

Gestion de projet et Agilité

3AGILEM

Partie 2

ESTIAM LES LILAS

Année 2021 - 2022

Bernard Le Rouzic

Le cursus

- **Introduction**

- Principe de l'agilité et rappels sur les méthodes classiques de gestion de projet
- Panorama des différentes méthodes « Agiles »
- Focus sur la méthode Scrum et vue d'ensemble pour bien commencer

- **Planification et organisation des itérations**

- Planification journalière : daily scrum ou standing meeting : l'objectif, l'organisation
- MEO de l'amélioration continue : animation des revues, rétrospectives de fin d'itération
- Principes d'ingénierie : conception simple, amélioration du code par la réécriture, intégration continue
- Indicateurs d'avancement

- **Formalisation des exigences**

- Techniques de description des besoins fonctionnels et des exigences qualité
- *Features*, les *Users Stories* ➔ Notion de *Backlog*
- Tests d'acceptation
- Construire des stories tests

Le cursus (suite)

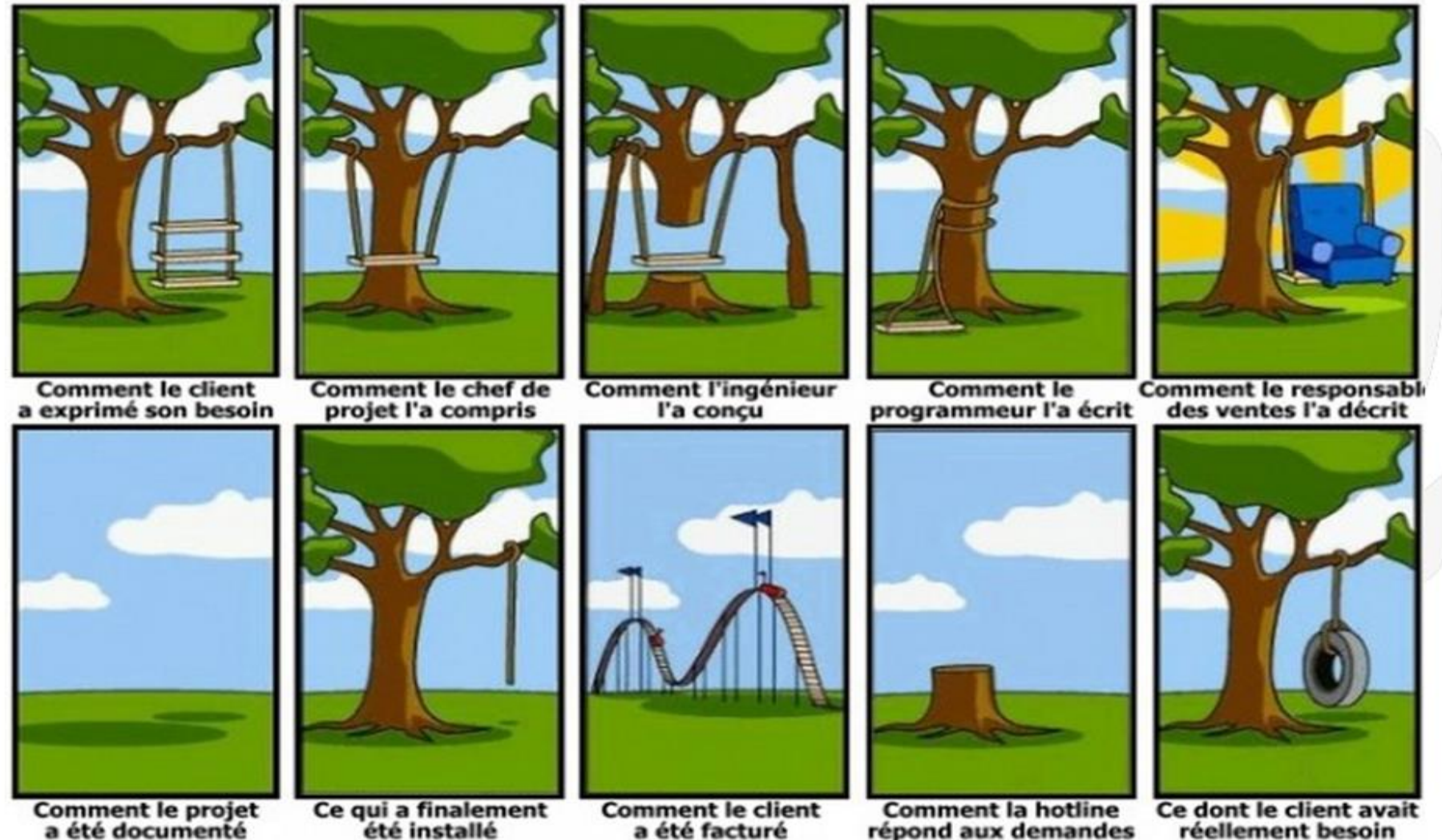
- **Priorisation des « *User Stories* »**
 - Planification basée sur la valeur
 - Modèle de Kano & Méthode Moscow
 - Calculer les valeurs ajoutées, valeur client pour chaque story à planifier dans la release
 - Priorisation des stories basée sur le risque et sur la valeur client
- **Planification des *releases* et des *sprints***
 - Découpage du projet en releases, construire la roadmap
 - Définir les sprints ou les itérations du projet
 - Évaluer des charges, évaluation de la taille des stories : le Planning Poker
 - Définition de la vélocité de l'équipe
- **Mise en œuvre des méthodes agiles**
 - Outils agiles, tableurs, outils spécialisés
 - Etapes de la transition d'une démarche classique vers une approche agile
 - Accompagnement du changement, contexte, objectifs du changement et rôle du coach

Introduction

- Principe de l'agilité et rappels sur les méthodes classiques de gestion de projet
- Panorama des différentes méthodes agiles
- Scrum : vue d'ensemble pour bien commencer

Principe et rappels sur la gestion de projet

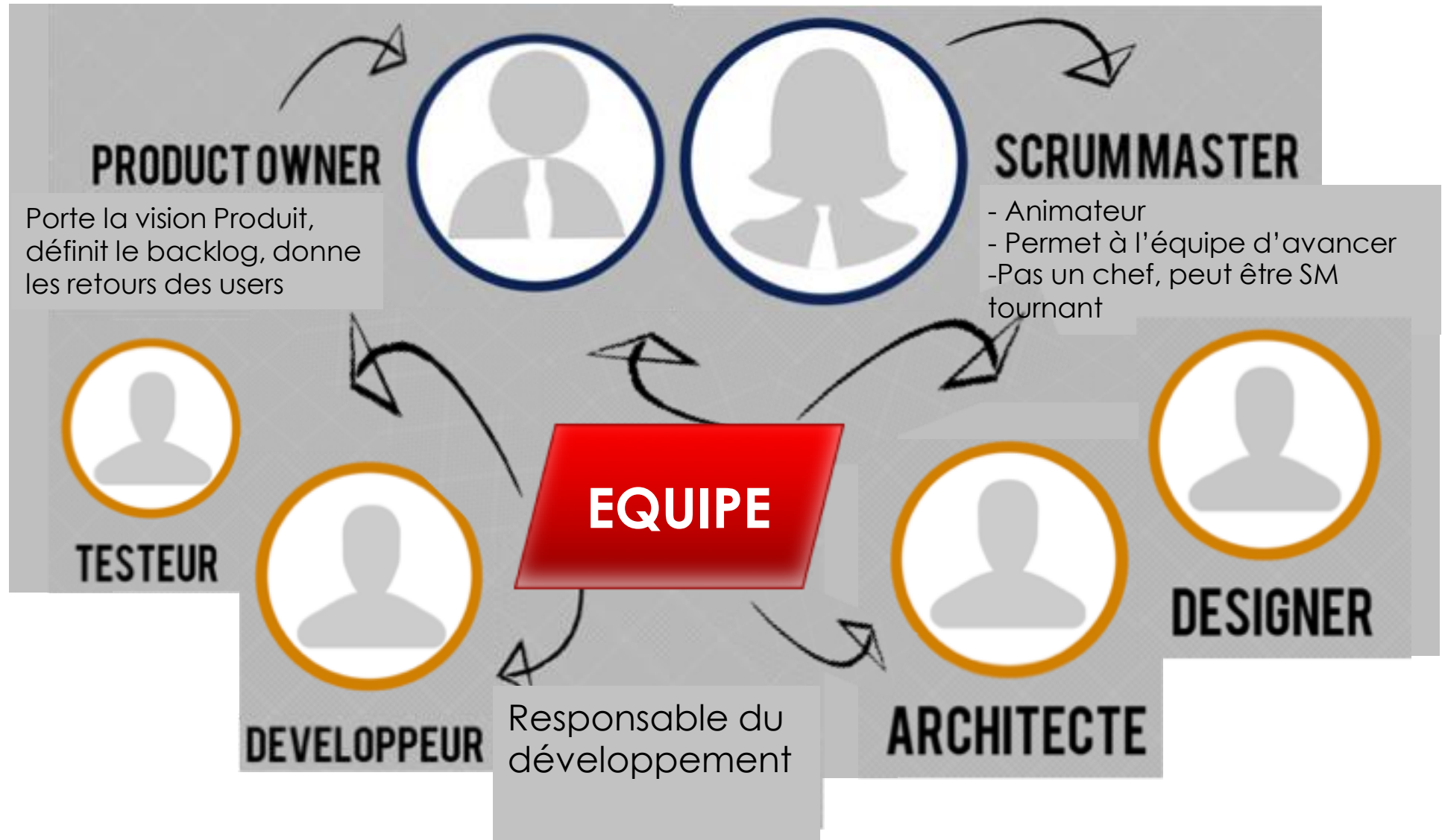
Le « Projet » : un certain point de vue...



Focus sur SCRUM

- La méthode agile Scrum est particulièrement destinée à la gestion de projets informatiques tient son nom du monde du rugby
- Le principe de Scrum est de pouvoir modifier la direction prise par le projet au fur et à mesure de son avancement. C'est exactement ce qui se passe lors d'un match de rugby, lors d'une mêlée (« scrum » en anglais)
- La mêlée est donc une phase essentielle au rugby comme dans la gestion de projet
- Si les conditions de réussite ne sont pas remplies, alors il faut réorienter le projet pour repartir sur de meilleures bases
- Le client est étroitement impliqué grâce à la livraison régulière de prototypes opérationnels permettant de valider les développements
- Cette gestion dynamique permet de s'assurer de la correspondance entre le besoin exprimé et le produit livré, et de réorienter au besoin les futurs développements

Focus sur SCRUM



Focus sur SCRUM

En gros, SCRUM fonctionne ainsi :

1. on distribue des rôles:

- **Product Owner** : représente les intérêts de l'utilisateur final et spécifie les besoins
- **Scrum Master** : aide l'équipe à avancer en suivant les principes de Scrum et la méthodologie Agile
- **Une équipe !**

IMPORTANT : chacun laisse de côté son égo, Scrum fonctionnant suivant un modèle de "Leader au service des autres"

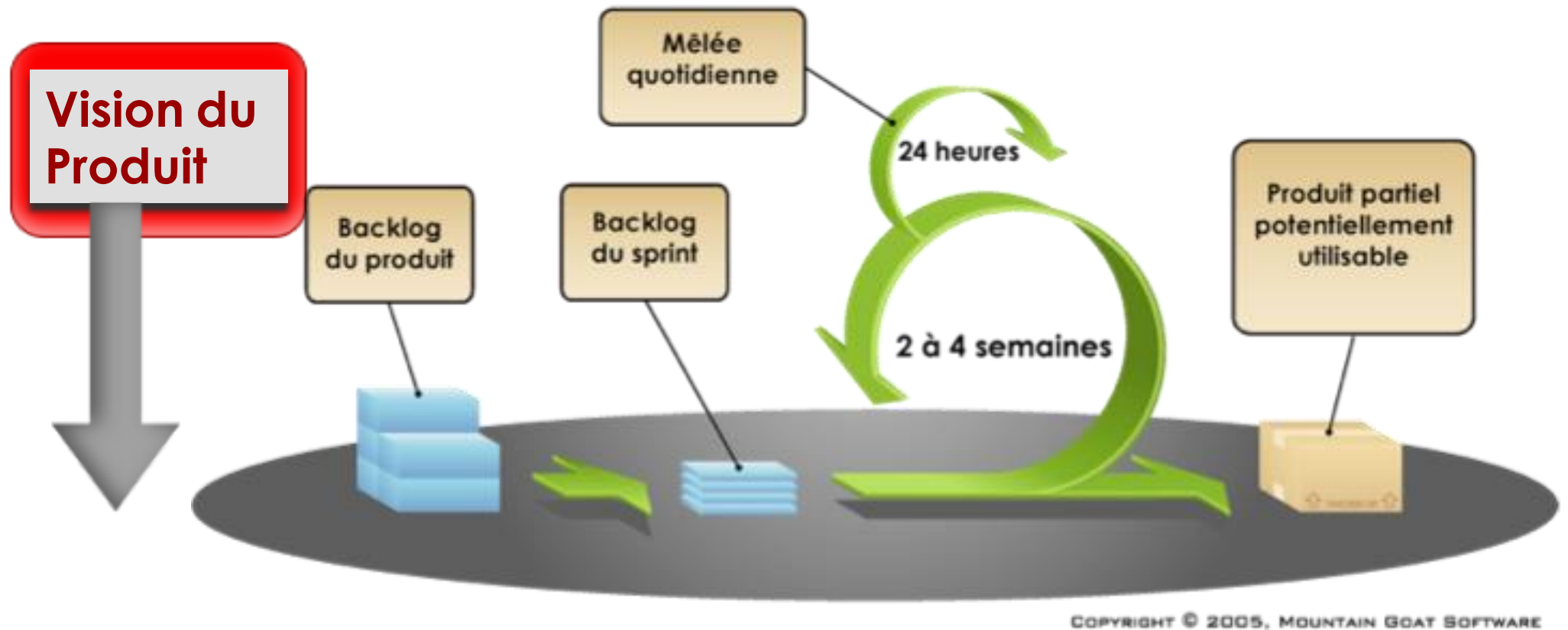
2. on crée le backlog du produit : contient la liste de toutes les spécificités du projet, classées par importance ; il n'est jamais complet, car quand le projet prend forme, de nouveaux besoins apparaissent sans cesse, ce qui oblige à actualiser la liste ➔ le Product Owner est le principal responsable de ce processus

3. on planifie un (1^{er}) sprint : à partir du backlog, on choisit les tâches devant être complétées lors du premier sprint ; un sprint est limité dans le temps ; on définit une durée compatible avec le projet et l'équipe ; la durée ne doit pas excéder un mois ; lors de sa planification, l'équipe définit les tâches et leur responsable

Focus sur SCRUM

4. **on met la main à la pâte** : chacun progresse sur ses tâches, rend compte de son avancement lors des *Daily Scrum Meeting* ne prenant pas plus de 15 minutes ; on se concentre sur trois questions :
 - qu'avez-vous fait hier ?
 - qu'allez-vous faire aujourd'hui ?
 - Avez-vous besoin d'aide pour débloquer quelque chose ?
5. **on fait une rétrospective du travail** : au terme du sprint, l'équipe présente les tâches réalisées et revient sur le travail accompli
6. **on fait une rétrospective du processus** : la réunion de rétrospective permet de revoir le processus de gestion de projet Scrum, pour détecter des leviers d'optimisation potentiels et être plus efficace lors du prochain sprint
7. **on repart pour un tour** : on choisit de nouvelles tâches depuis le backlog, puis on commence un nouveau sprint

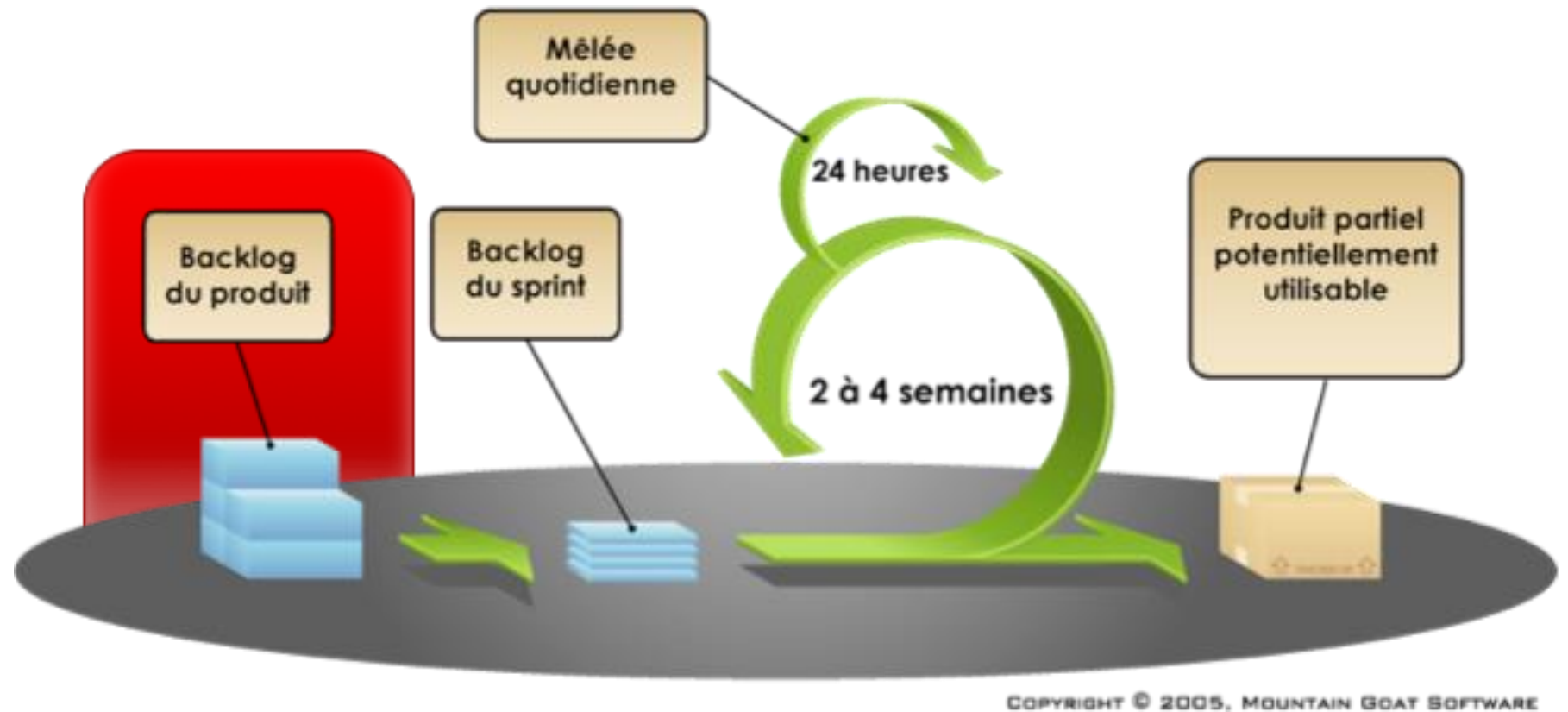
Les cycles de SCRUM : n° 1/6



1. Définition de la vision du produit

- le cap
- une ou deux phases prévues en avance

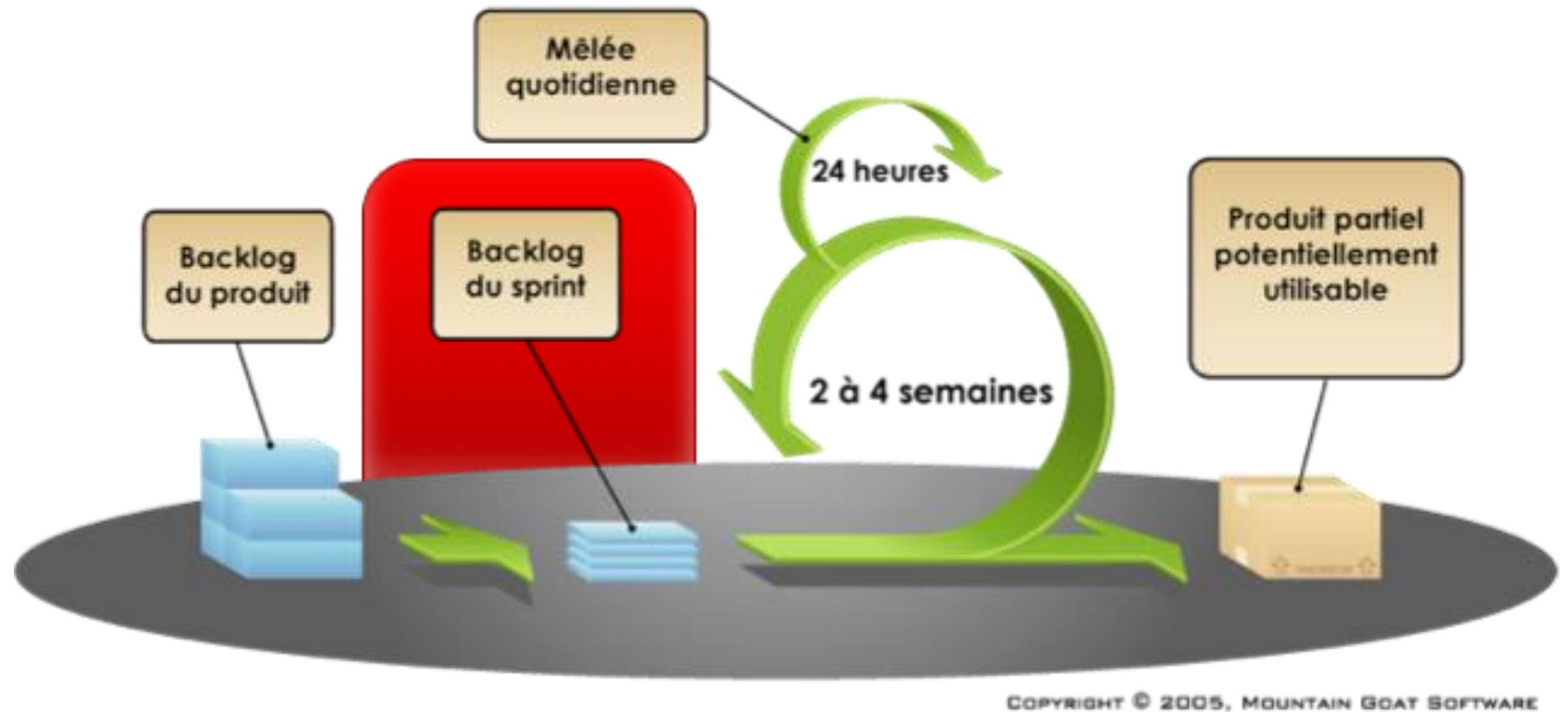
Les cycles de SCRUM : n° 2/6



2. Backlog produit (ou carnet de produit, catalogue des besoins)

- besoins priorisés par le product owner
- besoins estimés par l'équipe, qui évoluent et sont affinés

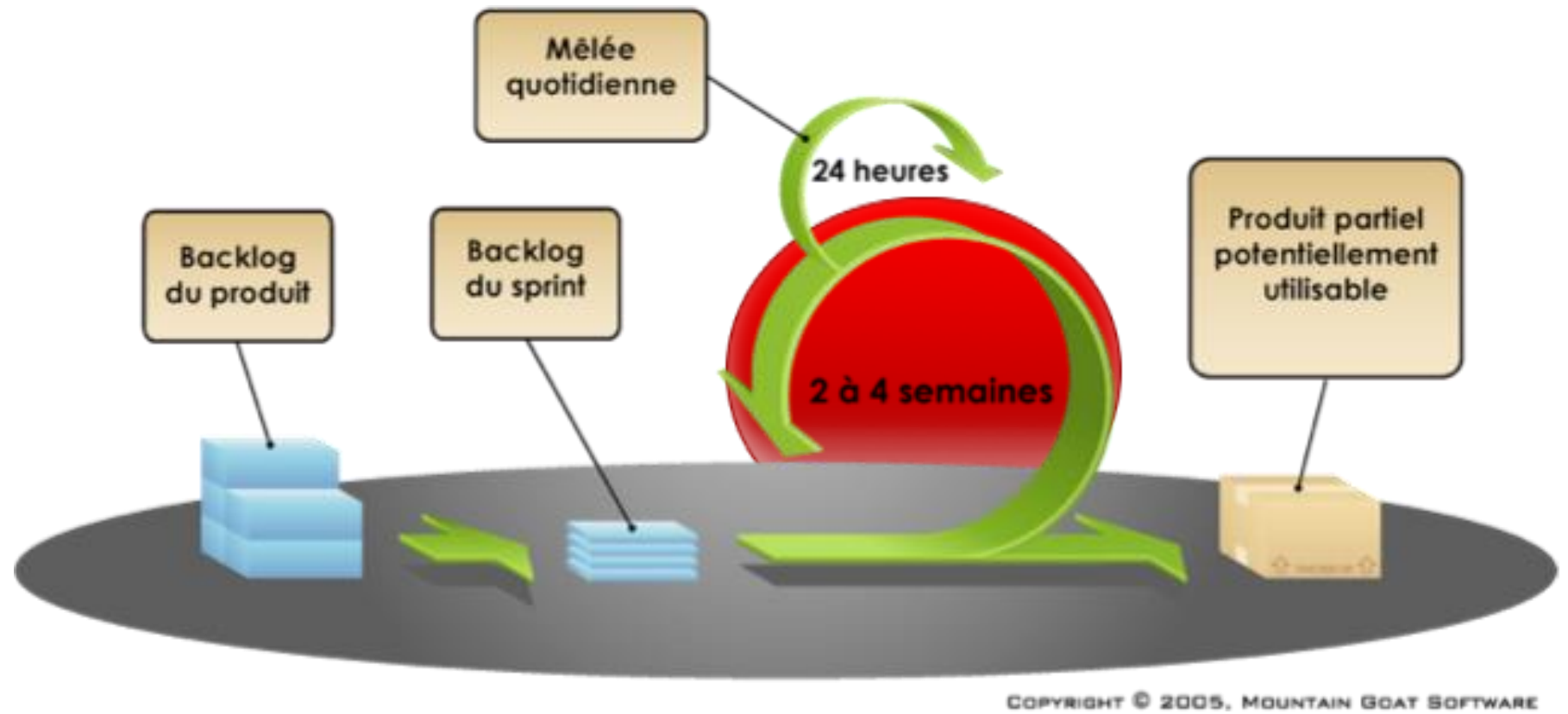
Les cycles de SCRUM : n° 3/6



3. Planning Game : élaboration du backlog de *sprint*

- **sélection** des besoins à faire sur le sprint, extrait du backlog produit
- **découpage** en tâches, répartition de l'effort, planification

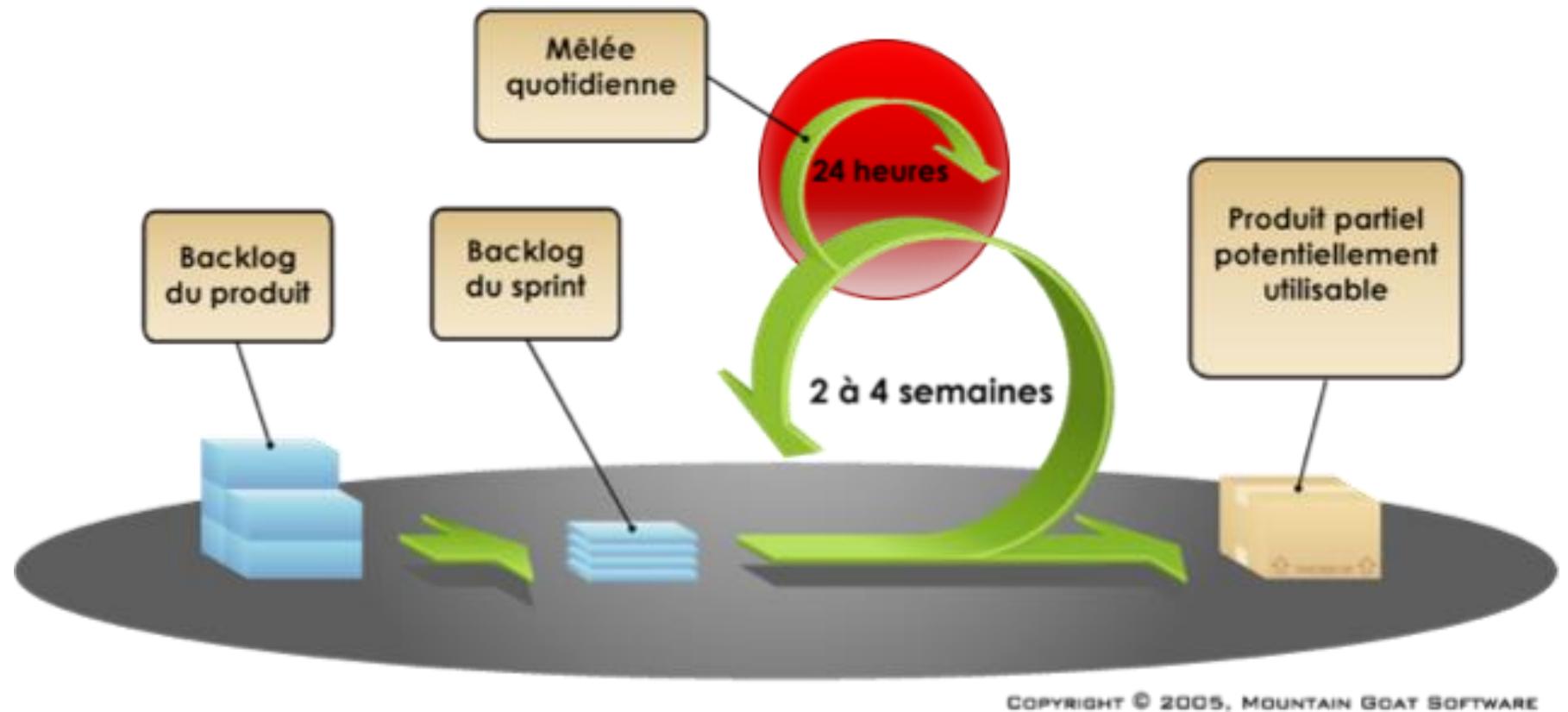
Les cycles de SCRUM : n° 4/6



4. Sprint

- développement des fonctionnalités du backlog de sprint
- pas de modification du backlog de sprint possible
- affinage du backlog Produit : une fois par semaine

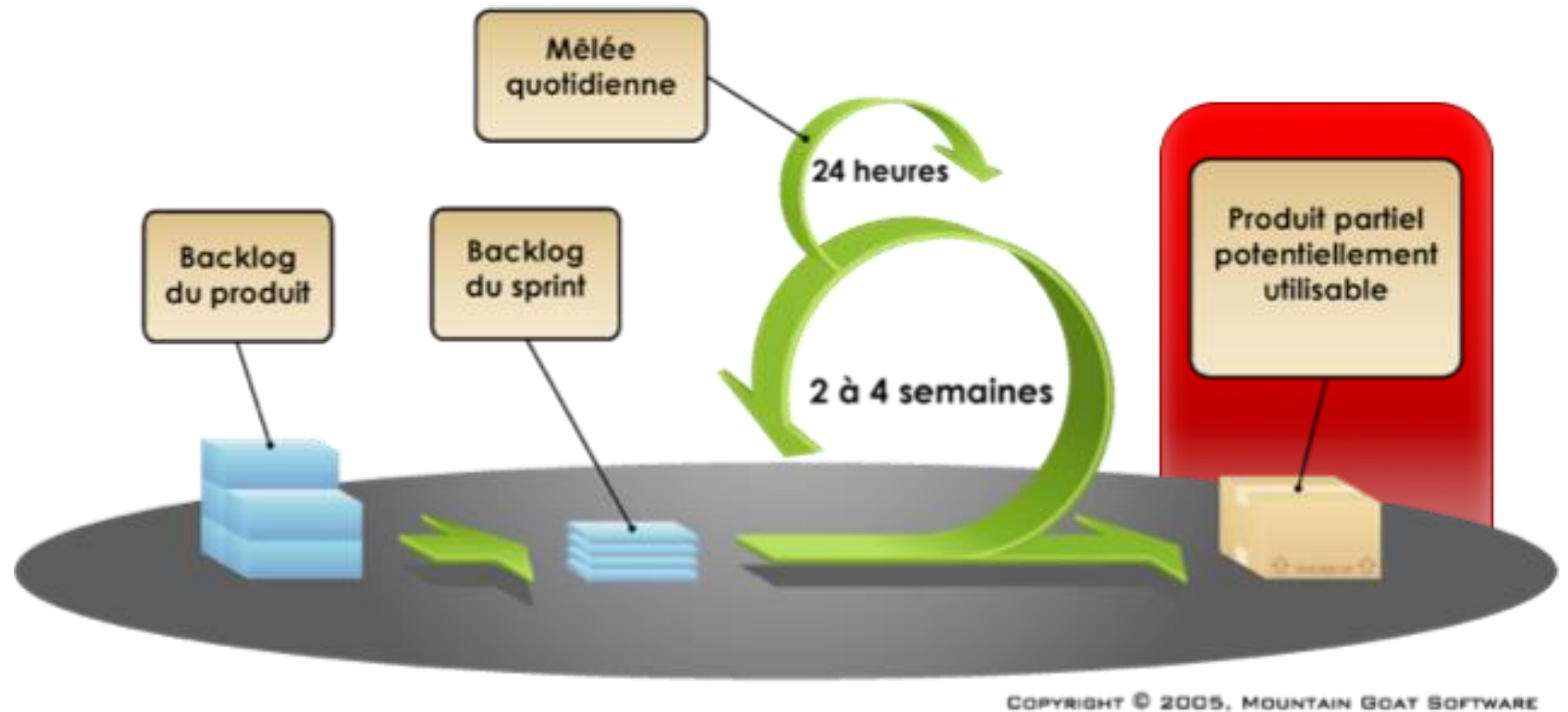
Les cycles de SCRUM : n° 5/6



5. Mêlée quotidienne

- point de contrôle quotidien de l'équipe
- interventions régulées – 2 min. par personne

Les cycles de SCRUM : n° 6/6



6. Incrément logiciel :

- livré au Product Owner à la fin du sprint
- donné aux commerciaux, prospects

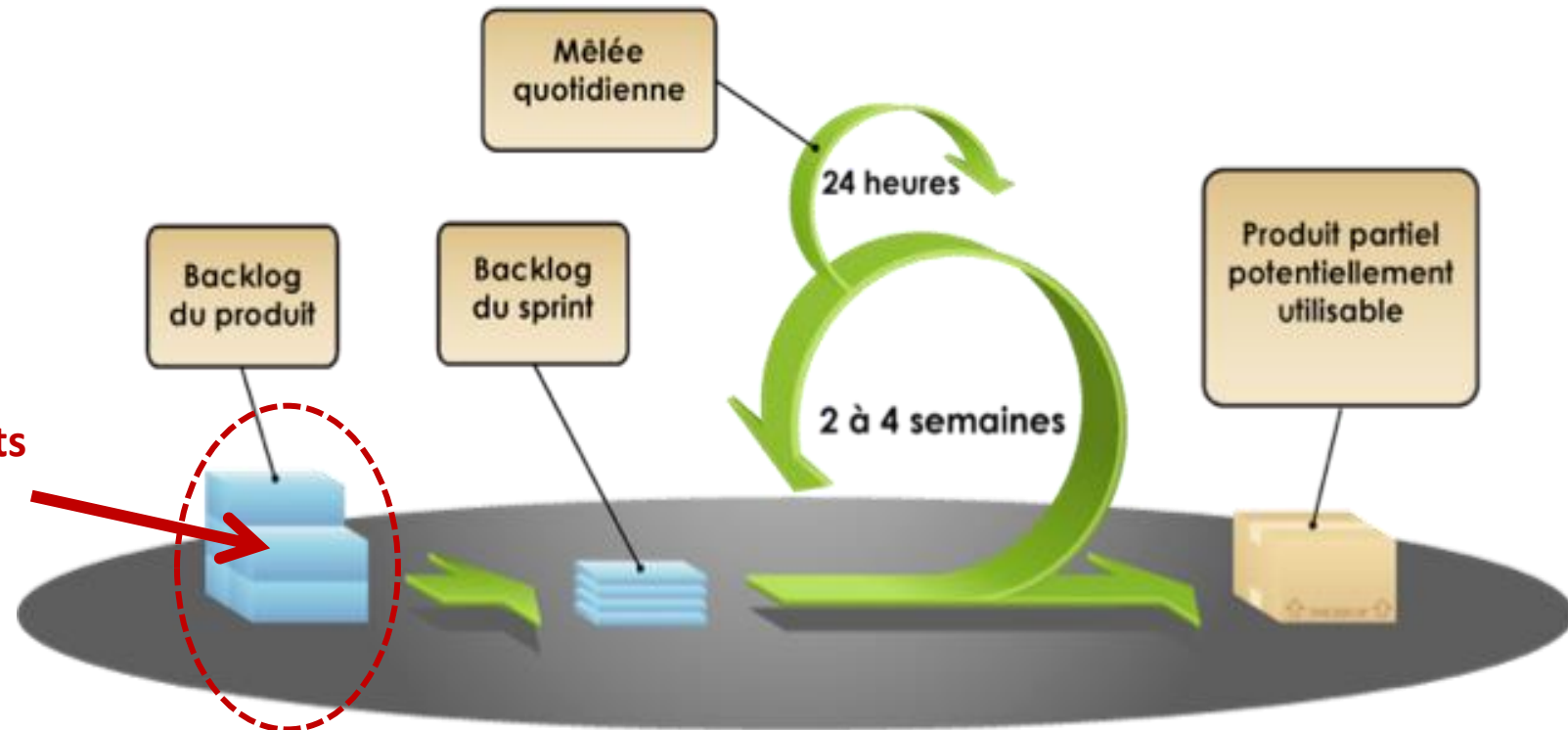
Planification et organisation des itérations

- Planification journalière : daily scrum ou standing meeting : l'objectif, l'organisation
- MEO de l'amélioration continue : animation revues, rétrospectives de fin d'itération
- Principes d'ingénierie : conception simple, amélioration du code par la réécriture, intégration continue
- Indicateurs d'avancement

Planification des Sprints

- **Quand ?** une fois le backlog hiérarchisé
- **En général** : 15 à 30 User Stories par sprint

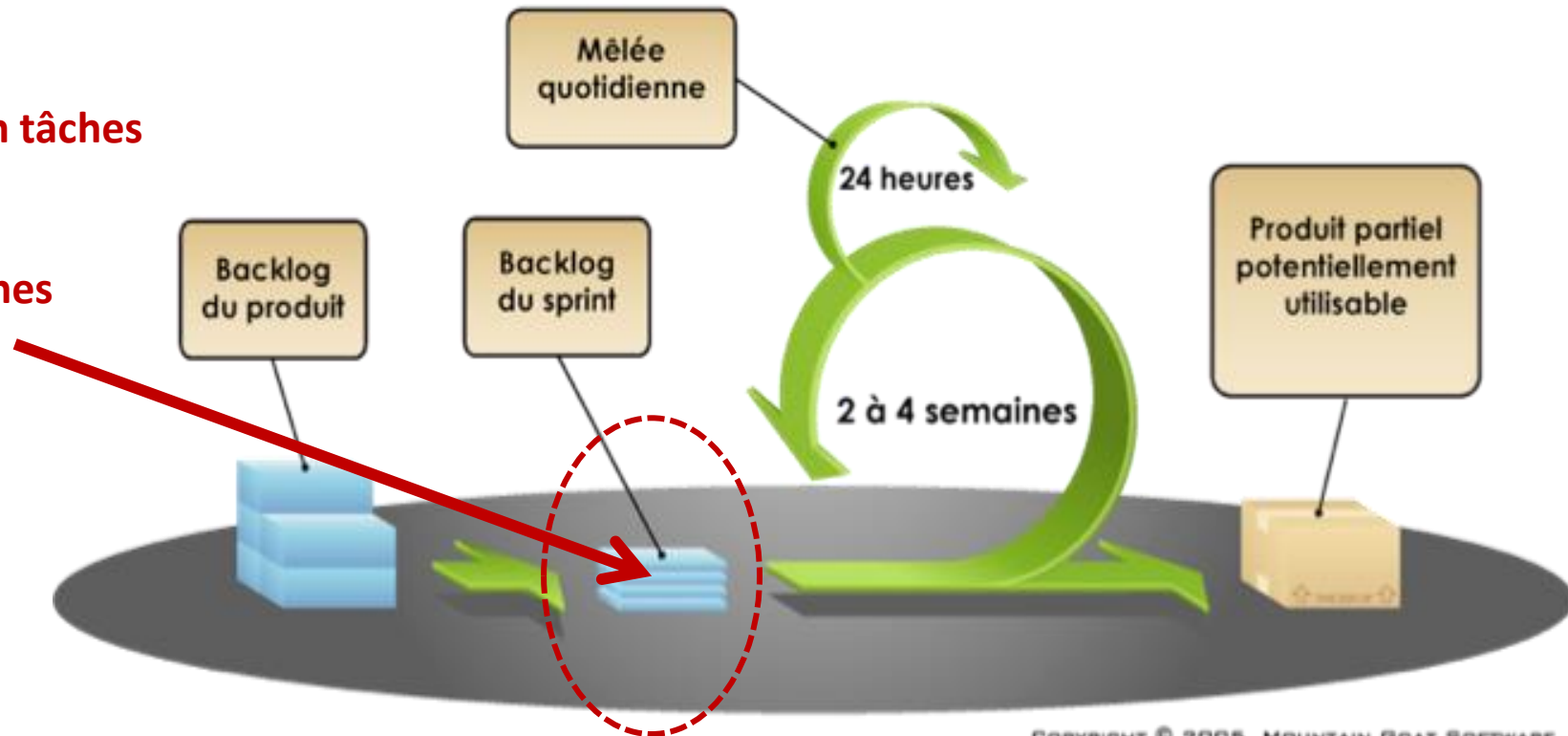
Priorisation Valeur
+
Estimation Effort
+
Planification des Sprints



Planning Game (2^{ème} planification SCRUM)

- **Quand ?** En début de sprint
- **Durée :** maximum 1 h pour un sprint de 1 semaine (2 h pour 2 semaines)

Planning Game :
découpage des US en tâches
+
répartition des tâches



Daily scrum/Standing meeting (objectif & organisation)

Tous les matins :

- une réunion, debout, qui dure tout au plus 15 minutes
- chacun s'exprime 2 à 3 minutes au maximum, et dit ce qu'il a réalisé la veille et compte/doit réaliser ce jour ; il indique au Scrum Master ses éventuels soucis ou ce qui le bloque dans son avancement



Formalisation des exigences

- Techniques de description des besoins fonctionnels et des exigences qualité
- *Features, Users Stories* ➔ Notion de *Backlog*
- Tests d'acceptation
- Construire des stories tests

Besoins fonctionnels, exigences qualité

Caractéristiques des méthodes Agiles

- Les **itérations** sont bornées dans le temps
- Les **réunions** sont bornées dans le temps
- Tout est borné !

Et donc :

- le **coût est prévisible**
- Il a moins de chance d'être **dépassé**
- On **optimise la qualité**



Besoins fonctionnels, exigences qualité

Etape 0 : LA VISION !

- Le cap : si vous ne savez pas où vous voulez aller, où irez-vous?
- Sans vision, les retards sont pris **en tout début** de projet ; l'équipe est alors hésitante, refusant de prendre **le risque de se tromper** de direction...



- **Objectifs :**
 - Situer le produit dans l'organisation
 - Lui donner un sens (raison d'être du produit)



Besoins fonctionnels, exigences qualité

La VISION comme piédestal

- Imaginer un exercice où on doit **spécifier un dessin à faire** → ex : une maison
- On va expliquer :
 - un toit rouge
 - 2 fenêtres avec une porte bleue
 - un banc et des fleurs à droite de la porte

*Vision
Modèle
architecte*



**On risque d'avoir
des interprétations
bien différentes !**

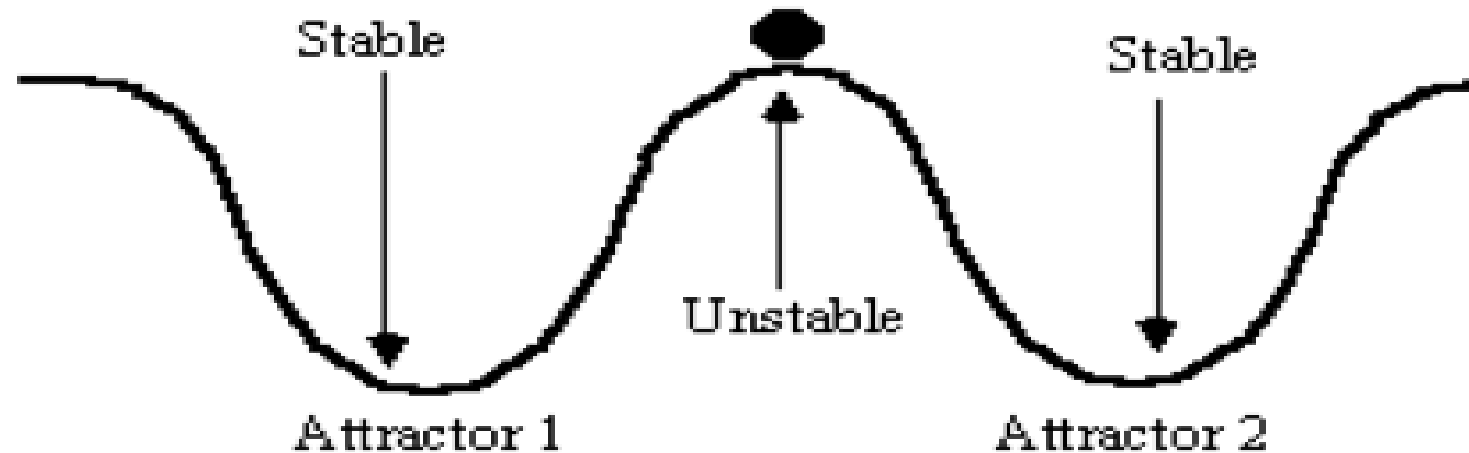


*Vision
Dessin
Animé*

Besoins fonctionnels, exigences qualité

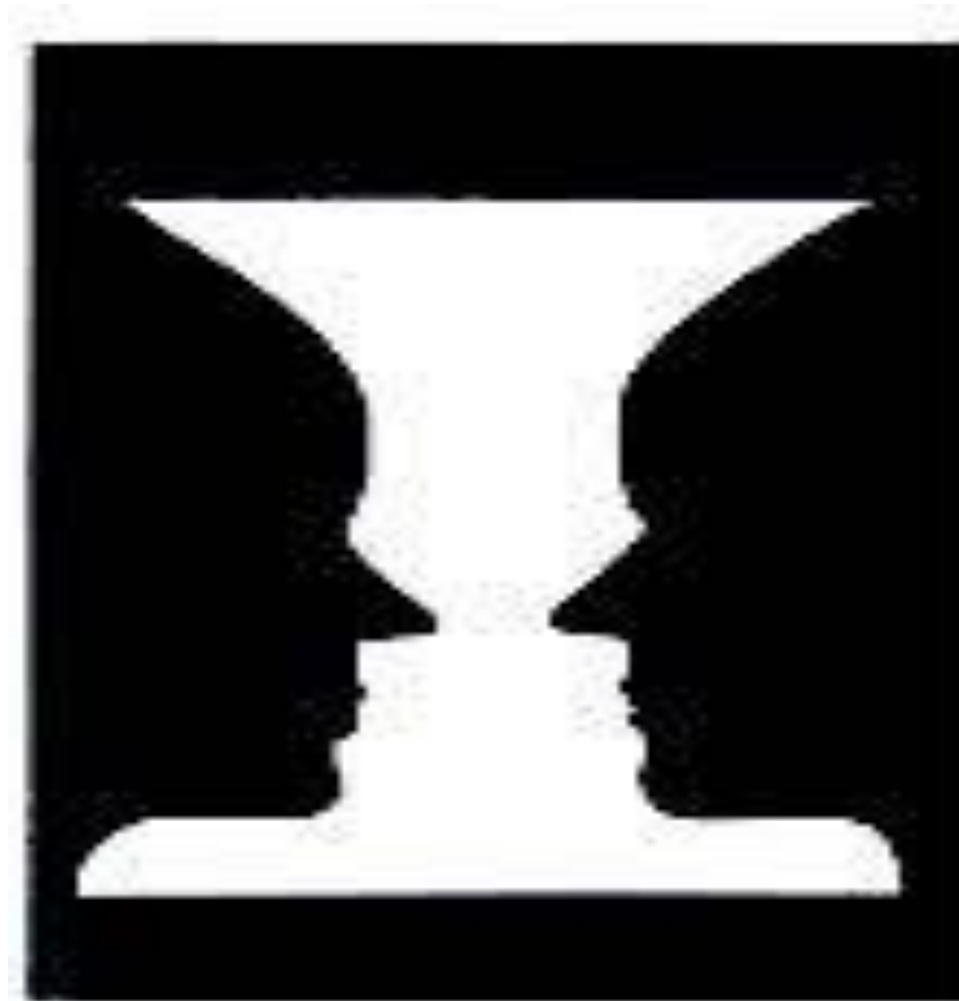
La vision est un ATTRACTEUR

- Selon la théorie de l'auto organisation, la vision est un attracteur de tous les éléments épars qui participent à réaliser une activité
- Elle réunit tous les points de vue vers un objectif commun connu de tous, même de façon inconsciente



Besoins fonctionnels, exigences qualité

Exemples d'attracteurs visuels → QUE VOYEZ-VOUS ? (1/3)



Besoins fonctionnels, exigences qualité

Exemples d'attracteurs visuels → QUE VOYEZ-VOUS ? (2/3)



Besoins fonctionnels, exigences qualité

Exemples d'attracteurs visuels → QUE VOYEZ-VOUS ? (3/3)



Besoins fonctionnels, exigences qualité

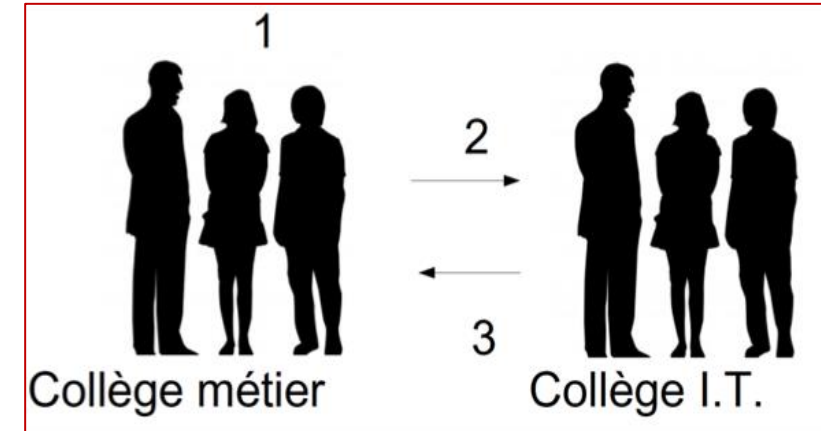
Un autre attracteur : LA MÉMOIRE !

- Un objet est identifié par le cerveau → les stimuli visuels activent la mémoire
- L'ensemble des stimuli s'organise alors par rapport à cet attracteur
- Un effort est nécessaire pour tenter de voir l'autre image
- Dans le cadre de la gestion de projet, la « **Vision Produit** » crée cet attracteur au sein de l'équipe et ce, pour toutes les étapes :
 - Planification, affinage du backlog
 - Choix des solutions techniques
 - Codage
 - Améliorations
 - Etc.

Features, Users Stories → Notion de Backlog

Le « Product backlog » ou le « Carnet de produits »

- Contient des **User Stories**
 - Une demande **fonctionnelle** d'un utilisateur
 - Qui apporte de la **valeur business** au produit
 - Ecrite en langage **naturel**
- Exemple : « **En tant que** client, **je souhaite** pouvoir ajouter un produit dans mon panier **afin de** pouvoir l'acheter »

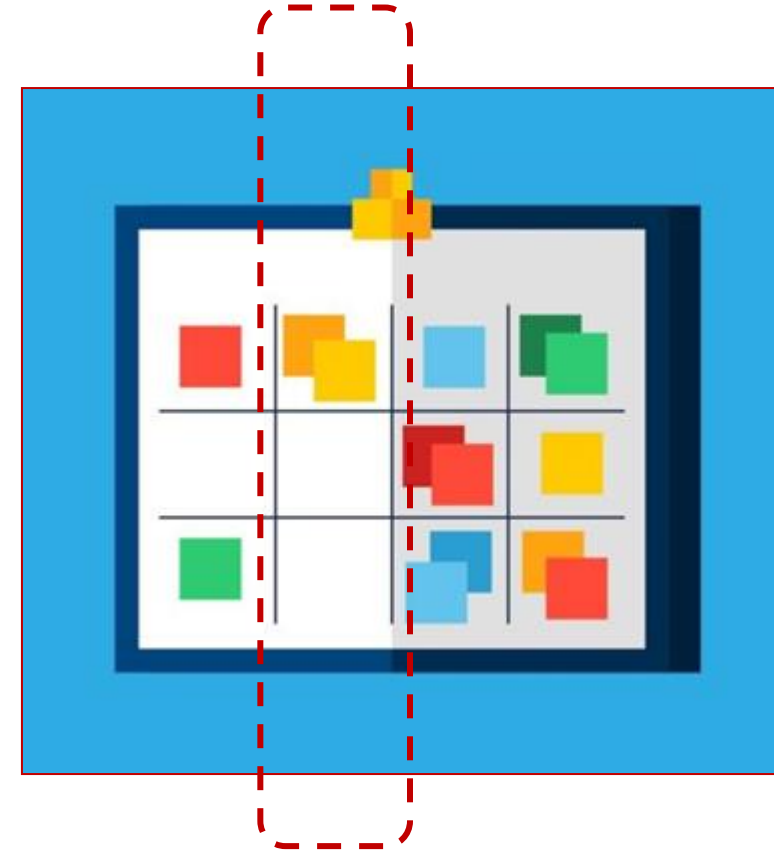


Features, Users Stories → Notion de Backlog

Management visuel des tâches

→ Tableau SCRUM avec des colonnes :

- **Ressources** : toutes les tâches récurrentes.
- **Backlog** : répertoire des tâches en cours ; si mon chef a une requête, je l'ajoute à cette liste
- **To do** : quand je planifie mon sprint, je déplace les tâches du backlog vers cette liste
- **En cours** : quand je commence à travailler sur une tâche je la déplace dans cette liste
- **Contrôle de Qualité** : lorsque les tâches sont complétées, elles sont déplacées dans la liste
- **Fait** : si le contrôle qualité est OK, on est OK pour livrer ; à partir de là, plus de modification !
- **Bloqué** : j'utilise cette liste lorsque la finalisation d'une tâche dépend d'un facteur externe (ex : réaliser un achat et obtenir l'aval de mon chef)



Tests d'acceptation

- Sont définis par le Product Owner et discutés
- Peuvent servir au TDD (*Test-Driven Development*)
- C'est là qu'on discute des **détails** !
- On va également devoir identifier les conditions permettant de dire que la tâche (ou la US) est terminée
- Exemple d'une User Story :

En tant que membre du site,
je peux annuler ma réservation d'hôtel
Afin de tenir compte d'imprévus dans mon voyage



Product Owner

Tests d'acceptation

Reprenons notre exemple de US :

En tant que membre du site,
je peux annuler ma réservation d'hôtel
Afin de tenir compte d'imprévus dans mon voyage

- Il faut donc :
 - vérifier qu'un email d'annulation est envoyé au voyageur et à l'hôtel
 - vérifier que si elle a lieu au moins 15 j. avant la date prévue, pas de charge imputée au voyageur
 - vérifier qu'un *premium* peut annuler n'importe quand sans charge supplémentaire
 - vérifier qu'un *non premium* paie 10% du montant de la résa si annulation moins de 15 j. avant la date prévue

Tests d'acceptation

Que penser de ce test d'acceptation ?



Tests d'acceptation

Que penser de ce test d'acceptation ?

The screenshot shows a 'testBoard' interface with a 'Board Settings' button. The board has several columns represented by colored squares (blue, red, yellow, white, green). A card in the yellow column reads: 'En tant qu'admin, je peux supprimer des users qui ne se sont pas connectés depuis plus de 2 ans'. A card in the green column reads: 'gestion d'un agenda collaboratif'. A 'Feedback' button is visible at the bottom left. A large yellow box on the right contains the following red text:

- Je réussis à supprimer un user qui ne s'est pas connecté depuis 2 ans
- Heu... d'abord, QUI est "Je" ?
- Suppression "d'un ou plusieurs" à la fois ?
- Peut-on annuler la suppression (Undo) ?

Tests d'acceptation

Que penser de ce test d'acceptation ?



The screenshot shows a web application interface for 'testBoard'. At the top, there's a header with the 'testBoard' logo and a 'Board Settings' button. Below the header is a row of colored squares (blue, red, yellow, white, green) and a 'Journeys' button. The main area is a Kanban board with a yellow sticky note on the left and a blue card on the right. The sticky note contains the text: 'En tant qu'admin, je peux supprimer des users qui ne se sont pas connectés depuis plus de 2 ans'. The blue card contains the text: 'gestion d'un agenda collaboratif'. A yellow box on the right side of the board contains a list of acceptance test cases. At the bottom left, there's a green 'Feedback' button and two search icons.

En tant qu'admin, je peux supprimer des users qui ne se sont pas connectés depuis plus de 2 ans

gestion d'un agenda collaboratif

- Vérifier qu'on est **Admin**
- Je peux lister les users ayant une date de dernière connexion > une **durée choisie**
- Je peux alors supprimer *un* user (ou un *bloc* d'users : décider !!)
- Je peux **annuler** (ne rien supprimer en fait)

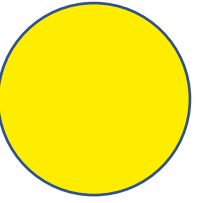
Et toutes ces sortes de choses... !

Construire des stories tests

Comment obtenir les User Stories ?

- C'est le problème du Product Owner
- Méthodes possibles :
 - **brainstorming**
 - **définition de personnas**
 - **identification des scénarios utilisateur, ...**
- Par exemple, pour un **logiciel de gestion de carnet d'adresses**, un scénario est :
 - "Je rentre un nom ou prénom, et le logiciel affiche la liste de toutes les personnes qui possèdent ce nom ou ce prénom"
 - "Je peux choisir d'exporter mon carnet d'adresses au format HTML ou XML"

Construire des stories tests



Et en « vrai », combien faut-il de US ?...

- Au **maximum 20** US au début (pour respecter le vœux agile « **peu de stock** »)
- Principe : « **être précis à court terme, grossier à long terme** »
- Attention : imprécision pour le **long terme** ne signifie pas “inutile”
 - Permet d’estimer le « reste à faire »
 - On ne les oubliera pas plus tard
- Une US = un **résumé formaté** pour avoir une vision rapide de la demande
- Elle sera **détaillée** ensuite avec l’équipe
- Une US doit pouvoir être **implémentée en une itération**
- Une itération doit comporter **au moins 4 US**



Priorisation des *User Stories*

- Planification basée sur la valeur
- Méthode Moscow
- Calcul des valeurs ajoutées & Valeur client de chaque story à planifier dans la release
- Priorisation des stories basée sur le risque et sur la valeur client

Planification basée sur la valeur

Les **étapes et cérémonies** du développement agile avec SCRUM :

- Définir des « user stories »
- Prioriser ces users stories
- Estimer l'effort pour les réaliser
- Hiérarchiser les tâches et planifier le(s) sprint(s)

Bref, en gros, il faut surtout **prioriser** , séparer le futile de l'important

Format d'une User Story type

Formalisme

En tant que... (rôle)
Je peux (tâche)
Afin de (but)

Les user stories proposées par le PO sont **discutées avec l'équipe** pour lever toute ambiguïté de compréhension

Exemple :

- **En tant que** Pilote,
- **Je peux** régler le commutateur en mode « horizontal »
- **Afin de** maintenir les ailes à l'horizontal pour que l'avion reste sur sa trajectoire

Format d'une User Story type

ETQ Assureur, **je peux**
récupérer des contrats
d'assurance sur le site web
afin de vérifier leur précision
et leur légalité

ETQ Assureur, **je peux**
chercher les infos du permis
de conduire du candidat chez
la préfecture concernée **afin**
de vérifier leur exactitude

Pour bien écrire des US correctes

Règles d'écriture

- **REGLE 1 : rester simple** (principe KISS)
- **REGLE 2 : parler du QUOI** (pas du COMMENT)
- **REGLE 3 : rester dans le périmètre** du projet (vision!), et dans le **champ de responsabilités** de l'organisation / du service
- **REGLE 4 : lever l'ambiguïté** des termes
- **REGLE 5 : pratiquer si possible la réécriture** des US

Rester simple

Principe KISS (Keep It Simple & Stupid)

Voici une user story définie pour un logiciel d'assurance auto :

En tant qu'internaute,

je peux naviguer sur le site, saisir mes informations personnelles et celles du véhicule, et soumettre une demande en ligne,

Afin d'obtenir une couverture d'assurance automobile.



➔ Qu'en pensez-vous exactement ?...

KISS sur un exemple

Analyse de la situation :

Il y a trop de « choses à faire » mentionnées
→ pas *simple*

REGLE :

- pas de User Story composée
 - Éviter les **ET** (et **OU**)
 - pour 'je peux'
 - sur les données : données personnelles, données véhicules
 - Éviter les 'à moins que', 'excepté pour'...
- Les scinder

En tant qu'internaute,
je peux naviguer sur le site, saisir mes informations personnelles et celles du véhicule, et soumettre une demande en ligne,
Afin d'obtenir une couverture d'assurance automobile.



KISS sur un exemple

Ici, nous allons donc découper notre US en 3 autres US :

1. « **En tant** qu'internaute, **je peux** naviguer sur le site, **afin** de choisir la couverture d'assurance automobile qui me convient »
2. « **En tant** qu'internaute, **je peux** saisir mes informations personnelles et celles du véhicule, **afin** de comparer les offres d'assurance automobile »
3. « **En tant** qu'internaute, **je peux** soumettre une demande d'assurance automobile en ligne, **afin** d'obtenir un contrat »

En tant qu'internaute,
je peux naviguer sur le site, saisir mes informations personnelles et celles du véhicule, et soumettre une demande en ligne,
Afin d'obtenir une couverture d'assurance automobile.



Lever les ambiguïtés

Versus la règle 4

- Ex., un Responsable du Stock écrit la US :

En tant que Resp. du Stock,
Je peux commander la bonne quantité de produits que nous allons vendre
Afin d'éviter d'avoir des coûts de stock trop élevés

- Qu'est-ce qui est ambigu ici ?
- « la bonne quantité » : quelle valeur ? Quelle unité ? (produit unitaire, palette, chargement camion,...)
- « coûts trop élevés » : idem, subjectif --> valeur seuil
- « nous allons vendre » : quand ? demain, la semaine prochaine ?

Pratiquer la réécriture des US

- **Versus la règle 5**

- A faire faire par un collègue très différent de soi, si possible...
 - **Consigne : ne reprendre aucun nom / verbe identique**
 - **On discute des différences**

- Ex: pour une agence de voyage, Tim écrit :

En tant qu'opérateur du Centre d'Appel,
Je peux saisir au moins 12 réservations par heure en période de forte activité
Afin de réduire le temps d'attente des clients



- Lea propose la réécriture suivante :

En tant qu'agent de voyage,
je suis capable d'effectuer un minimum de 12 demandes de voyages en 60 min durant la partie de l'année la plus chargée
Afin de minimiser les abandons



Estimer la valeur d'une US : méthode Moscow

On prépare 4 fiches « MoSCoW » :

- Must / Should / Could / Won't
- Méthode la plus facile et la plus rapide
- En équipe, on **place les User Stories** sur les fiches :
 - soit l'un après l'autre
 - soit par vote (argumentation, débat)
- Maximum **8 US** par fiche
- Règle : tous les items MUST devront être finis **sur les 2 premiers sprints**

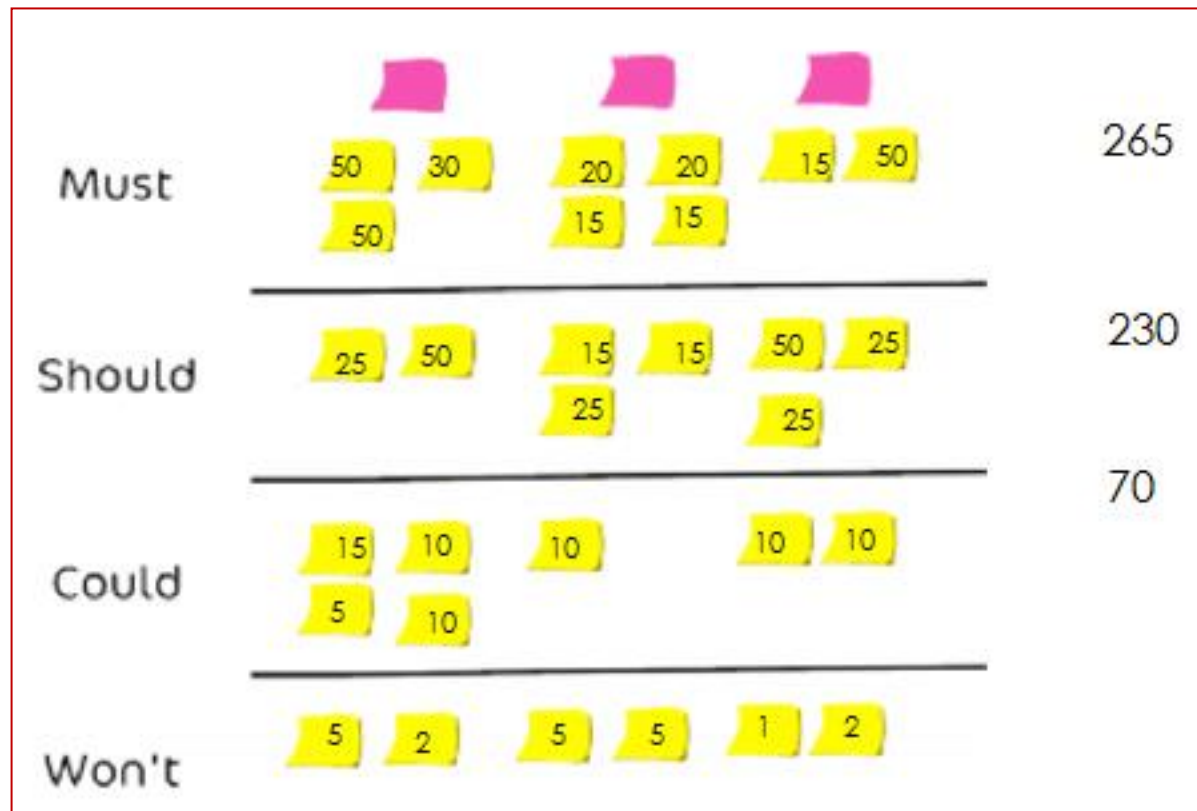


Il en existe d'autres, comme le « Modèle Kano » ou le « Story mapping » par exemple

Estimer la valeur d'une US : méthode Moscow

Quantifier la valeur client

- Une fois la liste priorisée des user stories, on attribue une valeur globale au projet (par ex. 100 ou 500 points) et on répartit les valeurs sur les US
- Il faut rester cohérent avec les priorités



On inscrit la valeur dans un angle de la US

Valeurs ajoutées & Valeur client de chaque story

Priorisation → on retient :

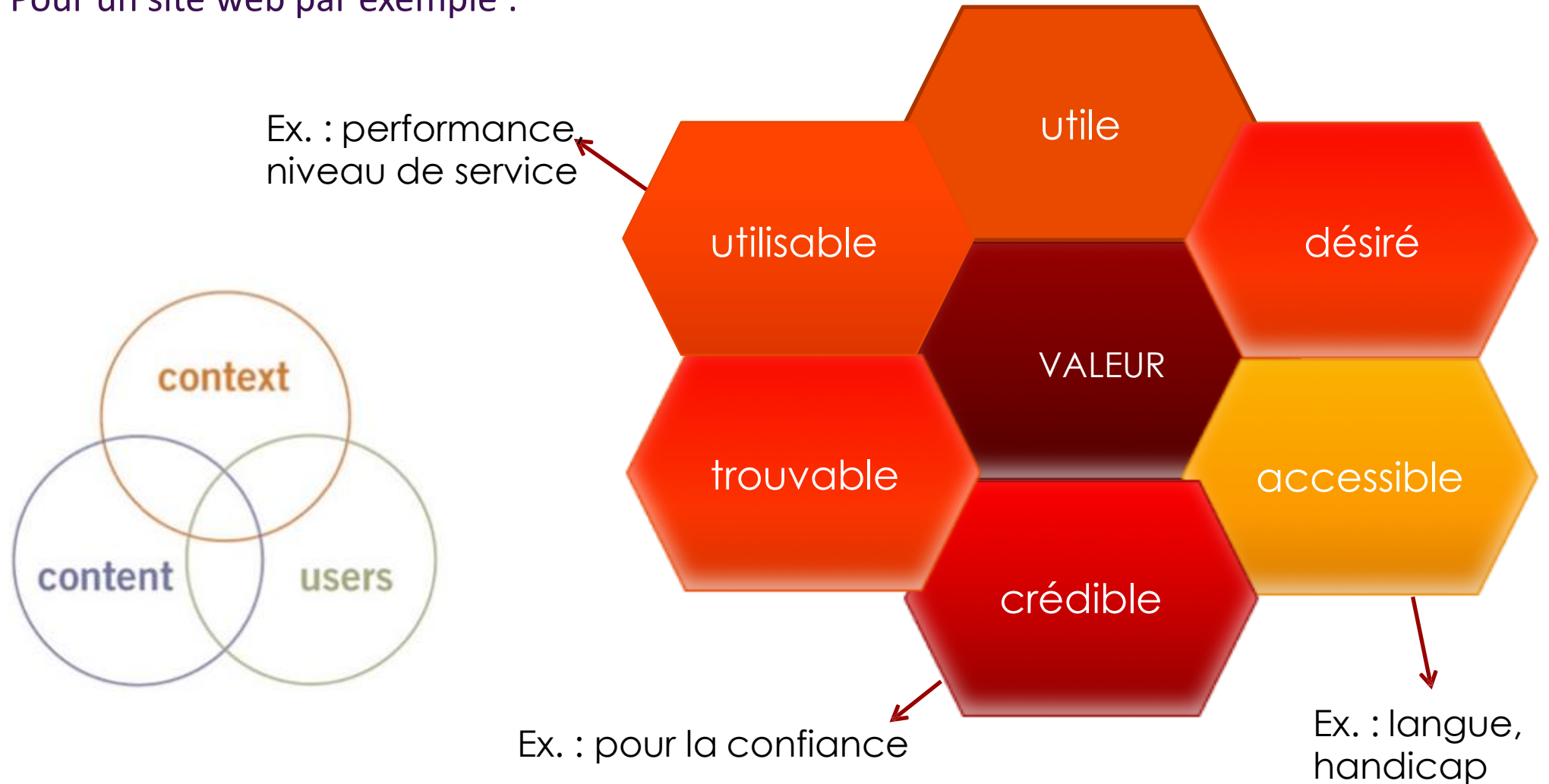
- La somme des points « Valeur » qui définit la **business value** du projet
- On trie les US selon la valeur Métier, définie avec les utilisateurs ou le client
- On hiérarchise le backlog en fonction de cette **VALEUR** et de l'**EFFORT**

Définir la valeur Métier n'est pas évident (*notion d'utilité, de désir, de facilité et fréquence d'utilisation etc.*)

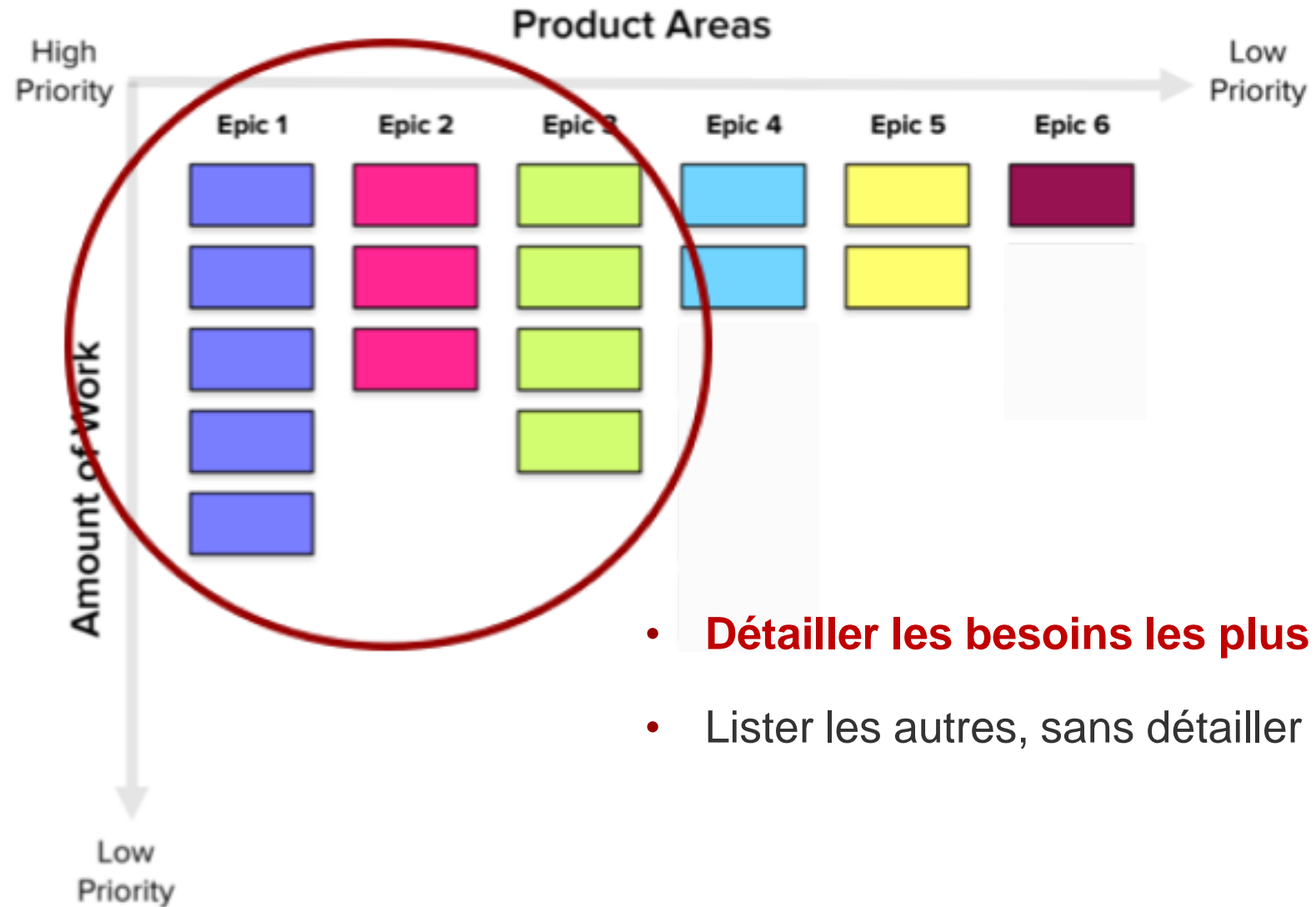
Valeurs ajoutées & Valeur client de chaque story

Qu'est-ce qui définit la valeur métier ? ➔ pas simple comme question !...

Pour un site web par exemple :



Priorisation des items (Cf. risque et valeur client)



- **Détailler les besoins les plus importants**
- Lister les autres, sans détailler

Planification des *releases* et des *sprints*

- Découpage du projet en sprints/releases, construire la roadmap
- Définir les sprints ou les itérations du projet
- Évaluer des charges, évaluation de la taille des stories : le Planning Poker
- Définition de la vélocité de l'équipe

Découpage en sprints/releases → roadmap

Durée des sprints ?

- Elle peut varier de 2 à 4 semaines
- Certains voudraient **1 semaine**, mais c'est court ! (à chaque Sprint, il faut faire 3 réunions = les « rituels de Scrum ») :
 - une Planification du Sprint (avec l'équipe)
 - une Revue de Sprint (équipe et parties prenantes)
 - une Rétrospective (équipe)
- Si les Sprints sont **trop courts** ce temps lié aux rituels risque de **pénaliser la productivité** du projet
- À l'inverse si les Sprints sont **trop longs** le risque de **ne pas détecter à temps** une anomalie est plus élevé

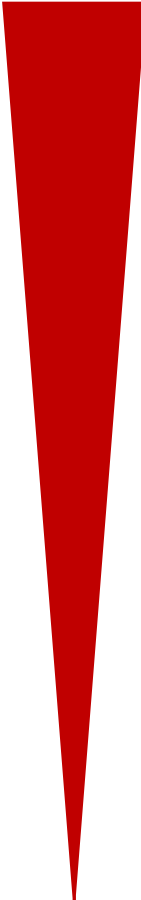
Découpage en sprints/releases ➔ roadmap

Durée des sprints ? (suite)

- Il faut donc un bon compromis sur la durée des Sprints qui est fonction :
 - du projet
 - de la maturité de l'équipe avec l'approche Scrum
- Attention au « Sprint 0 » et au « dernier Sprint »
- Leur durée peut être différente de celle des Sprints « normaux »

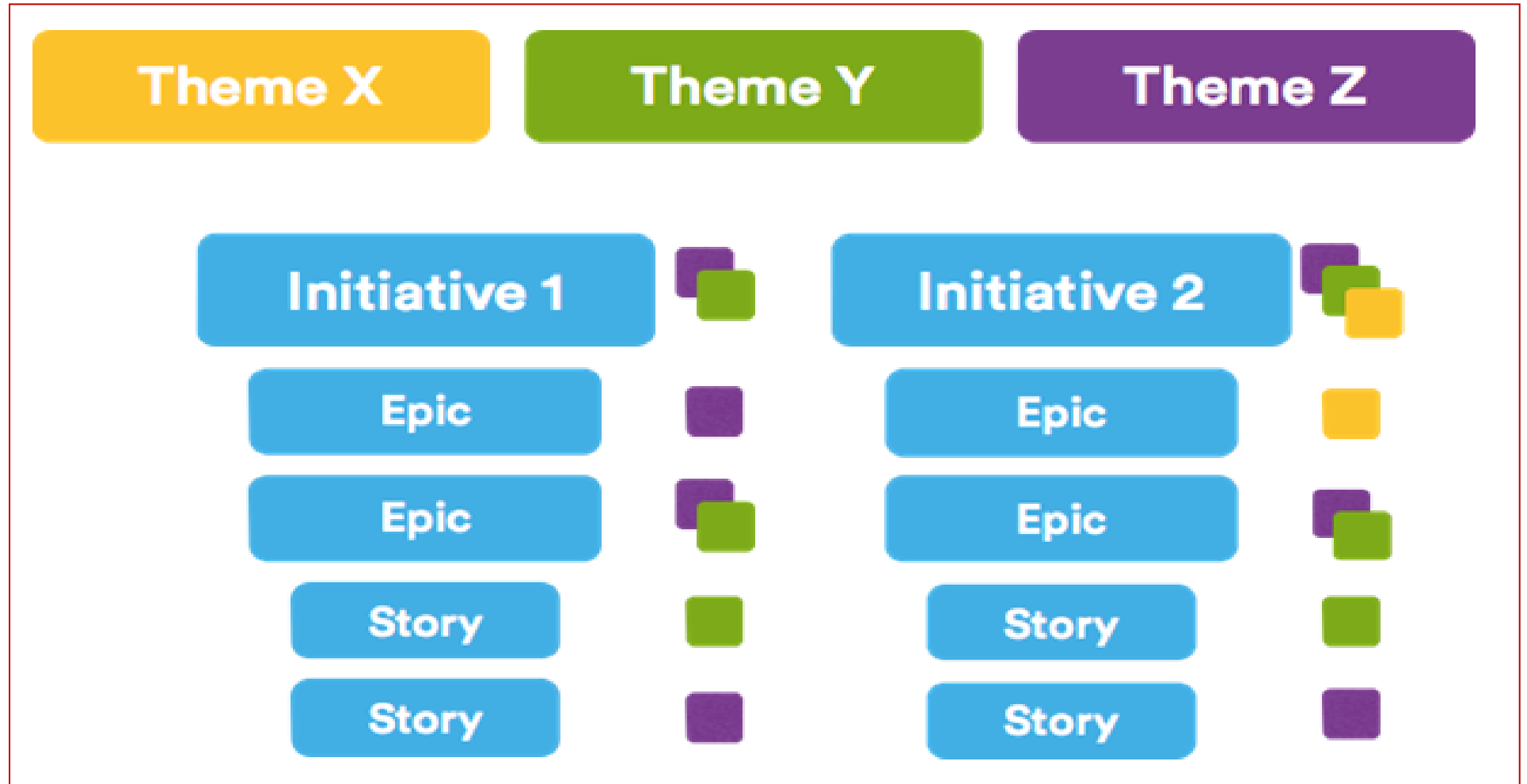
Définir les sprints ou les itérations du projet

Passer par la granularité du besoin

- 
- **Thème** : domaine Produit, grosse Feature, importants domaines d'amélioration à l'échelle de l'organisation
 - **Feature /Initiative**: grosse fonctionnalité Produit, collections d'epics axés sur un objectif commun
 - **Epic** : corpus de tâches importantes qui peuvent être subdivisés en plus petites tâches (appelées stories ou user stories)
 - **User Story** : besoin élémentaire d'un Utilisateur, brèves exigences ou requêtes écrites du point de vue de l'utilisateur final
 - **Task** : tâche (technique) pour faire la US

Définir les sprints ou les itérations du projet

Passer par la granulométrie du besoin (suite)



Définir les sprints ou les itérations du projet

Passe par la granulométrie du besoin (suite)



Notion de « Fini » (DoD)

« Definition of Done »

- Une user story pourra contenir :
 - une description
 - des **tests d'acceptation**
 - un résumé simple
 - etc.
- Les tests d'acceptation permettent de **définir le « DONE »**, c'est-à-dire expliciter le « **Quand** » une US sera considérée comme **terminée** :
 - « ça marche à peu près » : ???
 - « c'est OK pour 80% des data »
 - « c'est tout bon, mais la version Anglaise reste à faire »

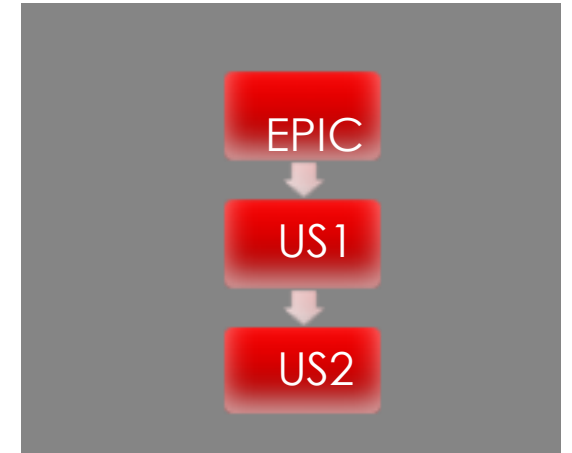
Notion de « Fini » (DoD)

- En Agile, le **DoD** est fondamental : *Definition of Done*
- On se met d'accord AVANT sur ce qu'est une US terminée :
 - Quand les Test Unitaires passent
 - ... Quand en plus, la doc est faite
 - ... Quand en plus, la version anglaise est terminée
 - ... Etc.
- Avec l'Agile, on veut éviter le **syndrome des 90%**, où tout est commencé mais pas encore complètement terminé...
!!!
- On cherche donc à éviter la dette technique



Découper une EPIC (épopée) en stories

- Si la story est trop grosse pour être développée en un seul sprint, ce n'est pas une story, mais une **EPIC** (une épopée !)
- Ex. EPIC : gestion d'un forum, gestion d'un agenda
- Souvent, les EPIC de priorité moindre ne sont pas même découpées en US



Techniques pour découper :

- Par scénario
- Par opération CRUD (*Create, Read, Update, and Delete*)
- Par type de données
- Par niveau de complexité, etc.

Rappel : une US tient sur un sprint, et au minium 4 user stories par sprint

Objectif : PRIORISER les US

- A l'issue du **brainstorming** avec les utilisateurs, de nombreuses idées ont été avancées. L'objectif est alors de *réduire le nombre d'éléments* : les fédérer et les regrouper en thèmes communs
- Comment identifier dans le **Backlog Produit** ce qui est le plus important pour l'organisation ?
 - La VALEUR METIER (business value)
 - Le BUT (« afin de ... ») des US
 - En **SCRUM** : priorisation effectuée par le **PO + utilisateurs**
 - En **XP** : priorisation effectuée par le **Client**

Estimer les efforts, la charge et la taille des US

Estimer l'effort

- Une fois les US affectées de valeur Métier avec le PO, l'équipe va **quantifier l'effort nécessaire pour les réaliser**
- On prend 2 méthodes d'estimation :

- **Planning POKER**



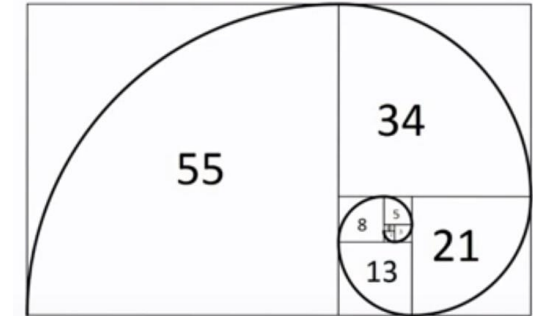
- **T-Shirt Size**



Estimer l'effort avec le Planning Poker

Fonctionne avec les points de Fibonacci :

- 1 (facile), 2, 3, 5, 8, ... (difficile, ... très difficile)
- Chacun dispose des cartes de Fibonacci



- Fonctionnement :
 - les US sont au centre : on en retourne une
 - chaque développeur retourne la valeur d'effort qu'il estime juste (tous en même temps !)
 - **on discute des valeurs extrêmes**
 - on se met d'accord sur l'effort de la story



Planning Poker : si désaccord sur l'estimation ?

- En cas de profond désaccord :
 - Discuter !
 - Origine du problème ?
 - incertitude sur le produit, la fonctionnalité, les technologies ?
- Dans tous les 2 cas → **une solution**
 - mettre la User Story de côté
 - creuser les choses pour diminuer l'incertitude
 - la ré-estimer lors de la prochaine séance
- *On peut aussi prendre la valeur moyenne !*



Estimer l'effort avec le T-Shirt Size

- Certains préfèrent attribuer un **niveau de difficulté** aux US et tâches plutôt que lui donner une *valeur d'effort*
- Plus facile de comparer un tâche par rapport à une autre
- Méthode du T-Shirt Size → on choisit les catégories :
 - **XS, S, M, L, XL**
 - soit définir une référence → ex : M = 2 jours
 - soit choisir une **User Story étalon**, connue et maîtrisée et lui affecter l'effort « qui va bien »
→ Ex : *développer l'interface de saisie de commande Client* estimé = **S**
 - Mesure de **l'effort relatif** → par exemple : l'interface de saisie des infos Client est deux fois plus long : disons **M**



Méthode Tshirt-size

- Les tâches **XL** sont en général **floues**
- Les découper et repositionner
- L'objectif est de trier les *items* en complexité
- On attribue ensuite des **points** de Fibonacci aux catégories
- Niveau difficulté faible ☐ effort de développement faible, par exemple :



XS : 1

S : 3

M : 5 L : 8

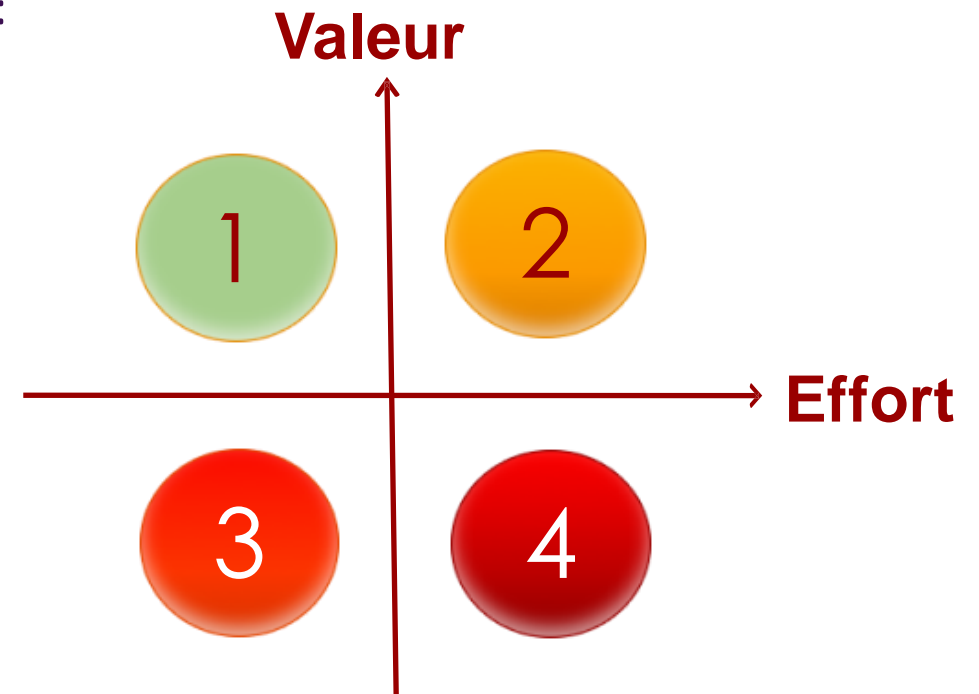
XL : 13

Exemple de Backlog Produit

Backlog item	Acceptance Criteria	Effort	Value
US1 - En tant qu'internaute, je peux réserver l'hôtel en ligne	<ul style="list-style-type: none"> Un email de confirmation est envoyé Réservation doit être faite au min 24h avant date 	8	25
US3 - Améliorer la gestion des exceptions	Pas de msg "Exception ..." même en cas de pb	21	10
US4 - En tant que membre, je peux modifier les dates d'une réservation	...	8	20
US6 - En tant que membre, je peux annuler une réservation	<ul style="list-style-type: none"> Un email de confirmation est envoyé Ne peut être annulée qu'au moins 15 jours avant la date 	11	25
US7 - En tant que Resp. Hotel, je peux voir les réservations à venir	...	21	15

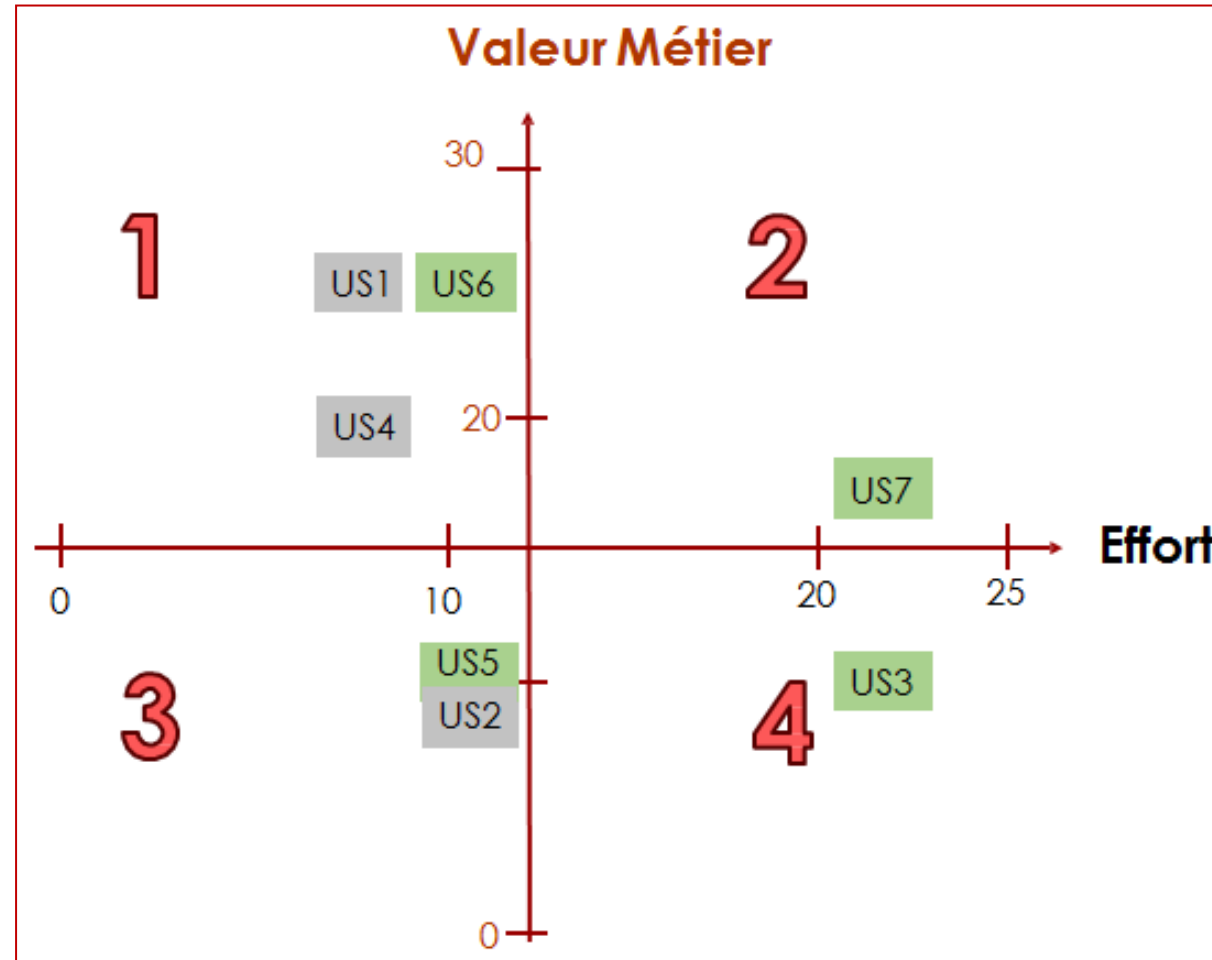
Hiérarchiser le Backlog Produit

- On ne hiérarchise pas en fonction de la VALEUR métier seule
- On hiérarchise en fonction du couple « **Valeur / Effort** »
- Objectif : faire ce qui est **le plus important et le plus facile** d'abord
- Deux méthodes pour hiérarchiser le backlog :
 - **Matrice Valeur / Effort**
 - **ROI**



Sur l'exemple du Backlog Produit

Centre du référentiel = valeur médiane de la Valeur Métier et de l'Effort



Hiérarchiser le Backlog Produit avec calcul de ROI

- Pour chaque User Story, on calcule le ratio :

Valeur Business de la US

Effort estimé

- C'est le ROI (Return On Investment)
- Pour le premier sprint, on choisit les US qui ont le ROI le plus élevé
- Avec les deux méthodes, on obtient un backlog de produit **HIÉRARCHISÉ**
- On sait donc par quoi on va commencer à développer...!

D'où le Backlog Produit hiérarchisé

Priority	Backlog item	Acceptance Criteria	Estimate	Value
1	US1 - En tant qu'internaute, je peux réserver l'hôtel en ligne	<ul style="list-style-type: none"> Un email de confirmation est envoyé Réservation doit être faite au min 24h avant date 	8	25
2	US6 - En tant que membre, je peux annuler une réservation	<ul style="list-style-type: none"> Un email de confirmation est envoyé Ne peut être annulée qu'au moins 15 jours avant la date 	13	25
3	US4 - En tant que membre, je peux modifier les dates d'une réservation	...	8	20
4	US7 - En tant que Resp. Hotel, je peux voir les réservations à venir	...	21	15
...				
7	US3 - Améliorer la gestion des exceptions	Pas de msg "Exception..." même en cas de pb	21	10



MEO des méthodes agiles

- Outils agiles, tableurs, outils spécialisés.
- Etapes de la transition d'une démarche classique vers une approche agile
- Accompagnement du changement, contexte, objectifs du changement et rôle du coach

Outils agiles, tableurs, outils spécialisés

The screenshot displays the Trello web interface. At the top, there's a navigation bar with icons for home, boards, and search. Below this, a specific board named 'ROD V2' is selected. The board is organized into columns: 'Backlog UX/UI', 'Backlog CODAGE', 'Backlog CONTENUS', 'Doing', 'To validate (QA)', and 'Done'. Each column has a '+ Ajouter une carte' button. The 'Backlog UX/UI' column contains several cards, including 'Modification de la page vitrine', 'Email confirmation de paiement', 'Retours V1 : Date des références / affaires', and 'Entreprise "Public vs Privé", "Siège vs Filiale" et "Prescripteur vs Non prescripteur"'. The 'Backlog CODAGE' column has cards like 'RGPD', 'Faire pointer les sites étrangers vers une page d'attente', 'Version Full Responsive : PC, tablettes, smartphones', 'Intégration de l'API SIREN', and 'Mise en place traduction des contenus via LEXIK ou https://localise.biz/'. A card titled 'Entreprise "Public vs Privé", "Siège vs Filiale" et "Prescripteur vs Non prescripteur"' is currently open, showing its details. The card's description includes a list of criteria for qualification: 'Public vs Privé', 'Siège vs Filiale', and 'Prescripteur vs Non prescripteur'. The card also shows an 'Activité' section with a comment from 'Blerouzi' dated 20 mars 2018 à 12:42. On the right side of the card, there are sections for 'SUGGÉRÉES', 'AJOUTER À LA CARTE', 'POWER-UPS', and 'ACTIONS'.

TRELLO, parfait pour gérer un dispositif de Kanban au sein d'un projet géré en mode SCRUM !

Transition : démarche classique → approche agile

- Tout part en général d'un besoin de **transformation digitale** dans l'entreprise
- Comme nous l'avons vu précédemment, cela nécessite :
 1. d'impliquer tous les intervenants
 2. former les équipes à la (ou aux) méthode(s) retenue(s)
 3. mettre en place et respecter des rituels
 4. communiquer et collaborer



Voir le document :

E4_Gestion de projet et Agilité - L'AMELIORATION CONTINUE.pdf

➔ 8 pages



Analyse et évaluation de la situation

7 Gemba : à travers l'approche Gemba (littéralement « là où se passent les choses »), on cherche à comprendre comment se déroulent les processus et comment surviennent les problèmes en les observant directement à la source, c'est-à-dire sur le terrain, et non pas à travers un tableau de bord.

8 Analyse de la valeur : méthode consistant à analyser un produit ou un procédé en vue d'optimiser ses coûts et ses fonctionnalités, dans une perspective de satisfaction du besoin associé à ce dernier.

9 FIPEC : outil de visualisation des données prenant la forme d'un diagramme et dont l'acronyme fait référence aux différentes sections qui le composent : Fournisseurs, Intents, Processus, Externs, Clients. Son équivalent anglo-saxon est le diagramme SIPOC.

10 7S de McKinsey : élaboré dans les années 1980, cet outil permet d'analyser la performance d'une organisation à travers 7 facteurs interdépendants : Strategy (stratégie), Structure (structure), Systems (systèmes), Style of management (style de management), Skills (savoir-faire), Staff (social) et Shared values (valeurs partagées).

11 7QOOOCP : acronyme de « Qui ? Quoi ? Où ? Quand ? Comment ? Combien ? Pourquoi ? ». la méthode 7QOOOCP est un outil de résolution de problème performant et flexible basé sur la collecte d'informations. Elle permet de décrire une situation, identifier un problème et définir des actions correctives.

Recherche des causes et des effets

12 Diagramme Ichikawa : aussi connu sous le nom de « diagramme en arêtes de poisson » ou « diagramme de cause à effet », cet outil est une méthode visuelle dont l'objectif est de trouver les causes d'un effet identifié. Ces causes sont rassemblées dans 5 catégories différentes, les 5M : milieu, méthodes, moyens, main d'œuvre, matière.

13 5 Pourquoi : cet outil participatif repose sur l'idée que pour trouver la cause racine d'un problème, il est nécessaire de se poser au moins 5 fois et de manière successive la question « Pourquoi ? ».

Indicateurs d'avancement

14 KPI (*Key Performance Indicator*) pour mesurer le suivi d'un projet

Pour utiliser efficacement les KPI, il faut que chaque indicateur :

- soit associé à un objectif précis
- soit à la fois réaliste, mesurable et défini dans le temps
- implique forcément une décision (même la décision de ne pas agir)
- Ne soit jamais muet
- soit simple et compréhensible par tous

Indicateurs d'avancement

14 KPI (*Key Performance Indicator*) pour mesurer le suivi d'un projet (suite)

On distingue quatre catégories de KPI :

1. **Les indicateurs de coûts** permettant de contrôler que le projet ne dépasse pas le budget alloué
2. **Les indicateurs de délais** sont indispensables pour vérifier que le projet respecte les délais et ne subit aucun retard
3. **Les indicateurs de qualité** qui aident à veiller à la qualité du travail fourni
4. **Les indicateurs d'efficacité et d'avancement du projet** permettant de savoir si le projet est géré efficacement , si le budget, les ressources et le temps sont utilisés à bon escient, et si la progression du projet est satisfaisante

Indicateurs d'avancement

14 KPI (*Key Performance Indicator*) pour mesurer le suivi d'un projet (suite)

Indicateurs de coûts

- **Coût réel** : calcul du coût réel du projet ; il se mesure en additionnant toutes les dépenses actuelles du projet
- **Coûts non planifiés** : il s'agit de l'ensemble des dépenses réalisées mais non prévues initialement dans le projet
- **Coût des retards**: ce KPI permet de savoir ce que coûte le retard ; il se calcule en faisant la somme des dépenses additionnelles liées au retard comme les heures supplémentaires de main d'œuvre, la maintenance, le dépannage, etc.
- **Ecart de coût du projet**: calcul de la différence entre le coût effectif du projet et le coût initialement prévu ➔ $(\text{coût réel} - \text{coût prévisionnel}) / \text{coût prévisionnel}$

Indicateurs d'avancement

14 KPI (*Key Performance Indicator*) pour mesurer le suivi d'un projet (suite)

Indicateurs de délais

- **Taux de retard** : ce KPI permet de connaître le pourcentage de retard par rapport au planning initialement prévu → $(\text{tâches non réalisées} / \text{tâches prévues}) \times 100$
- **Durée d'une tâche** : utile pour mesurer le temps nécessaire à la réalisation d'une tâche, notamment les tâches récurrentes
- **Ecart de durée** : mesure si une tâche ou un jalon est plus long à réaliser que ce qui avait été initialement prévu → $(\text{durée réelle} - \text{durée initiale}) / \text{durée initiale}$
- **Ecart de délai** : mesure si le projet est en avance dans les temps ou en retard sur le planning prévu au démarrage du projet ; un résultat négatif indique qu'il reste encore du temps pour continuer le projet → $\text{temps initialement prévu} - \text{temps actuellement utilisé}$

Indicateurs d'avancement

14 KPI (*Key Performance Indicator*) pour mesurer le suivi d'un projet (suite)

Indicateurs de qualité

- **Satisfaction client** : excellent indicateur de qualité qui peut se mesurer grâce à un questionnaire de satisfaction ainsi que par la fidélité du client ou du consommateur
- **Nombre d'erreurs** : indique le nombre de fois où une tâche a dû être refaite ou un élément repris ; les erreurs ont un impact sur le budget et le calendrier du projet
- **Les plaintes du client** : tout comme la satisfaction du client, il s'agit d'un bon indicateur de la qualité du projet et du travail fourni

Indicateurs d'avancement

14 KPI (*Key Performance Indicator*) pour mesurer le suivi d'un projet (suite)

Indicateurs d'efficacité et d'avancement du projet

- **Taux d'avancement** : comme son nom l'indique, ce KPI permet de savoir quelle est la progression de votre projet ➔ $(\text{tâches accomplies} / \text{tâches prévues}) \times 100$
- **Nombre de tâches ou de jalons réalisés** : il s'agit d'un autre bon indicateur de l'avancement du projet
- **Temps passé sur le projet** : à tout moment dans le projet, il faut comparer les heures de travail déjà effectuées par rapport aux heures initialement planifiées pour savoir si l'équipe projet est efficace ; si la quantité d'heures passées est supérieure à celle prévue, peut-être faut-il revoir l'estimation du temps nécessaire à la réalisation du projet