

Formation Agile: “SCRUM PRODUCT OWNER PSPO”

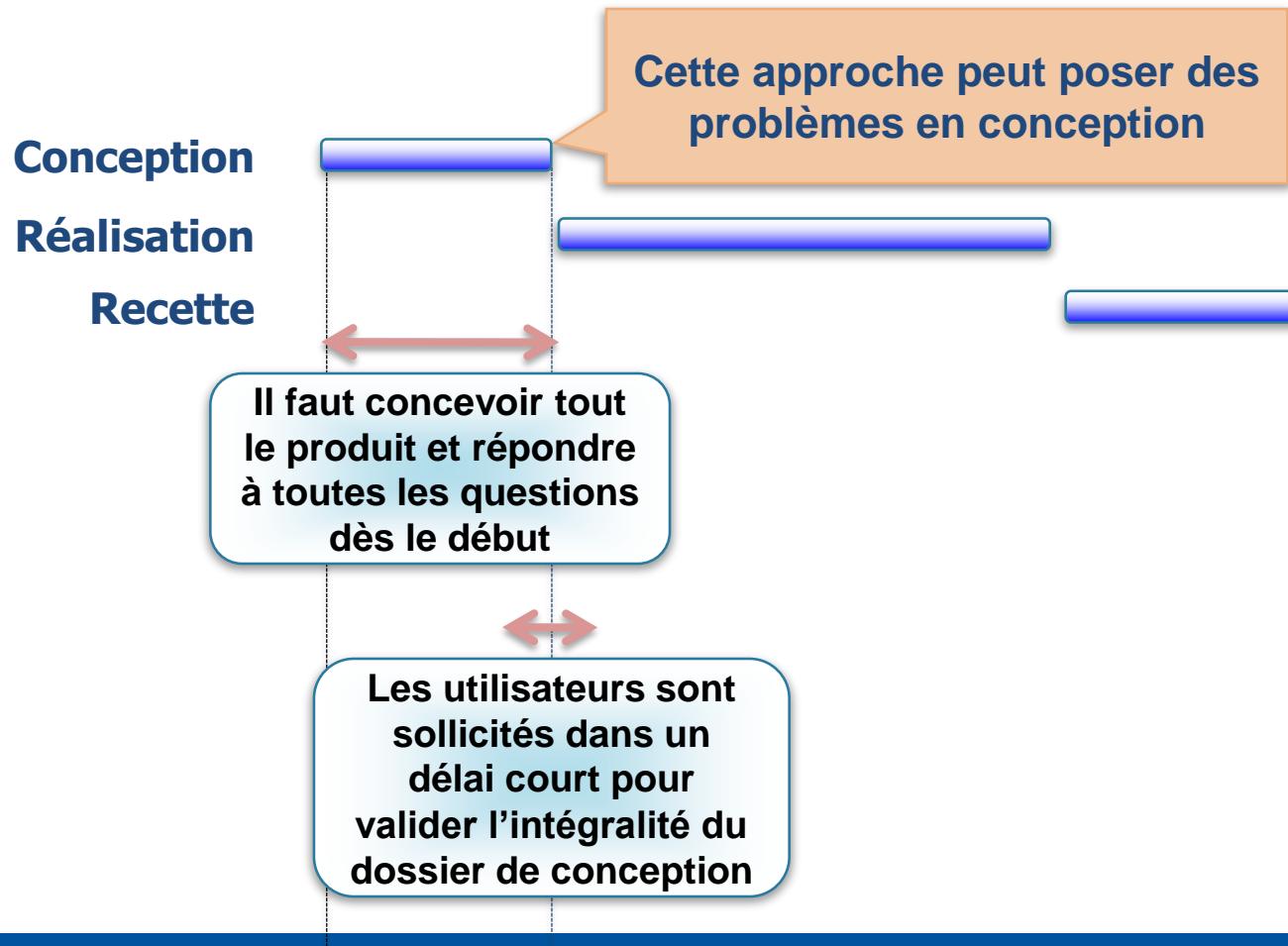


Plan

- Vérification des acquis
 - Rappels sur la démarche Agile.
 - Contexte et origines.
 - Le Manifeste Agile.
 - Principes de SCRUM.
 - Les différents rôles (Scrum Master, Equipe de développement, Product Owner).
 - Les cérémonies (Sprint Review, Sprint Retrospective, Daily
- Scrum).
- Gestion de produit
 - Développement piloté par la valeur
 - Plan de livraison
- Gestion des exigences
- Planification des releases
- Atelier : Scrum@Play
- Préparation à l'examen PSPO 1

L'approche classique

L'approche classique se décompose en trois phases :



L'approche classique

Conception

Réalisation

Recette

Cette approche peut poser des problèmes en réalisation

Les travaux sont fortement parallélisés et sont souvent bloqués par des questions fonctionnelles

Toutes les fonctionnalités sont intégralement développées sans arbitrages sur le R.O.I.

Les utilisateurs sont sollicités dans des démonstrations trop longues leur montrant de nombreuses fonctionnalités

L'approche classique

L'approche classique se décompose en trois phases :

Conception

Réalisation

Recette

Cette approche peut poser des problèmes en recette

Les utilisateurs sont fortement sollicités sur une période très courte

Certains points soulevés en recette peuvent remettre en cause profondément d'autres fonctionnalités

L'approche classique

- En synthèse les principaux inconvénients de la méthode classique sont :
- Il est très difficile de faire une conception exhaustive au démarrage du projet.
- La phase de réalisation est basée sur cette exhaustivité et elle n'intègre pas la notion de complément de conception ni de variation du besoin.
- Les erreurs de conception ou de programmation ou variation du besoin sont détectées au dernier moment ce qui aggrave leurs conséquences.
- Il n'est pas efficace de solliciter les utilisateurs de manière intense sur des périodes courtes.
- L'application opérationnelle n'est disponible qu'à la fin du projet.
- Il est difficile de communiquer directement avec les utilisateurs

L'approche Agile



- Approche réactive et itérative d'organisation de travail
- Focalisée sur la fonctionnalité et satisfaction client
- Construit en adéquation avec les capacités et limites humaines

Manifeste Agile

www.agilemanifesto.org



Manifeste pour le développement Agile de logiciels

Nous découvrons comment mieux développer des logiciels
par la pratique et en aidant les autres à le faire.

Ces expériences nous ont amenés à valoriser :

Les individus et leurs interactions plus que les processus et les outils

Des logiciels opérationnels plus qu'une documentation exhaustive

La collaboration avec les clients plus que la négociation contractuelle

L'adaptation au changement plus que le suivi d'un plan

**Nous reconnaissons la valeur des seconds éléments,
mais privilégiions les premiers.**

Le Manifeste Agile – une déclaration de valeurs



Individus et interactions

plutôt
que

Processus et outils

Un logiciel qui fonctionne

plutôt
que

Documentation complète

Collaboration avec le client

plutôt
que

Négociation à partir d'un contrat

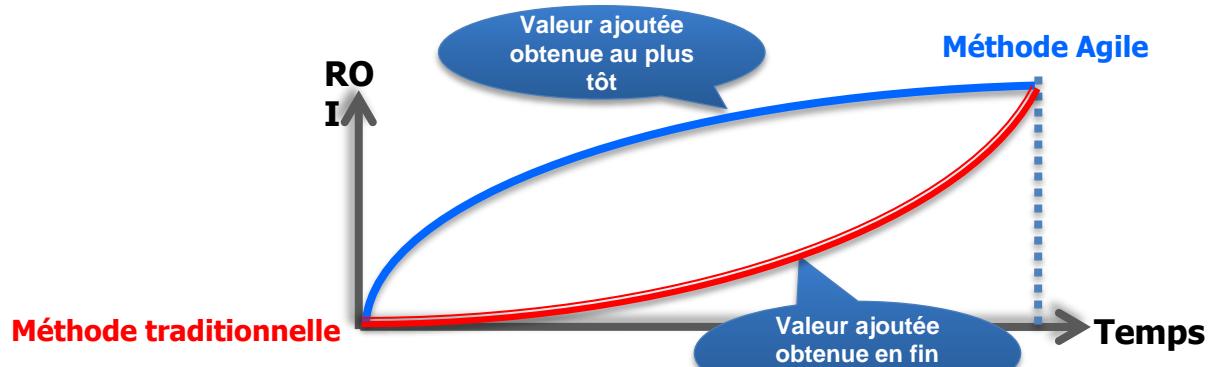
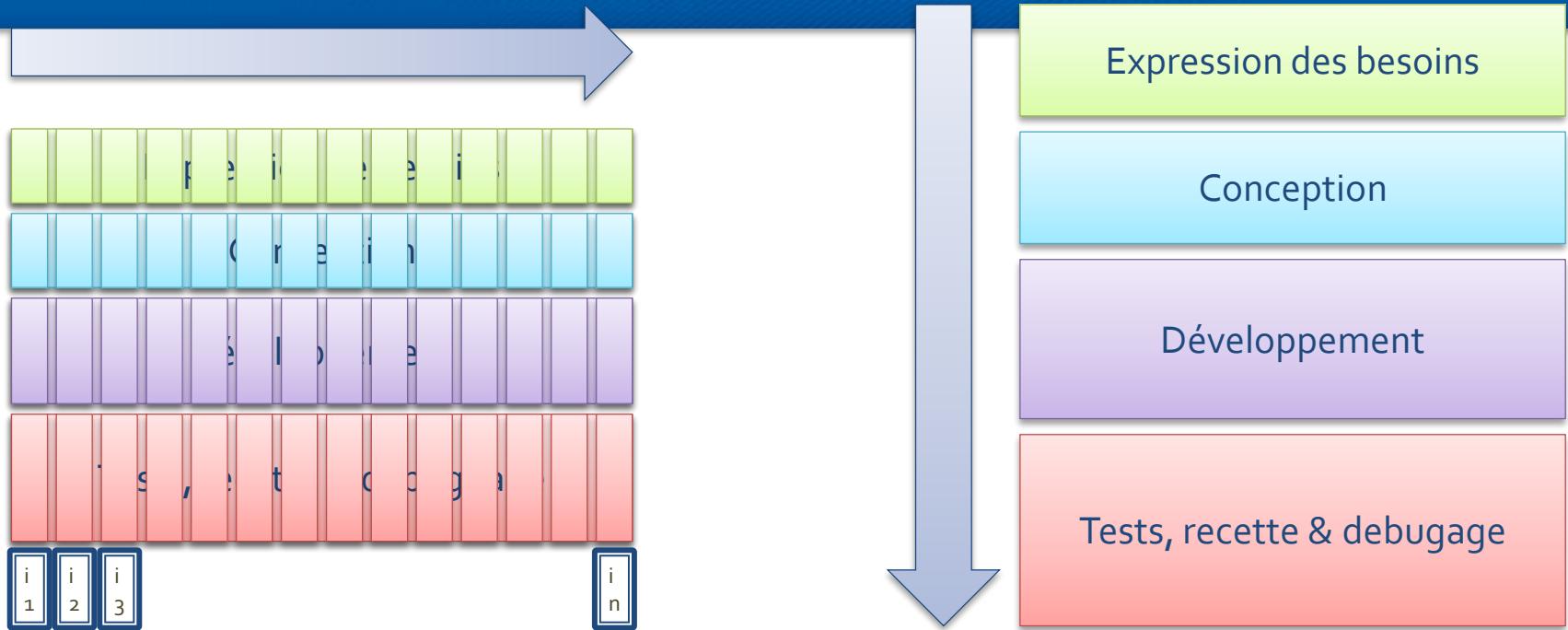
S'adapter au changement

plutôt
que

Suivre un planning

Source: www.agilemanifesto.org

Agile vs Itératif



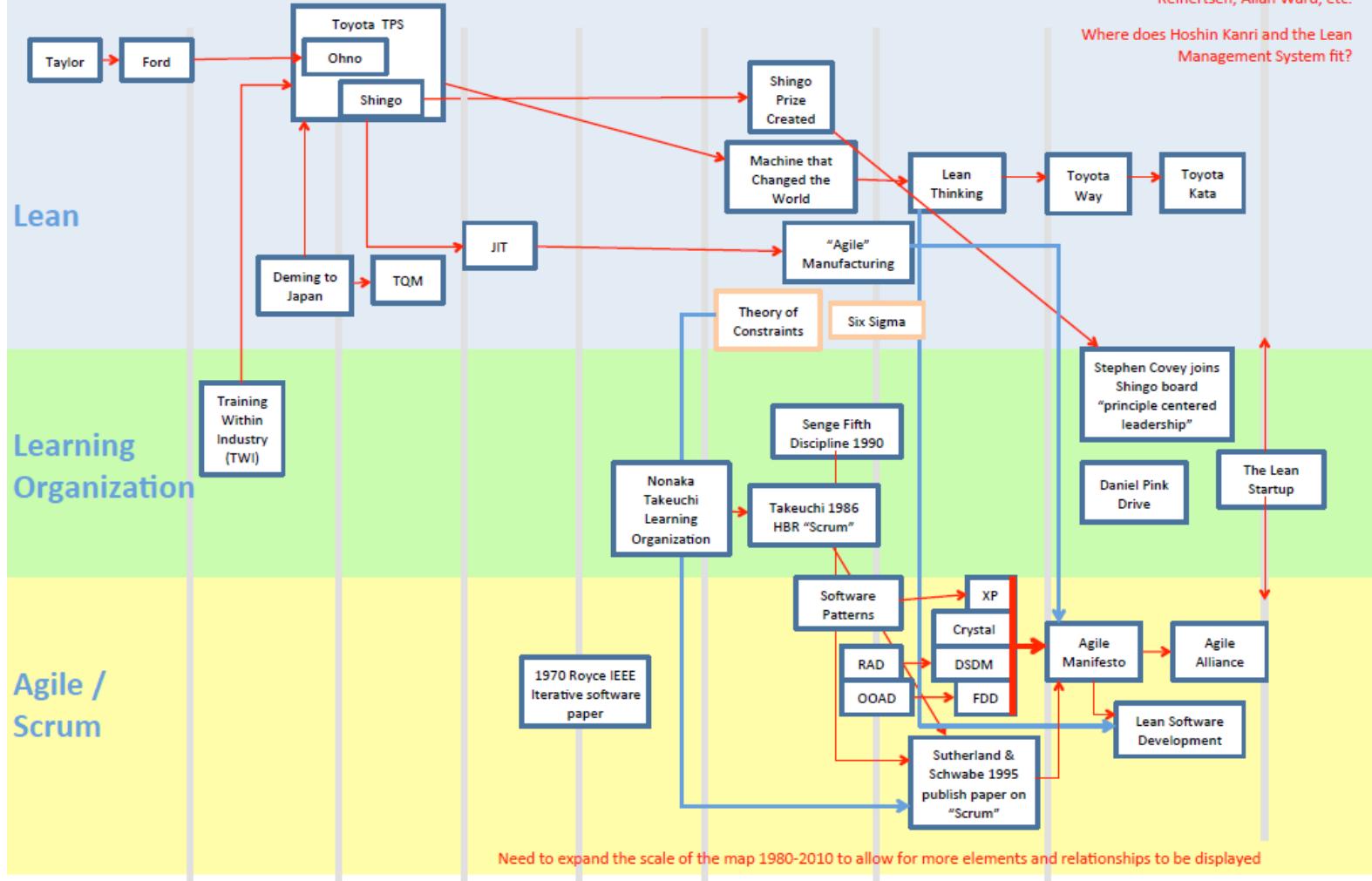
D'où vient l'agilité ?

Lean and Agile/Scrum Lineage: a visual work in progress

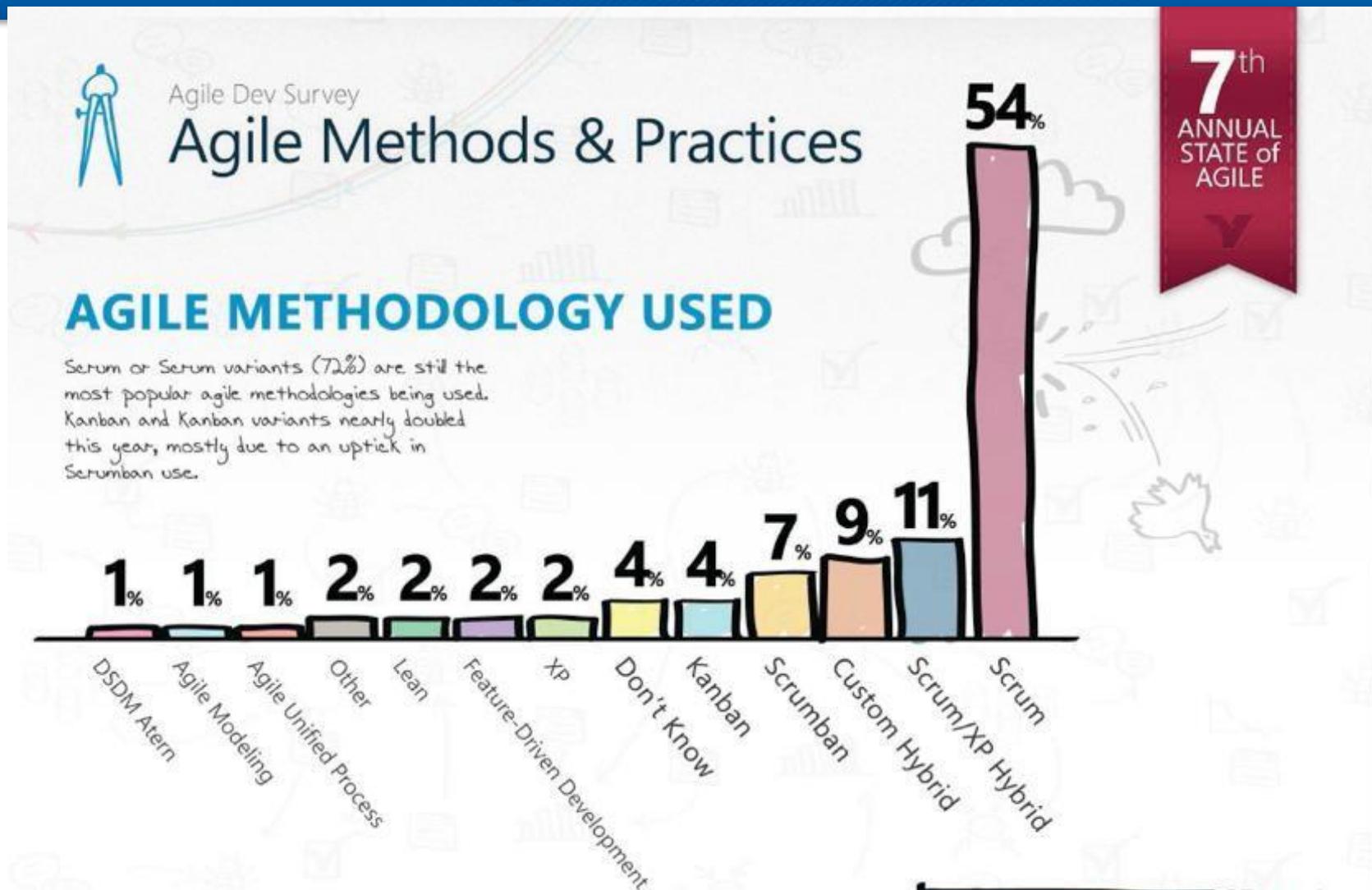
Copyright 2012 Lean IT Strategies, LLC and Scrum, Inc.

Consider how to fit Lean Product Development in here: Toyota, Reinertsen, Allan Ward, etc.

Where does Hoshin Kanri and the Lean Management System fit?



Scrum = la + utilisée des méthodes Agiles



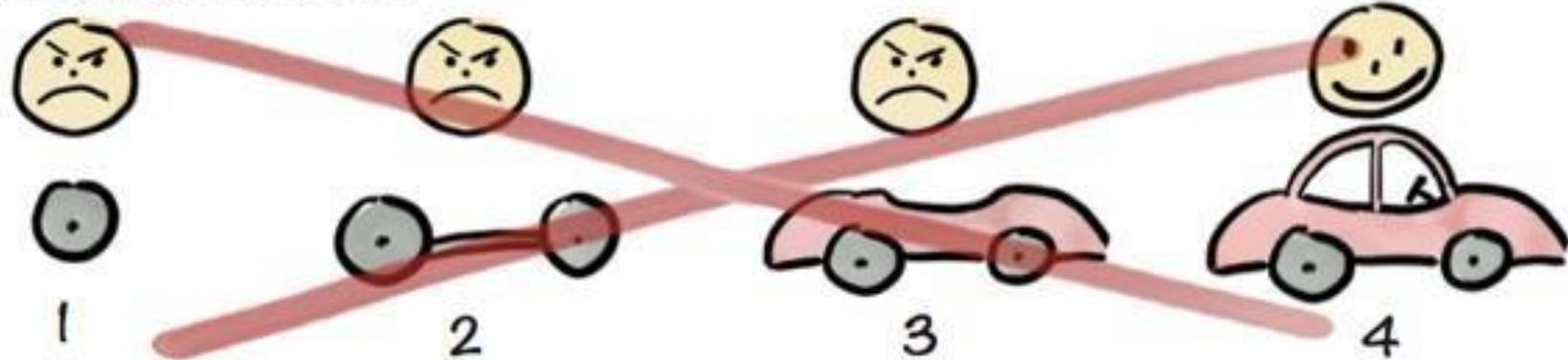
Scrum en 100 mots

- Scrum est un processus agile qui permet de se concentrer sur la livraison de la plus grande valeur métier dans le délai le plus court.
- Il nous permet de vérifier rapidement et régulièrement un logiciel qui fonctionne réellement (toutes les 2 à 4 semaines).
- Le métier définit les priorités. Les équipes s'organisent elles-mêmes pour déterminer la meilleure façon de produire les fonctionnalités les plus prioritaires.
- Toutes les 2 à 4 semaines, tout le monde peut voir réellement fonctionner un logiciel et décider soit de le livrer dans l'état, soit de continuer à l'enrichir sur le sprint suivant.

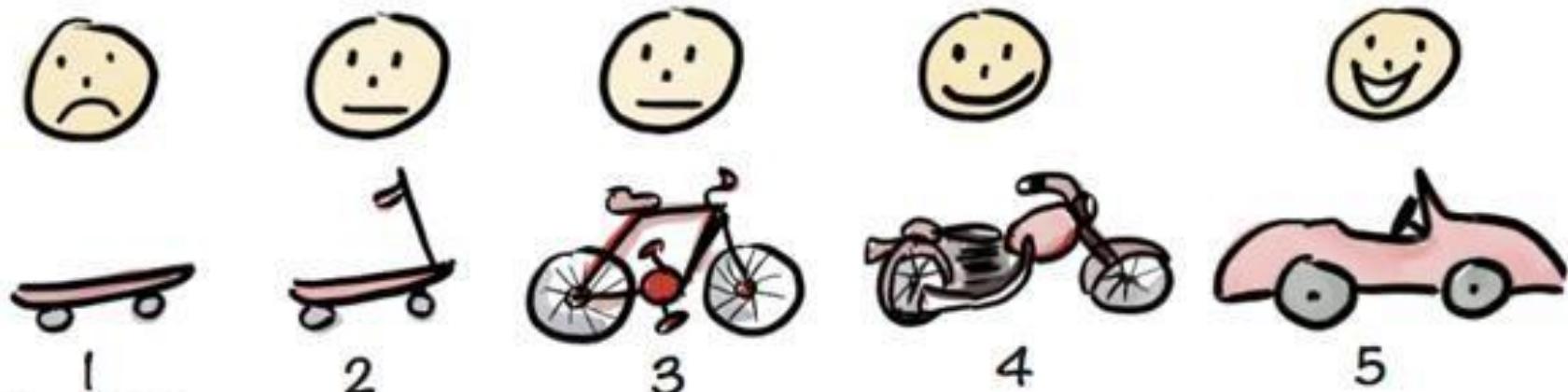
Iterative mais aussi Incrémentale



Not like this....



Like this!



by Henrik Kniberg

Caractéristiques



- Equipes auto-organisées
- Gestion empirique
- Orientation sur les buts (valeurs d'affaires)
- Mesure d'avancement en fonction de fonctionnalités terminées
- Le produit progresse par “sprint” de 2 à 4 semaines de dev.
- Les fonctionnalités à développer sont listées dans un “carnet de produit” (product backlog)
- Pas de pratiques spéciales de développement interdites ou obligatoire
- Ce n'est qu'une des méthodes Agile mais la plus populaire

Framework Scrum



Rôles

- Product owner
- ScrumMaster
- Équipe

Cérémonies

- Planification de Sprint
- Revue de Sprint
- Rétrospective de Sprint
- Scrum quotidien

Artefacts

- Backlog de produit
- Backlog de Sprint
- Burndown charts

En mettant tout ensemble



Framework Scrum



Rôles

- Product owner
- ScrumMaster
- Équipe

Cérémonies

- Planification de Sprint
- Revue de Sprint
- Rétrospective de Sprint
- Scrum quotidien

Artefacts

- Backlog de Produit
- Backlog de Sprint
- Burndown charts

Rôle du chef de produit



Product owner

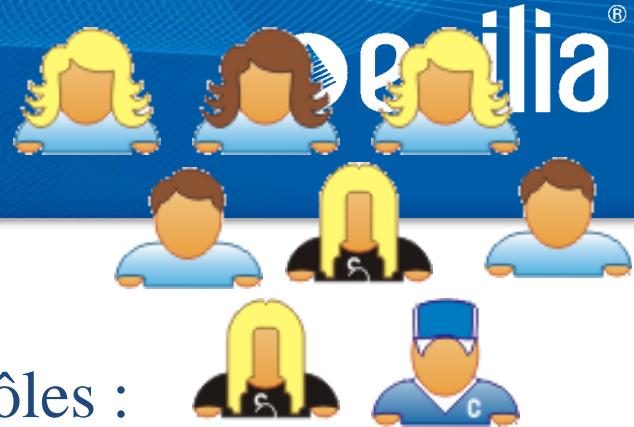
- Définir les fonctionnalités du produit
- Décider de la date de livraison et de son contenu
- Responsable de la profitabilité
- Priorise les fonctionnalités en fonction du client/marché
- Ajuste les fonctionnalités et les priorités à chaque sprint, si nécessaire
- Accepte ou refuse les livraisons de chaque Sprint

Rôle du Scrum Master



- Ce n'est pas un chef de projet mais un facilitateur !
- Responsable des procédures et valeurs Agile
- Résout les problèmes de l'équipe
- S'assure que l'équipe est pleinement fonctionnelle et efficace
- Permet une proche collaboration entre tous les intervenants
- Protège l'équipe des perturbations extérieures

Equipe de développement



- Typiquement 5-9 personnes
- Pluridisciplinaire: Regroupant tous les rôles :
 - Développeurs, testeurs, ergonomes, etc.
- Les membres de l'équipe doivent l'être à plein temps
 - Sauf exceptions (ex : administrateur BD)
- Equipes auto-organisées
- Les changements d'équipes si absolument nécessaires ne doivent avoir lieu qu'entre 2 sprints

Rôles

- Product owner
- ScrumMaster
- Équipe

Cérémonies

- Planification de Sprint
- Revue de Sprint
- Rétrospective de Sprint
- Scrum quotidien

Artifacts

- Backlog de Produit
- Backlog de Sprint
- Burndown charts

- Les projets progressent par “Sprint”
- Durée typique 2–4 semaines
- Une durée constante tout au long du projet est préférable pour garder un même rythme
- Des sprints plus courts permettent de voir et corriger les problèmes de fond plus rapidement
- Le produit est défini, codé et testé au cours du sprint

Planification de Sprint



L'équipe choisit les cas d'utilisation qu'elle peut s'engager à finir dans l'ordre des priorités du carnet de produit

- **Le backlog du sprint est créé**
- Les cas d'utilisation sont identifiés et décomposés en tâches
 - Chaque tâche est créée de façon à ne pas durer plus d'1 ou 2 jours
 - Cela de façon collaborative par l'équipe (pas slt le ScrumMaster)
- **La valeur business est prise en compte**

(1): rôle utilisateur, (2): que faut il faire ?, (3): valeur business

En tant que "touriste"(1),
je veux "voir des photos
de l'hôtel"(2) afin "d'être
sur qu'il est près de la
mer"(3)



Coder l'accès à la base de données (8h)
Coder l'interface utilisateur (4)
Ecrire les tests (4)
Coder la classe X (6h)
Mettre à jour les tests de perf (2h)

Capacité de l'équipe

Backlog de produit

Conditions métier

Produit actuel

Technologies

Planification du Sprint

Périmètre

- Analyser et évaluer le backlog du produit
- Définir le but du sprint

Planning

- Décider comment s'y prendre (conception)
- Créer la liste des tâches à partir des éléments du backlog du produit (user stories)
- Estimer les tâches en heures

But du Sprint

Backlog du Sprint

Le but du sprint

- Un bref énoncé de ce sur quoi le travail va essentiellement porter pendant le sprint

Base de données

Faire tourner l'application
sur SQL Server en plus
d'Oracle.

Sciences de la vie

Offrir les fonctionnalités nécessaires
à l'étude génétique d'une population

Services financiers

Offrir davantage d'indicateurs
techniques que l'entreprise ABC avec
un flux de données diffusé en continu
et en temps-réel.

Le regroupement journalier (Daily Scrum)



- Caractéristiques
 - Quotidien
 - 15 minutes
 - Debout
 - Tout le monde peut venir écouter
 - SEULS les membres de l'équipe (incluant le ScrumMaster) peuvent parler
- But
 - Garder l'équipe synchronisée et focalisée
 - Noter le travail fini et ce qu'il reste à faire
 - Créer un esprit d'équipe



Tout le monde répond à 3 questions



1

Qu'ai je fait hier ?

2

Que vais je faire aujourd'hui?

3

Est-ce que j'ai des problèmes ?

- Ce ne sont **pas** des statuts de suivi/flicage pour le Scrum Master
- Ce sont des engagements vis à vis de ses pairs

Sprint Demo/Revue



- L'équipe démontre ce qu'elle a accompli durant le sprint
- Typiquement cela prend la forme de démonstration des nouvelles fonctionnalités
- Informel
 - Pas plus de 2 heures de préparation
 - Pas de slides ! Des démos !
- Toute l'équipe participe + PO
- Tout le monde peut venir et discuter



Sprint retrospective



- Réfléchir régulièrement à ce qui marche et ne marche pas
- Dure en général de 15 à 30 minutes
- Fait à la fin de chaque sprint
- Toute l'équipe participe :
 - ScrumMaster
 - Product owner
 - Equipe
 - Eventuellement clients et intervenants

Sprint retrospective



- Toute l'équipe se réunit et discute sur qu'elle aimeraient :

Commencer à faire

Arrêter de faire

Juste une façon
parmi d'autres
de faire la
rérospective
d'un sprint.

Continuer à faire

Framework Scrum



Rôles

- Product owner
- ScrumMaster
- Équipe

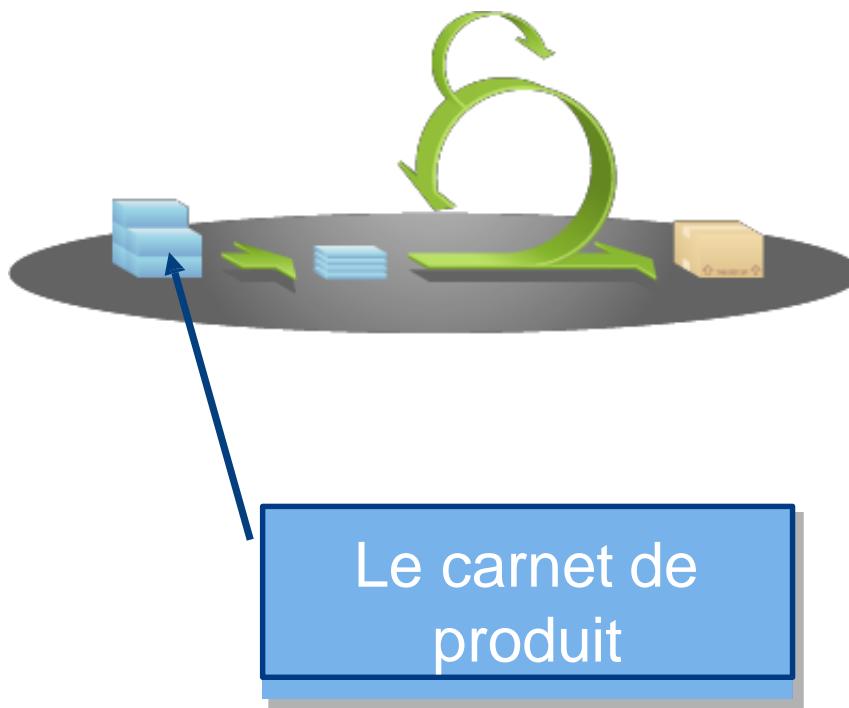
Cérémonies

- Planification de Sprint
- Revue de Sprint
- Rétrospective de Sprint
- Scrum quotidien

Artefacts

- Backlog de Produit
- Backlog de Sprint
- Burndown charts

Backlog de produit



- Les fonctionnalités demandées
- Une liste de tout ce qui est demandé pour le produit
- Exprimé en montrant la valeur attendue par le client
- Classé par priorité par le responsable produit
- Re-priorisé à chaque sprint et même entre sprint par le responsable produit suite à discussion avec le client

Exemple de carnet de produit



Cas d'utilisation	Estimations (Points d'histoire : Story Point)
En tant que client, je veux pouvoir annuler ma réservation, pour récupérer mon argent.	5
En tant que client, je veux pouvoir changer mes dates de réservations, en cas d'imprévu.	3
En tant qu'employé de l'hôtel, je veux pouvoir calculer la rentabilité des chambres tous les jours, pour adapter le tarif	8
...	30
	50

Le backlog de sprint

- L'ensemble des items sélectionnés pour le Sprint
- Identifier toutes les fonctions à développer
 - besoins utilisateur : user stories
 - fonctions induites : technical stories
- Evaluer les charges de travail

Gérer le backlog de sprint



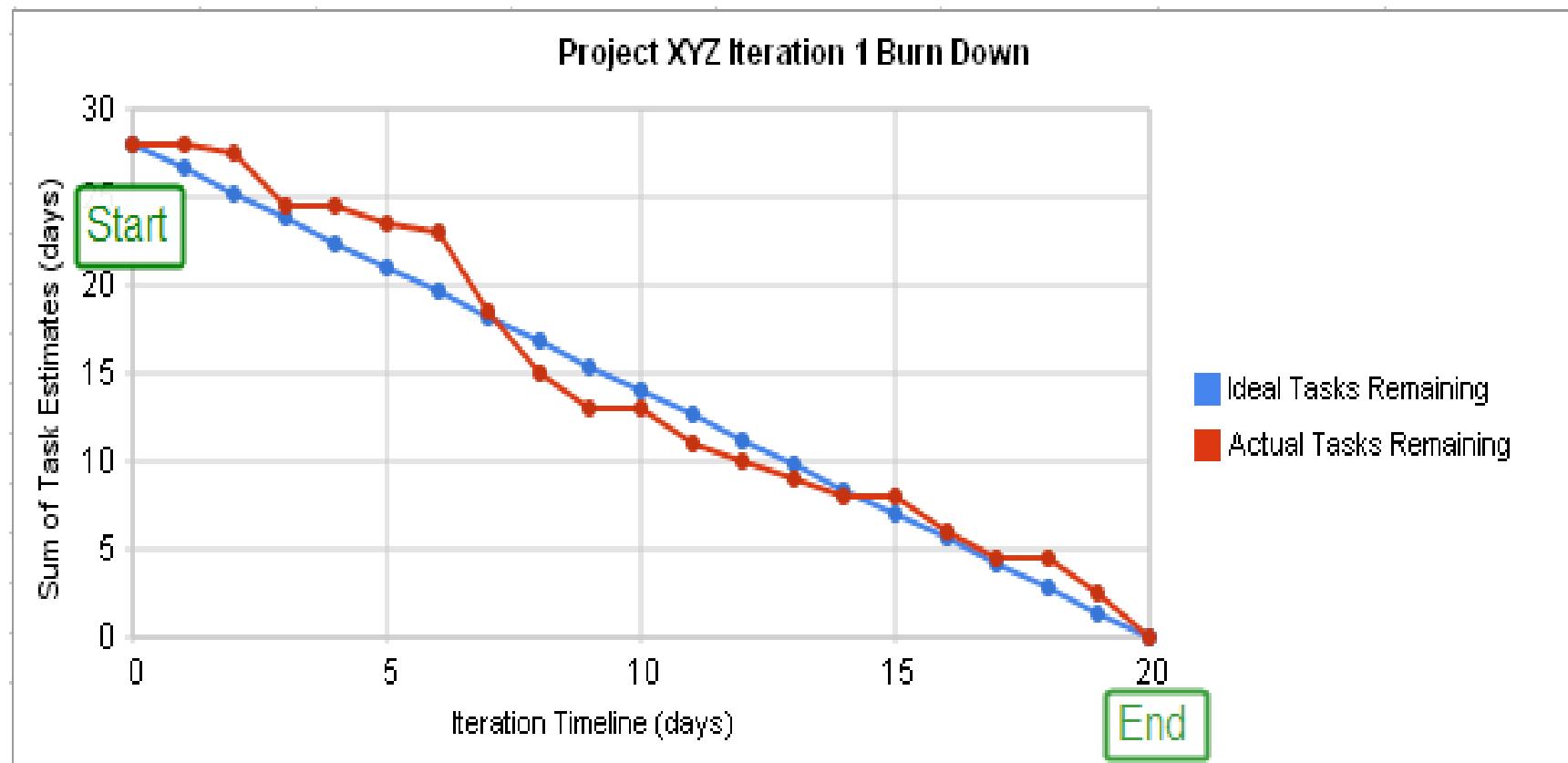
- Chacun s'engage sur un travail qu'il choisit lui-même
 - Le travail n'est jamais assigné par un autre
- L'estimation du reste à faire est réactualisée tous les jours
- Chaque membre de l'équipe peut ajouter, supprimer ou changer un élément du backlog de sprint
- Le travail du sprint émerge progressivement
- Si le travail n'est pas clair, définir une tâche dans le backlog du sprint en estimant une grosse charge et la décomposez plus tard
- Affinez le reste à faire au fur et à mesure

Diagramme de consommation : burndown chart de sprint



Le Burndown chart permet de matérialiser l'avancement global du projet.

Les courbes représentent la taille total du back log ainsi que le nombre d'user stories restant à réaliser (en points de complexité).



Jeu de Rôle

- Différence entre une User Story et une Tache par un exemple estival.
 - Vous avez votre Release composée de Sprints, composés de Users Stories, composées de Tâches.
 - La Release étant : "Passer une semaine au bord de la mer"

[Différence entre une User Story et une Tache par un exemple estival.docx](#)

La définition de « Fini » (Done)

- Lorsqu'un item du Backlog produit ou un Incrément est décrit comme «Fini», tout le monde doit comprendre ce que «Fini» signifie.
- Bien que cela varie considérablement d'une équipe Scrum à une autre, les membres doivent avoir une compréhension commune de ce que signifie que le travail soit complet, afin d'assurer la transparence.

La définition de « Fini » (Done)

- Les critères d'achèvement peuvent être résumés comme suit :
 - « Code écrit » – tel que défini par l'organisation / les équipes
 - « Testé » – tel que défini par l'organisation / les équipes
 - « Revu par les pairs » (qualité) – tel que défini par l'organisation / les équipes
 - « Aligné avec les attentes » – telle que défini par l'organisation / les équipes
 - « Documenté » (selon les besoins, déterminé par l'équipe Scrum au début de Sprint)

La vélocité

La vélocité est le nombre de points recettés pour chaque itération.

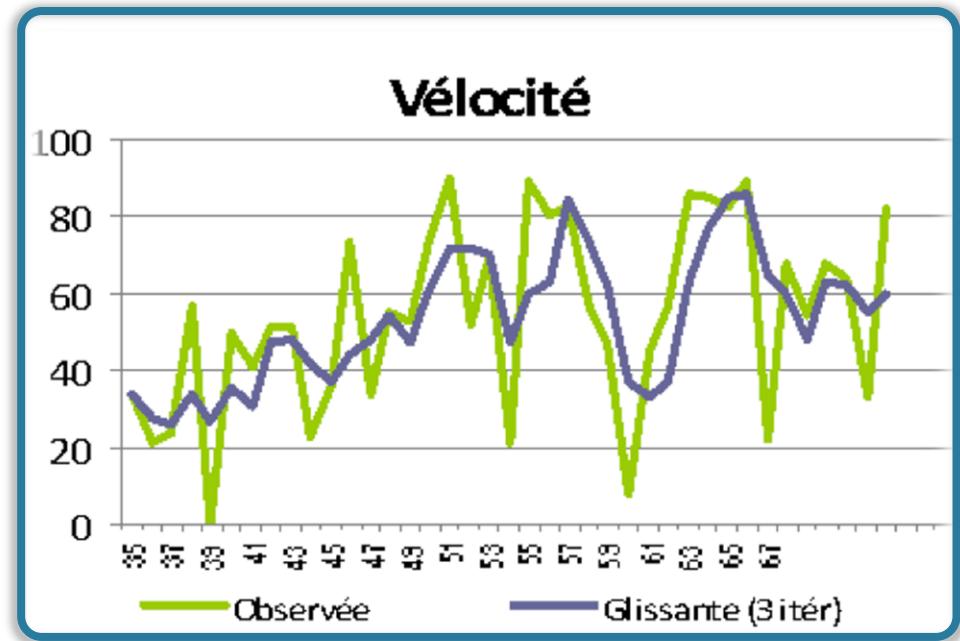
Cet indicateur permet d'avoir une bonne vision de la productivité globale de l'équipe projet.

La vélocité en pourcentage est cette formule "ultra-simple" :

Soit :

- NDPT = Le **Nombre De Point Terminé** sur le Sprint : Terminé et donc livré au Product Owner
- CDEP = Capacité De l'Equipe Prévue au début du Sprint

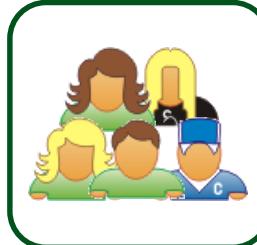
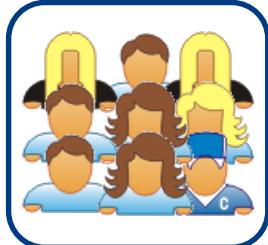
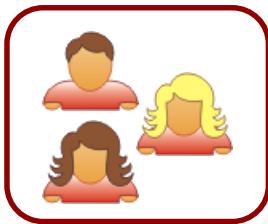
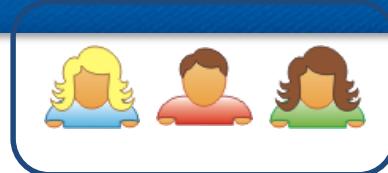
$$\rightarrow \text{Vélocité} = 100 * \text{NDPT} / \text{CDEP}$$



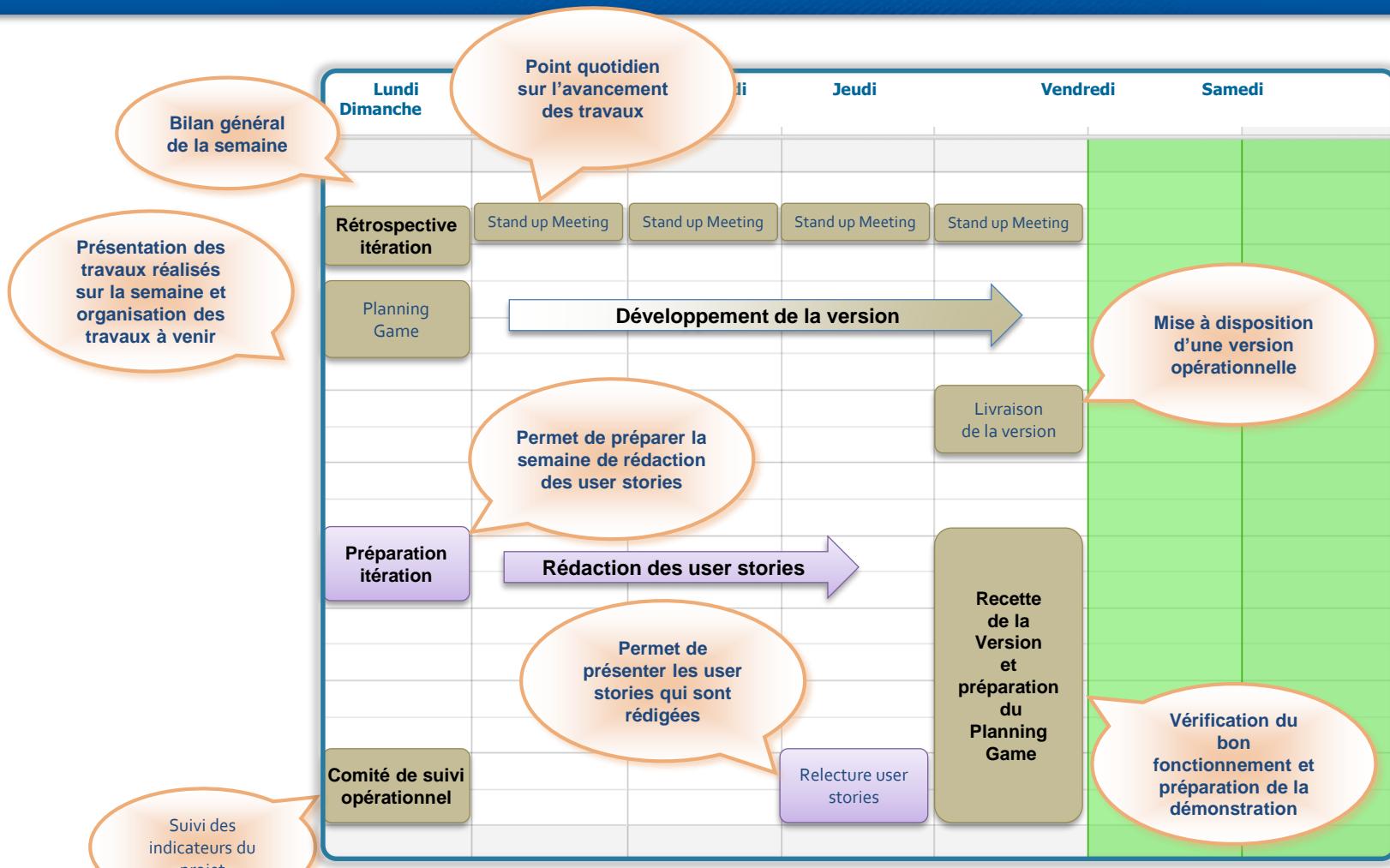
Scalabilité

- Une équipe comprend généralement 7 ± 2 personnes
 - La scalabilité est obtenu avec plusieurs équipes
- Facteurs de scalabilité :
 - Type d'application
 - Taille de l'équipe
 - Répartition géographique des équipes
 - Durée du projet
- Scrum a été utilisé sur de nombreux projets de 500 personnes et plus

Scrum de scrums de scrums



L'agenda du sprint



User story : Fonctionnalité unitaire de l'application décrite d'un point de vue utilisateur.



VISION « THÉORIQUE » DU RÔLE ET DES DEVOIRS DU PRODUCT OWNER



Fiche du poste

- Expert de la méthodologie Agile,
- Le Product owner est une sorte de Chef de projet
- Organisé, rigoureux, diplomate et pédagogue
- Responsable de l'aboutissement du projet ou de la conception du produit.
- Le principal responsable de la conception ou de la définition d'un produit en faisant l'interface entre les clients et les équipes de développement.

Les activités du Product Owner

■ Les activités du Product Owner

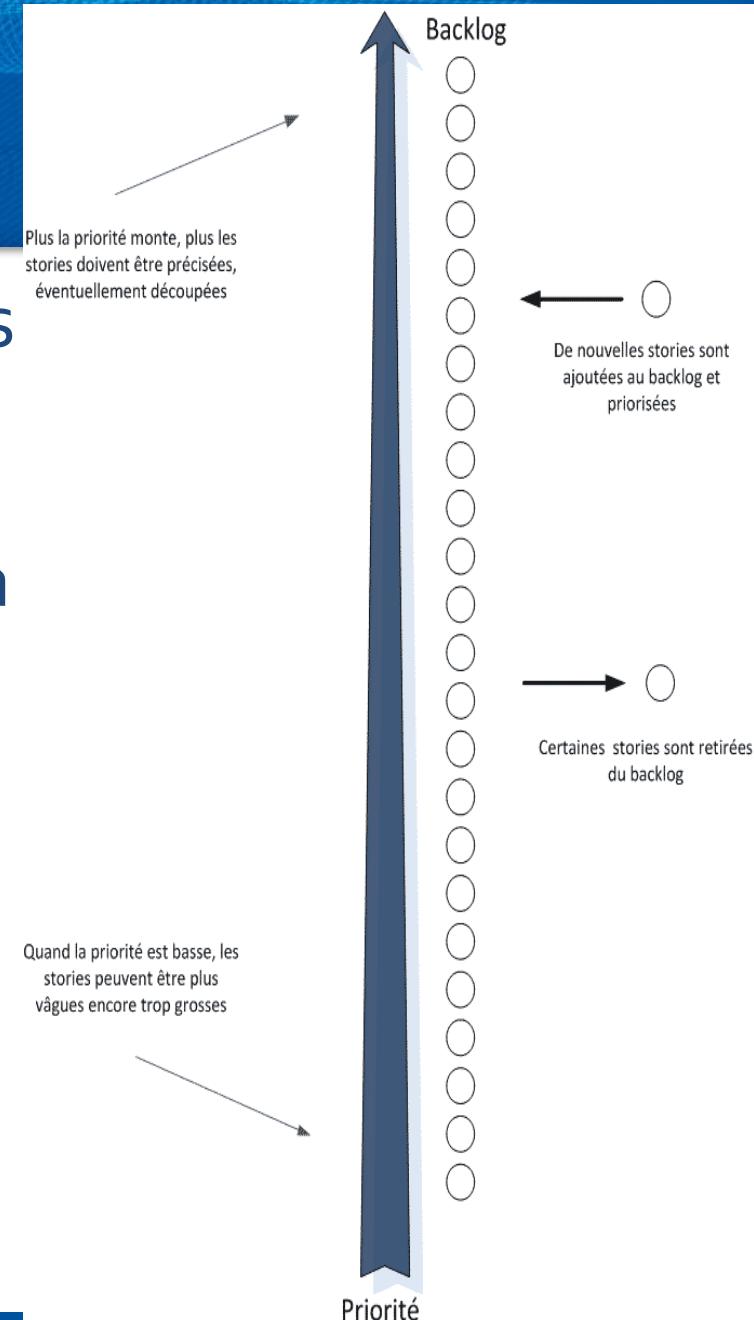
- Savoir élaborer et partager la vision du produit.
- Communiquer sur l'utilité et l'avancement du produit.
- Recueillir les retours utilisateurs.
- Participer aux réunions avec l'équipe de développement et le Scrum Master.
- Organiser les tests utilisateurs de la release.
- Evaluer les versions précédentes. Réfléchir sur les versions à venir.
- Gérer le Product Backlog.

Cumul de fonctions

- Le PO cumule :
 - Un savoir-faire technique dans la conception et la définition des caractéristiques du produit → afin d'apporter le maximum de valeur métier aux utilisateurs (dans le temps et le budget impartis au projet)
 - Un savoir professionnel dans la gestion des équipes de réalisation et dans les échanges avec les clients.
 - Le Product Owner possède également un talent certain pour le marketing stratégique ainsi qu'un œil formidable pour les détails.

Product Backlog

- Il s'agit d'une liste ordonnée des User Stories constitutifs du produit (ou des différents objectifs secondaires) menant à l'aboutissement du projet.
 - 4 caractéristiques :
 - valeur métier,
 - effort de réalisation,
 - risque
 - la connaissance technique ou métier.



Product Backlog

- Il est recommandé de le construire en équipe(marketing, des gens de métier et de stakeholders [sponsors]),
- Fréquemment le Product Owner fera le travail seul.
- L'élaboration du Product Backlog permet de répondre aux 3 questions suivantes :
 - Quelles seront les fonctionnalités à créer ?
 - Dans quel ordre devront-elles être livrées ?
 - À qui sont-elles destinées ?

- Etablir la vision du projet
 - Définir quel est l'objectif que le projet devra atteindre avant que le Product Owner n'en arrête le développement.
 - Cette vision doit être acceptée et comprise de tous
 - Exemple de vision :
« Offrir une solution de gestion complète, incluant suivi, statistiques et facturation ainsi qu'un module de prélocation en ligne et d'analyse statistique pour les vidéoclubs. »

Développement piloté par la valeur



- Lister les acteurs / personas :
 - Détailler tous les intervenants, utilisateurs du système dans tous ses aspects. Pour chaque intervenant on voudra préciser les informations suivantes :
 - Surnom : donner un Surnom aux acteurs rendra plus agréable leur utilisation et il sera plus simple de s'y identifier.
 - Icône / image : ajouter une représentation graphique de l'acteur rend l'identification encore plus facile.
 - Rôle : c'est en fait une description courte souvent juste un mot ou nom commun.
 - Description : on décrit pourquoi et/ou comment cet acteur utilisera le système.
 - Critères de satisfaction : ce qui rendra cet acteur satisfait de l'utilisation qu'il fait du système.
 - Valeur commerciale : élevée, moyenne, basse, bloquante (les autres acteurs ne pourront utiliser le système).
 - Fréquence d'utilisation : permanente, quotidienne, occasionnelle, rare.
 - Nombre d'instances : combien d'intervenants comme celui-ci utiliseront le système. 1, 10, 100+.
 - Niveau de connaissance technologique : élevée, moyenne, basse.
 - Niveau de connaissance métier : élevée, moyenne, basse.

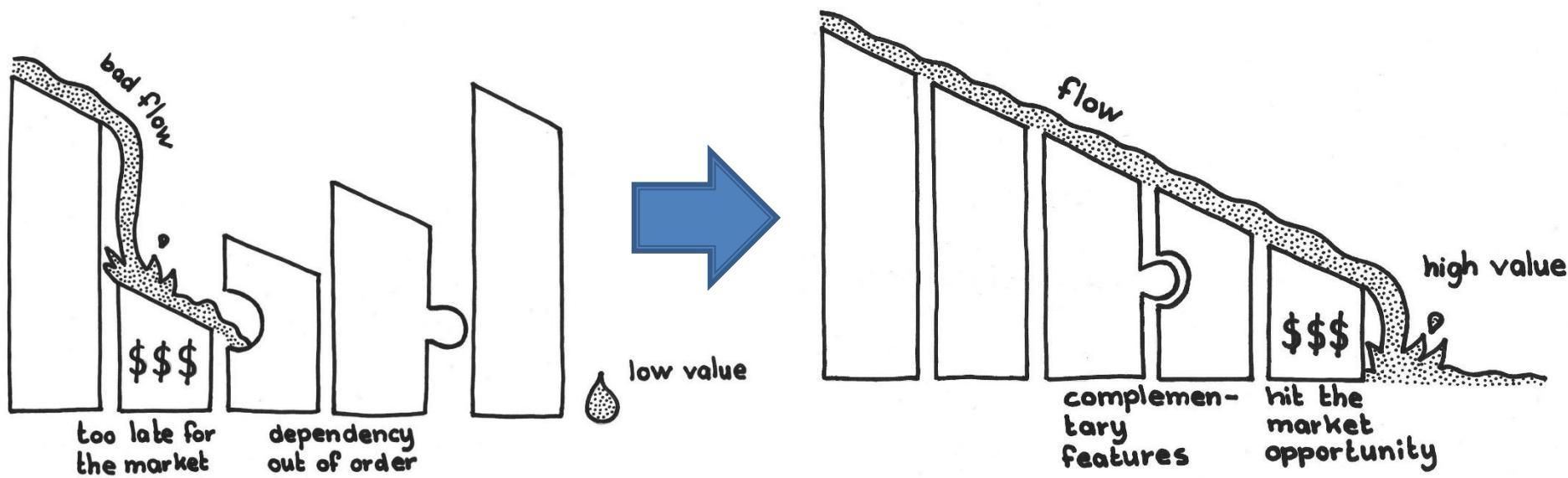
Développement piloté par la valeur

- Les thèmes ou regroupement de fonctionnalités
 - Pour chacun des thèmes, il faut préciser certains éléments :
 - Description : plus de détail concernant le thème, peut être même une liste de sous thèmes
 - Valeur commerciale : c'est un indicateur très important qui guidera vos choix et priorités
 - Critère de satisfaction : les critères qui feront de ce thème un succès
 - Couleur de post-it : permettra de facilement identifier les stories associées
 - Utilisateurs : la liste des utilisateurs que ce thème rejoint

- Définir la valeur commerciale
 - Compter le nombre de thèmes et attribuer 300 points par thème.
 - Pour chaque thème, accorder une valeur commerciale sur 1000 points.
- Rapidement, on remarque qu'il n'y a pas assez de points pour tout. Cela oblige à différencier en 3 groupes :
 - les incontournables,
 - les bons thèmes
 - et ceux dont on pourrait se passer.

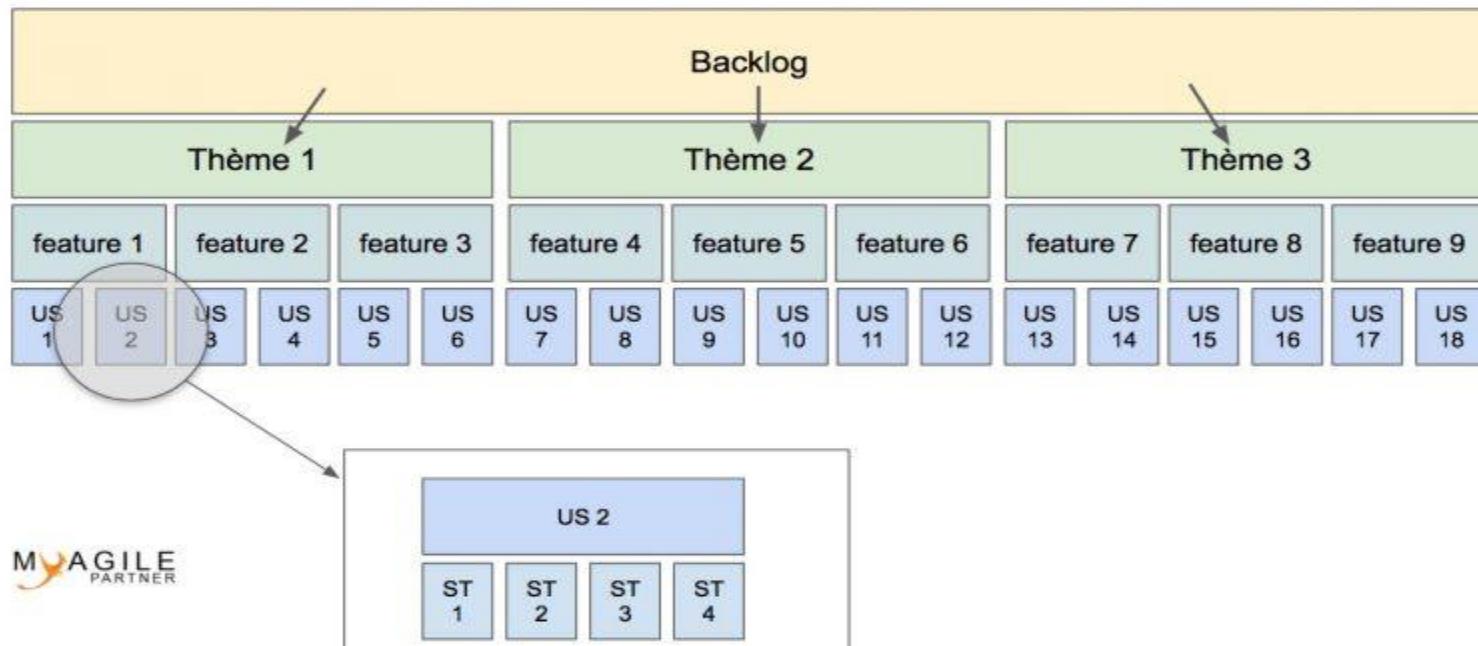
Développement piloté par la valeur

- Définir la valeur commerciale



BACKLOG PRODUIT, DU THÈME AUX egilia® USER-STORIES !

- Un backlog produit par produit
- Un backlog se découpe en thèmes
- Les thèmes découpés en Features
- Les features découpées en User-stories
- Les user-stories découpées en sous-tâche techniques



Scénarios utilisateurs

- Les USs sont les spécifications du projet présentées sous forme d'histoires utilisateurs qui décrivent les interactions de l'utilisateur avec le système de façon plus ou moins détaillées.
- Elle s'écrit sous la forme suivante :
« En tant que ...,
Je peux ...
Afin de ... »
- La rédaction d'une user story se doit de répondre aux critères INVEST (independant, negotiable, valuable, estimable, small, testable).

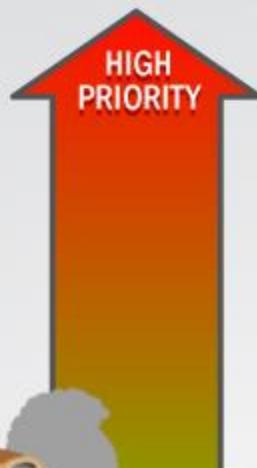
Scénarios utilisateurs

- Certaines informations utiles peuvent être ajoutées pour la définition de la story:
 - Thème : le thème auquel cette story est reliée. Utiliser un post-it de la couleur définie dans le thème permet de rendre le backlog plus visuel.
 - Acteur : l'acteur ou personas utilisé pour rédiger la user story. Là encore, si c'est possible, utiliser l'icône rend l'ensemble plus visuel.
 - Story : utilisée lorsqu'il s'agit d'une user story.
 - Description : utilisée lorsqu'il s'agit d'une story technique ou d'un bug.
 - Valeur : la valeur métier. On pourra utiliser le même genre de principe que celui présenté pour les thèmes.
 - Complexité : sera évaluée par l'équipe lors des sprints plannings ou des rencontres de travail du backlog produit (optionnelles).
 - ROI : le retour sur investissement se calcule simplement = Valeur/Complexité. C'est un indicateur intéressant pour prioriser le product backlog.
 - Tests d'acceptation : permet de définir quels seraient les tests d'acceptation de la story.
 - Commentaires.

LA PRODUCT BACKLOG REFINEMENT



Product Backlog
Refinement Meeting,
a.k.a. Backlog Grooming



Grades

Attendance

Event Calendar

Alumni Archives

Scholarship Award

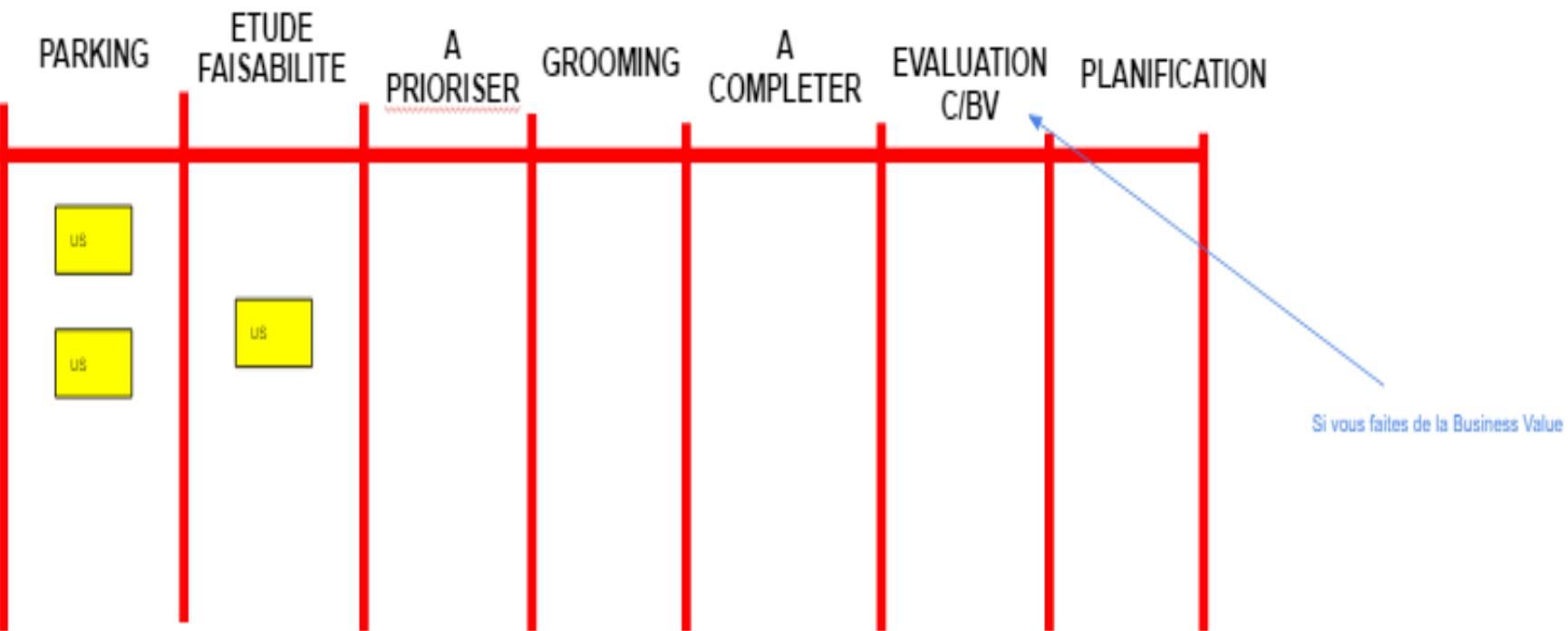
Project Backlog

Backlog

Backlog



LA PRODUCT BACKLOG REFINEMENT



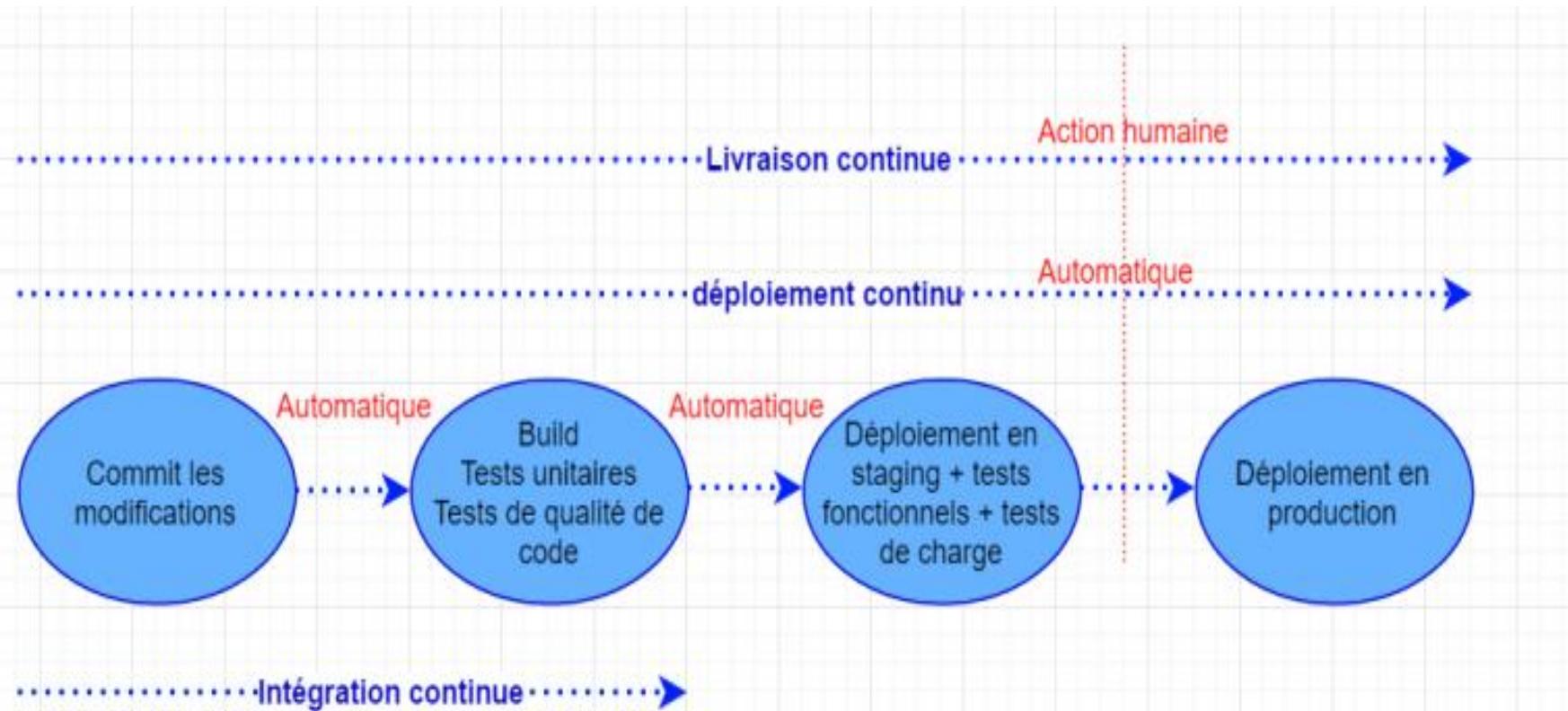
Le principe du planning poker scrum

- Tout l'intérêt du planning poker est que chacun est libre de s'exprimer comme il l'entend, et qu'il sera donc possible de croiser les différentes sources d'informations et les différents avis.
- Le planning poker a donc lieu avant le démarrage d'un sprint afin de déterminer le coût de chaque fonctionnalité et lesquelles pourront effectivement être réalisées.

Agile Planning Poker									
0	1	3	8	20	100				
0	1	3	8	20	100				
1/2	2	5	13	40	100				
1/2	2	5	13	40	100				

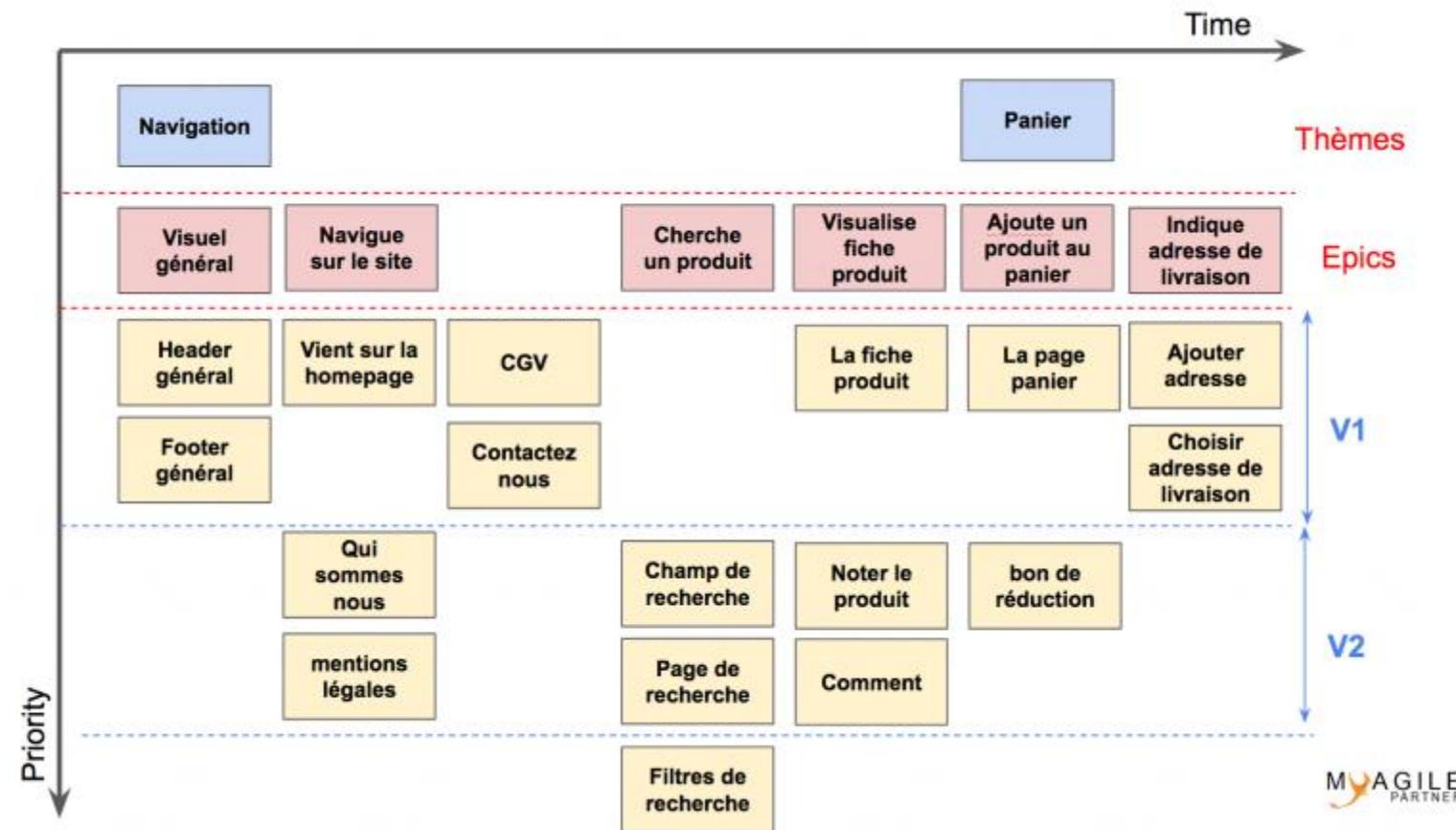
LES DIFFÉRENTS TYPES DE RELEASE POSSIBLES EN SCRUM

- Quelles sont les types de mise en production ?



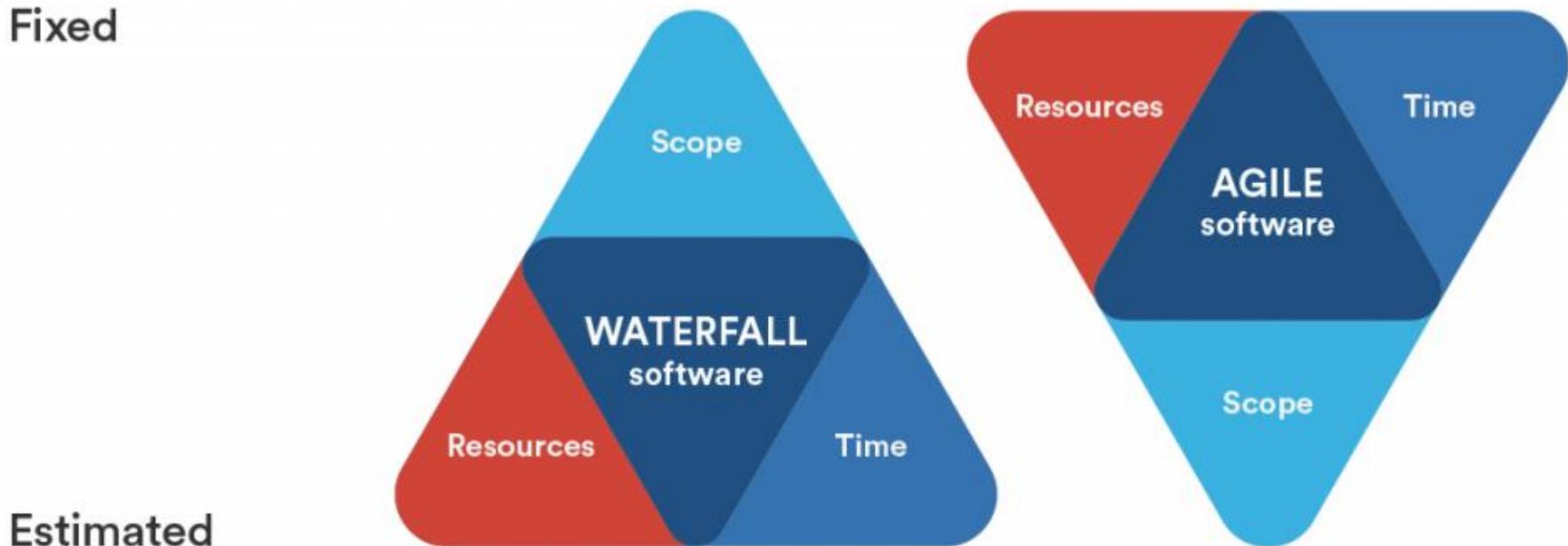
LES DIFFÉRENTS TYPES DE RELEASE POSSIBLES EN SCRUM

■ Livraison par version vs Livraison par Sprint



LES DIFFÉRENTS TYPES DE RELEASE POSSIBLES EN SCRUM

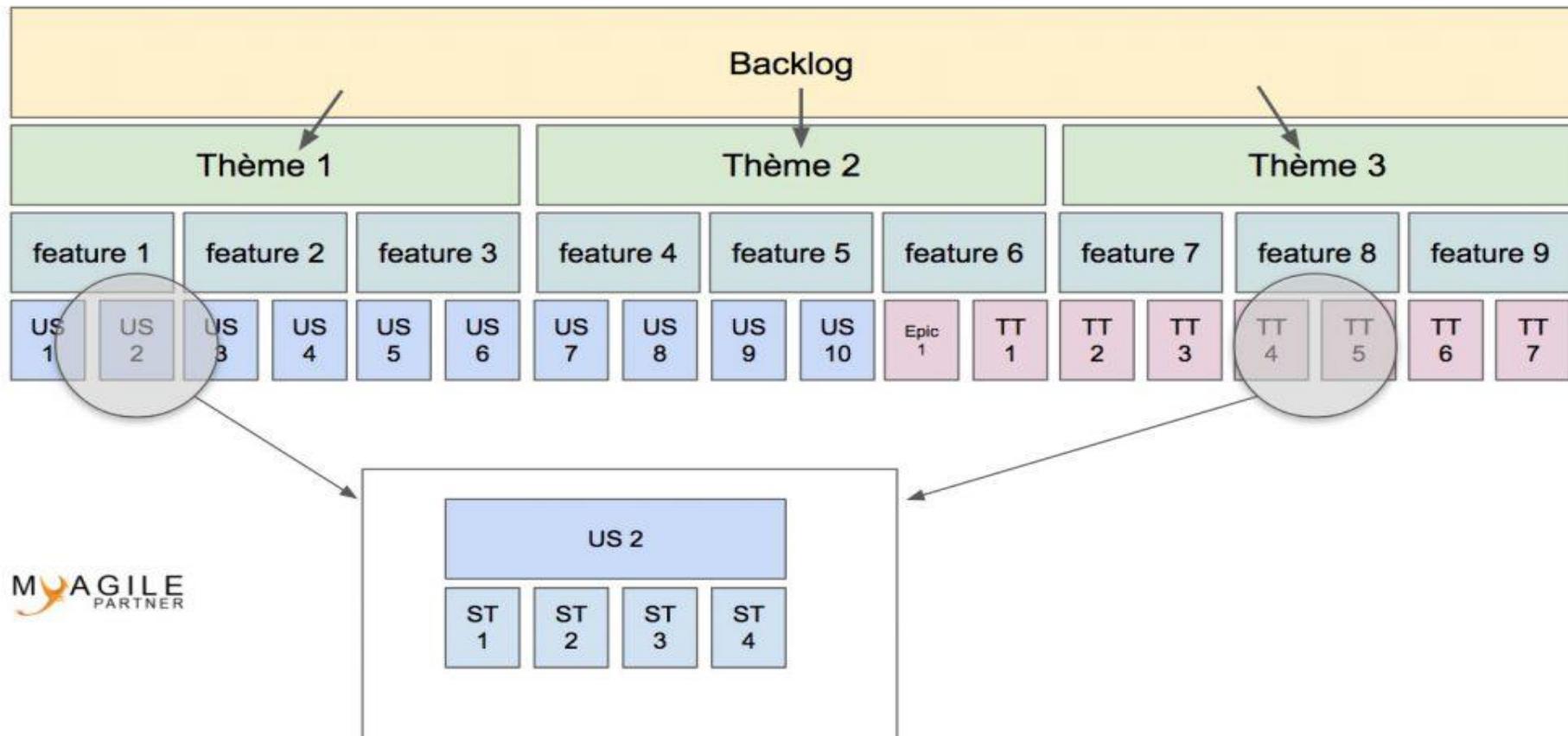
- Livraison à une date T



LES DIFFÉRENTS TYPES DE RELEASE POSSIBLES EN SCRUM



■ Livraison par feature



MYAGILE
PARTNER

LES DIFFÉRENTS TYPES DE RELEASE POSSIBLES EN SCRUM

■ Récapitulatif des 4 cas

Voici une façon simple de se rappeler des cas présentés :

Déploiement continue >

« Mise en production » pour être « Done »
Pas de teste humain

Livraison continue >

« Mise en production » pour être « Done »
Le Product Owner teste en continue puis mise en production

Livraison fin de sprint >

« Mise en production » pour être « Done »
Le Product Owner teste qu'en fin de sprint

Livraison par release >

« Mise en recette » pour être « Done » – Mise en production plus tard
Le Product Owner teste en continue en recette
> puis aussi lors de la grosse release en période de freeze



LES DIFFÉRENTS TYPES DE RELEASE POSSIBLES EN SCRUM



- Attention:
 - Faire des release qu'en fin de Sprint
 - La stratégie de l'entreprise peut amener à interdire le déploiement continu pour différentes raisons possibles :
 - associer des livraisons de fonctionnalités complètes à un acte marketing
 - respecter des raisons légales
 - valider la mise en production avec une instance de l'entreprise

8 outils pour animer une rétrospective Agile Scrum



- Quelques rappels sur la rétrospective agile scrum
 - Au terme de la rétrospective, un plan d'actions est mis en place, de façon à consolider ce qui a bien fonctionné, et à corriger les différents points négatifs.
 - il est préférable qu'aucun manager n'assiste à la rétrospective, de façon à ce que les membres de l'équipe ne soient pas freinés dans ce qu'ils pourront exprimer et qu'ils décident librement et sans contraintes des actions à mettre en place.
 - La rétrospective se déroule généralement en trois grandes phases :
 - la récupération des données,
 - l'identification des problèmes et des solutions,
 - et enfin, la définition d'un plan d'actions.

8 outils pour animer une rétrospective Agile Scrum

- Outils pour la récupération des données
 - Le Mood Board
 - L'objectif ici est de mesurer les émotions et le ressenti de l'équipe.
 - 1 mot
 - Chaque participant exprime son sentiment sur le dernier sprint en un seul mot et l'écrit sur un post-it.
 - Le quadrant
 - Diviser un tableau en quatre parties de couleurs différentes qui regrouperont respectivement :
 - Les points positifs.
 - Les points à améliorer.
 - Les « merci ! ».
 - Les idées d'amélioration.

8 outils pour animer une rétrospective Agile Scrum



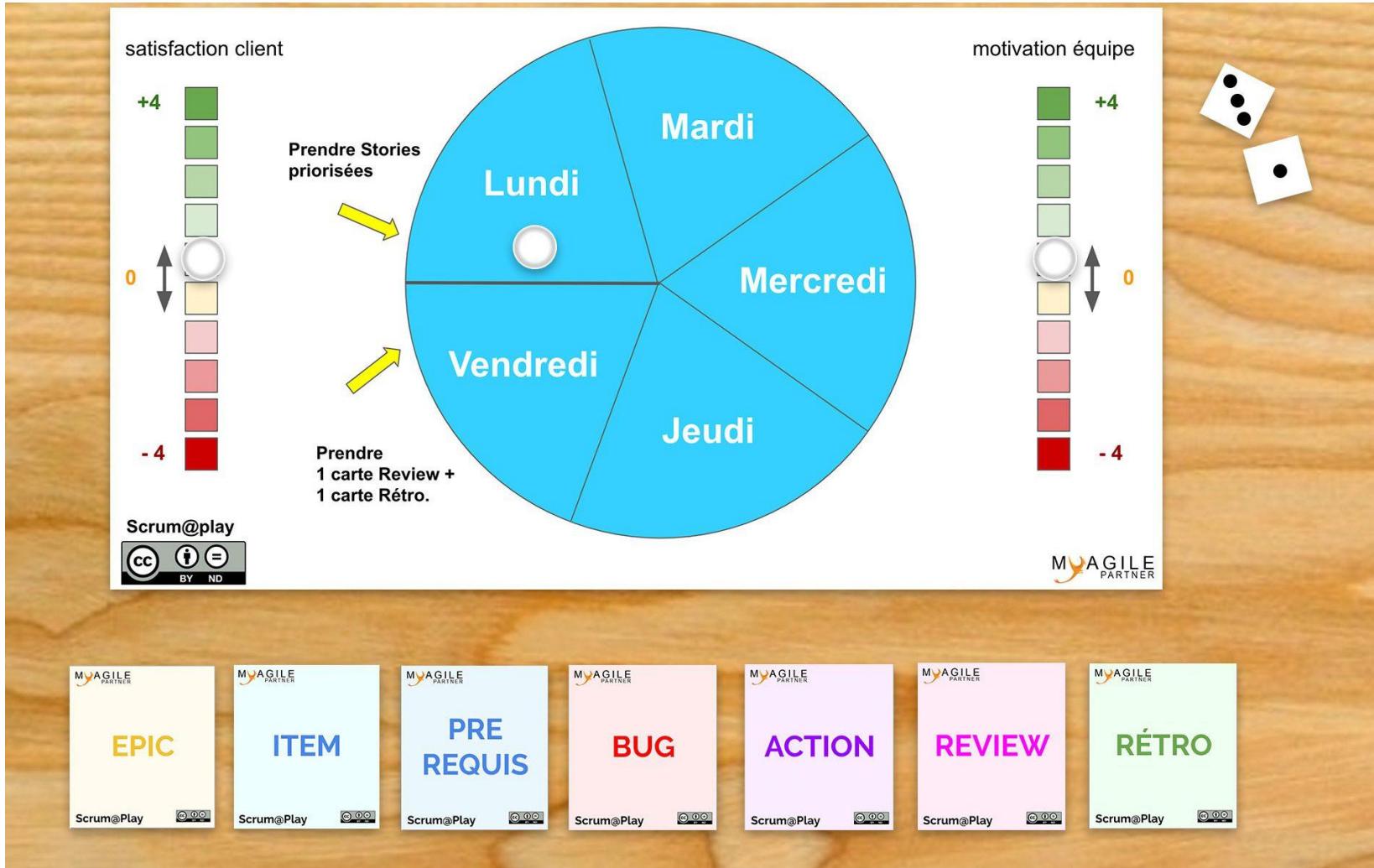
- Outils pour identifier les problèmes et leurs solutions
 - Le questionnement ORID
 - « O »bjective : on identifie précisément des faits, ce qui s'est passé.
 - « R »eflective : on identifie le ressenti de chacun pour les faits énoncés.
 - « I »nterpretive : on partage son interprétation des faits, ce qu'ils signifient.
 - « D »ecisional : on identifie les différentes actions qui peuvent être entreprises pour éviter que les faits décrits ne se reproduisent à l'avenir.
 - Les 5 pourquoi ?
 - Pourquoi le développeur a-t-il été dans l'incapacité de travailler ?
Parce qu'il ne pouvait pas utiliser sa version de travail de l'application.
 - Pourquoi ne pouvait-il pas utiliser sa version de travail de l'application ?
Parce que sa version de l'application ne fonctionnait plus.
 - Pourquoi sa version de l'application ne fonctionnait-elle plus ?
Parce que ses tests ont fait apparaître une anomalie majeure et bloquante.
 - Pourquoi n'a-t-il pas restauré la version précédente de l'application pour repartir sur des bases saines ?
C'est ce qu'il a fait, mais il n'y a pas de procédure automatisée, juste une procédure manuelle qui peut prendre plusieurs heures.
 - Pourquoi n'y a-t-il pas une procédure automatique de restauration d'une version de l'application ?

8 outils pour animer une rétrospective Agile Scrum



- Outils pour la définition du plan d'actions
 - Start / Stop /Continue
 - Start : les actions que l'équipe souhaite mettre en place.
 - Stop : les actions qu'elle souhaite arrêter (parce qu'elles n'ont pas porté leurs fruits par exemple).
 - Continue : les actions déjà en place, mais que l'on souhaite poursuivre, soit parce qu'elles sont efficaces, soit parce qu'elles n'ont pas encore apporté tous les bienfaits qu'elles auraient dû.
- Le starfish
- Speed boat

LE SCRUM@PLAY



Se préparer à la certification PSPO1 en 8 étapes



1. Lisez le Guide Scrum (16 Pages)
2. Reprenez vos notes, afin de compléter cette première lecture avec des exemples et des conseils.
3. Mettez vos connaissances en pratique
4. Lisez attentivement le Guide Scrum en version originale (Vocabulaire)
5. Entraînez-vous grâce au test d'entraînement
6. Échangez sur les questions pour lesquelles vous avez proposé une réponses incorrecte
7. Inscrivez-vous à la certification PSM 1
8. Vous disposez de 60 minutes pour répondre à 80 questions. Le test est réussi si vous obtenez 85% de bonnes réponses.

Glossaire



Sigle	Signification
ST	Scrum Team
PO	Product Owner
DT	Developpement Team
SM	Scrum Master
PP	Parties Prenantes (Stakeholder)
PBi	Product Backlog Items (exigence fonctionnelle ou pas)
PBlog	Product Backlog
SBlog	Sprint Backlog

Il est important de bien comprendre que



- Scrum est :

- Un framework (pas une méthodologie ni un livre de cuisine,...)
- Utiliser pour gérer des projets complexes dans un environnement complexe
- Basé sur une approche (process) empirique - ou l'empirisme
- Prévu (par la scrum.org) pour des projets de logiciels que l'on va livrer de manière Itérative (Sprint) et Incrémentale :
 - Incrément de logiciel constitué sprint après sprint et que l'on mettra en production quand cela fera sens pour le Product Owner
 - Seraproche de la notion de Release / Road Map mais ce « n'est pas Scrum »
- Trois piliers qui soutiennent toute mise en œuvre d'un contrôle de processus empirique: la transparence, l'inspection et l'adaptation.
- Basé sur 5 valeurs : engagement, courage, focus, ouverture et respect

Il est important de bien comprendre que



- « Scrum » conseille :

- Le recourt à une **équipe pluri-fonctionnelle** (qui a tous les talents nécessaires pour faire le job) composée de « généralistes éclairés » (profil de compétence en T) et qui travaille à un rythme soutenable (sustainable pace)
- L'utilisation **d'équipe fonctionnelle** (qui peut intervenir sur toutes les couches de l'application (en opposition avec les équipes dites « component »))
- (Scrum a dans l'idée) Que l'équipe de développement est en charge du travail jusqu'à la mise en production. Quant c'est Done il n'y a rien d'autre à faire !
- Que tous les membres de la Dev Team soient nommés des Developpeur (même et surtout si ils ne codent pas)
- De s'améliorer dès que possible et pas seulement lors des rituels prévus par le framework (retrospective par exemple)
- De mettre à jour les artefact et leurs informations dès que nécessaire et pas seulement lors des rituels prévus par le framework (Daily Scrum parex)

Il est important de bien comprendre que



- « Scrum » pense :

- Que la qualité / maturité / succès d'une équipe ne se mesure pas :
 - Au fait qu'elle ait des Release Sprint (c'est pas scrum), par contre qu'elle est des Sprint released (shipped) en production ca c'est bien !
 - Si le Done est non tenu et/ou inconsistante alors on y perd en transparence (vitesse,...), en qualité (il y a risque de bug) et les projections du Product Owner sur l'avancement deviennent illusoires
- Que s'engager, vouloir, pouvoir faire des choses c'est bien mais moins bien que de s'engager, vouloir, pouvoir les faire **DONE** en toute transparence
- Que l'incrément ne mérite son nom que si il contient de réelle nouveauté à la version N-1 (pas de correction de bug ou de refonte de fonctions déjà livrées)

Il est important de bien comprendre que



- « Scrum » interdit :

- Toute « perte de temps » non directement productrice de valeur :
 - Sprint 0 (préparation) ou Sprint Hou de Hardening (intégration avant release)
 - Sprint dédié à l'architecture / infrastructure ou à la QA
 - Pause, phase, travail entre les sprints
- Tout ce qui n'est pas Scrum :
 - Les rôles types : Project Manager, Tester, Business Analyst, QA
 - Les rituels types : Release Sprint, Sprint Testing,...
- De changer sans le dire (même si cela peut rendre moins anxieux le management)
- En conséquence si c'est « interdit » alors cela ne fait rien (role) ou ne sert à rien (rituel) ou n'existe pas (artefact)

Il est important de bien comprendre que



- Product Owner (PO) :

- C'est un rôle de « management » non hiérarchique : Leader Fonctionnel
- Ses fonctions sont de:
 - Maximaliser la valeur du travail fourni par l'équipe (ROI, Value, TCO,...)
 - Manager et Ordonner le Product BackLog
 - Envisager les objectifs des itérations puis les caler en coordination étroite avec la Dev Team
 - De décider quand cela fait sens de mettre en production l'incrément mis à sa disposition par la Dev Team
 - Rendre compte de l'avancement et de la stratégie de livraison au management et aux parties prenantes
 - Engager (pas au sens embaucher) les parties prenantes dans le projet et manager leurs attentes
 - Répondre aux questions que pourrait se poser la DT, lui donner du Feedback

Il est important de bien comprendre que



- Développement Team (DT) :

- C'est un rôle de « management » non hiérarchique : Leader technologique
- Entre 3 et 9 développeurs (hors Product Owner et Scrum Master)
- Ses fonctions sont:
 - De s'auto-organiser pour effectuer au mieux le travail qui lui incombe
 - De s'auto-organiser pour résoudre les Pb et les conflits qui naissent en son sein
 - De fabriquer un incrément de logiciel comprenant des fonctionnalités réalisées de telle manière :
 - Qu'elles satisfassent les critères d'acceptance
 - Qu'elles soient conformes à la Définition du Done
 - De s'assurer que l'incrément en cours de fabrication est bien « stable » pour ce qui est des nouvelles fonctionnalités MAIS AUSSI des anciennes
 - De s'assurer qu'à la fin de chaque sprint l'incrément livré soit de qualité suffisante (c'est obligatoire) pour pouvoir être mis en production (ce n'est pas obligatoire) si le Product Owner le souhaite

Il est important de bien comprendre
que



- Scrum Master (SM) :

- C'est un rôle de « management » non hiérarchique : Leader Organisationnel
- Ses fonctions sont de:
 - Promouvoir l'agilité et scrum
 - Faire en sorte que les rituels et les artefacts obligatoires se tiennent ou soient utilisés (pas négociable) mais que ce soit profitable à la Scrum Team (utile, efficace, adapté) en facilitant, enseignant, accompagnant les autres rôles de la Scrum Team et les parties prenantes
 - Lever les blocages
 - Faire respecter les timebox
 - Favorisant la prise de décision et la collaboration commune par / dans la Scrum Team

Il est important de bien comprendre que



- le Management :

- Ses fonctions sont de:
 - Faciliter la vie de la Scrum Team:
 - Communiquer les informations / décisions conduisant à un produit de haute valeur au Product Owner
 - Accompagner le Scrum Master dans ses actions de changement organisationnels
 - Habiliter la Dev Team à être auto-organiser
 - Respecter les rôles et devoir de Scrum
 - Si ils souhaitent des changements pas de Pb mais il faut passer par le Product Owner (et le Product backlog)

Il est important de bien comprendre que



- TimeBox = Temps maximum pour tenir un objectif / un rituel

Rituel	Time-Box Max	Durée liée à la durée du Sprint	Boucle Feed-Back	Observation / → Objectif
Sprint	4 semaines / 30 jours			Durée fixe jusqu'à décision de changement → Increment done et shippable
Sprint-Planning	8 heures	Oui		→ Sprint Backlog & Sprint Goal
Daily Scrum	15 minutes	Non	Oui	→ Actualisation Liste Blocage et Stratégie pour attendre le sprint Goal
Sprint-Review	4 heures	Oui	Oui	→ Validation Incrément avec les parties prenantes après analyse des résultats du sprint
Sprint Retrospective	3 heures	Oui	Oui	→ Plan Action Amélioration
Backlog Refinement	-10% de la capacité DT	Non		→ Product Backlog DÉPét BP pour le(s) prochain(s) sprint(s) Ready

Il est important de bien comprendre que



- Acteurs des rituels Scrums :

- Les Parties Prenantes sont toujours les biens venues mais... en mode « Poule » sauf à la review où l'on souhaite qu'elle s'exprime

Rituel	« Leader »	« co-producteur »	« facilitateur »	« observateur actif / inactif »
Sprint	DT	PO	SM	Partie Prenante (PP) sinécessaire
Sprint-Planning - Goal: - Quoi : Pbi - Comment : Task	P O P O D T	D T D T P O	SM	Partie Prenante (PP) si nécessaire
Daily Scrum	DT		DT	PO,SM et PP → non obligatoire et tous en mode inactif !
Sprint-Review	PO	DT& PP	SM	
Sprint Retrospective	SM	PO& DT	SM	Partie Prenante (PP) sinécessaire
Backlog Refinement	PO	DT	SM	Partie Prenante (PP) sinécessaire

Il est important de bien comprendre que



- les 3 seuls Artefact OBLIGATOIRES:

Artefact	Owner	Composé de	« Créer » et « Manager »	Observation
Product Backlog (PBLog)	Product Owner	Product Backlog Items	« Créer » et « Manager » par PO avec appui DT durant les activités de Backlog Refinement « Manager » avec PP durant Sprint-Review	Le PBLog évolue tout au long du projet, il attendance à contenir des PBi + gros que ceux présents dans le SBLog
Sprint Backlog (SBLog)	Dev Team	Tout type de décomposition utile des PBi (Test, US, Task, ...)	« Créer » et « Manager » par DT avec appui PO durant le sprint planning et le sprint	Doit (si il existe) être maintenu en cohérence avec le Sprint Goal Si il y a problème / question cela doit se traiter entre PO et DT
Increment	Product Owner	PBi Done	« Créer » par DT durant le sprint	L'incrément livré par la DT à la fin d'un sprint est composé : - Des nouvelles fonctionnalités - ET des anciennes Il doit être Done et Shippable Si + d'une DT l'incrément doit être GLOBAL

Il est important de bien comprendre que



- les autres Artefact sont facultatifs (mais fortement conseillés) :

Artefact	Owner	Objectif	« Créer » et « Manager »	Observation
Definition du Ready (DoR)	Product Owner	Fixe les critères d'un PBI suffisamment transparent pour être mis en développement	Fixer conjointement par : - PO et DT Sous la surveillance du SM pour qu'elle soit SMART	Souvent l'acronyme INVESTsert de base à cette définition
Definition du Done (DoD)	Dev Team	Fixe les critères pour qu'un PBI puisse être intégré dans l'increment	Fixer conjointement par : - la DT et le PO Doit être tenu par la DT Sous la surveillance du SM pour qu'elle soit SMART	Il devrait toujours y avoir une définition du Done Elle doit être bâtie sur ce que l'équipe sait faire à un instant T Elle doit progresser (+ rigoureuse) dès que la connaissance de l'équipe le permet Il peut y avoir plusieurs niveau de Done (PBI, Sprint, Release,...)

Il est important de bien comprendre que



- les autres Artefact sont facultatifs (mais fortement conseillés) :

Artefact	Owner	Objectif	« Créer » et « Manager »	Observation
Velocite	Dev team	« Mesurer » la production de valeur fournie DT	La vélocité ne se mange pas elle se constate sprint après sprint	Lavélocité d'une DTne doit (normalement) pas se comparer à la vélocité d'une DT Lavélocité et sastabilité sont un bon signe de maturité de la DT mais pas un signe de succès du projet
Burn Up	Product Owner (*)	Mesurer la progression de « l'effort » déjà réalisé	Créer par le PO et MAJ à l'issue de chaque Sprint Review	Sert à visualiser / envisager la progression du projet dans le temps Lesmétriques suivis (#US,#Valeur, #Effort) sont à déterminer selon les besoins
Burn Down	Dev Team (*)	Mesurer « l'effort » restant à accomplir	Créer par la DTà l'issue du Sprint Planning et MAJ chaque jour à l'issue du Daily Scrum	Sert à visualiser la tendance « d'avancement » de l'équipe dans sa tenue (ou non) de l'objectif du sprint Donne en tendance la date de fin si rien ne change

Il est important de bien comprendre
que



- le Product Backlog:

- Est ordonné normalement avec ce qui a le plus de priorité en tête de liste (top)
- Mais qu'en fait il peut être ordonné comme le veut le Product Owner qui est seul maître à bord
- Que si les besoins et/ou les conditions liés au produit alors il évolue

Il est important de bien comprendre que



- Product Backlog Items :

- Ils peuvent / doivent être de tout type :
 - Fonctionnel
 - Non Fonctionnel
 - Correctif...
- Tout ce qui représente du travail pour l'équipe doit être dans le Product Backlog
- Un PBi est terminé lorsqu'il ne reste pas de travail à faire avant de le mettre à la disposition des clients
- Pour les Exigences non fonctionnelles (architecture, temps de réponse, sécurité...) deux stratégies possibles (et/ou) :
 - Les inclure dans la définition du done
 - Les mettre dans le Product Backlog (PBi)

Il est important de bien comprendre que



- Sprint Planning :

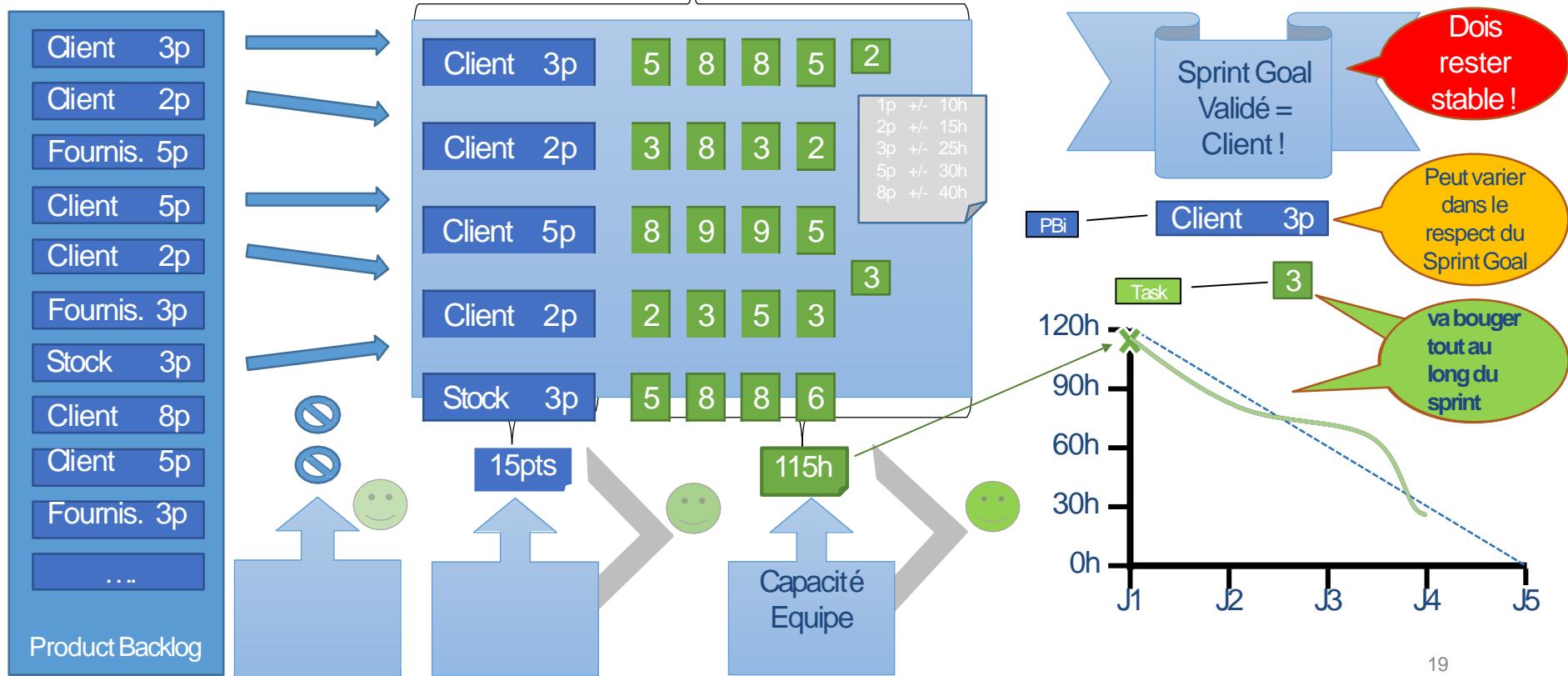
- Les objectifs sont :

- Permettre à l'équipe de constituer son plan d'action pour le sprint qui commence en tirant (to pull) dans le sprint backlog les Product Backlog Items (PBi) qu'elle estime pouvoir livrer dans l'incrément (Donne et Shippable) à la fin de la période
- L'ordre de prise en compte des Product Backlog Items est plutôt de la responsabilité du Product Owner mais il est négocié avec l'équipe de Développement (optimisation de la capacité)
- Seule l'Equipe de développement est responsable des estimations de charge et du découpage en action à faire (Task) des PBi. Bien sur c'est fait après discussions avec le PO
- Dans le Product Backlog peuvent être stockés tout type de décomposition des PBi sous quelques formes que ce soit dès l'instant où cela est utile pour la DT
- Si le travail de décomposition n'est pas totalement fini avant la fin du timebox : on commence à « coder » et l'on poursuit la décomposition en tache de fond « over time »
- Si la DT pense qu'elle a pris trop de travail on revoit le Sprint Backlog et/ou elle informe de PO du risque de ne pas pouvoir tout faire

Il est important de bien comprendre que



- Sprint Planning : créer le Sprint Backlog (PBi décomposés) & le Sprint Goal



Il est important de bien comprendre que



- Dailly Scrum :

- Il doit :
 - Se tenir tous les jours à la même heure au même endroit (réduit la complexité) et ne doit pas obligatoirement être tenu debout
 - Durée 15 mn ou moins
 - Il est organisé et managé par l'équipe de développement pour elle-même (et elle seule)
 - Le Dev Team est le seul acteur actif du Dailly Scrum. Elle fait en sorte que tout le monde réponde aux 3 questions et que l'on vérifie que la liste des problèmes est à jour et que l'on peut consolider/partager la vision de la stratégie pour venir à bout du reste à faire
 - En cas de remise en cause des engagements alors il doit y avoir concertation avec le Product Owner

Il est important de bien comprendre que



- Sprint :

- Contient tous les autres rituels de Scrum
 - Sprint planning
 - Daily Stand-Up
 - Sprint Review
 - Sprint Retrospective
- Débute immédiatement après la fin du sprint précédent, il ne se passe donc rien entre les sprints
- Setermine quand le time-box est terminé (4 semaines / 30 jours max)
- Doit servir à obtenir un incrément de logiciel done et potentiellement shippable (à la limite si il y a quelques bugs mineurs... c'est bon aussi)
- Sadurée doit être adaptée au contexte (risque, time to market,...) mais n'est directement piloté par la taille des PBi
- Seul le PO peut décider de l'arrêter avant la fin du timebox

Il est important de bien comprendre que



- Sprint-Review :

- On y parle (que) du produit à travers les artefact qui le caractérisent :
 - Product Backlog / Sprint Backlog
 - Incrément
- Ou ceux qui rendent compte de son avancement :
 - Burn Up – Vélocité
- Seul ce qui est done est présenté et discuté, le reste est « caché »
- Il n'y a pas que la demo dans la review mais aussi :
 - La récolte du feedback des parties prenantes (et du PO)
 - L'analyse des résultats de la demo : toujours Done (→ vitesse et BurnUp) ou KO (→ PBi de correction dans la Product Backlog)
 - La revue du Product Backlog pour envisager ce qui pourrait être fait ensuite

Il est important de bien comprendre
que



- Sprint-Retrospective :

- On y parle de tout (sauf sans doute du produit lui-même)
- L'objectif après une phase d'inspection (introspection en fait) est d'adapter un plan d'action (liste de changement que l'équipe souhaiterais mettre en œuvre) :
 - Peut porter sur n'importe quel point du fonctionnement du projet et de l'équipe dès l'instant que cela ne remet pas en cause Scrum (valeurs, Rituels, Artefact,...)
 - Les activités à réaliser liées à ce plan doivent être incluses dans le Product BackLog sous forme de PBi

Il est important de bien comprendre que



- Divers :

- Lorsque la composition d'une équipe change (et elle peut changer) alors la productivité va baisser quelque temps (il faut le temps de se réorganiser).
- De même si il y a plusieurs équipes et que leurs compositions changent alors les équipes voient leur productivité régresser
- Les causes qui peuvent avoir un impacts sur la performance de la DT ou la bloquer sont multiples (changement scope, techno, Teammembers,...)
- Il faut prendre le verbe « To Create » au sens finaliser pas initialiser
- Chaque Scrum Team doit comprendre un PO et un SM (qui peuvent être partagés entre plusieurs Scrum Team) leurs disponibilités n'a pas à être 100% mais doit être suffisante pour éviter des pertes de productivités de l'équipe (blocage, question, Acceptance test,... non traité / fourni à temps)

Il est important de bien comprendre que



- C'est sans doute pas bon si on pense que:

- Scrum ne marche pas ou ne peut pas marcher
- Scrum, ses artefact ou ses rituels sont des sources d'informations ou des occasions pour le management de blamer un des membre de la scrum team
- C'est une bonne chose que d'aller « baver » auprès du management, ou de « taper » sur quelqu'un ou un rôle auprès des autres
- De même si il y a plusieurs équipes et que leurs compositions changent alors les équipes voient leur productivité régresser
- Que quelque chose (tache, programme,..) peut appartenir à l'un des membres de l'équipe (to own) tout appartient à tout le monde

Il est important de bien comprendre que



- Scrum à l'échelle (scalabilité) reste du Scrum:

- Si il devait y avoir plusieurs équipes pour réaliser le projet / produit :
 - Il n'y a qu'un Product Backlog
 - Il n'y a qu'un Product Owner
 - Il y a un (ou plusieurs) Scrum Master qui intervient (viennent) en tant que Servant Leader (cf slide sur le SM)
 - Que les équipes de développement sont collectivement auto-organisées :
 - Elles choisissent comment elles se structurent
 - Elles définissent un Done commun visant à ce que leurs livrables soient directement (sans sprint d'intégration) mettables en production
 - Elles choisissent ensemble comment elles sélectionnent (to Pull) et se répartissent le travail à faire durant le sprint (qui peut être conseillé) avoir les mêmes dates / durées
 - Elles doivent se synchroniser pour être en mesure de délivrer l'incrément (done et shippable)

Il est important de bien comprendre
que



Word	Signifie	Ne pas confondre avec
Over Time	au fil du temps	Extra Time (heure supp)

Il est important de bien comprendre que



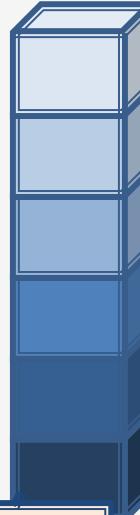
- Par exemple qui a le droit/devoir de virer un développeur qui ne fait pas l'affaire...
 - Perso je dirais le manager
 - Mais la Scrum.org attend la dev team comme bonne réponse



Récapitulatif animé

Mêlée quotidienne

- 15 minutes, tous les jours
- Trois questions pour chacun
 - Qu'avez-vous fait hier
 - Qu'allez-vous faire aujourd'hui
 - Quels sont vos problèmes
- Mettre à jour le Backlog du Sprint
- Le reste à faire total pour le Sprint -> burndown chart



Backlog
du produit



Produit



Questions ?