

Gestion de projet

Méthodes agiles

Emmanuel CHAUVET

Planning

- lun. 02/10/23
- ven. 20/10/23
- mar. 31/10/23
- ven. 24/11/23
- lun. 27/11/23

01

Les fondamentaux de l'agilité

Objectifs des méthodes agiles



Réduire le cycle de vie du logiciel
et accélérer son développement



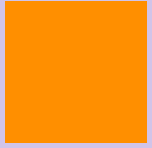
Développer une version minimale
via l'intégration de fonctionnalités



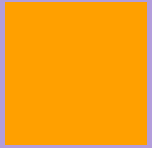
Processus itératif et adaptatif basé
sur une écoute client



Tests tout au long du cycle de
développement



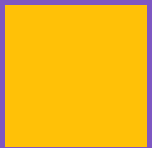
Instabilité de l'environnement
technologique



Client incapable de définir les besoins de
manière exhaustive en début de projet



Adaptation aux changements durant le
projet : évolution des spécifications en
cours de projet



2001 : Manifeste agile constitué par 17
personnes, avec une volonté de rupture
avec la gestion de projet classique

Origines des méthodes agiles

Le Manifeste Agile

<http://agilemanifesto.org/iso/fr/manifesto.html>

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

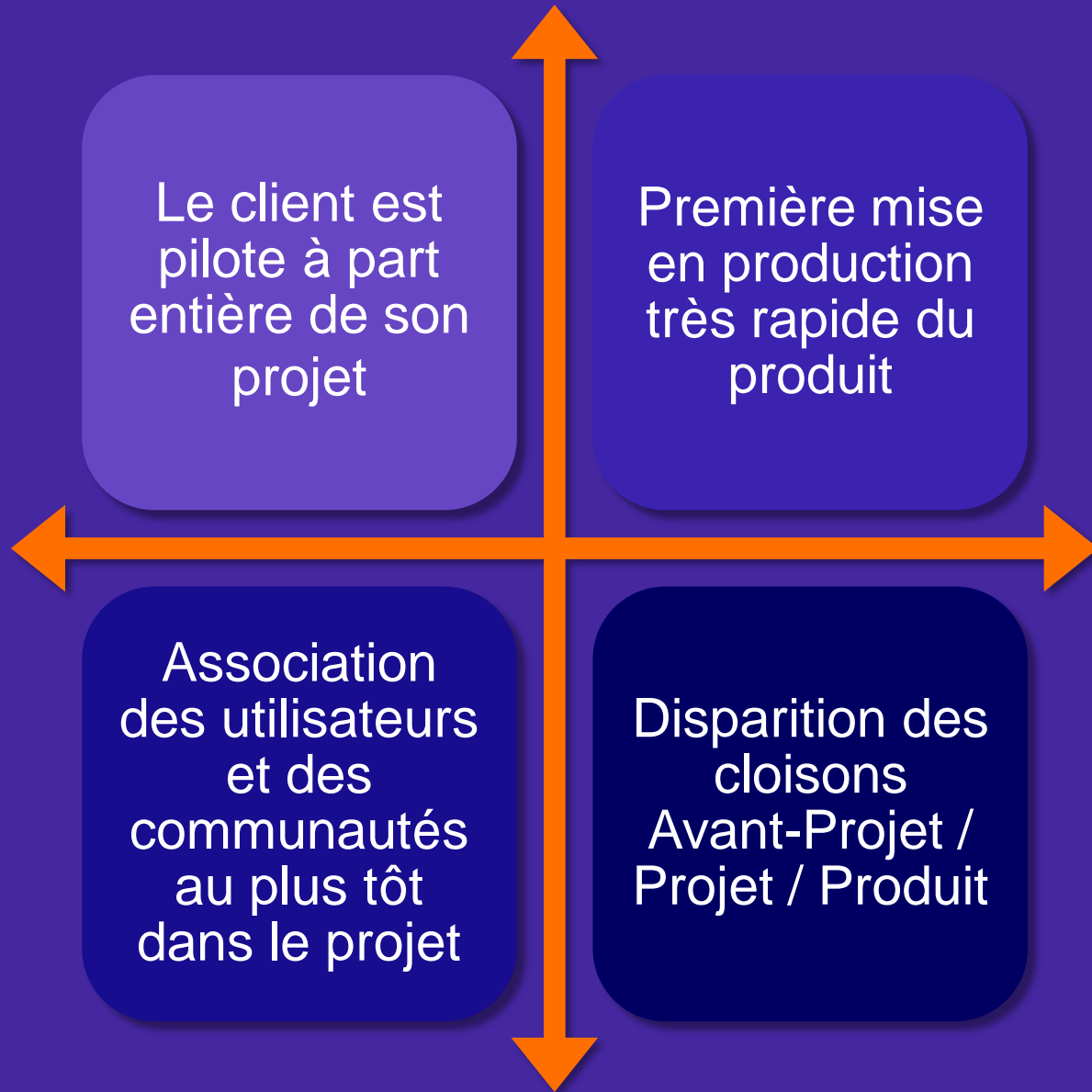


La collaboration avec les clients plus que la négociation contractuelle



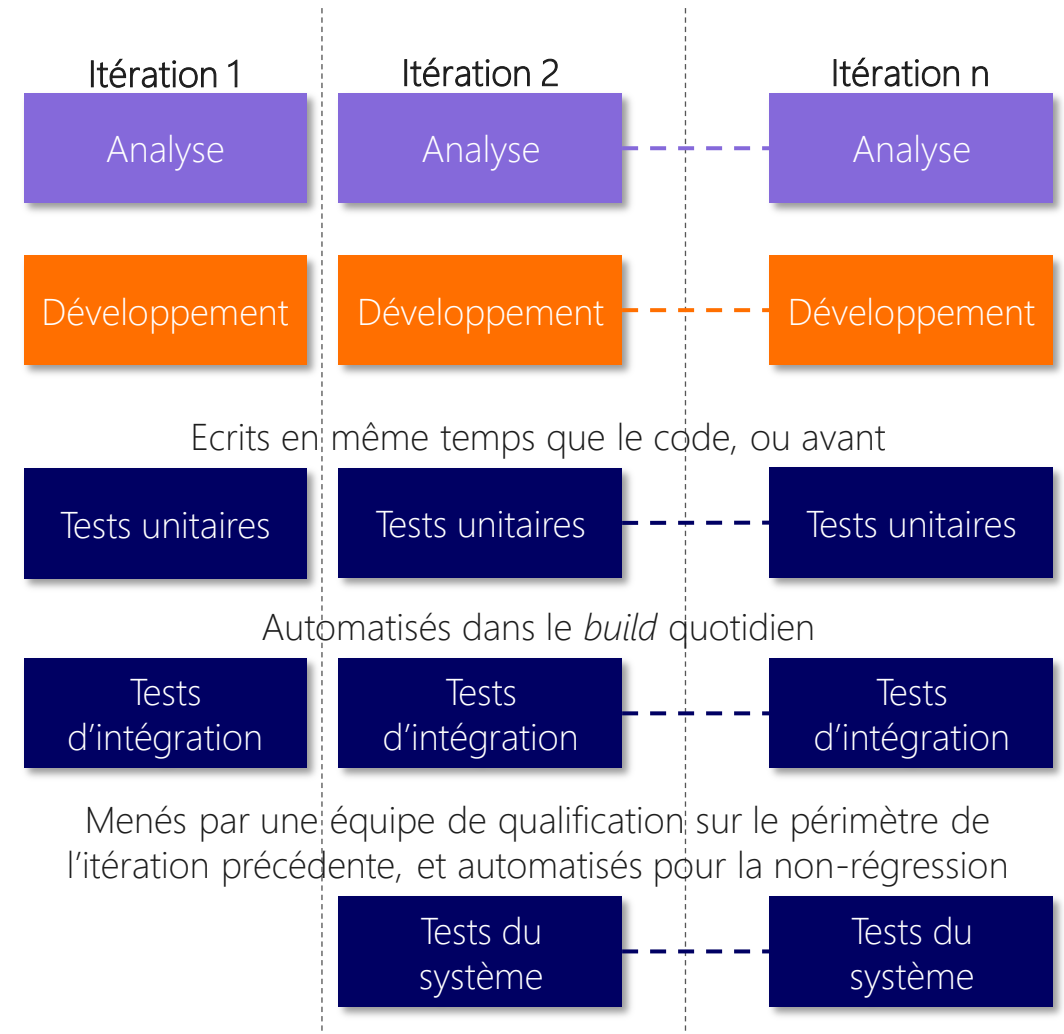
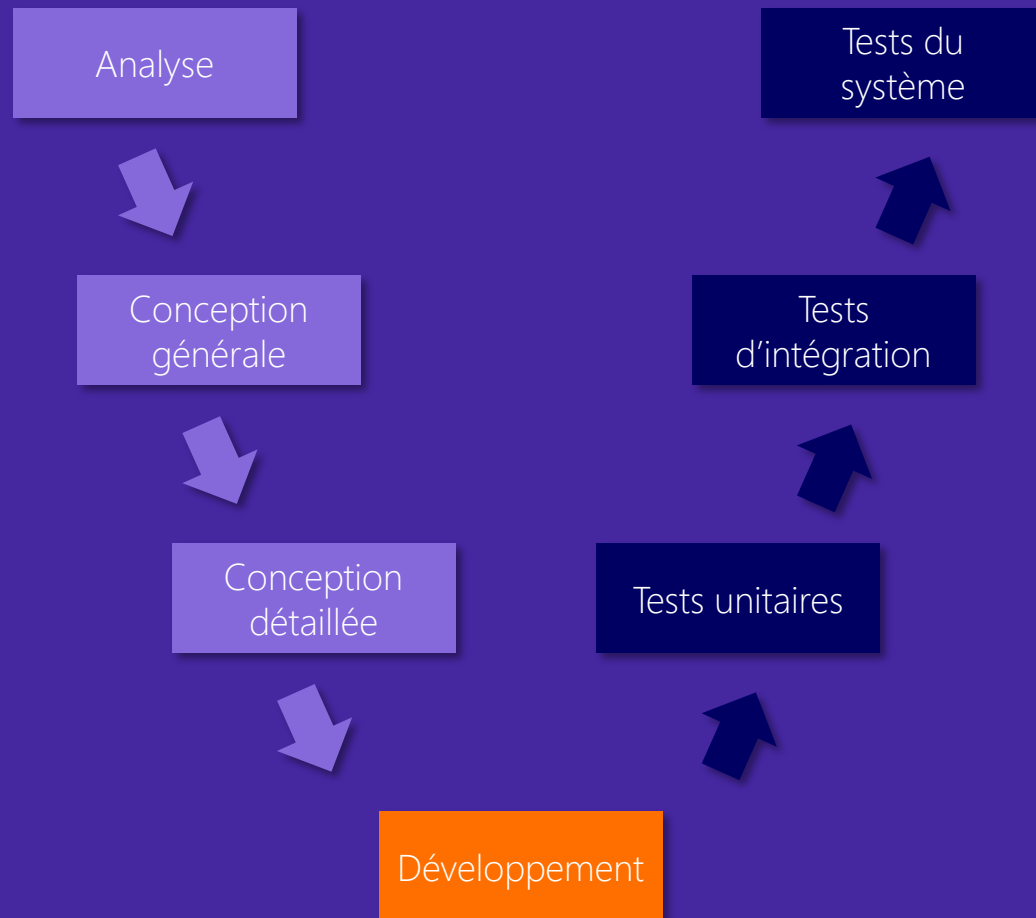
L'adaptation au changement plus que le suivi d'un plan

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



Bénéfices pour le client

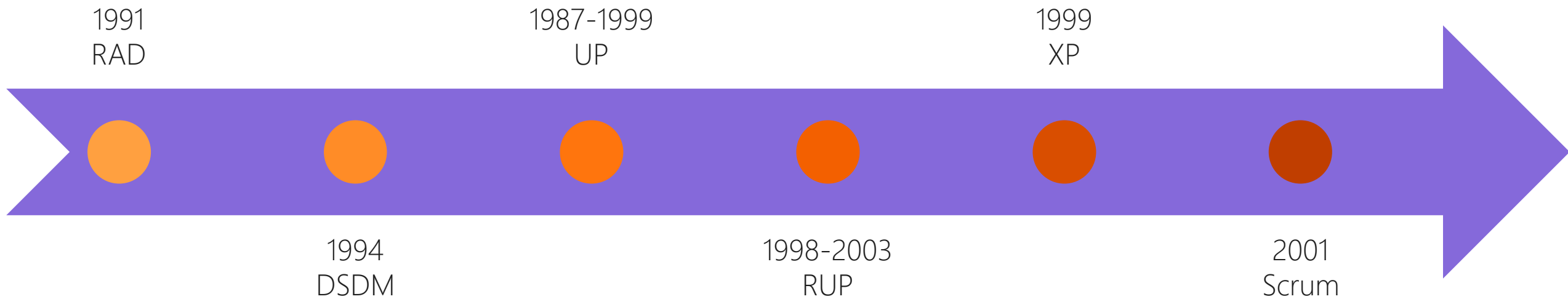
Cycle en V vs Agilité



2

Inventaire (partiel) des méthodes agiles

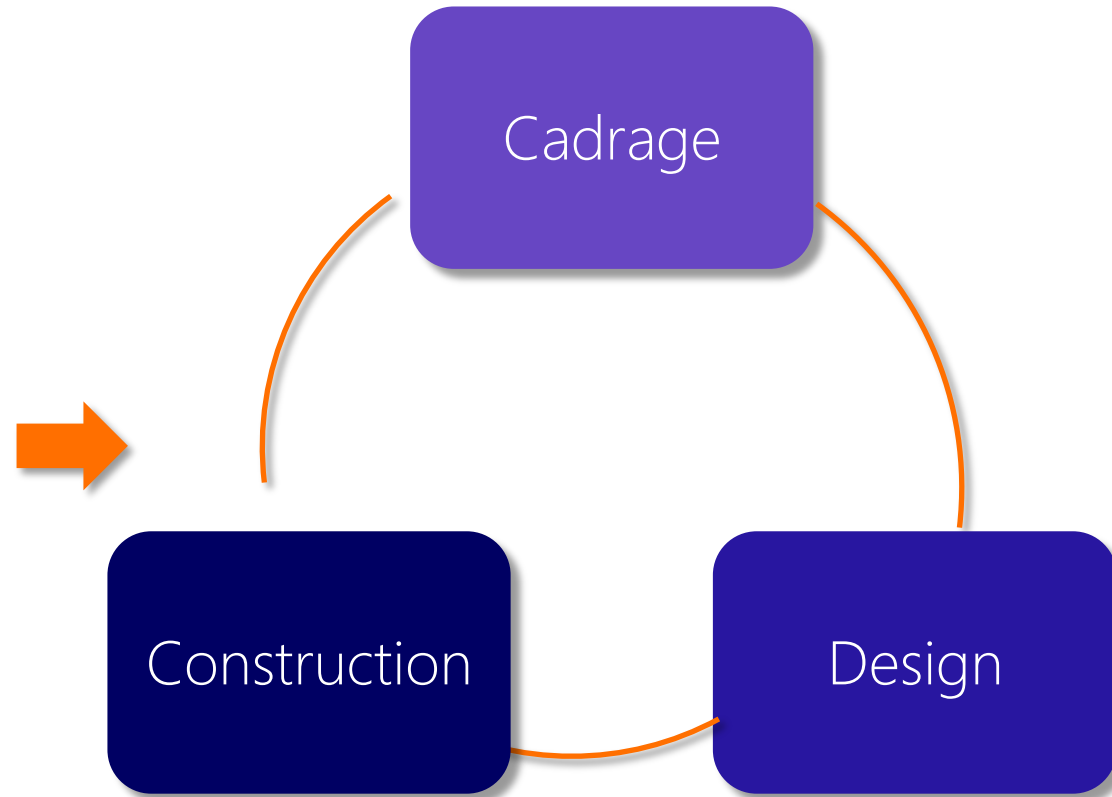
Un peu d'histoire



RAD – *Rapid Application Development*

James Martin (1991)

Cycle de développement
court (90 à 120 jours
max)



DSDM – *Dynamic Software Development Method*

Basé sur RAD (UK, 1994)

Implication

des utilisateurs durant tout le cycle de développement

Autonomie

L'équipe projet a un pouvoir de décision sur l'évolution des besoins

Visibilité du résultat

Application livrée le plus souvent possible pour avoir un feed-back rapide

Adéquation

Livrer une application en adéquation avec le besoin du client

Développement itératif et incrémental

Evolution du développement basée sur le feed-back des utilisateurs

Réversibilité

Toute modification effectuée durant le développement est réversible

Synthèse

Schéma directeur défini au préalable pour fixer le périmètre du projet

Tests

continus durant le développement pour garantir le fonctionnement de l'application

Coopération

Acteurs du projet faisant preuve de souplesse sur les modifications demandées

UP – *Unified Process*

Ivar Jacobson (initié dès les années 60, première formalisation en 1987)



Développement **itératif** et **incrémental**



Segmentation en **phases courtes**



Livraison systématique en **fin de phase** d'une nouvelle version **incrémentée**



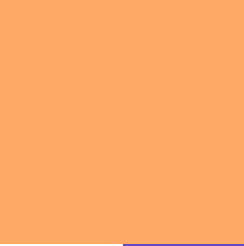
Modélisation **UML** pour l'**architecture** (fonctionnelle, logicielle et physique)



Cas d'utilisation (*use case*) pour les **besoins** et **exigences** des utilisateurs

RUP – *Rational Unified Process*

Implémentation d'UP proposée par Rational Software (racheté depuis par IBM)



Gestion et
composition
en rupture
avec
l'approche
traditionnelle
des équipes



Approche
temporelle
par deadline



Modèles de
documents

XP – *eXtreme Programming*

Kent Beck, Ward Cunningham, Ron Jeffries &alleja Xavier (1999) – Systématisation extrême de pratiques pré-existantes

Revue de code

bonne pratique

faite en permanence (par un binôme)

Tests

utiles

faits systématiquement avant chaque mise en œuvre

Conception

importante

faite tout au long du projet

Simplicité

avancer plus vite

solution la plus simple

Intégration des modifications

cruciale

plusieurs fois par jour

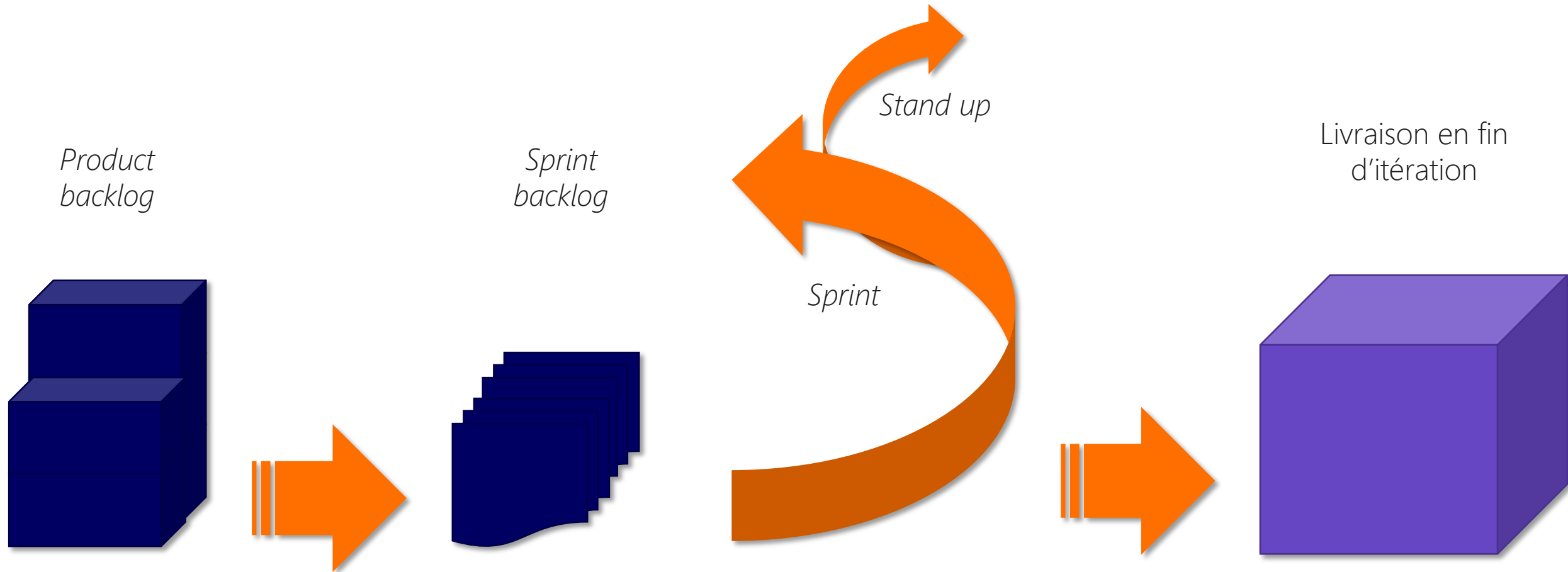
Besoins

évoluent vite

cycles de développement très rapides

Scrum (= mêlée de rugby)

Présentation détaillée dans la prochaine partie



Synthèse et recommandations

Ne pas choisir une méthode

mais s'inspirer des existantes

Garder en tête les principes agiles

- Le client est au centre du projet
- Moins de documentation et plus de livraisons
- Deadline et découpage en phases courtes
- Implication et autonomie des équipes
- Tests tout au long du projet

Une question à se poser

« Le client est-il agile ? »

Corollaire : s'il ne l'est pas, comment faire de l'agilité ?

03

Scrum

Une méthode agile adaptable à tous types de projets

Un peu d'histoire

1986

Hiroataka Takeuchi & Ikujiro Nonaka (*The New New Product Development Game*)

2001

Ken Schwbaer & Mike Beedle (*Agile Software Development With Scrum*)

2011

Jeff Sutherland et Ken Schwaber (*The Scrum Guide*)

1991

Ken Schwaber
(description des fondements de la future méthode)

2004

Ken Schwaber (*Agile Software Management with Scrum*)

Notions fondamentales & vocabulaire

Sprint

Découpage du projet en itérations de 2 à 4 semaines selon le projet

Livraison

à la fin de chaque *sprint*

User story

Description d'une fonctionnalité attendue

Product backlog

Liste de l'ensemble des fonctionnalités attendues

Sprint backlog

Liste des fonctionnalités à produire pendant un *sprint*, non modifiable pendant le *sprint*

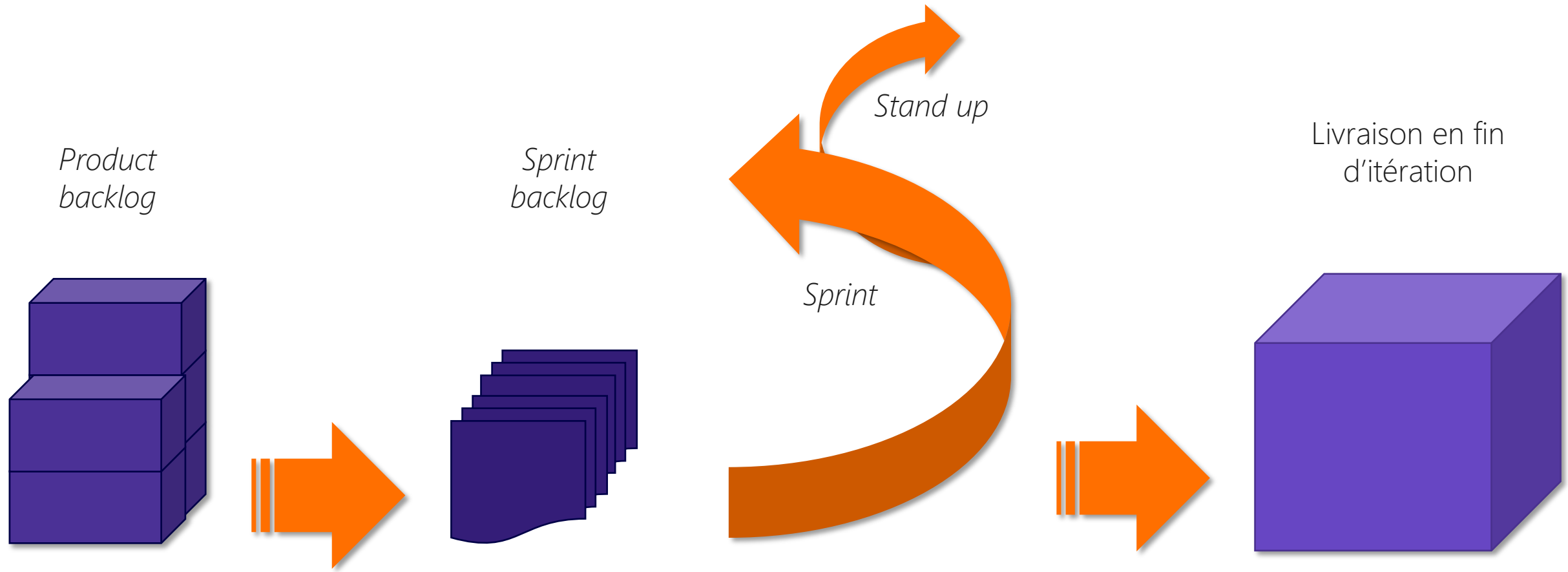
Planning poker

Estimation de charge des tâches avant chaque *sprint*

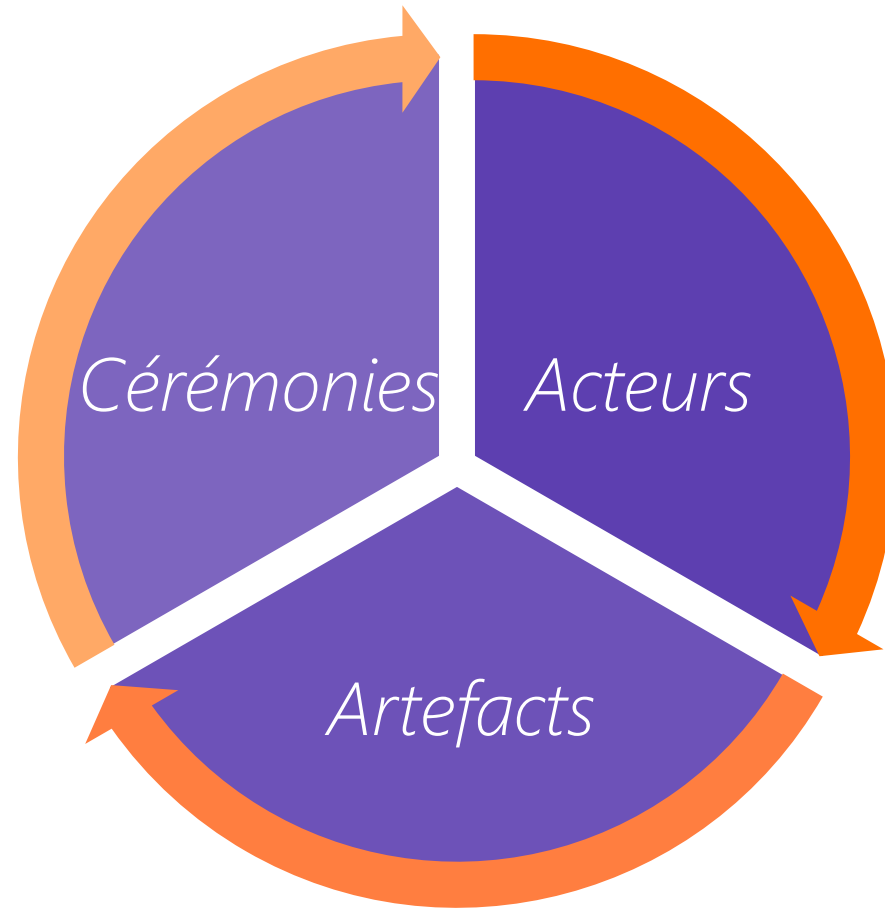
Stand up meeting ou Morning meeting

Réunion quotidienne de tous les acteurs du projet

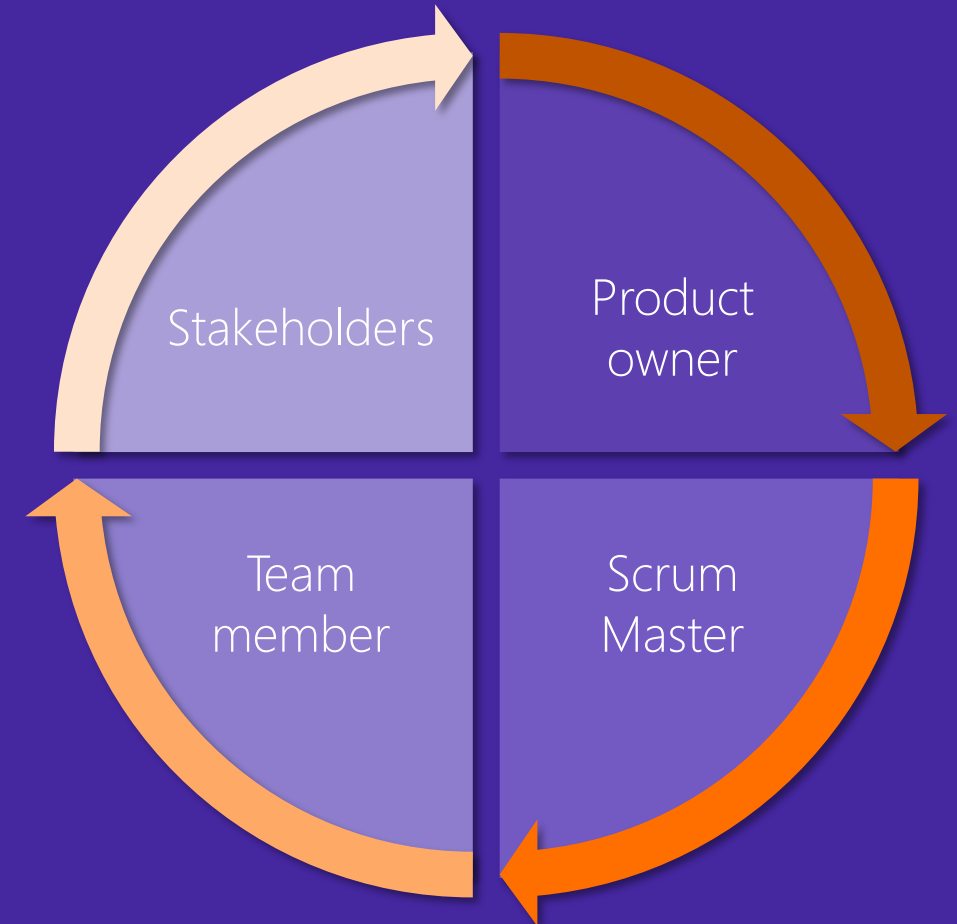
Déroulement d'un *sprint*



3 piliers

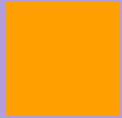


Les acteurs du projet





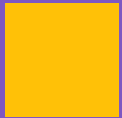
Représentant officiel du client, présent avec l'équipe



Interlocuteur principal du *scrum master* et des *team members*
Répond aux questions qui surviennent (*stories*, maquettes...)



Formalise les besoins du produit et rédige les spécifications



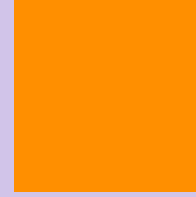
Se fait aider de responsables fonctionnels pour la rédaction des spécifications



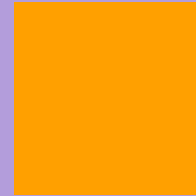
Définit et priorise les *users stories* pour chaque *sprint*

Product owner

Scrum master



Veille à la mise en application de la méthode et au respect de ses objectifs

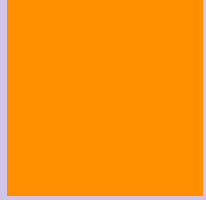


Lève les obstacles éventuels qui empêcherait l'avancement de l'équipe et du projet pendant les *sprints*



Ce n'est pas un chef de projet





Chargé de la
réalisation du *sprint* et
d'un produit utilisable
en fin de *sprint*

Développeur,
Architecte,
Testeur,
Infographiste,
Intégrateur...



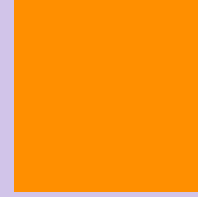
Impliqué très en amont dans le
projet



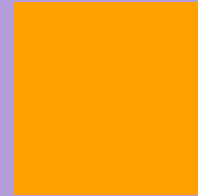
Participe à l'évaluation de charge,
à la conception, aux arbitrages
techniques...

Team member

Stakeholder



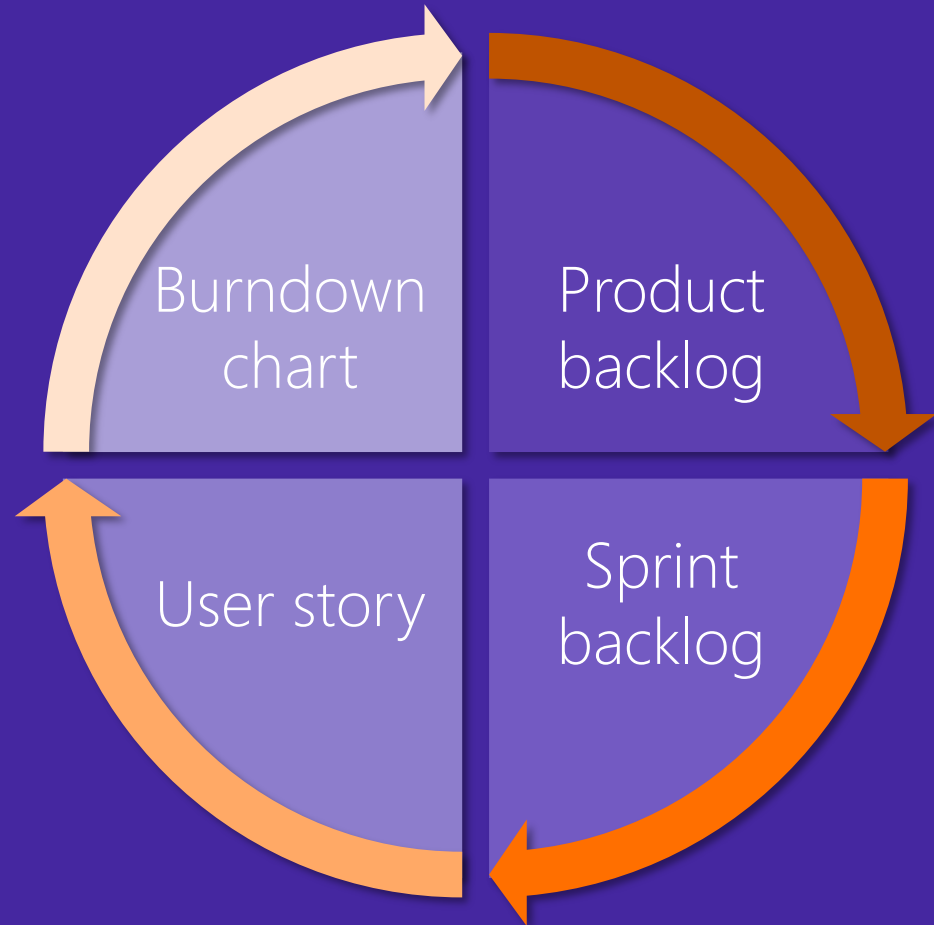
Ensemble des autres parties prenantes du projet



Principalement les utilisateurs finaux ou clients du projet



Expriment leurs attentes/besoins au *Product owner*



Les artefacts

Le *Product backlog*

« Simple » inventaire des *user stories*

Valeur métier : priorisation selon les critères métier (la valeur accordée à la *story*)

Story points : estimation de l'effort nécessaire à la production de la *story*


<i>User Stories</i>	Valeur métier	<i>Story points</i>
Story A	1	5
Story B	2	8
Story C	3	1
Story D	4	8
Story E	5	2
Story F	6	2
Story G	7	2

Le *Sprint backlog*

Sélection de *stories*

- à produire pendant un sprint
- par le *product owner*
- en accord avec le *scrum master* et les *team members*
- construit au fur et à mesure de l'avancement (1 à 2 *sprints* à l'avance)

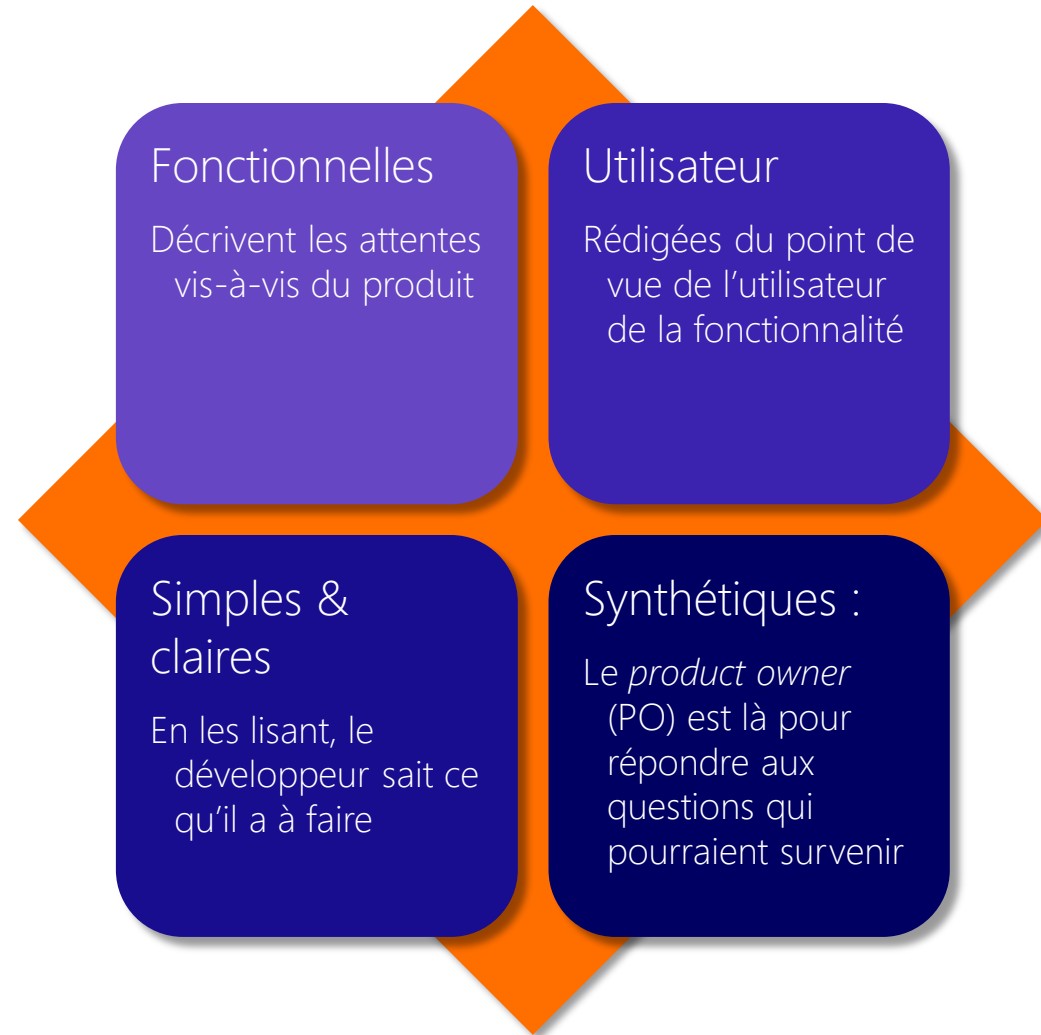
<i>User Stories</i>	Valeur métier	<i>Story points</i>
Story A	1	
Story B	2	
Story C	3	
Story D	4	8
Story E	5	2
Story F	6	2
Story G	7	2



<i>User Stories</i>	Priorité métier	<i>Story points</i>
Story A	1	5
Story C	3	1

Les *User stories*

Décrivent les fonctionnalités à produire



Modèle de *user story* (format A5)

Titre	Rattachement
Valeur métier	Propriétaire
Effort	Affectation
Type de carte	Statut
Objectif <i>En tant que...</i> <i>Je souhaite...</i> <i>Afin de...</i>	Critères d'acceptation BDD <i>Given that (Etant donné que)</i> <i>When (Quand)</i> <i>Then (Alors)</i>

Titre	Rédigé par le PO
Rattachement	Défini par le PO et l'équipe
Valeur métier	Définie par le PO
Propriétaire	Défini par le PO, sauf pour les US <i>technical</i> ou <i>defect</i>
Effort	Estimé par l'équipe
Affectation	Définie par l'équipe
Type de carte	Défini par le PO (US fonctionnelle) ou par l'équipe (US <i>technical</i> ou <i>defect</i>)
Statut	Défini par l'équipe
Objectif	Rédigé par le PO
Critères d'acceptation	Définis par le PO, enrichis par l'équipe
BDD (<i>Behaviour Driven Development</i>)	Définis par le PO, enrichis par l'équipe

Les différents champs d'une *user story*

Titre	identifie la <i>US</i> au sein du <i>backlog</i> , donc clair et concis	Objectif	description de l'action utilisateur, purement fonctionnelle (pas de technique, pas d'IHM) respecte la formulation rituelle « En tant que » (caractéristiques utilisateurs) « Je veux » (exigence utilisateur) « Afin de » (bénéfice utilisateur)
Rattachement	généralement à une <i>feature</i> (ensemble de fonctionnalités liées)	Critères d'acceptation	description des critères que le propriétaire de la <i>US</i> va contrôler pour valider ou rejeter la livraison (ex : règles métiers à appliquer, textes à afficher...) peut contenir des informations/contraintes techniques (ex : regexp email, captcha, dédoublonnage...)
Valeur métier	indicateur d'importance et de priorité (de 0 à 100 par ex.)	BDD (<i>Behaviour Driven Development</i>)	description des cas de test ; permettent au d'écrire les scénarios des test en même temps (voire avant) qu'il développe respecte la formulation rituelle « Etant donné que » (contexte) « Quand » (action utilisateur) « Alors » (comportement/résultat attendus)
Propriétaire	désigne le responsable de la validation de la <i>US</i> produite (généralement le <i>product owner</i>)		
Effort	estimation de l'effort nécessaire pour la réalisation de la <i>US</i> (cf. <i>planning poker</i> , <i>t-shirt sizing</i>)		
Affectation	qui va développer la <i>US</i>		
Type de carte	<i>user story</i> fonctionnelle (la base), <i>technical story</i> (ex : traitement auto), <i>defect story</i> (bug)		
Statut	état à l'instant <i>t</i> (ex : à faire, en cours, en test, validé, à livrer...)		

Des *user stories* « INVEST »

I

Indépendante

Aucune dépendance avec les autres *stories* afin d'éviter des problèmes de tests et de planification

N

Négociable

Discussion entre PO et équipe... tant que la *story* n'est pas dans un *sprint*

V

Valeur

Apporte une valeur à l'utilisateur, donc exprime l'objectif de l'utilisateur

E

Estimable

Suffisamment claire pour que l'équipe puisse estimer l'effort nécessaire

S

Small

Suffisamment petite pour être planifiée simplement et sur un seul *sprint*

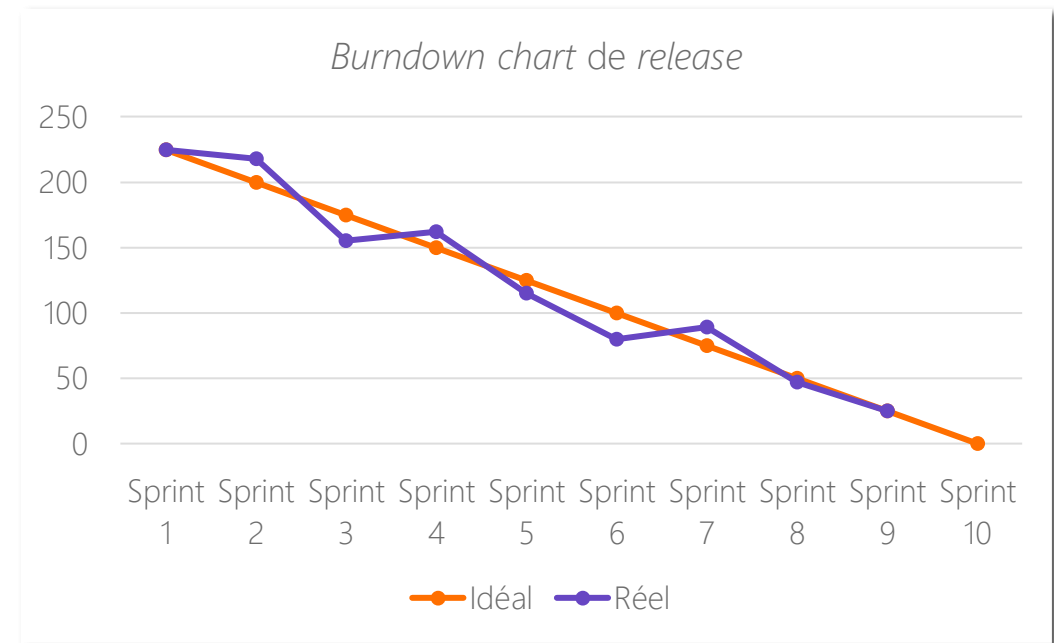
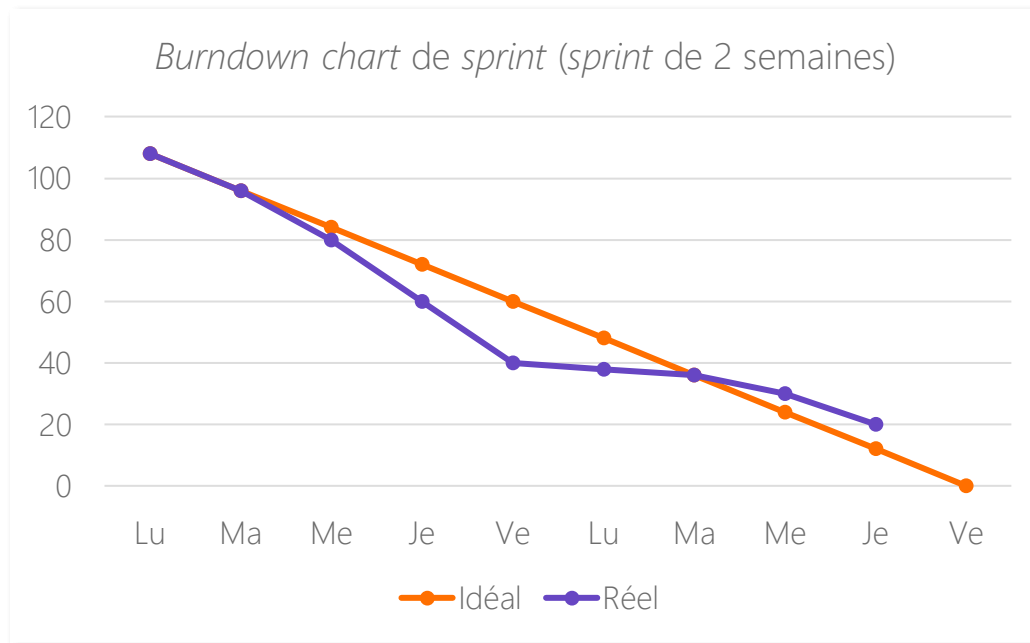
T

Testable

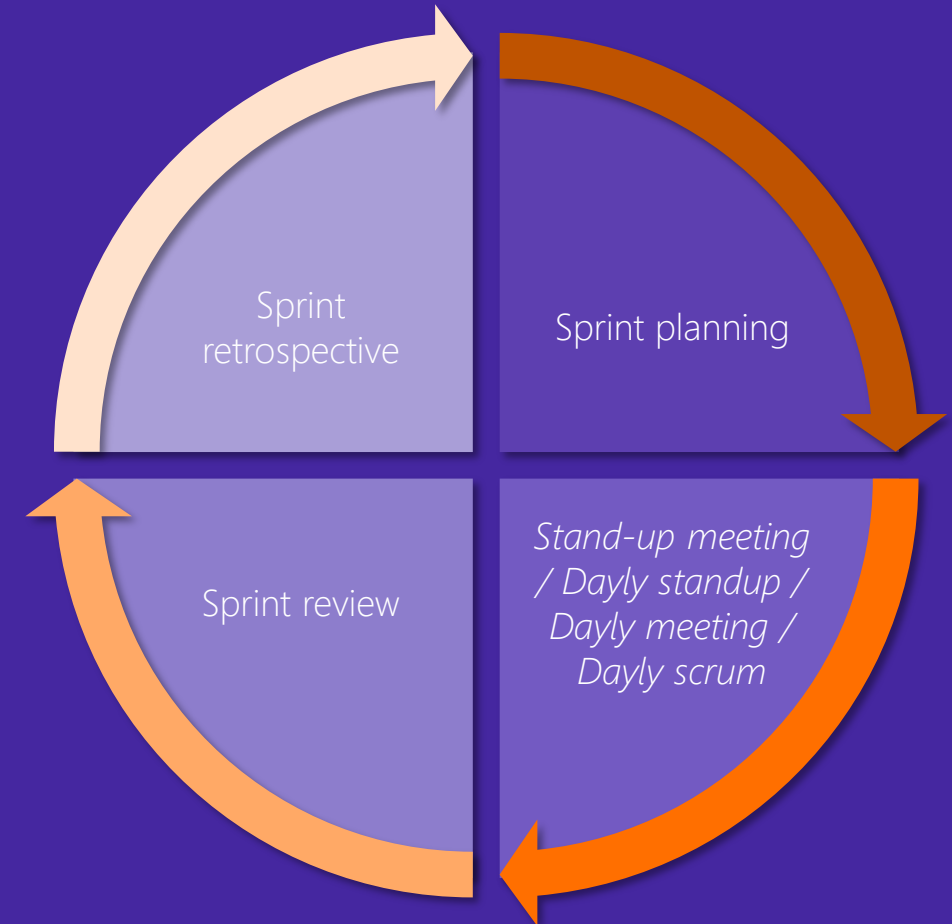
Les tests doivent découler de la *story* de façon évidente

Le *Burndown* chart

- Graphique d'avancement basé sur le « reste à faire »
- Indique l'effort en fonction du temps restant (jours, *sprints*...)
- Souvent utilisé pour piloter les *sprints* ou les *releases*
- Mise à jour fréquentes : quotidienne pour les *sprints*, fin de *sprints* pour les *releases*



Les cérémonies



Le *Sprint planning*

Qui ?

Team members, Scrum Master, Product Owner

Quand ?

Début de *sprint*

Combien de temps ?

Environ 1 h par semaine de *sprint*

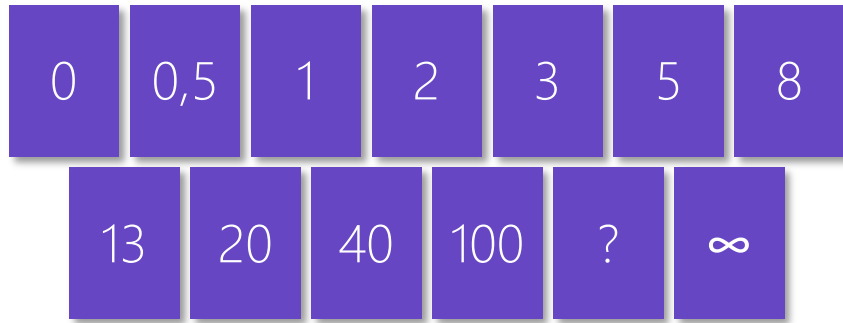
Quoi ?

- Construction du *sprint backlog* (*sprints* longs) ou basée sur le *sprint backlog* (*sprints* courts)
- Définition, estimation (en heures) et attribution des tâches en fonction des *user stories* du *sprint backlog*

Contraintes

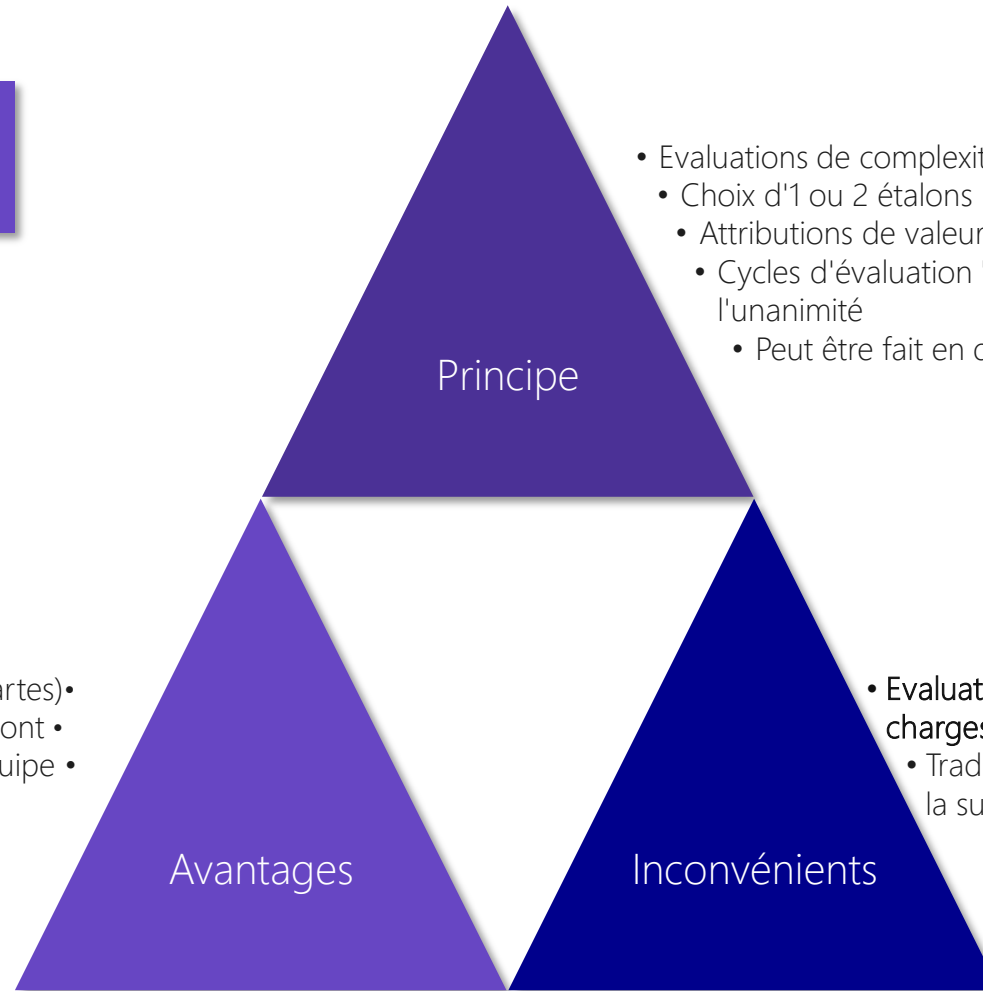
- Respect du cycle agile :
- Spécifications détaillées des *user stories* (tests d'acceptation)
 - Remaniement de l'architecture si nécessaire
 - Conception, implémentation et tests des composants
 - (tests unitaires)
 - Intégration et des composants (tests d'intégration)
 - *Package* produit (livraison)
 - Passage des tests d'acceptation

Le *Planning poker*



- Ludique (utilisation des cartes)
- Communication en amont
- Implication de l'équipe

<https://scrumpoker.online/>
<https://www.pointingpoker.com/>

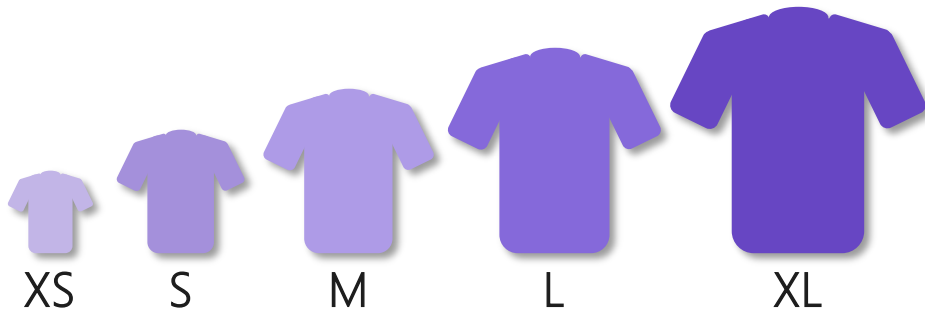


- Evaluations de complexités relatives de *user story*
- Choix d'1 ou 2 étalons (fonctionnalités connues)
- Attributions de valeurs (non extrêmes) aux étalons
- Cycles d'évaluation "secrète"/débat jusqu'à obtention de l'unanimité
 - Peut être fait en début de projet, puis affiné ensuite

- **Evaluation de complexités relatives, pas de charges**
 - Traduction en charge de travail à réaliser par la suite en début de sprint

Le *TShirt Sizing*

Variante simplifiée et plus « grossière » du *Planning poker*



- Ludique (utilisation des tailles de t-shirts)
- Communication en amont
- Implication de l'équipe

Avantages

Principe

- Evaluation de complexités relatives
- Choix d'un étalon de durée (ex : « M » = 2 semaines)
- Attributions de valeurs (non extrêmes) à l'étalon
- Évaluations « grossières » à affiner par la suite

Inconvénients

- **Evaluation de complexités relatives, pas de charges**
- Traduction en charge de travail à réaliser par la suite (souvent difficile)
- Ne remplace pas les autres méthodes

Le *TShirt Sizing*

Variante simplifiée et plus « grossière » du *Planning poker*

XS

- Très simple voire anodin
- Ce n'est même pas la peine d'en parler (ou presque)

S

- Simple et maîtrisé
- On a déjà fait ça souvent et on sait comment s'y prendre

M

- Taille moyenne
- La majorité des tâches/fonctionnalités devrait être M

L

- Plus complexe et moins maîtrisé
- Il va falloir y réfléchir avant de s'y attaquer

XL

- Très complexe et potentiellement très long
- On n'a jamais fait... et on ne sait pas vraiment comment s'y attaquer
- Est-ce qu'on peut redécouper en éléments plus petits et plus facilement maîtrisables ?

Le *Stand-up meeting* / *Dayly standup* / *Dayly meeting* / *Dayly scrum*

Qui ?

Team members, Scrum Master, Product Owner

Quand ?

Tous les jours, en début de journée

Combien de temps ?

15 mn **maximum**

Quoi ?

Tout le monde debout

Contraintes

Chaque *team member* expose en 1 mn :

- Ce qu'il a fait la veille pour l'atteinte des objectifs du projet
- Ce qu'il va faire aujourd'hui pour l'atteinte de ces objectifs
- Les éventuels freins pour lui ou pour l'équipe

Daily en vidéo

<https://youtu.be/EpysjhYBEfg>



La *Sprint review*

Qui ?

Team members, Scrum Master, Product Owner
Eventuellement certains *Stakeholders* du projet

Quand ?

Fin de *sprint*

Combien de
temps ?

30 à 60 mn

Quoi ?

Présentation du produit du *sprint* (démonstration)
Le tout en 5 étapes

- Rappel des objectifs du *sprint* définis pendant le *sprint planning*
- Démonstration du produit (*user stories* terminées uniquement, et par ceux qui les ont réalisées)
- Bilan du *sprint*, intégrant les retours en temps réel du *Product Owner* (y compris modifications du *product backlog*)
- Calcul de la vélocité réelle (capacité de production de l'équipe)
- Mise à jour du planning de release et du *burndown chart*

La Sprint review

En détails

À la **fin du sprint** juste avant la rétrospective

Adapter au fur et à mesure avec ce qui fonctionne l'équipe

Revoir – Évaluer – Adapter à partir du sprint écoulé

Présenter et
contrôler
l'incrément
réalisé

Donner du
contexte sur
les résultats
obtenus

Aborder les
éventuels
problèmes
rencontrés

Échanger
sur les
solutions à
mettre en
place pour
les éviter à
l'avenir

Ajuster le
Product
backlog, si
nécessaire

La Sprint review

Conseils

Préparer la réunion

- Lister les fonctionnalités qui vont être présentée
- Définir le scénario de la démo
- Noter les éléments nécessaires à la démonstration:
 - emplacements des fonctionnalités démos (URL...)
 - identifiants et mots de passe des cas de tests pour accéder
 - ...

Informelle... en apparence !

«Il s'agit d'une réunion informelle, pas d'une réunion de mise en état, et la présentation de l'incrément vise à susciter des feedbacks et à favoriser la collaboration.»

Le Scrum guide

Inclure les bonnes personnes

- Celles qui s'intéressent au produit
- Celles qui ont des avis tranchés et des suggestions qui aideront à faire avancer le produit
- Celles qui ne comprennent pas le produit, pour le challenge
- Celles qui vont entendre parler du produit sans réellement travailler dessus.

Noter toutes les suggestions

- Noter tous les retours faits pendant la review
- Ne pas prendre de décisions hâtives
- Echanger ensuite avec l'équipe sur d'éventuels modification du *Product backlog*

Les **feedbacks** augmentent les chances de succès

La *Sprint review* est conduite en toute **transparence**

La *Sprint review*

Suggestion de présentation

Bienvenue (5 mn – PO)

- Accueil des participants
- Rappel des objectifs du *sprint* et de l'engagement pris par l'équipe
- Partage des événements pertinents qui ont eu lieu pendant le *sprint*
- Focus sur le produit : pas de décisions managériales ou opérationnelles

Introduction (5 mn – Scrum Master)

- Présentation de l'agenda de la *Sprint Review*
- Rappel de l'importance d'échanger, commenter et débattre

Présentation de chaque fonctionnalité

- 5 mn par fonctionnalité
- Pitch par le dev
- Démo de la fonctionnalité

Échange sur chaque fonctionnalité présentée

- 10 mn par fonctionnalité
- 1 à 2 mn de réflexion pour tout le monde
- Lancement des échanges par le PO (choisir des participants au hasard pour amorcer si besoin)
- S'assurer qu'un maximum de feedbacks est récolté

Conclusion (15 mn – PO)

- Résumé des commentaires pertinents recueillis
- Partage de son avis sur les retours
- Explication de ses choix

Ce n'est qu'un exemple
Chaque projet/équipe/*review* est différent

La *Sprint* retrospective

Qui ?

Team members, Scrum Master, Product Owner

Quand ?

Fin de *sprint*

Combien de
temps ?

60 mn

Quoi ?

Bilan du *sprint* en mode « amélioration continue » :

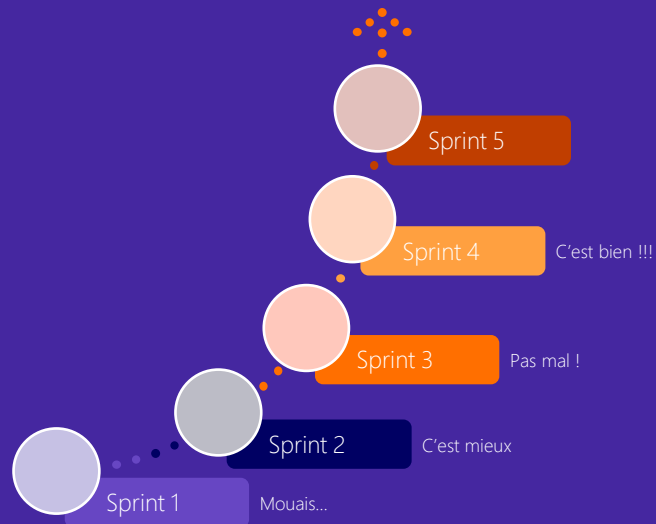
- Ce qui a bien fonctionné et comment capitaliser dessus => communiquez vos *best practices*
- Ce qui n'a pas fonctionné et comment le corriger et s'améliorer => imaginez des solutions

La Sprint retrospective

En détails

À la **fin du sprint** juste après la revue

Orientée **process** et non produit



Améliorer le fonctionnement de l'équipe et son quotidien

Relations
entre les
membres

Efficacité des
processus ou
des outils

Problèmes
intervenues
dans le sprint

Ce qui **n'a pas**
fonctionné et
ce qui **a bien**
fonctionné

La Sprint retrospective

Conseils

Organiser

- Ambiance :
 - Ouverture d'esprit
 - Collaboration
 - Constructif
- Logistique :
 - Nourriture & boissons
 - De quoi écrire
 - Post-it
 - ...

Communiquer

- Inciter les participants à s'exprimer
- Ne pas laisser de non-dits ni de sous-entendus

Synthétiser

- Inciter l'équipe à choisir les sujets à traiter... surtout s'il y en a beaucoup !

Prioriser

- Prioriser les actions à engager
 - Forte valeur
 - Rapide à mettre en œuvre
- Comme pour une *user story* !

Adapter la cérémonie aux **besoins** de l'équipe

Proposer des solutions

Sprint retrospective en vidéo

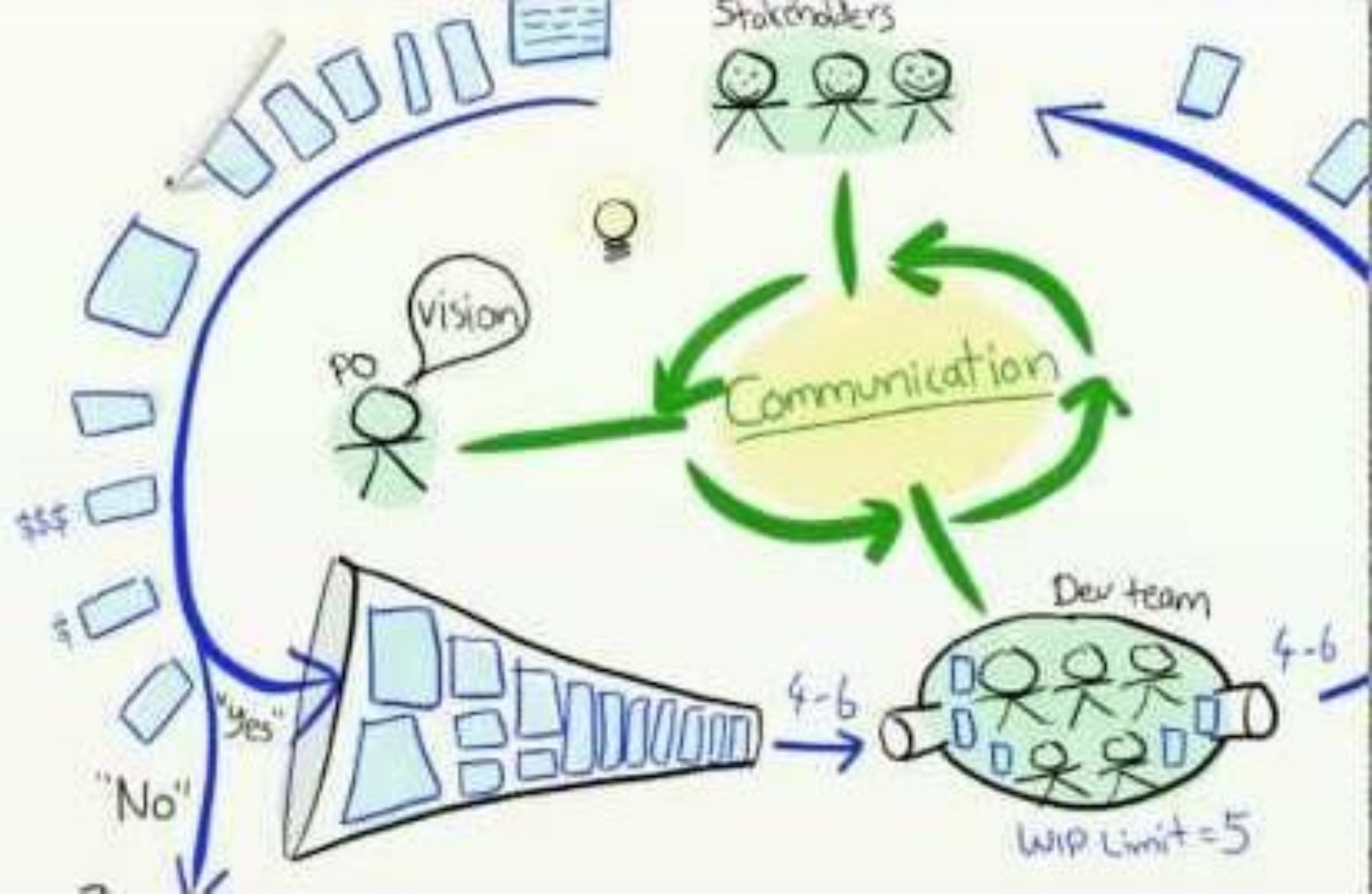
<https://youtu.be/oJp8dQxMNeo>





Synthèse en vidéo

<https://www.youtube.com/watch?v=3qMpB-UH9kA>



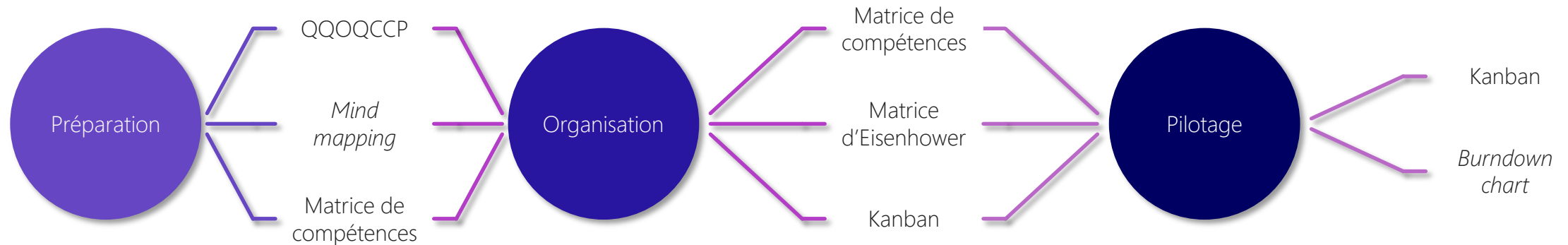
04

Des outils et des techniques

Inventaire (très) partiel (encore !)

Des outils et des techniques pour tous

Préparation, organisation, pilotage... quelques outils pour se simplifier la vie



QQOQCCP : les questions à se poser sans cesse

Analyse systématique d'un projet, d'une problématique, d'une situation

Pousse à envisager les différents aspects et à identifier des pistes de solutions

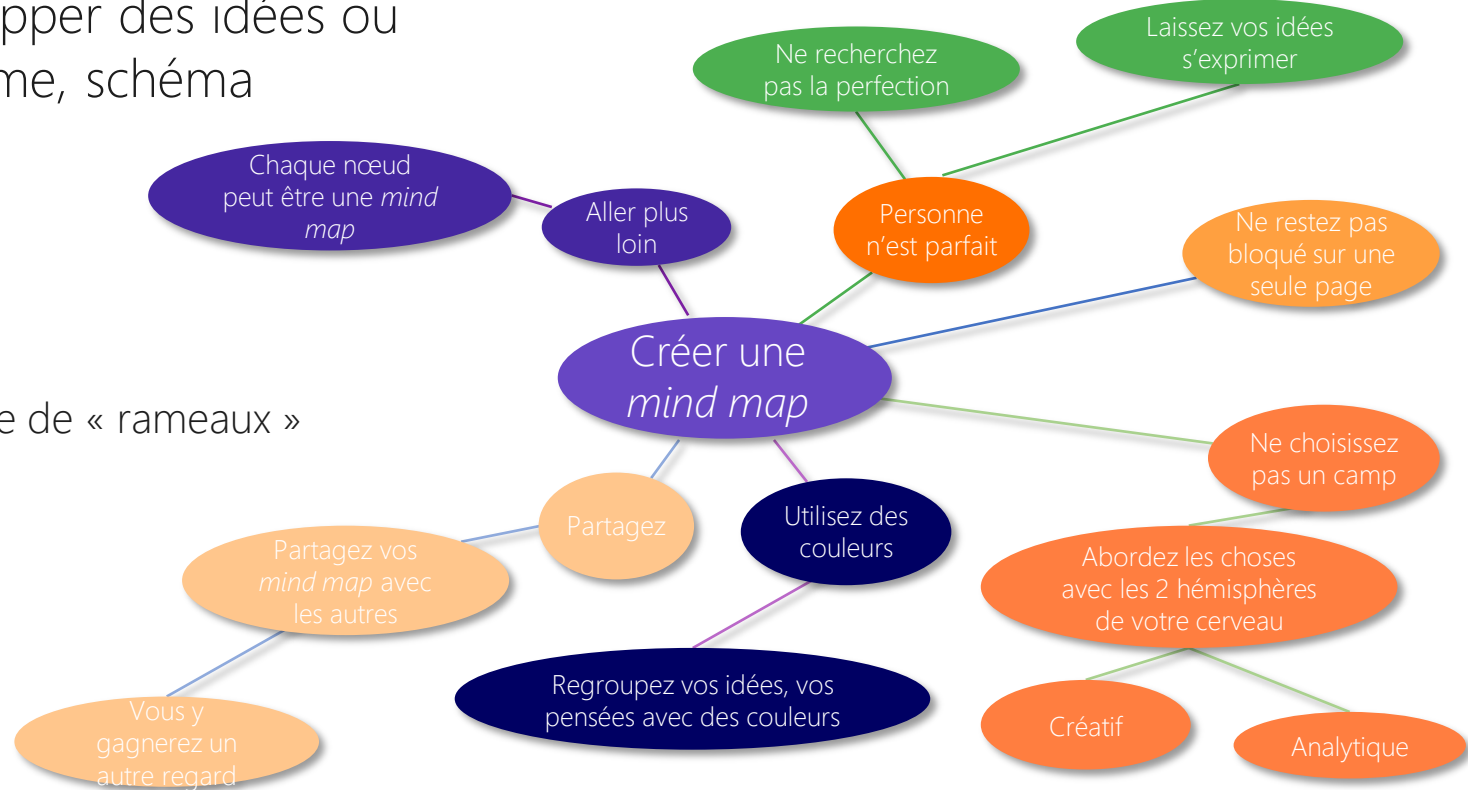
Adaptation des questions au contexte (projet à réaliser, incident à corriger, situation à améliorer...)

Quoi ?	Description du projet	De quoi s'agit-il ? Que produit-on ? ...
Qui ?	Client, parties prenantes	Qui est concerné ? Qui intervient ? ...
Où ?	Localisation, contexte	Où le produit sera-t-il utilisé ? Dans quel environnement ? Dans quel contexte ? ...
Quand ?	Planification, contraintes temporelles	Quand doit-on livrer ? Quels jalons ? Quelles contraintes temporelles ? ...
Comment ?	Méthodes, outils, solutions	De quelle manière va-t-on procéder ? Quels outils seront mis en œuvre ? ...
Combien ?	Moyens et ressources	Quel coût ? Quels moyens ? Quelles ressources ?
Pourquoi ?	But et objectifs du projets	Dans quel but ? Quelle finalité ? A quoi cela servira-t-il ?

Mind mapping

Collecte d'idées ludique
Représentation par nœuds, flexible, permettant d'associer, capitaliser, enrichir, développer des idées ou des concepts (*mind map*, topogramme, schéma heuristique...)

Sujet ou thème au centre
Idées ou rubriques principales réparties autour comme des branches
Idées « secondaires » représentées sous forme de « rameaux »
Utilisation de libellés les plus courts possibles



Freeplane : www.freeplane.org
Freemind : freemind.sourceforge.net
XMind : www.xmind.net
Lucidchart : www.lucidchart.com

Matrice de compétences

Utile pour les recrutements, la constitution d'équipe...

Tableau synthétisant le niveau des compétences techniques et fonctionnelles des personnes

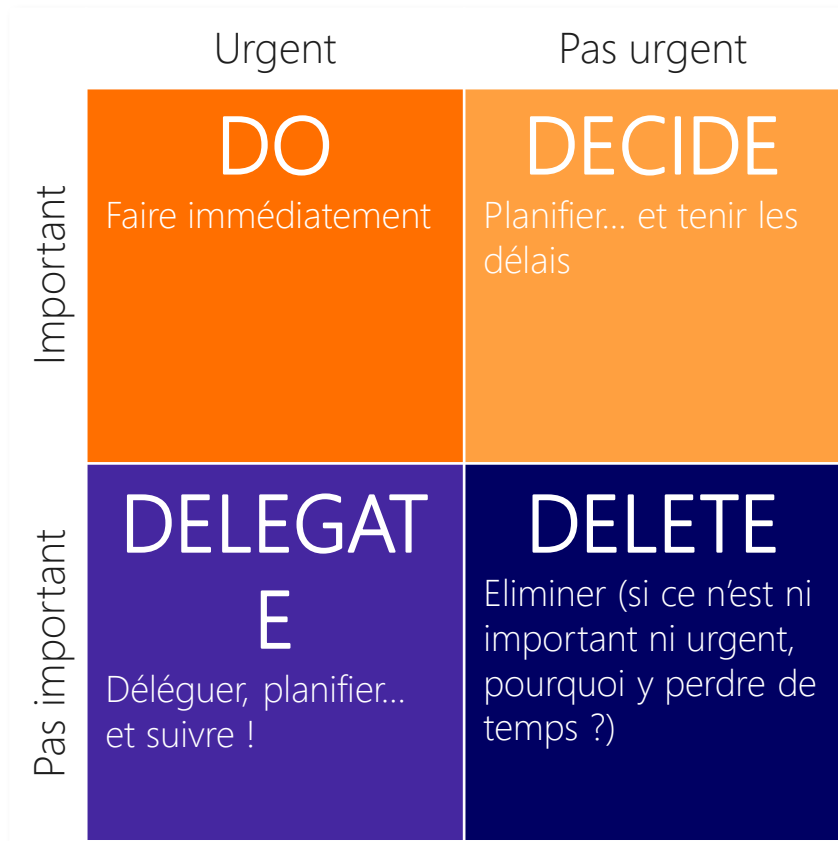
Facilite la constitution d'équipes projets (matching compétences nécessaires vs compétences disponibles)

Permet d'identifier les manques

Compétences	Gandalf	Frodo	Galadriel	Arwen	Aragorn	Éowyn
Equitation	4	2	3	3	4	4
Magie (feu)	4	0	0	0	0	0
Magie (Eau)	0	0	4	0	0	0
Escrime	3	1	2	3	4	4

Niveaux : 0 = pas de compétences | 1 = connaissances théoriques (scolaires) | 2 = première expérience | 3 = bonne expérience | 4 = expert

Matrice d'Eisenhower – aide à la priorisation



Aide visuelle à la priorisation de tâches, projets...

Faire le tri entre l'urgence et l'importance des items

Arbitrer entre faire soi-même et déléguer

Kanban (signe en japonais)

Outil de management visuel des tâches d'une équipe
(représentation visuelle du *Product backlog* et du *Sprint backlog*)

Tableau des tâches : ce qu'il y'a à faire, qui le fait, où ça en est

Principe très adaptable en fonction du contexte, des besoins

Utilisable « en physique » (tableau & post-it) aussi bien qu'en digital

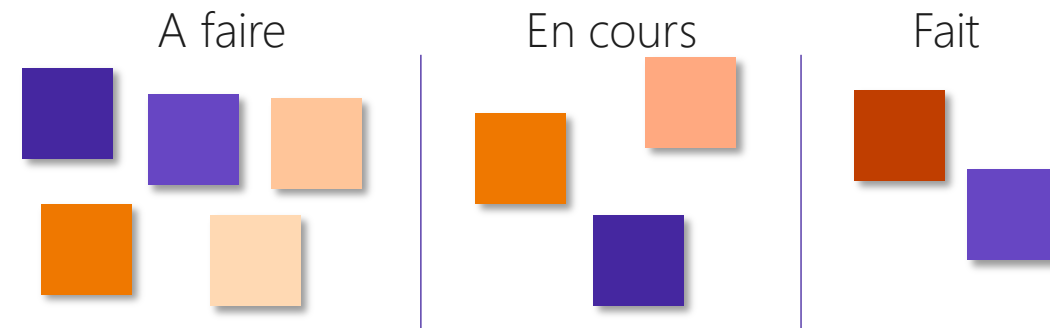


Tableau Kanban « de base »

Kanban – exemples

