# Agilité et pilotage de projets

## Approche classique

### Méthode Waterfall (méthode en cascade)

• **Méthode Waterfall :** méthode de gestion de projet linéaire et séquentielle dans laquelle chaque phase du projet doit être terminée avant que la phase suivante ne puisse commencer

• **Différentes étapes :** définition des exigences, conception, mise en œuvre, tests, maintenance

• **Avantages :**

— Facile à comprendre, à utiliser et à gérer

— Discipliné

— Documentation de qualité et détaillée

— Implication minimale du client

— Départementalisation

• **Inconvénients :**

— Pas bien adaptée aux projets longs ou à grande échelle

— Peu ou pas de modifications ou de révisions

— Livraison tardive du produit

— Impossibilité de rassembler des exigences précises et complètes

### Cycle en V

• **Cycle en V :** méthode de gestion de projet qui consiste à effectuer des activités de façon séquentielle, en allant de la phase de spécification à la phase de validation, avant de remonter dans la phase de réalisation et de tester les différentes composantes du produit

• **Différentes étapes :**

— Définition des besoins, étude de faisabilité : compréhension des attentes et exigences

— Spécification : rédaction du cahier des charges fonctionnel

— Conception générale : rédaction du cahier des charges technique

— Conception détaillée : structuration par composants

— Réalisation

— Tests unitaires : test du fonctionnement des composants Équipe « de développement »

— Tests d’intégration : tests du fonctionnement des composants entre eux

— Validation : vérification que le produit répond bien au besoin initial

— Recette : validation du produit par rapport aux exigences initiales

• **Avantages :**

— Facile à mettre en œuvre

— Rigoureux et intuitif

— Chaque étape est validée avant la suivante

— Temporalité plus facilement repérable

• **Inconvénient :**

— Le client ne voit pas l’avancée du travail avant l’étape de validation

— Manque de souplesse

— Manque de feedback car tout repose sur l’analyse du besoin

— Cloisonnement des équipes

— Travail en silo : approche organisationnelle où chaque département, chaque équipe ou chaque personne travaille de manière indépendante et isolée, sans communiquer ni collaborer efficacement avec les autres

— Méthode rationnelle mais pas adaptée à la logique humaine

## Problématiques de l’approche classique

• **Effet tunnel :** pendant toute la durée du projet, le maitre d’ouvrage et le maitre d’œuvre ne se parlent pas. Conséquence : le résultat est le plus souvent extrêmement décevant, quand le projet n’est pas complètement raté

• **Limites des méthodes classiques :**

— Nécessite absolue d’une spécification complète et immuable

— Rigidité du contrat

— Traitement discontinu de chaque fonctionnalité

— Production de valeur déportée à la fin du projet

— Mode de communication dégradé

• **L’approche classique et l’évolution :**

— L’approche classique n’est pas adaptée aux évolutions technologiques.

— La loi de Moore stipule que les ordinateurs deviennent au fil du temps plus petits, plus rapides, moins chers, à mesure que les transistors sur circuits intégrés deviennent plus efficaces.

— La loi de Nielsen postule que la vitesse de connexion au réseau pour l’utilisateur final augmente de 50% par an, soit le double tous les 21 mois.

— Mais il y a toujours des produits qui n’évoluent pas comme le stylo BIC.

## L’Agile

• L’agilité n’est pas une méthode de travail mais une philosophie.

• **Quatre préceptes :**

— **Les individus et leurs interactions, plus que les processus et outils :** les personnes et leur communication doivent être la priorité, plutôt que de se concentrer uniquement sur les processus ou les outils

— **Des logiciels opérationnels, plus qu’une documentation exhaustive :** il est important de fournir rapidement des résultats opérationnels, plutôt que de passer beaucoup de temps à écrire une documentation détaillée qui pourrait ne pas être pertinente dans le futur

— **La collaboration avec les clients, plus que la négociation contractuelle :** les clients doivent être impliqués dans le processus de développement et il faut travailler ensemble de manière étroite et collaborative

— **L’adaptation au changement plus que le suivi d’un plan :** les méthodes agiles privilégient la flexibilité et l'adaptabilité, plutôt que de suivre strictement un plan préétabli. Les équipes doivent être en mesure de s'adapter rapidement aux changements de situation et aux besoins des clients

• **12 principes Agile :**

— Satisfaire le client est la priorité

— Accueillir les demandes de changement

— Livrer le plus souvent possible les versions opérationnelles de l’application

— Assurer une coopération permanente entre client et équipe projet

— Construire des projets autours d’individus motivés

— Privilégier la conversation en face à face

— Mesurer l’avancement du projet en termes de fonctionnalités de l’application

— Faire avancer le projet à un rythme soutenable et constant

— Porter une attention continue à l’excellence technique et à la conception

— Favoriser la simplicité

— Responsabiliser les équipes

— Ajuster, à intervalles réguliers, son comportement, ses processus pour être plus efficace

## Les méthodes Agile

• **RAD (Rapid Application Development) :**

— Considérée comme étant la première des méthodes agiles

— Développement incrémental : on ajoute des fonctionnalités au fur et à mesure du développement

— Mode de développement semi-itératif

— L’expression du besoin est déterminée en début de projet

• **DSDM (Dynamic Systems Developmennt Method) :**

— Mise en place au milieu des années 90 en Royaume-Uni

— Se base sur RAD et apporte 9 principes :

> se concentrer sur le besoin métier

> respecter les délais

> coopérer et collaborer activement entre toutes les parties prenantes

> ne jamais compromettre la qualité

> contrôler le projet en continu, notamment en utilisant des livraisons fréquentes et régulières

> donner un rôle clé aux utilisateurs finaux dans le projet

> prévoir des livraisons incrémentales et itératives

> favoriser une approche centrée sur le produit plutôt que centrée sur le projet

> adapter la méthode DSDM en fonction des spécificités de chaque projet

• **FDD : Feature Driven Development :**

— Itérations courtes et grandes

— Meilleure communication durant l’ensemble du développement

— Livraisons fréquentes avec de vrais travaux terminés

— Informations de progression et d’état précises et significatives pour un minimum de cout et de perturbation pour les développeurs

— Processus appréciés par les clients, développeurs et manageurs

— Se divise en cinq étapes :

> créer le modèle du système avec un diagramme de classes UML

> faire la liste des fonctionnalités à réaliser

> assigner les fonctionnalités aux développeurs

> créer le modèle de chacun des fonctionnalités

> développer chacune des fonctionnalités

— Contrairement au SCRUM et à l’Extrême programming, le FDD recommande fortement d’assigner les fonctionnalités à un ou des développeurs précis.

• **Méthodes agiles Crystal :**

— Créées au milieu des années 90 par Alistair Cockburn

— Centrées sur les individus plutôt que sur les processus

— 5 points clés pour une bonne gestion de projet agile : communication, réflexion, amélioration, adaptation et travail en équipe

— Plusieurs méthodes agiles Crystal selon la taille d’équipe et le type de projet : un code de couleur est utilisé pour identifier le poids de la méthode agile à utiliser en fonction du projet

• **XP (eXtreme Programming) :**

— Né en 1996 et publié en 1999

— Elle s’appuie sur :

> des cycles de développement courts

> des tests unitaires automatisés

> une intégration continue

> une conception simple

> un refactoring régulier du code

> un travail en binôme (pair programming)

> une forte implication du client

> un planning et une estimation en termes de semaines plutôt que de mois

> une adaptation continue au changement

— 5 valeurs fondamentales : communication, simplicité, feedback, courage, respect

## Scrum

• **Scrum :** cadre de travail (framework) au sein duquel les acteurs peuvent aborder des problèmes complexes et adaptatifs, en livrant de manière efficace et créative des produits de la plus grande valeur possible

### Pilotage par la valeur

— Le client définit son besoin et le priorise.

— Avant chaque itération, le client et l’équipe définissent ensemble ce qui sera développé durant l’itération selon les priorités fixées.

— Le besoin et les priorités peuvent évoluer **entre et pendant** chaque itération.

— La valeur acquise du logiciel est d’autant plus importante sur les premières itérations.

### Incrémental et itératif

• Scrum se base sur une approche itérative et incrémentale du développement :

— **Itératif :** le processus de développement se déroule en plusieurs itérations successives, chacune produisant un résultat partiel fonctionnel et testé. À chaque itération, l'équipe de développement travaille sur un ensemble de fonctionnalités à réaliser et à livrer. Chaque itération permet de recueillir des retours et des feedbacks sur le travail effectué, qui permettent de réajuster les priorités et d'orienter la suite du développement.

— **Incrémental**: le développement avance par petites étapes progressives (appelées incrément), en ajoutant des fonctionnalités au fur et à mesure. Cela permet de livrer des versions partielles fonctionnelles régulièrement tout en minimisant les risques et les erreurs.

### Les 3 piliers de l’Empirisme

• Pratiquer l’empirisme revient à se baser sur l’expérience acquise pour optimiser l’environnement de production et le produit final.

• Voici les 3 piliers :

— **Transparence :** présenter les faits tels qu'ils sont, de manière ouverte et honnête, entre les membres de l’équipe et les clients, afin de favoriser la confiance et de prendre des décisions éclairées en se basant sur des informations fiables.

— **Inspection :** vérifier le travail effectué tout en le réalisant, afin de détecter rapidement les écarts entre les prévisions et la réalité

> L’inspection n’est pas réalisée par un inspecteur ou un auditeur, mais par tout un chacun dans l’équipe Scrum.

> L’inspection peut être appliquée au produit, aux processus, aux aspects humains, aux pratiques, et à l’amélioration continue.

> L’équipe présente ouvertement et de manière transparente le produit au client à la fin de chaque Sprint, afin de recueillir de précieux avis (sprint review). Si le client modifie son besoin pendant l’inspection, l’équipe ne s’en plaint pas mais exploite cette opportunité de collaboration en s’adaptation à cette clarification des exigences et à l’expérimentation de cette nouvelle hypothèse.

— **Adaptation :** capacité à changer de direction tactique en fonction des résultats de l'inspection, afin de limiter les risques liés aux écarts détectés et de garder le projet sur les bons rails en réduisant le risque de nouvelles dérives. Si un écart sort des limites de l’acceptable et entraine une production jugée médiocre, des ajustements doivent être décidés. L’adaptation est assurée par les cérémonies Scrum (daily scrum meeting, sprint planning meeting, sprint review meeting, sprint retrospective meeting).

### 5 valeurs

• **Courage :** les membres de l’équipe Scrum ont le courage de faire les bonnes choses et de travailler sur les problèmes difficiles

• **Focus :** chacun se concentre sur le travail du Sprint et les objectifs de l’équipe Scrum

• **Engagement :** les gens s’engagent personnellement pour atteindre les objectifs de l’équipe Scrum

• **Respect :** les membres de l’équipe se respectent les uns les autres pour être des personnes compétences et indépendantes

• **Ouverture :** l’équipe Scrum et ses parties prenantes acceptent d’être ouverts au sujet de tout le travail et des défis liés à l’exécution de ce travail

### La maitrise du temps

• **Timeboxing (temps limité ou boite de temps) :** définir des limites de temps strictes pour chaque activité ou cérémonie de manière à respecter un rythme régulier et à encourager une meilleure planification et organisation du travail

• En respectant les durées prévues, l'équipe est encouragée à se concentrer sur les tâches essentielles et à éviter les gaspillages de temps. De plus, en limitant le temps consacré à chaque activité, Scrum permet de mieux contrôler le temps de développement du produit, d'adapter rapidement la stratégie en cas de besoin et de faciliter l'engagement des parties prenantes.

• Durées typiques (à adapter proportionnellement selon la durée de l’itération) :

— Une itération doit durer au maximum 4 semaines.

— La planification d’itération dure 8 heures pour une itération de 4 semaines.

— La mêlée dure ¼ heure.

— La démo dure 4 heures pour une itération d’un mois.

— La rétrospective dure 3 heures pour une itérations d’un mois.

### Rôles principaux

#### Équipe Scrum

• **Équipe Scrum :**  constituée de tous les membres impliqués dans le projet, avec pour objectif de satisfaire les utilisateurs et de fabriquer le meilleur produit possible

#### Scrum Master

• **Scrum Master :** responsable de faciliter les échanges et l’amélioration

##### Différence entre Scrum Master et chef de projet

• Le chef de projet participe à la création du projet, tandis que le Scrum Master garantit la bonne tenue du projet qu’il n’a pas créé.

• On peut le différencier sur 3 points :

— **le résultat :** contrairement au chef de projet technique, le Scrum Master n’est pas garant de la qualité technique ou fonctionnelle du produit

— **la communication :** lorsque le Scrum Master s’adresse à l’écosystème, c’est pour éduquer sur la méthodologie ou régler les problèmes qui barrent la route de l’équipe Scrum. Il ne discute pas des points techniques du projet, communications réservées à l’équipe de développement

— **le planning :** le Scrum Master n’intervient pas dans la planification du projet, il s’assure que l’équipe fournisse un produit de la meilleure qualité possible, il répond à une logique d’amélioration continue et non de résultat.

##### Méthodologies et organisation

• **Focus sur l’efficacité de l’équipe Scrum :** il assure la bonne application de Scrum et de l’agilité, l’amélioration des process de l’équipe : il fait en sorte que la Scrum team apprenne à s’autogérer

• **Organisateur :** accompagne l’équipe sur tous les évènements Scrum, s’assure que les réunions soient constructives : inspection et adaptation

• **Facilitateur :** fluidifie la communication entre les acteurs, s’occupe de tout ce qui pourrait déranger l’avancée du projet

• **Parfois un médiateur :** interne au sein de l’équipe ou externe avec le client/parties prenantes

##### Compétences et comportements

• **Compétences :**

— une base de connaissances Scrum et Agile

— une capacité à transmettre et à évaluer les acquis

— une intelligence relationnelle

• **Comportements :**

— Il n’exploite pas les compétences, il motive les individus.

— Il n’est pas au-dessus, il est dans l’équipe.

— Il ne dirige pas, il guide.

— Il ne cherche pas qui a fait la faute, il accompagne jusqu’à la solution.

##### Les outils du Scrum Master

###### Le Burn Down Chart

• **Burn Down Chart :** indicateur de mesure qui montre le travail restant sur une période donnée

• Il permet de suivre le projet dans le temps afin de contrôler sa livraison dans les délais souhaités.

• Sur l’axe X, s’affiche le nombre de jours que dure le sprint. Sur l’axe Y, s’affiche l’effort en termes de story-point (ou de tickets) de l’équipe.

• Le taux de progression de l’équipe exprime le nombre de stories-points (ou de tâches) complétés par itération.

• Seules les tâches terminées à la fin de l’itération sont comptées.

###### Le Burn Up Chart

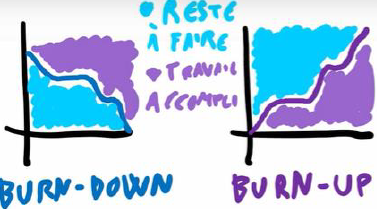
• Il est utilisé pour suivre l’avancement d’une Release (horizon moyen ou long terme).

• Il permet de suivre l’évolution de la quantité de travail terminée en fonction du temps.

• Le but consiste donc à atteindre la cible (haut du graphique) le plus tôt possible d’où le terme « Up ».

###### Burn Up Chart Vs le Burn Down Chart

• Les deux charts sont complémentaires. Le travail accompli et le reste à faire sont directement liées.



• Si le périmètre est figé (la quantité de travail à faire ne varie pas) les deux charts montrent la même information.

• Si le périmètre est variable, le Burn Up permet de :

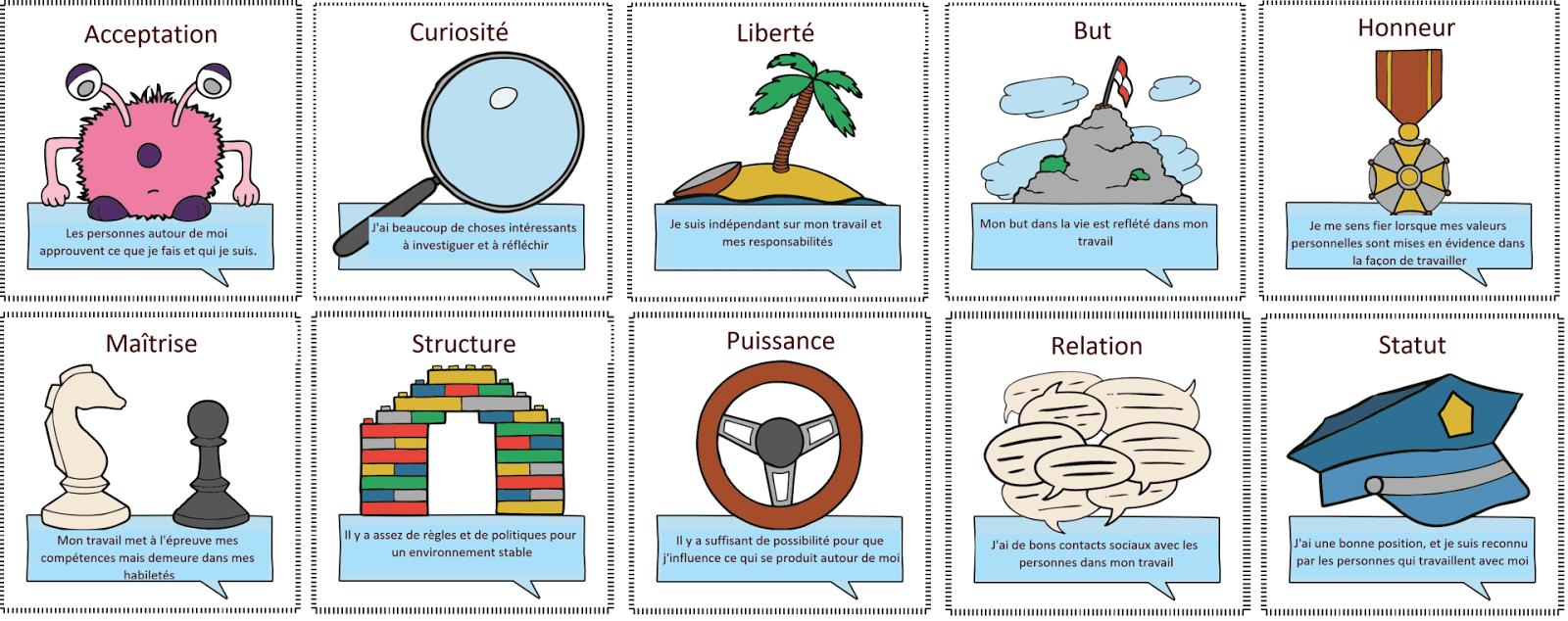
— **déculpabiliser l’équipe** qui fait son maximum avec un périmètre variable

— **de faire comprendre** aux personnes extérieures au projet l’impact d’un ajout de tâches

###### Le Moving Motivator

• **Moving Motivator :** exercice de management 3.0 créé par Jurgen Appelo qui permet de connaitre les motivations réelles des personnes

•L’outil a pris pour référence les 10 désirs intrinsèques :



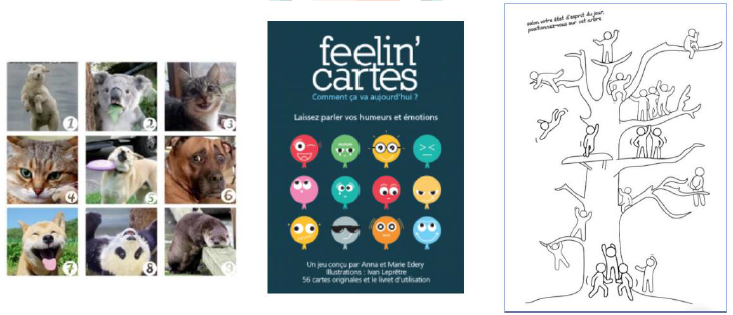
•Déroulé :

**1.** Demander au participant de placer de gauche à droite les cartes de la plus importante à la moins importante qu’il accorde à chaque carte de moving motivators

**2.** Vous allez proposer au participant de pousser les cartes si le sujet abordé à un impact positif sur l’une des motivations et de tirer vers lui les cartes s’il a un impact négatif sur lui. Si la transformation a selon lui aucun impact, il ne bouge par la carte.

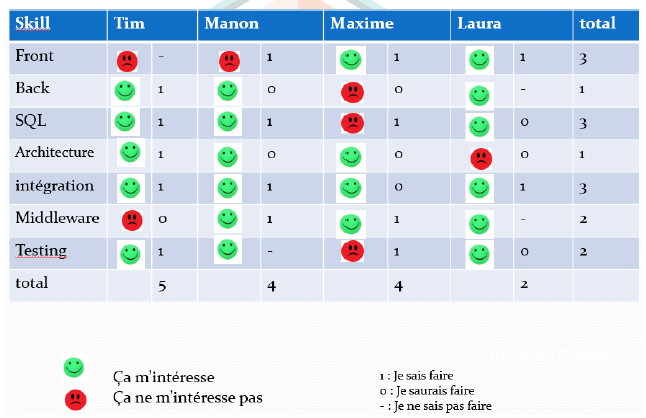
###### La météo de l’humeur

• **Météo de l’humeur :** outil utilisé pour aider les membres de l'équipe à communiquer efficacement et à travailler ensemble pour maintenir un environnement de travail positif et productif. L'équipe peut utiliser cette métaphore lors des réunions de revue de sprint, des rétrospectives, ou tout au long du sprint pour surveiller l'état d'esprit de l'équipe et trouver des moyens de résoudre les problèmes ou de maintenir un état d'esprit positif.



###### Matrice de compétences

• Matrice de compétences : outil utilisé pour les gestionnaires de projet et les responsables des ressources humaines pour évaluer les compétences de leur équipe, identifier les lacunes en matière de compétences et de formation, et élaborer des plans de développement pour améliorer les compétences de l'équipe



#### Product Owner

• **Product Owner :** responsable de définir le produit répondant aux attentes utilisateurs

##### Rôles

• Il fait le lien entre la partie métier et la partie technique du projet.

• Il doit porter la vision du produit.

• Il est l’interface entre l’utilisateur, le Scrum Master et les équipes chargées du développement.

• Il manage le backlog, pas une équipe.

• Il est responsable de l’établissement des Users Stories et de leur priorisation.

##### Responsabilités

• Collecter les attentes du marché

• Construire et partager la vision produit

• Être le relai entre le client et l’équipe de développement

• Planifier le travail en petites itérations

• Suivre l’avancement de la réalisation du produit

• Concilier les objectifs de l’entreprise et les besoins utilisateurs

• Récolter les feedbacks clients

• Maintenir un backlog priorisé et à jour

##### Rôles principaux

• **Responsable du produit :** il définit le produit, sa roadmap et veille à ce que celui-ci réponde aux attentes des utilisateurs mais également aux besoins du client

• **Validateur :** il s’assure que le développement du produit est en phase avec le besoin

• **Traducteur de besoin :** il va apporter une expertise métier pour s’assurer que le développement du produit est en phase avec le besoin

• **Maximiser la valeur du produit :** il priorise les fonctionnalités avant chaque planification, qui accepte ou rejette ces dernières pour une livraison

##### Compétences

• Être rigoureux

• Avoir une capacité d’immersion

• Être à l’écoute et ouvert à la discussion

• Être synthétique

• Connaitre le marché, les utilisateurs, le métier, les couts, les délais, etc.

##### C’est un garant au sein de l’équipe

• Il est toujours représenté par une personne au sein de l’équipe. Il ne peut exister qu’un seul Product Owner.

• Ce rôle peut, **même si le Scrum Guide ne le conseille pas**, être incarné par un membre de l’équipe de développement ou par le Scrum Master lui-même.

• L’objectif est d’avoir un seul et unique point d’entrée « humain » entre le client et l’ensemble de l’équipe.

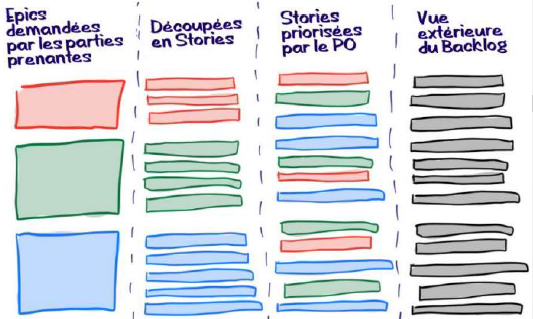
• Contrairement à un chef de projet tout puissant, il échange autant avec le client que l’équipe de développement.

##### Les outils du Product Owner

###### Le Product Backlog

• C’est l’artéfact primordial dans l’application de Scrum.

• Liste composée de tâches à réaliser afin de livrer un ensemble de fonctionnalités : les items



• Caractéristiques :

— Unique

— Considéré comme la référence du projet

— Partagé

— Parler une langue commune

— Chaque item est atomique

— Vit avec le projet

• Pour résumer, le backlog est une liste de ce qu’il y a à faire. Cette liste est évaluée et priorisée pour être partagée au sein de l’équipe Scrum. Dans l’état, elle est utilisable et l’équipe fait vivre ce backlog en ajoutant ou déplaçant des tâches.

###### L’item du product backlog (PBI) ou User Story

• C’est l’élément unitaire de backlog. Il est défini par :

— une description

— des critères d’acceptation

— une estimation

— un ordonnancement

• **Peut être une User Story si rédigé de la façon suivante :** En tant que <rôle>, je peux <action> afin de <résultat>

• **Formalisation d’une User Story :**

— **Narrative :** En tant que <utilisateur>, je peux <verbe d’action> afin de <bénéfice attendu>

— **Notes :** le contexte, les règles de gestion, les maquettes graphiques, les cas nominaux non passant et aux limites, la documentation à disposition, les contraintes techniques, la sécurité attendue, etc.

**Critères d’acceptation « Gherkin » :**

* + Lorsque <contexte>, quand je <verbe d’action> alors je <résultat attendu>.
  + Lorsque <contexte>, quand je <verbe d’action> alors je <résultat attendu>.
  + Lorsque <contexte>, quand je <verbe d’action> alors je <résultat attendu>.

• **Comment formaliser une bonne User Story ?** Le critère INVEST :

— I : indépendante des autres

— N : négociable initialement, plutôt qu’un engagement ferme

— V : verticale, ou ayant de la valeur en soit

— E : évaluée en termes de complexité relative

— S : suffisamment petite (en anglais Small)

— T : testable en principe, ce qu’on vérifie en écrivant un test

• La grille des critères INVEST permet de juger de la qualité d’une User Story : elle conduira éventuellement à reformuler son énoncé, voire à modifier en profondeur la Story.

###### Story Mapping

• Ce workshop permet de définir le parcours utilisateur et d’identifier toutes les fonctionnalités (« Stories ») d’un produit digital

• Son objectif :

— Faire communiquer toute l’équipe (technique et métier)

— Discuter des points risqués

— Prioriser (MVP, V1, V2, etc.)

— Partager une vision globale

• Généralement, il réunit l’ensemble de l’équipe Scrum et le(s) métier(s). Les parties prenantes voire les utilisateurs peuvent être conviés.

• **Les étapes d’un bon Story Mapping :**

— Identifier les utilisateurs :

> Personnes : client, gestionnaires… Ex : CRM : commerciaux, recruteurs, consultants, RH, paie, formation, direction (KPI), communication, etc.

> Automates (= autre systèmes)

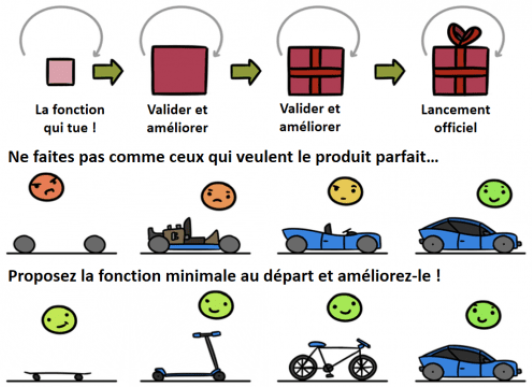
— Identifier les besoins ou epics

— Identifier les User Stories pour chaque Epic

— Prioriser les User Stories pour identifier le MVP et des Releases

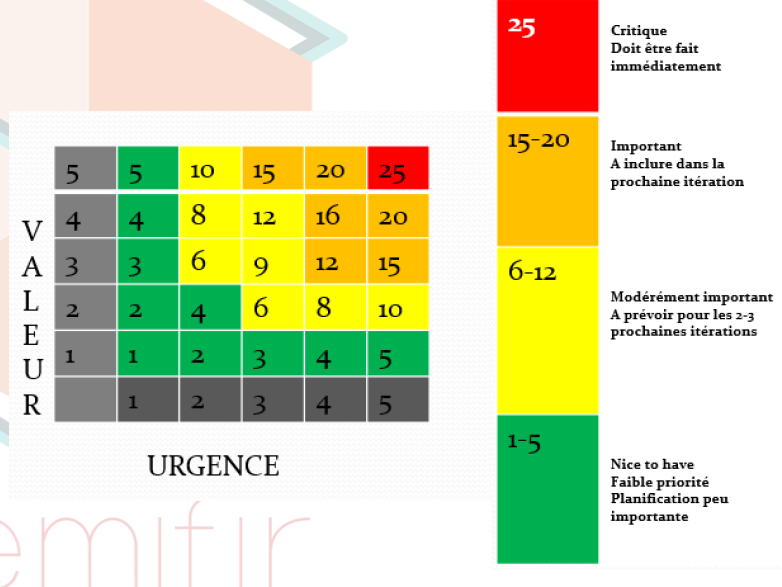
###### MVP

• **MVP (minimum viable product ou Produit minimum viable) :** méthode qui a pour objectif de sortir d’abord un produit avec uniquement la fonction la plus attendue (la Killer Feature ou la fonction qui tue) par un public cible, et de proposer le plus rapidement un produit afin de le confronter au marché.



###### Gérer les priorités

• Les priorités doivent être calculées selon cette formule : Priorité = Urgence x Valeur Métier



• Exemple de priorisation selon deux axes :

|  |  |  |
| --- | --- | --- |
|  | **URGENCE** | **VALEUR** |
| 5 | • Très forte contrainte de temps  • Très fortes dépendances avec les autres éléments  • Si pas fait immédiatement, il n’y a n faible intérêt de le faire | • Extrêmement important pour la plupart ou tous les clients  • Très fort impact sur la marque ou la réputation  • Critique pour le succès du business |
| 4 | • Forte contrainte de temps  • Fortes dépendances avec les autres éléments  • À faire dans la prochaine itération à cause des clients ou des exigences contractuelles | • Important pour beaucoup de clients  • Impact significatif sur la marque ou la réputation  • Avantage compétitif significatif |
| 3 | • Contrainte de temps modérée  • Dépendances modérée avec les autres éléments  • Préférable d’être terminé dans les prochaines itérations | • Important pour plusieurs clients  • Impact modéré sur la marque ou la réputation  • Avantage compétitif modérément important |
| 2 | • Faible contrainte de temps  • Faible dépendance avec les autres éléments  • Acceptable d’être terminé dans les 2-3 prochaines itérations | • Important pour seulement quelques clients  • Faible impact sur la marque ou la réputation  • Avantage compétitif faible |
| 1 | • Pas de contraintes de temps  • Pas de dépendances avec les autres éléments  • Peu ou pas d’impact | • Important pour peu ou aucun client  • Peu ou pas d’impact sur la marque ou la réputation  • Peu ou pas d’avantage compétitif |

• US, Epic, Task... mais quels sont les repères ?

On peut répartir les éléments du backlog product selon des fonctions ou thèmes, eux-mêmes faisant partie d’Epics :

|  |  |  |
| --- | --- | --- |
| **Grandes fonctionnalités** | **Mois** | **Produit** |
| Epics | Semaines | Version/release |
| Stories | Jour | Itération |
| Tâches | Heures |  |

• Les erreurs à éviter :

— Ne pas avoir de backlog de produit

— Avoir plusieurs backlogs pour un seul produit (ou d’autres sources parasites, comme un gestionnaire d’anomalies)

— Ne pas partager le backlog avec toute l’équipe

— Ne jamais actualiser le backlog

— Confondre avec le Sprint Backlog en y mettant des tâches

— Avoir plus de 150 éléments actifs à faire dans le backlog

— Ne pas prioriser les éléments ou donner des priorités égales

#### Dev Team

• **Équipe de développement et de tests :** responsable de la réalisation du meilleur produit possible

##### Principe d’autogestion

• Manifeste Agile, principe n°11 : « Les meilleures architectures, spécifications et conceptions émergent d’équipes autoorganisées. »

• Elle est la seule à décider la manière dont elle va réaliser les fonctionnalités du sprint backlog.

##### Mission

• Transformer les besoins transmis par le Product Owner en nouvelles fonctionnalités dans l’application.

• L’objectif sera toujours de maximiser la valeur.

##### Rôles principaux

• La Dev Team est :

— responsable de l’incrément

— chargé de l’estimation

— réalisateurs

##### Qui sont-ils ?

• Membre d’une équipe pluridisciplinaire (cross functional)

• 3 à 10 développeurs

##### Definition of Done ou critère d’acceptation

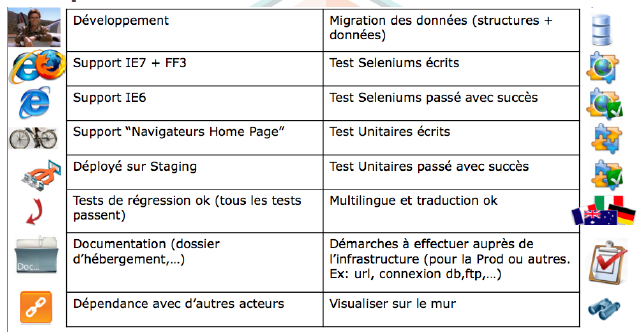
• C’est une checklist constituée de tout ce qu’il faut faire pour considérer les fonctionnalités / User Story comme terminées.

• Elle doit être liée au contexte de la fonctionnalité et peut contenir des éléments : de tests, techniques, fonctionnels, liés à la sécurité, etc.

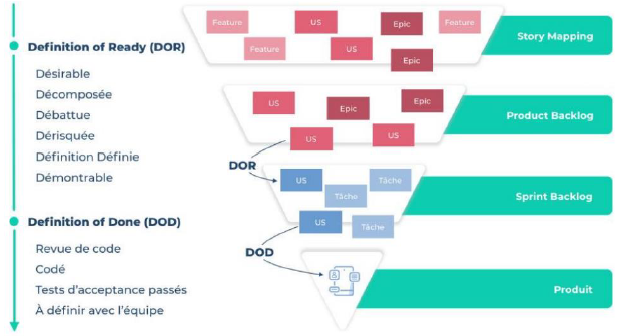
• La DoD peut évoluer d’un Sprint à l’autre, cependant **une DoD mal exprimée est une des causes d’échec majeur dans la gestion de projet**.

• La DoD responsabilise et implique l’équipe de développement.

• Exemple de Definition of Done :



• DoD vs DoR (Definition of Ready)



##### La dette technique

• **Dette technique :** accumulation de « crasses » laissée dans le code à cause des délais à tenir.

• On peut y retrouver :

— Duplication de code

— Non-respect de la nomenclature

— Algorithme trop complexe

— Classes qui font des kilomètres de long

— Code mort

— Code en commentaire « qui pourrait servir un jour »

— Code non commenté et non documenté

— Etc.

##### Le planning poker ou Scrum Poker

• Objectif : faciliter l’estimation collective de l’effort à investir dans les **différents scénarios utilisateurs (users stories)**.

• **On l’utilise non pas pour planifier mais pour estimer**.

• Cette technique favorise les échanges entre les membres de l’équipe projet, par exemple entre le responsable produit et les développeurs. L’avantage principal du planning poker est de permettre à tous de s’exprimer librement.

• Les membres de l’équipe projet s’installent autour d’une table, placés de façon que tout le monde puisse se voir. Au minimum, il faut que tous les membres de l’équipe de développement soient présents.

• Déroulement du jeu :

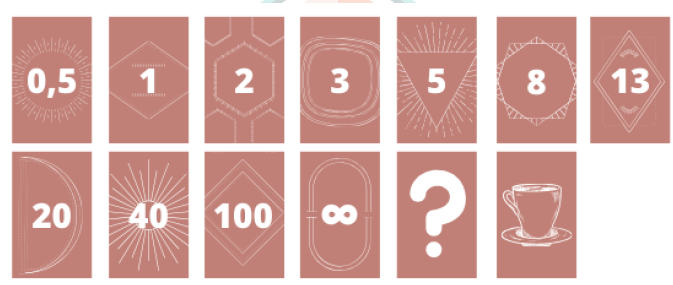
— Explication de l’US : Le PO explique aux autres participants l’US

— Questions/réponses sur l’US : les participants posent des questions PO, discutent du périmètre du scénario, évoquent les conditions de satisfaction qui permettront de le considérer comme « terminé »

— Estimation individuelle : Chacun des participants évalue la complexité de ce scénario, choisit la carte qui correspond à son estimation et la dépose, face vers le bas, sur la table devant lui

— Estimation collective : au signal du facilitateur, les cartes sont retournées en même temps. S’il n’y a pas unanimité, la discussion reprend. On répète le processus d’estimation jusqu’à l’obtention de l’unanimité

— Valeur d’un « story point » : à la fin, on établit la valeur en temps d’un « story point ».



*Avec la suite de Fibonacci.*

##### Spike

• **Spike :** type spécial de User Story qui est utilisé pour acquérir les connaissances nécessaires pour réduire le risque d’une approche technique, mieux comprendre une exigence ou augmenter la fiabilité d’une estimation de story.

• Elle est **issue de l’Extreme Programming et non de Scrum.**

• On l’utilise quand :

— on ne sait pas où on va techniquement

— l’US est floue et ne peut pas être estimée

• Avantages du Spike :

— Le spike est timeboxé (maximum 2 jours).

— Il permet de faire de la recherche ou du reciblage métier.

— Il ne fait pas partie des points d’efforts.

— Il se termine par la production d’un éléments concret (prototype, documentation, présentation, etc.).

• Le résultat d’un spike est partagé lors de la rétrospective du sprint.

##### Vélocité d’une équipe

• C’est **le nombre de points d’efforts traités en une itération**.

• Elle traduit la capacité de l’équipe :

— en nombre de points par itération

— permet de fixer un périmètre réaliste à l’itération à venir en se fondant sur l’extrapolation des itérations passées (aspect empirique)

— repère quantitatif

• C’est un outil de planification pour le reste du projet. Le développeur n’a pas à s’en préoccuper.

• **Vélocité = moyenne des points d’efforts embarqués sur les 8 derniers sprints**.

• Points de vigilance :

— La vélocité n’est pas un outil de management pour optimiser la capacité de l’équipe.

— L’augmentation de la vélocité n’est pas toujours une bonne nouvelle.

— Ne pas comparer les vélocités des équipes.

##### Le Sprint Backlog

• Composé de fonctionnalités issues du Product Backlog en fonction de la priorisation exprimée par le PO.

• Le nombre de ces Items est décidé par l’équipe (après estimation).

• Ces fonctionnalités/items/User Stories sont découpées en tâches à réaliser par les développeurs.

• Il contient au moins un item d’amélioration continue (supprimé en 2020).

• Une mise à jour de l’état de chaque tâche permet d’être transparent sur l’avancement.

• Le Sprint Backlog évolue en permanence durant le Sprint.

• Le Sprint Backlog est la responsabilité de l’équipe de développement.

##### Le Sprint Goal

• Décidé par l’ensemble de l’équipe scrum

• Le Scrum Master peut guider le PO pour faire ressortir un objectif :

— « Pourquoi construire ces fonctionnalités dans le sprint ? »

— « Quelle est LA fonctionnalité urgente et importante ? »

— « Quel est le sens de notre sprint ? »

• La Scrum team est responsable d’atteindre le sprint goal.

• Le sprint goal doit être connu et affiché.

• N’évolue pas pendant le sprint

• L’objectif d’un Sprint Goal est de définir la raison de l’exécution d’un sprint, et ils fournissent un objectif commun.

• Les objectifs de sprint énoncent souvent les histoires à mettre en place plutôt que la raison de l’exécution de l’itération.

• Voici un template pour réaliser le Sprint Goal :



#### Les parties prenantes ou stakeholders (clients)

• **Rôles :** s’investir dans la vue produit sans participer à la réalisation

• **Droits :**

— Assister aux démonstrations

— Avoir une vue sur l’avancement

• **Devoirs :**

— Fournir les moyens de réalisation

— Faire confiance à l’équipe

— Donner du feedback

### Les évènements dans Scrum

#### Le sprint

• **Sprint :** évènement timeboxé comportant des tasks issues du sprint backlog

• Il a une durée calendaire d’un mois maximum et permet d’avoir un découpage du projet temporisé.

#### Le sprint planning

• L’objectif du sprint planning est de :

— le sprint goal (la vision macroscopique)

— le sprint backlog (la vision microscopique)

• Toute l’équipe Scrum (PO, Scrum Master et développeurs) est présente à cette réunion.

• Le temps de cet évènement Scrum est de 8 heures pour un sprint de 4 semaines.

#### Le sprint review : l’inspection

• **Sprint review :** cérémonie Scrum qui permet d’effectuer la phase d’inspection de l’itération

• Le temps de cet évènement est de 4h pour un Sprint de 4 semaines.

• Objectifs :

— Montrer en quoi le sprint est bel et bien terminé et répond aux exigences posées lors de la planification

— Faire le constat de ce qui s’est bien ou mal passé durant le sprint et réfléchir aux potentiels solutions

— Faire le point sur le backlog product, du planning de la réalisation et du budget

— Prendre en compte les contraintes externes telles que l’évolution du marché, les objectifs des parties prenantes, etc.

— Anticiper les priorités du prochain Sprint

• Les participants sont : le Scrum Master, le PO, la Dev Team et les Stakeholders.

• Objectif de la review : présenter l’incrément et collecter les feedbacks

— Présenter l’incrément produit

— Échanger autour des fonctionnalités

— Récolter les feedbacks

#### Le sprint retrospective

• **Sprint retrospective :** cérémonie Scrum en fermeture du Sprint qui se réalise juste après la Sprint Review.

• Elle propose à l’équipe Scrum au complet d’avoir l’opportunité de mettre en place des axes d’amélioration pour les prochains sprints. L’amélioration continue est la clé de la réussite des projets agiles.

• Le temps de cet évènement est de 3h pour un sprint de 4 semaines.

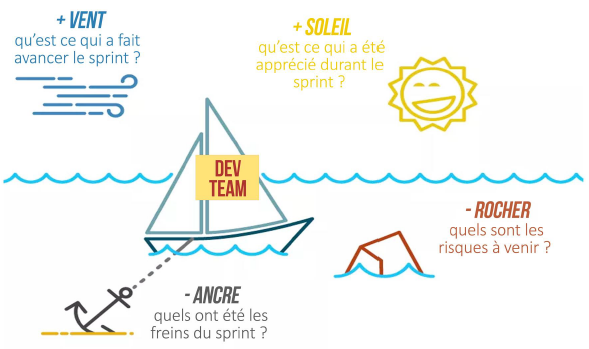
• Le Scrum Master organise et y participe.

• Là où la review se concentre sur le produit, la rétro se focalise sur l’équipe sur plusieurs niveaux possibles :

— Relationnel et humain

— Processus de réalisation du produit





• Objectifs :

— tirer les enseignements de l’itération

— Analyser le plan d’actions précédent

— Prioriser ensemble chaque point

— Déterminer ensemble un plan d’actions réaliste pour la prochaine itération

— DOD : definition of done

— Composition de l’équipe

— Point sur la Sprint Review

— Durée du Sprint

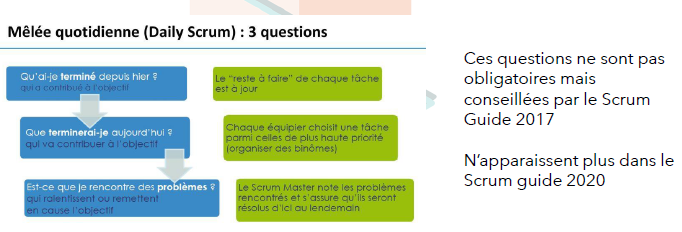
#### Daily Meeting : la mêlée

• Elle doit se dérouler à la même heure et dans le même lieu pour faciliter l’organisation.

• **Elle ne doit pas durer plus de 15 minutes MAXIMUM.**

• **3 questions :**

* Qu’ai-je terminé depuis hier ?
* Que terminerai-je aujourd’hui ?
* Est-ce que je rencontre des problèmes ?



• **Les acteurs de la mêlée :**

— Seul l’équipe de développement peut y participer.

— Tout le monde peut y assister.

— Le Scrum Master veille à sa tenue et peut s’y greffer uniquement sur demande de l’équipe de développement et pour aider à l’organiser.

• **Objectif :**

— Planifier et rythmer les développements

— Libérer la communication et renforcer l’esprit d’équipe

— Garder le cap et rectifier en cas de dérive

— Anticiper les problématiques

#### Le backlog refinement (depreciated)

**•** Dans la nouvelle version de Scrum 2020, cette technique a disparu.

• **Objectifs :**

— affiner et ordonner le Backlog de produit

— s’assurer que les items du backlog seront discutés et prêts au démarrage du sprint suivant

— anticiper les impacts et activités à moyen terme

• **Qui participe ?**

— Product Owner

— Équipe de développement

— Scrum Master (pas obligatoire)

• **Quand ?**

— Pour le PO : autant que nécessaire

— Pour les développeurs : pas plus de 10% du temps de développement