagement : classement des user stories par ordre de priorité.
  - phase de direction : mise à jour du planning
- Petites releases
- Utilisation de métaphores pour décrire l'architecture du système
- Conception simple : toujours développer la solution le plus simple possible
- Tests (unitaires et fonctionnels)
- Refactoring du code : retravailler le code pour le rendre plus lisible et plus robuste
- Programmation en binôme
- Propriété collective du code
- Intégration continue (plusieurs fois par jour)
- Pas de surcharge de travail : ne pas dépasser 40 heures de travail par semaine
- Client sur site : présence sur site d'une personne minimum à temps plein pendant toute la durée du projet
- Standards de code (normes de nommage et de programmation)

XP et Kanban

- La première itération peut être longue
- Les itération suivantes peuvent être plus courtes.



XP et Kanban

L'Approche Kanban

- À la base, Kanban a été développé à la fin des années 1940 chez Toyota afin d'optimiser l'efficacité de ses chaînes de montages. Basé sur la méthode de remplissage des supermarchés, cette méthode visait à la base à optimiser les niveaux de stocks pour les garder aux niveaux nécessaires et éviter tout genre de « sur-stockage », sans pour autant manquer de pièces.

- Il faut savoir que la méthode Kanban pour le développement logiciel est basée sur les principes et les valeurs Agile. Quels avantages cela nous apporte-il?

XP et Kanban

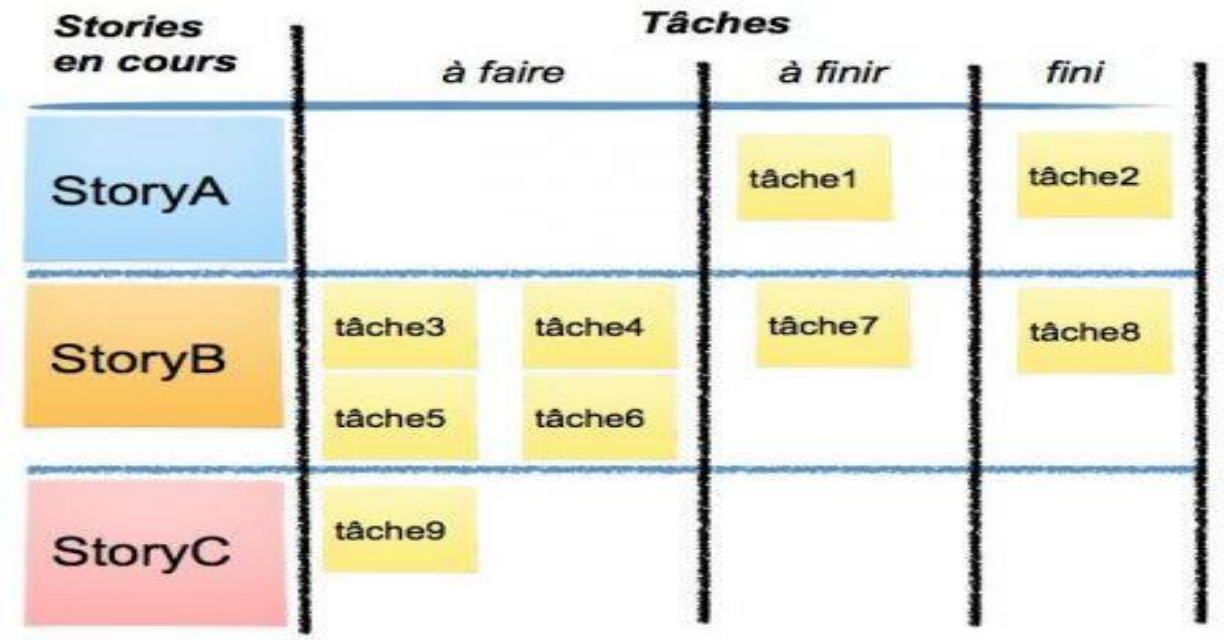
- **Planification flexible** : Il faut savoir accepter les changements en tout temps. Le client change d'idée durant le développement logiciel et il faut savoir intégrer ces changements tant que cela n'a pas d'impact sur les tâches en cours. De plus, le Product Owner peut se permettre de changer l'ordre des tâches futurs en toute temps sans jamais impacter le travail en cours.
- **Durée de cycle raccourcies :** La durée du cycle est une des métriques les plus importantes dans la méthode Kanban. Elle représente la durée pour qu'une équipe de travail passe à travers son « workflow » ou sa charge de travail avant la prochaine livraison. Les équipes suivent de bonnes pratiques de bases lors de la production et ensuite applique le « code review » pour s'assurer que la qualité ne sera pas impactée par ce laps de temps raccourcie

XP et Kanban

- **Réduction des goulots d'étranglement :** Avec l'utilisation du Kanban Board (nous le verrons plus tard), nous pouvons suivre l'état de chacune des tâches en état. Par exemple : « À faire (Backlog) », « En cours », « En révision/Test » et « Complété ». La méthode Kanban propose de limiter le nombre maximal de tâches pouvant être dans un certain état à tout moment. Par exemple : On peut décider de mettre un maximum de 4 tâches en cours et 2 tâches en révision en même temps afin d'éviter un bouchon et que certains employés n'aient rien à faire.
- **Métriques visuelles :** L'une des valeurs de Kanban est la constante amélioration de l'efficacité de l'équipe. Pour y arriver Kanban utilise plusieurs métriques visuelles afin de s'assurer de l'amélioration constante des processus: Tableau des tâches verticales avec des post-it
- **Livraison continue :** Kanban est une méthode de travail qui prône l'intégration continue : C'est-à-dire, les builds automatisés et les tests incrémentiels tout au long du développement. Ceci nous permettra de livrer des fonctionnalités au client régulièrement.

XP et Kanban

- Le Task Board: simple et visuel, destiné principalement aux membre de l'équipe le Task Board représente l'avantage de fournir un moyen efficace de s'organiser et de voir en un clin d'oeil le reste à faire (Utilisé par SCRUM également)



XP et Kanban

Kanban VS SCRUM

- Le tableau des tâches verticales: Pour SCRUM, il est important que toutes les tâches de la première colonnes se retrouvent dans la dernière colonne à la fin d'u sprint. Pour Kanban, le nombre de tâches dans chaque colonne doit être contrôlé.
- Plus de planification dans SCRUM: Sprint, Release. Pour SCRUM la notion de sprint (itération) est importante. Pour Kanban même si le processus est itératif , on s'attend à ce que l'amélioration se produit de manière évolutive. Au début, on s'entend sur une certaine cadence et on tente de l'améliorer
- Les responsabilités: Dans SCRUM, le PO et le Scrum Master ont un rôle important. El l'équipe joue aussi un grand rôle. On peut donc identifier trois rôles importants. Pour Kanban… l'équipe a un rôle est de manières implicite on s'entend à ce qu'un membre de l'équipe joue le rôle de chef de projet.
- Évidement … il y a beaucoup de similitudes entre les deux approches. Tous deux accordent une grande importance à l'amélioration continue, à l'optimisation du travail et au processus. L'équipe est au centre du développement. Toutes les tâches à réaliser sont visibles de tous les membres de l'équipe.