
product
terminology
values
retrospectives
productive
ways
roles
understand
Scrum
teams
time
team
work
ceremonies
progress
Product
dynamic
scrum
form
done
share
look
negotiation
planning
broken
processes
Successful
sprint
framework
completed
prioritized
one
burndown
right
items
meets
refined
artifacts
value
backlog
Team
agile
deliver
choose
embrace
priority
tasks
project
set
day
comprehensive
struggles
owner
complete
chart
Three
Items
Sprint
Manifesto
three

SOMMAIRE

- MÉTHODES PRÉDICTIVES
- L'AGILITÉ
- SCRUM

OBJECTIFD

A la fin de ce séminaire, l'élève doit :

- Connaître les limites des méthodes prédictives
- Comprendre les principes fondateurs de l'agilité
- Connaître les acteurs de Scrum ainsi que leurs rôles
- Connaître les times-boxes Scrum et leurs utilités
- Savoir définir des US INVEST ainsi que leur complexité

I

SURVOL DES METHODES PREDICTIVES

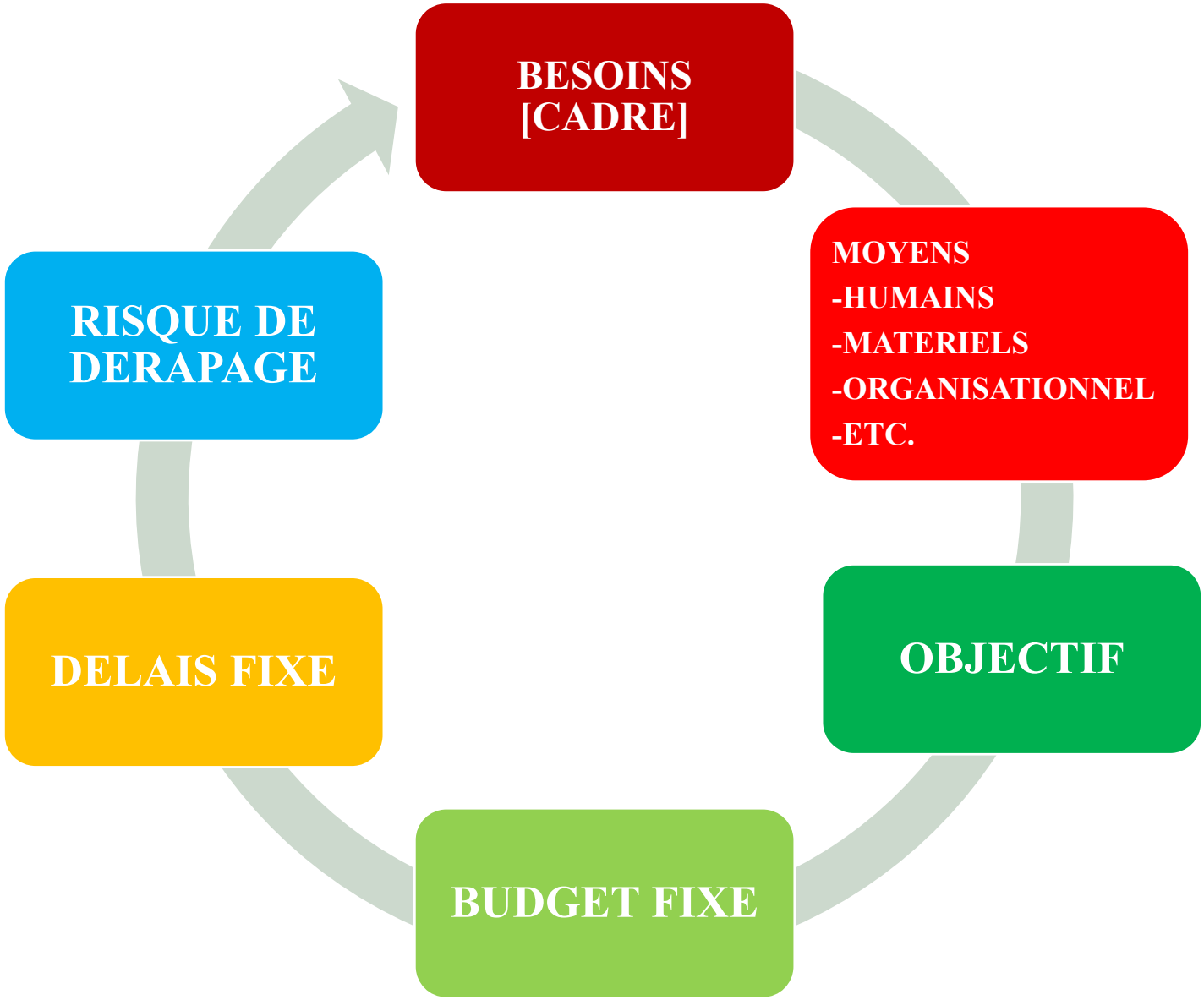
SOMMAIRE

- Introduction
- Définition
- Exemples de projets
- Démarche
- Limites

INTRODUCTION

Quand on parle de projet, cela vous dit quoi ?

DEFINITION


BESOINS
[CADRE]
MOYENS
-HUMAINS
-MATERIELS
-ORGANISATIONNEL
-ETC.
OBJECTIF
BUDGET FIXE
DELAIS FIXE
RISQUE DE
DERAPAGE

EXEMPLE DE PROJET

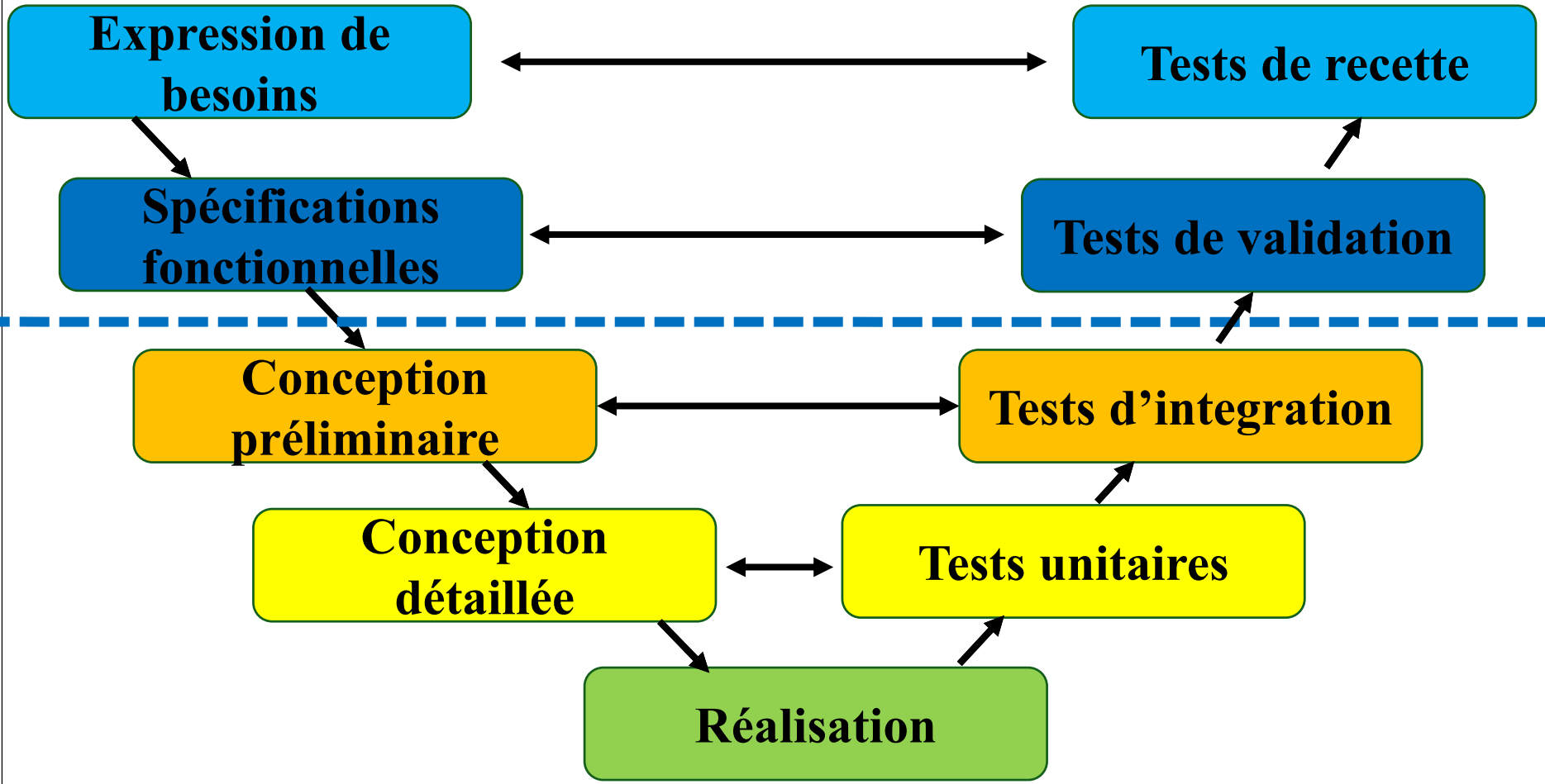
- Votre prochain mariage
- Aller faire du shopping
- Servir le café à 10h
- Organiser la remise de diplomes
- Mettre en place une plateforme ENT
- Etc.

DEMARCHE

Dans la demarche classique de gestion de projet, nous retrouvons un certains nombre de phases plus ou moins figées notammant :

- **Expression de besoins**
- **Specifications fonctionnelles**
- **Conception preliminaire**
- **Conception detaillée**
- **Realisation**
- **Tests unitaire**
- **Tests d'integration**
- **Tests de validation**
- **Tests de recette**

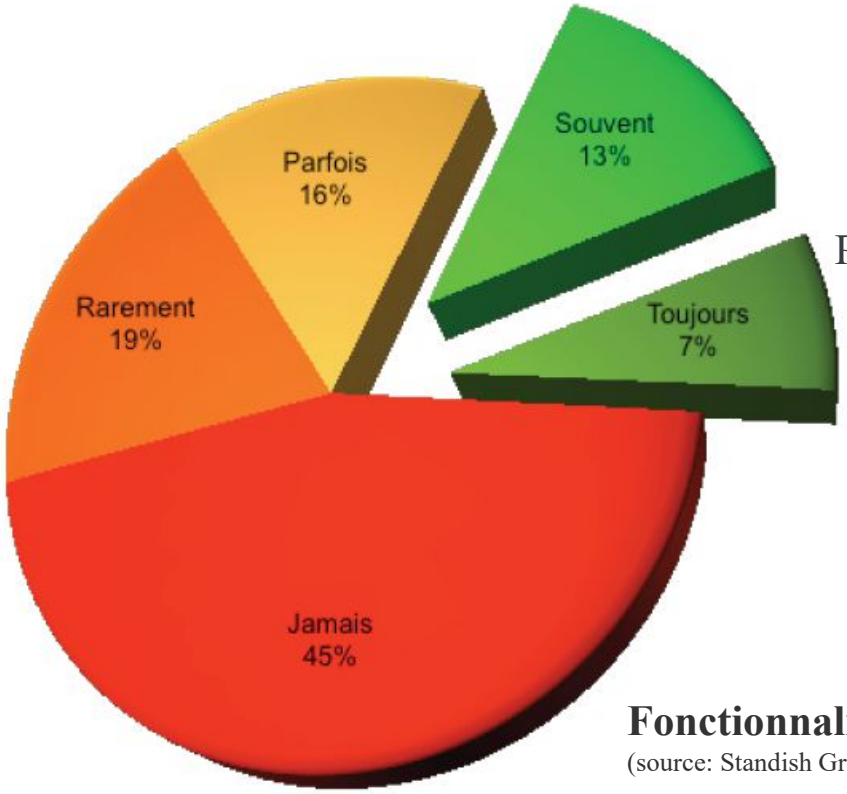
RESUME


Expression de besoins
Tests de recette
Spécifications fonctionnelles
Tests de validation
Conception préliminaire
Tests d'integration
Conception détaillée
Tests unitaires
Réalisation

LIMITES (1/3)

Taux de succès des projets informatiques en 2009 de 32%

- Source : enquête Standish Group sur 8000 projets



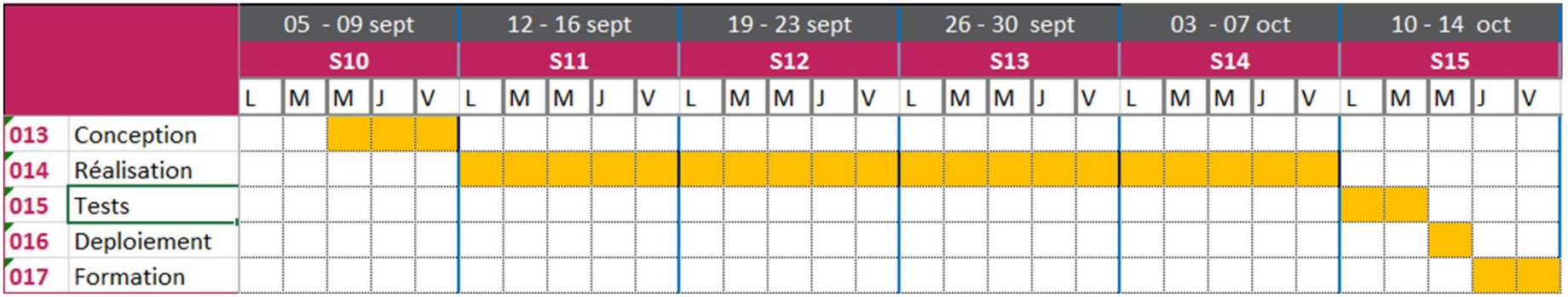
Peu de fonctionnalités développées réellement utilisées

- **45% de fonctionnalités jamais utilisées**

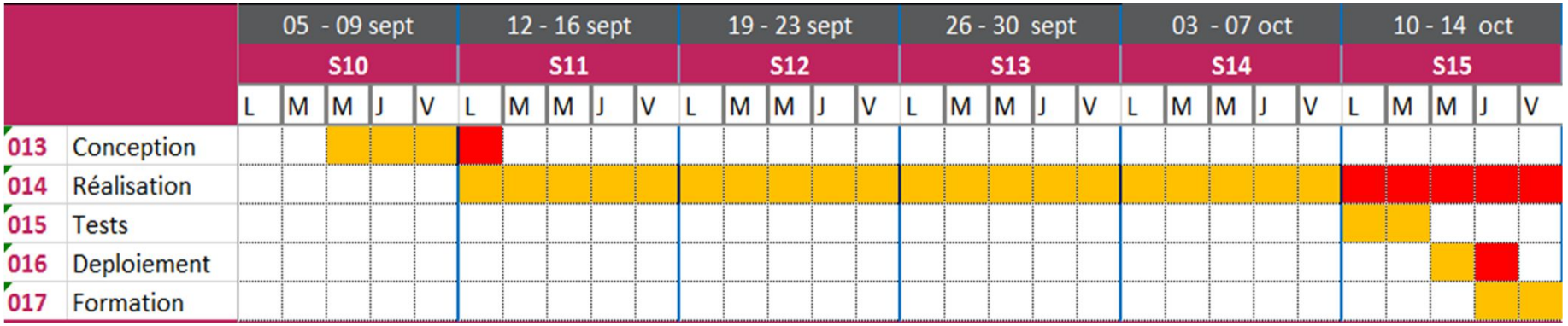
Fonctionnalités utilisées d'un SI en %
(source: Standish Group Study reported at XP 2002 by Jim Johnsonn, chairman)

LIMITES (2/3)

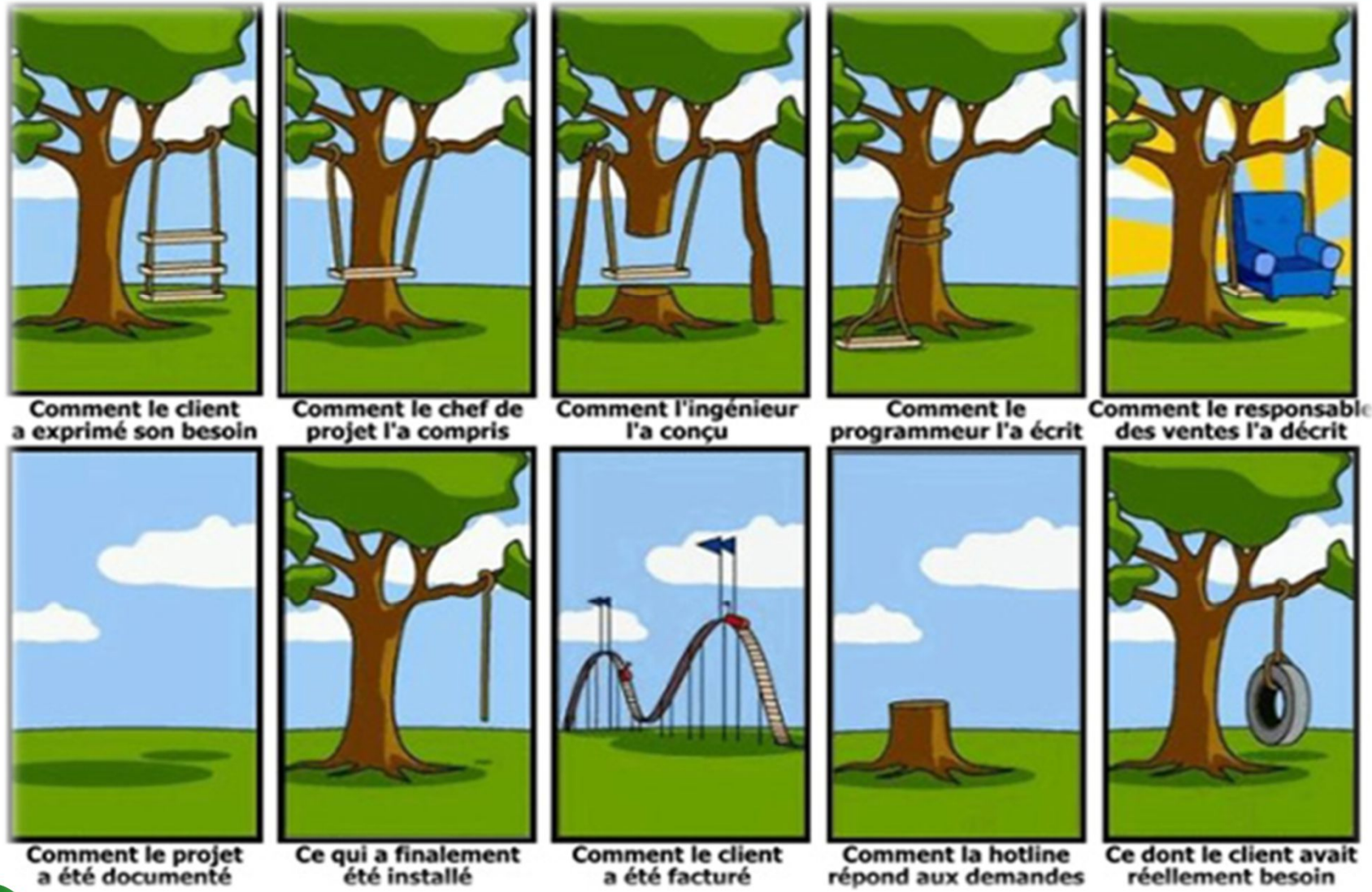
Projet idéal



Le cauchemar



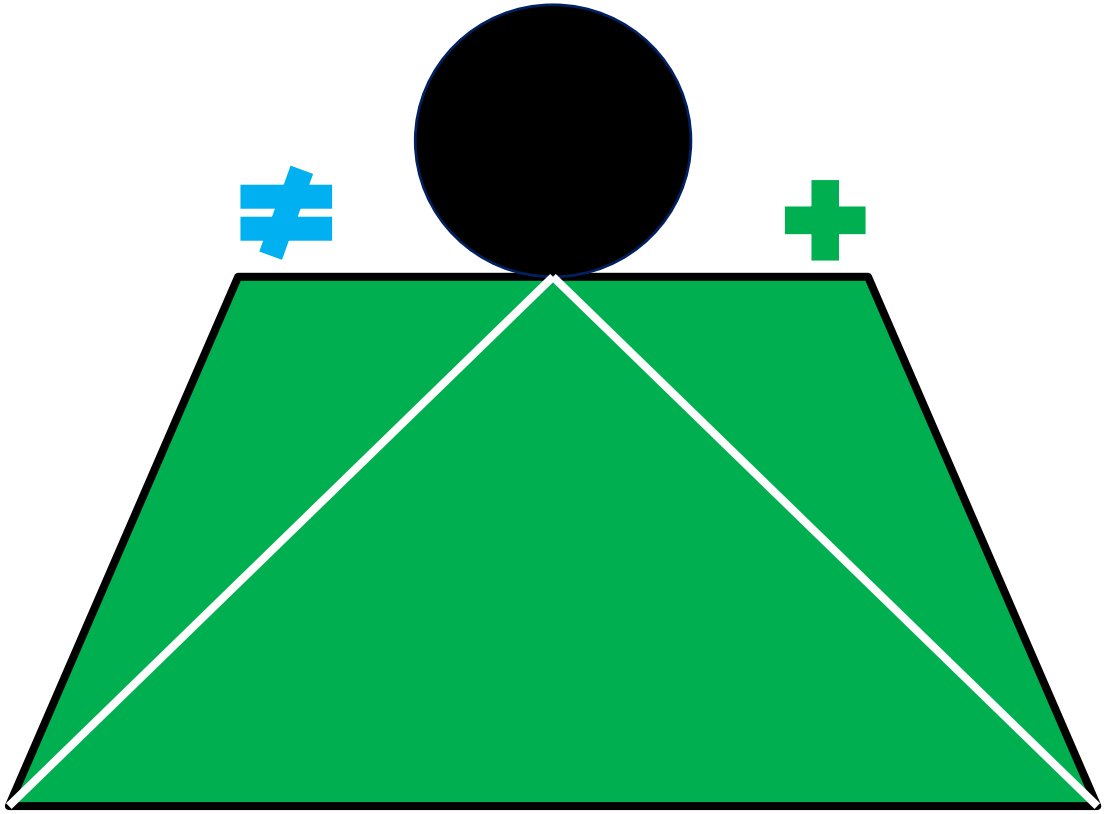
LIMITES (3/3)


Comment le client a exprimé son besoin
Comment le chef de projet l'a compris
Comment l'ingénieur l'a conçu
Comment le programmeur l'a écrit
Comment le responsable des ventes l'a décrit
Comment le projet a été documenté
Ce qui a finalement été installé
Comment le client a été facturé
Comment la hotline répond aux demandes
Ce dont le client avait réellement besoin

LES CRITÈRES QUI POUSSENT VERS L'AGILITÉ

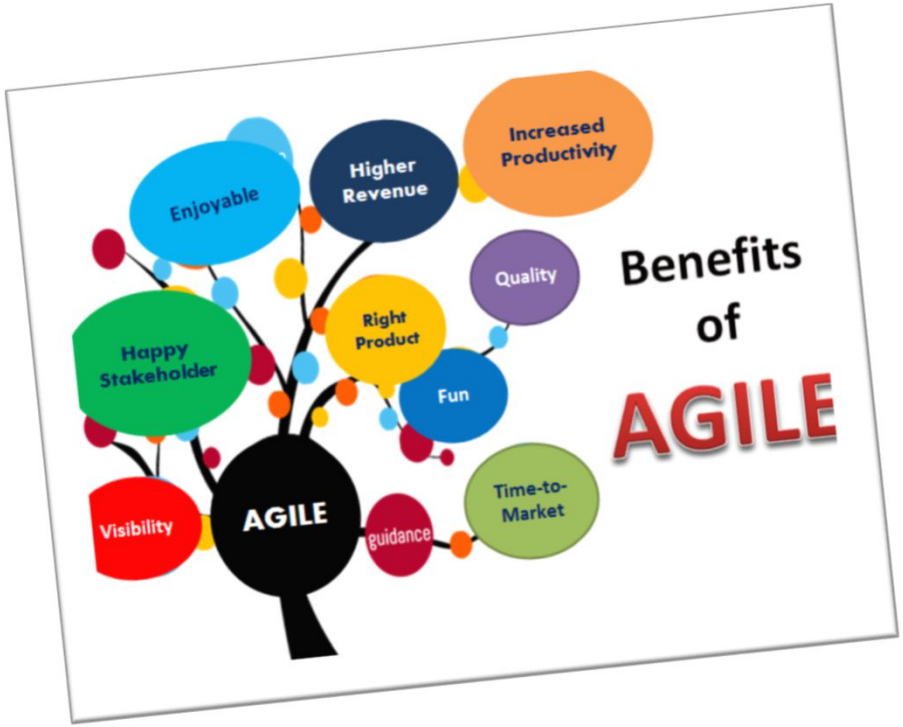
- **Le périmètre fonctionnel du projet n'est pas très clair au début et risque de bouger au cours du temps**
- **Il y a de forts risques de ne pas réussir facilement à répondre au besoin du client, et il peut être salutaire de valider régulièrement avec lui ce qui est réalisé par l'équipe**
- **Il est nécessaire de livrer très rapidement une première version, quitte à livrer une version ne contenant que les fonctionnalités primordiales**
- **Etc.**

IDENTIFICATION DE BESOIN



II

AGILITE


Benefits
of
AGILE
AGILE
Enjoyable
Higher Revenue
Increased Productivity
Quality
Right Product
Fun
Happy Stakeholder
Visibility
guidance
Time-to-Market

SOMMAIRE

- Pour quoi l'agilité
- Définition
- Historique
- Organisation
- Pilotage par valeur métier
- Quelques méthodes agiles
- Quelques chiffres

POUR QUOI L'AGILITE

Les méthodes de gestion de projet classiques ont de nombreux inconvénients.

- Trop d'efforts sur la phase de planification
- Beaucoup de documentations
- Le plan pas adapté aux éventuels changements
- Détection tardive des écarts

Il est finalement apparu de nouvelles méthodes de développement regroupé sous le terme : **Agile Software Development**. L'**Agile Alliance regroupe** des personnes promeuvent l'utilisation de ces méthodes.

DEFINITION (1/2)

Depuis un dictionnaire on définit l'agilité comme :

- Habilité à bouger rapidement de façon légère et gracieuse
- Caractéristique de ce qui s'adapte facilement
- Synonyme de légèreté de souplesse, facilité à se mouvoir
- Capacité à réagir promptement

Dans le monde industriel, l'agilité se définit :

- Par la capacité à s'adapter et à réagir à l'environnement.

Le concept de méthode Agile met en avant l'interactivité entre les acteurs d'un projet. Le dialogue a pour but d'augmenter le niveau de satisfaction du client.

DEFINITION (2/2)

- L'agilité c'est une composante majeure d'un large mouvement d'auto-management, où la résolution de la complexité de détail est confiée à la compétence et à la motivation rationnelle du personnel d'exécution.
- L'agilité a émergé d'une recherche d'amélioration continue se basant sur l'intelligence collective des équipes qui la pratiquent
- Etre agile est le fait de développer un logiciel d'une grande qualité fournissant le maximum de valeur ajoutée le plus tôt possible
- **Agile = Itératif + Incrémental + Adaptatif**

HISTORIQUE (1/2)

Février 2001, aux USA, 17 spécialistes du développement logiciel se sont réunis pour unifier leurs méthodes respectives. Les plus connus d'entre eux étaient :

- **Ken Schwaber** et **Jeff Sutherland**, fondateurs de **Scrum**,
- **Kent Beck**, père de **XP** et cofondateur de **JUnit**,
- **Ward Cunningham**
- **Jim Highsmith**, prônant l'**ASD**
- **Alistair Cockburn** pour la méthode **Crystal clear**,
- **Martin Fowler**, **Dave Thomas** et **Arie van Bennekum** pour **DSDM** , la version anglaise du **RAD**
- **Robert Cecil Martin** pour **Clean Architecture**.
- **Etc.**

HISTORIQUE (2/2)

Ces 17 experts venant tous d'horizons différents réussirent à extraire de leurs concepts respectifs des critères pour définir une nouvelle façon de développer des logiciels : A l'issue de cette réunion devait émerger le **Manifeste Agile**, considéré comme la définition canonique du **développement agile** et de ses principes sous-jacents.

Le Manifeste agile est constitué de 4 valeurs et de 12 principes fondateurs.

AGILITE : 4 VALEURS

Manifeste agile commence ainsi :

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 :

Individus et interactions plutôt que processus et outils.

Logiciel fonctionnel plutôt que documentation complète.

Collaboration avec le client plutôt que négociation de contrat.

Réagir au changement plutôt que suivre un plan.

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

De ces quatre valeur, découlent douze principes.

AGILITE : 12 PRINCIPES (1/4)

- **Notre plus haute priorité est de satisfaire le client en livrant rapidement et régulièrement des fonctionnalités à grande valeur ajoutée.**
- **Accueillez positivement les changements de besoins, même tard dans le projet. Les processus agiles exploitent le changement pour donner un avantage compétitif au client.**
- **Livrez fréquemment un logiciel opérationnel avec des cycles de quelques semaines à quelques mois et une préférence pour les plus courts.**
- **Les utilisateurs ou leurs représentants et les développeurs doivent travailler ensemble quotidiennement tout au long du projet.**

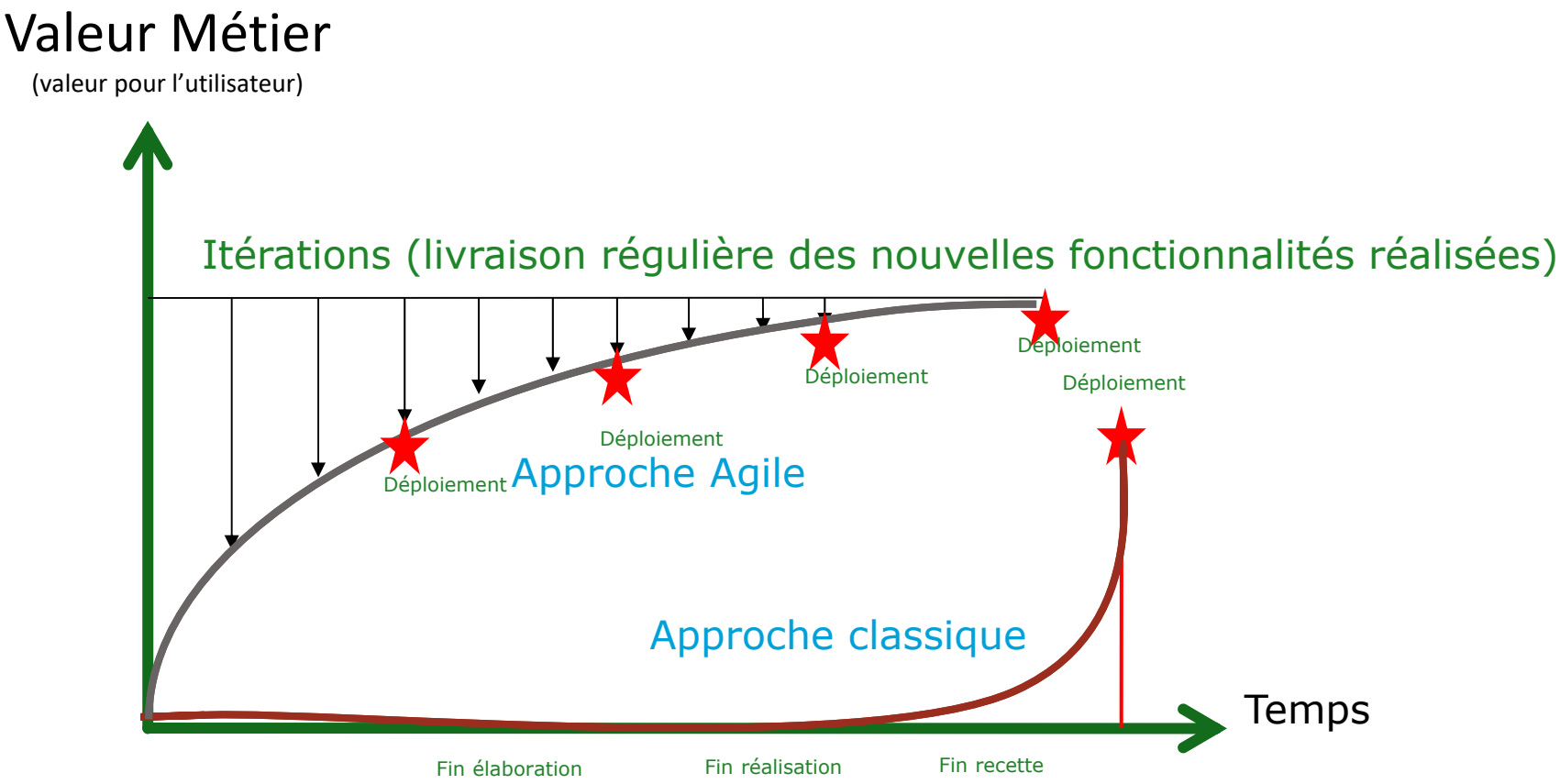
AGILITE : 12 PRINCIPES (1/4)

- Réalisez les projets avec des personnes motivées. Fournissez-leur l'environnement et le soutien dont ils ont besoin et faites-leur confiance pour atteindre les objectifs fixés.
- La méthode la plus simple et la plus efficace pour transmettre de l'information est le dialogue en face à face.
- Un logiciel opérationnel est la principale mesure d'avancement.
- Les processus agiles encouragent un rythme de développement soutenable. Ensemble, les commanditaires, les développeurs et les utilisateurs devraient être capables de maintenir indéfiniment un rythme constant.

AGILITE : 12 PRINCIPES (1/4)

- Une attention continue conduit à l'excellence technique et une bonne conception renforce l'agilité.
- La simplicité – c'est-à-dire l'art de minimiser la quantité de travail inutile – est essentielle.
- Les meilleures architectures, spécifications et conceptions émergent d'équipes auto-organisées
- À intervalles réguliers, l'équipe réfléchit aux moyens de devenir plus efficace, puis règle et modifie son comportement en conséquence.

PILOTAGE PAR LA VALEUR METIER



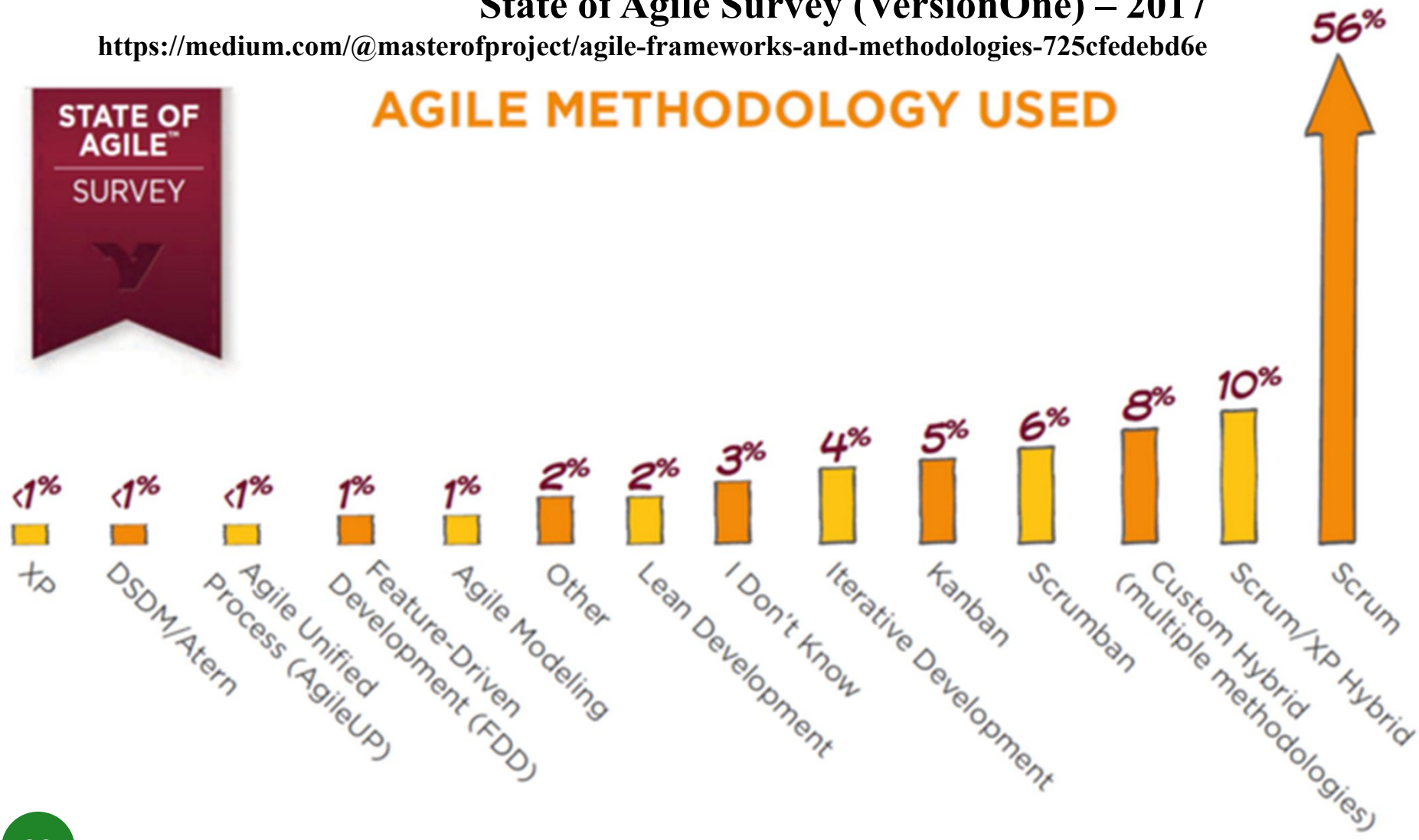
QUELQUES METHODES AGILES

- **RAD** (Rapid Application Development)
- **DSDM** (Dynamic Software Development Method)
- **UP** (Unified Process)
- **RUP** (Rational Unified Process)
- **XP** (eXtreme Programming)
- **Scrum**
- **Lean**
- Etc.

STATISTIQUES

State of Agile Survey (VersionOne) – 2017

https://medium.com/@masterofproject/agile-frameworks-and-methodologies-725cfedebd6e



III

SCRUM

SCRUM

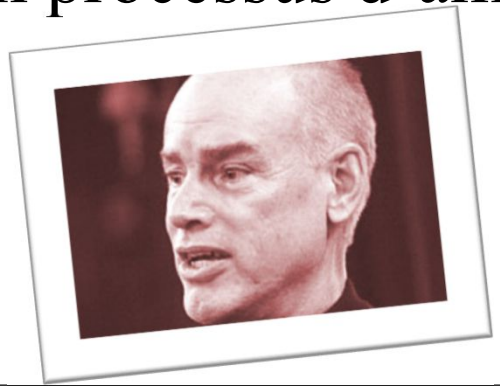
- Introduction
- Définition
- Qui utilise Scrum ?
- Composition
- Processus empirique
- Scrum vs Cycle en V
- Piliers
- Rôles
- Artéfacts
- Meetings
- Code review

INTRODUCTION

- Créée en 20O2 par Ken Schwaber et Jeff Sutherland,
- SCRUM tient son origine du terme sportif de rugby signifiant mêlée.
- La méthodologie demande à ses acteurs d'être soudés dans l'atteinte d'un but
- SCRUM veux avant tout produire un logiciel fonctionnel.
- SCRUM veux mettre en avant la satisfaction client.
- Ce dernier doit pouvoir donc juger l'avancement de son produit.
- L'équipe de développement s'organise elle-même pour déterminer la meilleure façon de produire les exigences les plus prioritaires

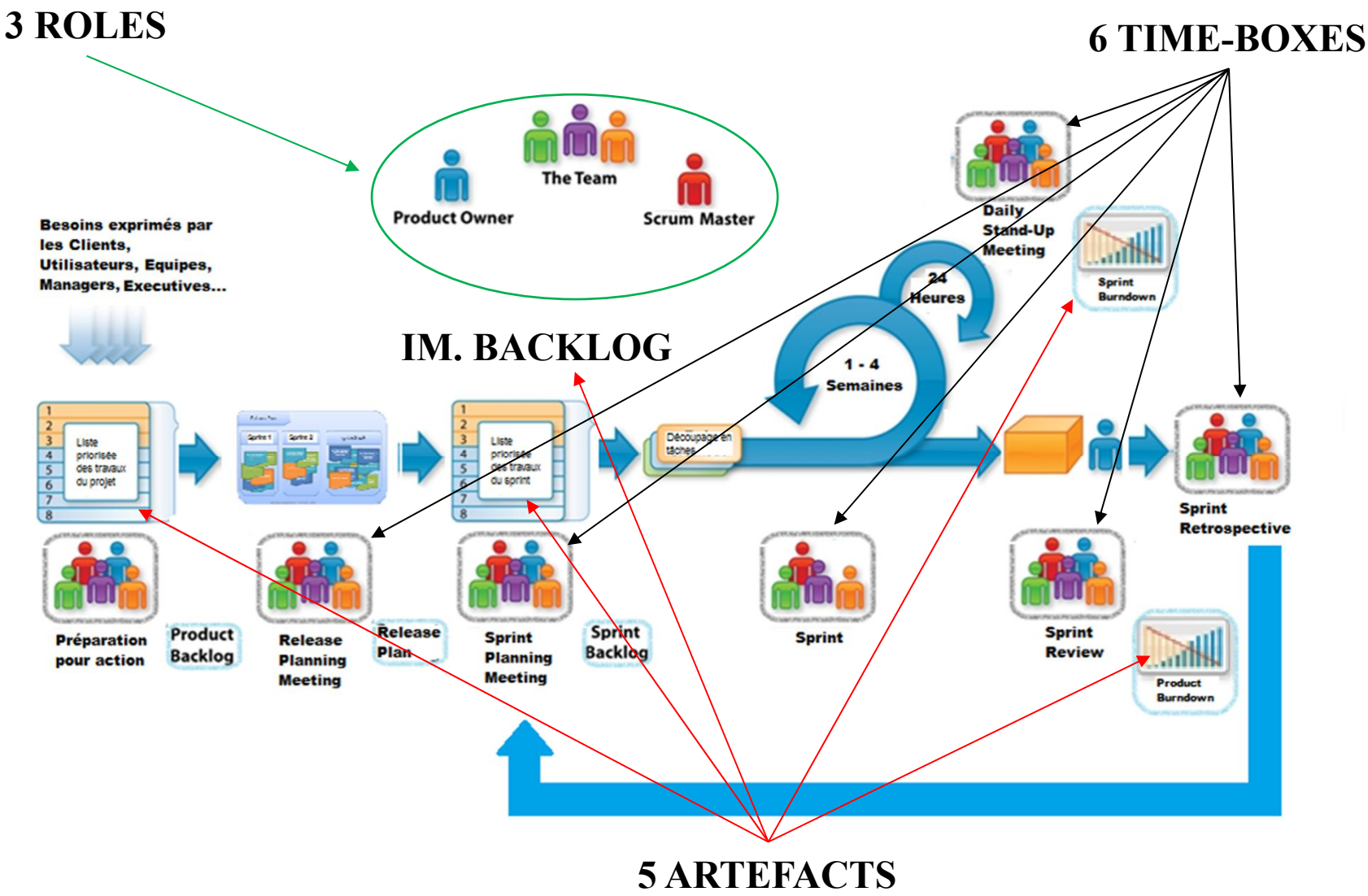
DEFINITION

- Une implémentation de l'agilité
- Scrum n'est pas une méthodologie. Scrum ne fournit pas les réponses à la manière de construire des logiciels de qualité plus rapidement.
- Scrum est un cadre dans lequel le jeu du développement de produit est joué.
- Votre équipe joue et, le bon ou le mauvais deviennent très visibles.
- Votre équipe est dans un processus d'amélioration continue.

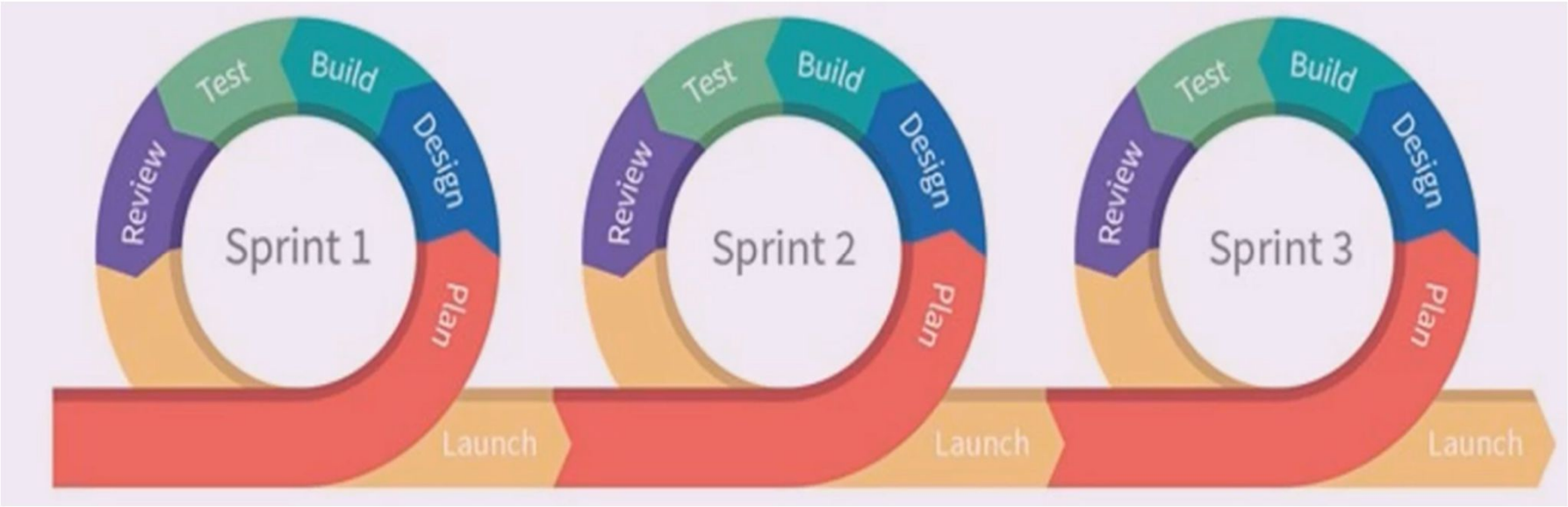


QUI UTILISENT SCRUM ?

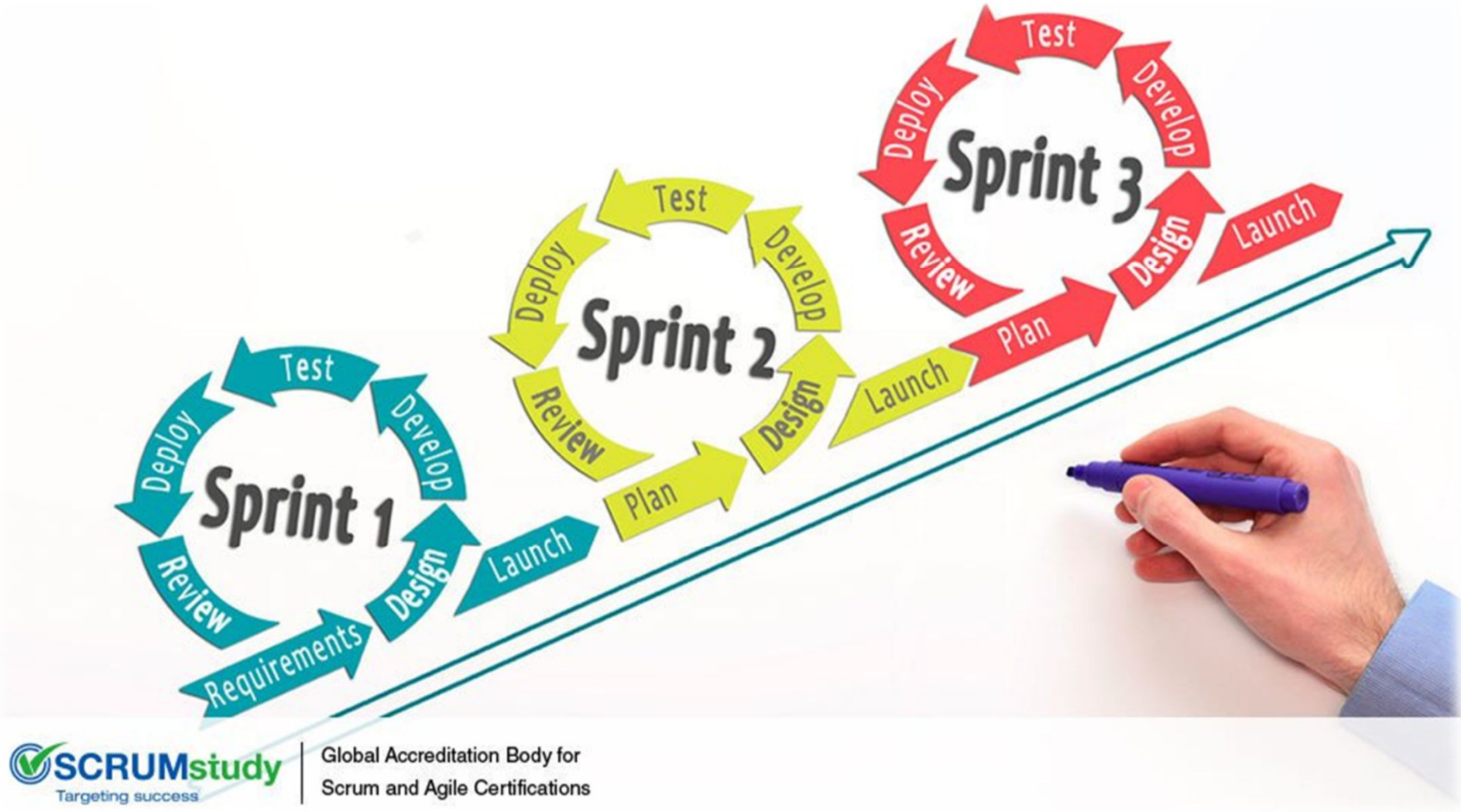
COMPOSITION



SCRUM : PROCESSUS EMPIRIQUE



SCRUM VS METHODES EN CASCADE



SCRUMstudy
Targeting success

Global Accreditation Body for
Scrum and Agile Certifications

PILIERS

Scrum repose sur 3 piliers

Transparence

Inspection

Adaptation

ROLES

- PRODUCT OWNER (PO)
- SCRUM MASTER (SM)
- SCRUM DEVELOPMENT TEAM (SDT)

PRODUCT OWNER (PO)

- Représentant du client
- Responsable du ROI (Retrun On Investment)
- Priorise le product backlog
- Planifie les releases et les livraisons
- Dirige le projet d'un point de vue du métier.
- Il communique une vision claire du produit
- Il accepte ou rejette le produit à la fin d'un sprint.
- Veille à ce que l'équipe travaille seulement sur la partie la plus importante du Backlog.
- Il donnant rapidement les informations dont l'équipe a besoin.
- Etc.

SCRUM MASTER(SM)

- Ce n’est pas un chef de projet
- Ce n’est pas un manager pour l’équipe de développement
- Protège l’équipe contre toutes perturbations extérieures
- « Organise et préside les réunions »
- Il s’assure que l’agilité est respectée par l’équipe.
- Il est en charge des obstacles remontées par l’équipe.
- Serviteur de l’équipe
- Etc.

SCRUM DEVELOPMENT TEAM(SDT)

- Pluridisciplinaire
- Auto-organisée
- Délivre le produit et est responsable de sa qualité.
- Travaille avec l'ensemble des demandeurs pour créer le Backlog de produit.
- Analyse le Backlog du produit afin que ses membres aient toutes les informations pour développer.
- Est responsable de la conception et fournit les fonctionnalités prévues.
- Est responsable de son travail.
- Travail continuellement avec le PO
- Etc.

ARTEFACTS

- PRODUCT BACKLOG
- SPRINT BACKLOG
- IMPEDIMENT BACKLOG
- BACKLOG BURNDOWN CHART
- SPRINT BURNDOWN CHART

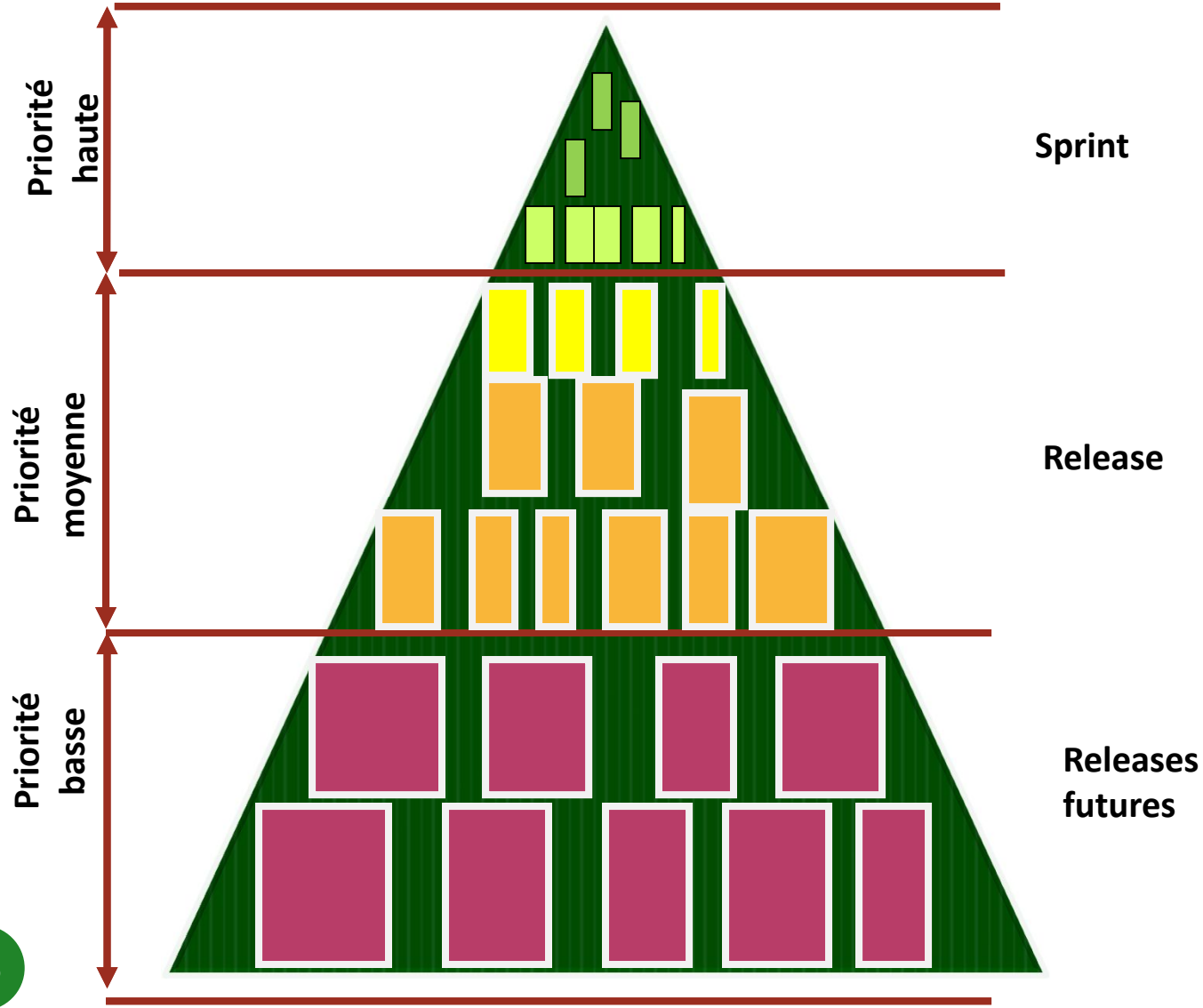
ARTEFACTS

- Les équipes utilisent des Artefacts durant le cycle de vie du projet.
- Ce sont des outils pour suivre et améliorer l'efficience de l'équipe durant un projet.
- Ils sont basés sur les bonnes pratiques du management visuel.

PRODUCT BACKLOG

- La liste des éléments que l'on veut réaliser dans son produit.
- Les éléments sont triés sur la valeur ajoutée pour l'application
- Les fonctionnalités sont connues sous les nom de PBI (Product backlog items)
- Les PBI sont mis à jour par le Product Owner
- Chaque élément doit apporter de la valeur aux utilisateurs finaux
- Les priorités sont définies par le Product Owner
- Les priorités sont revues avant chaque nouveau sprint

PRODUCT BACKLOG



USER STORY

C’est la traduction d’une exigence client en une phrase simple.

Elle respecte le format:

As a «*ROLE*» I’d like to «*ACTION*» [so that « *PURPOSE* »]

Exemples:

- En tant qu’administrateur, je dois pouvoir créer un élève afin qu’il existe dans le système
- En tant qu’élève, je dois pouvoir visualiser mes notes
- En tant que professeur, je dois pouvoir réserver une salle pour y faire cours

EPIC, US et TACHES

- Une Epic est une très grande fonctionnalités ou grosse US
  - **Exemple** :Gestion des Etudiants
- Une User Story est le nom d'une fonctionnalité
  - **Exemple**
    - ✓ Créer un Etudiant
    - ✓ Supprimer une note
    - ✓ Réserver une salle
- Un tâche est une étape atomique d'une US
  - **Exemple** :Créer étudiant
    - Formulaire
    - Service
    - Controller
    - Test unitaire
    - Test d'intégration
    - Test QA (Quality assurance)

BIEN DEFINIR SES USERS STORIES

- Selon Bill william (2003)
- Une bonne User Story doit être **INVEST**.

INDEPENDENT

NEGOCIABLE

VALUABLE

ESTIMABLE (EVALUABLE)

SMALL

TETSTABLE

VALORISER LES US

Permet de classer les **US** selon la valeur métier.
Sont possibles les valeurs **FISPE aussi MuSCoW**

F ONCTIONNALITE (*)
I NDISPENSABLE (100) (Must)
S OUHAITABLE (050) (Should)
P POSSIBLE (025) (Could)
E LIMINE (000) (Won't)

DEFINITION DONE ?

Définition de DONE: Un exemple basic

- Code documenté et commité (push)
- Respectant les règles définies dans le PAQL (s'il y en a)
- Tests écrits et effectués avec succès
- Build effectué sans erreur et intervention humaine
- Respect de la qualité de code (Sonar)
- Présenté lors de la revue de Sprint
- Accepté par l'utilisateur
- Etc.

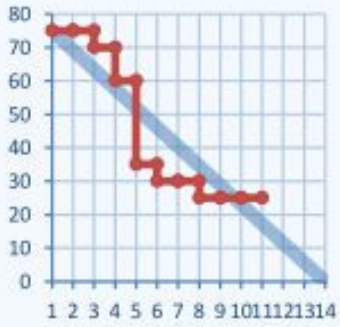
SPRINT BACKLOG

- US pour transformer le Product Backlog en fonctionnalités
- Contient la liste de toutes les fonctionnalités à réaliser dans le Sprint courant
- Crée lors du Sprint Planning Meeting
- Les tâches ne sont pas assignées
- Le Sprint Backlog est mis à jour quotidiennement

SPRINT BACKLOG

Sprint Goal	To Do	Doing	Done
The goal of this sprint is To make the purchasing part of the website mature enough to be able to handle the whole process and users can experience a full purchasing process, through which other functionalities of the website will be more meaningful.			Item #1 t.1.6 t.1.1 t.1.3 t.1.5 t.1.2
	Item #2 t.2.7	t.2.6 t.2.5	t.2.1 t.2.3 t.2.2 t.2.4
	Item #3 t.3.4 t.3.5 t.3.3 t.3.2	t.3.1	
	Item #4 t.4.4 t.4.2 t.4.1 t.4.5 t.4.3		
	Item #5 t.5.4 t.5.1 t.5 t.5.5 t.5.2		

Sprint Burn-Down Chart

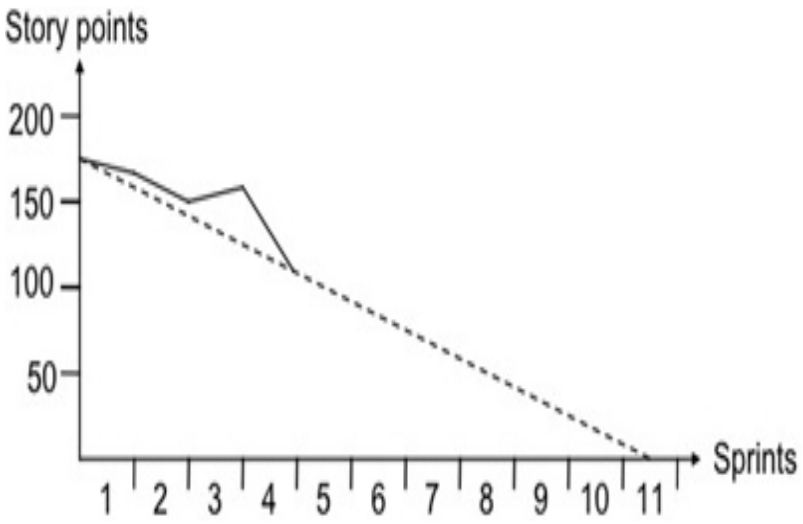


IMPEDIMENT BACKLOG

- Liste des blocages
- Le Scrum Master l'utilise pour visualiser les blocages qui impactent sur la productivité de l'équipe.
- Il reflète les éléments qui doivent être débloqués le plus rapidement possible.

BURNDOWN CHART

Example de Burndown Chart



Le Release burndown chart rend les tendances des progrès visibles.

Le rapport est basé sur les informations suivantes:

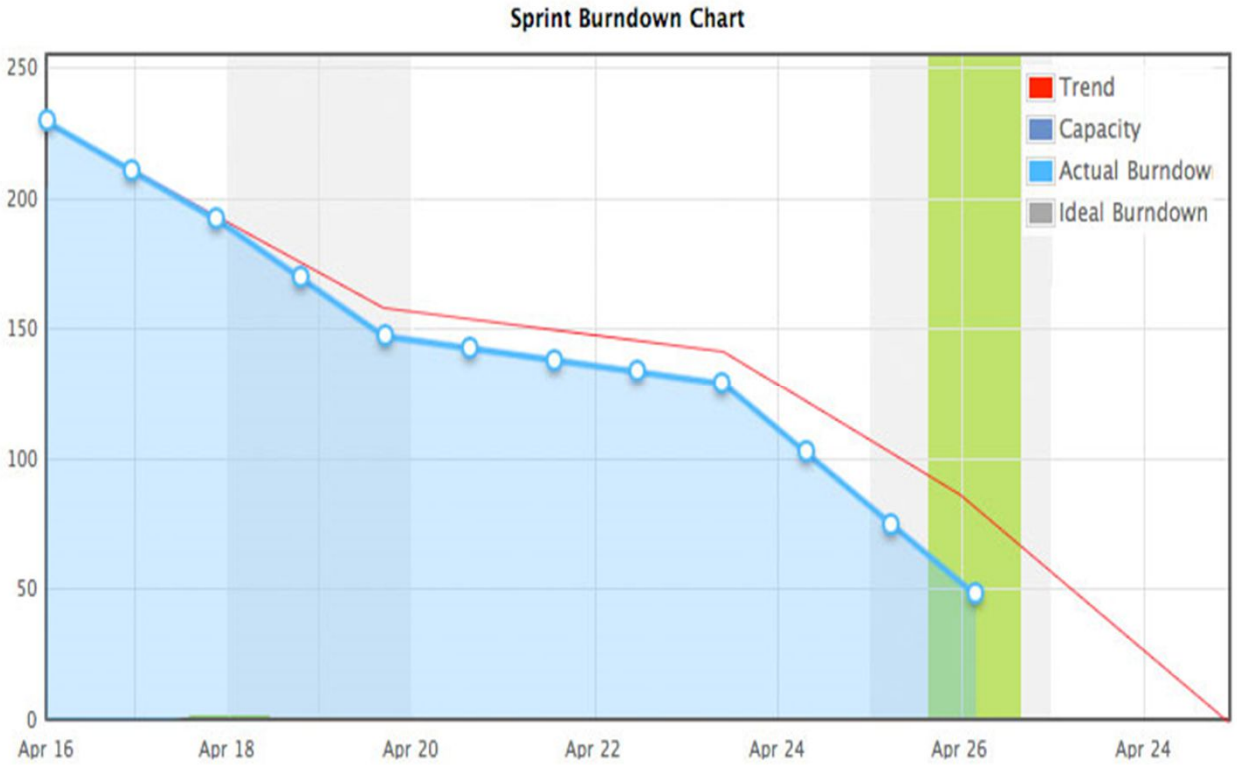
- *Le reste à faire du Product Backlog pour transformer la vision en un produit gagnant.*
- *Le nombre de Sprints nécessaires ou restants.*
- *La vélocité.*

Le Release burndown regarde le passé pour comprendre ce que l'avenir est susceptible de détenir. Nous déterminons le taux d'avancement des sprints passés.

SPRINT BURN-DOWN CHART

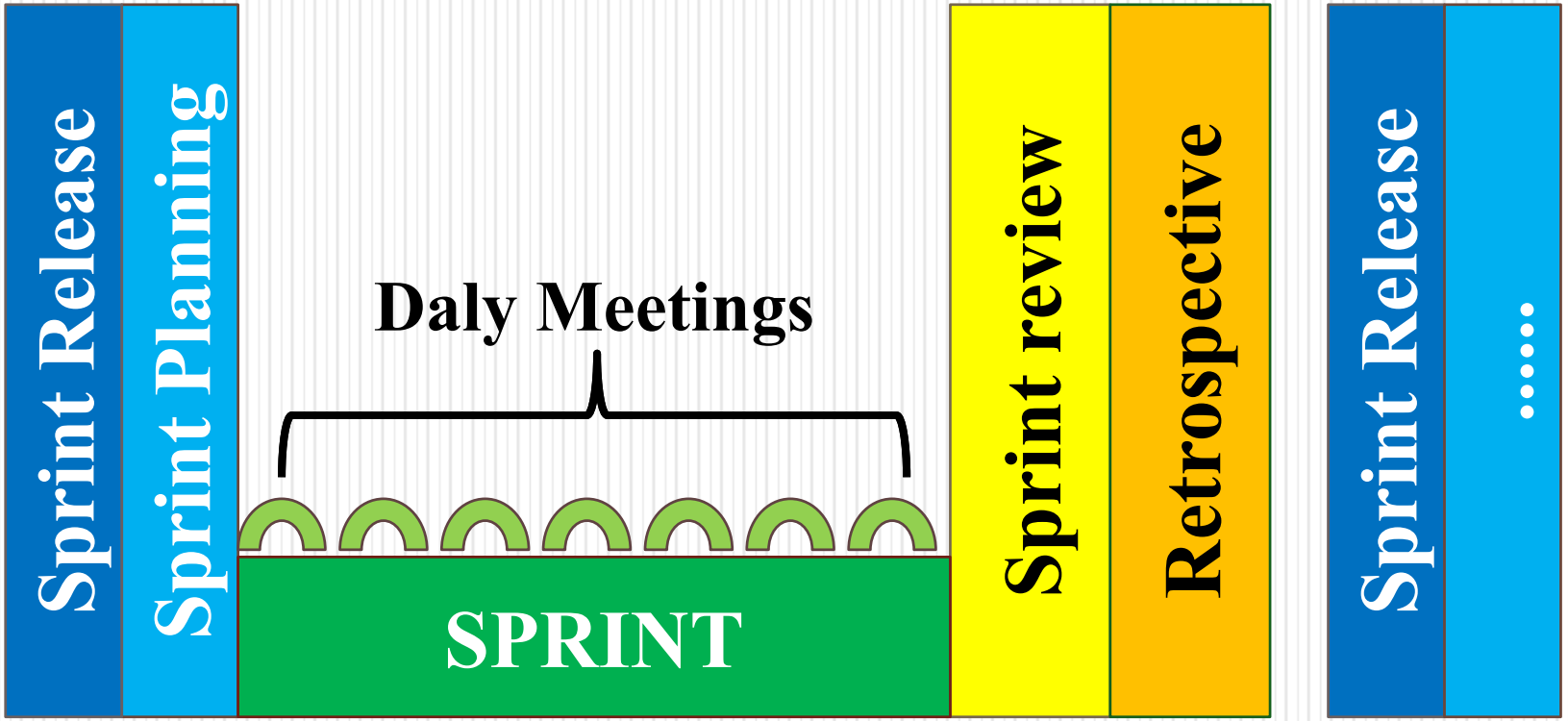
Le Sprint Burn-down chart montre :

- Combien d'efforts a été déployé en travaillant sur les tâches contenues dans le Sprint Backlog
- Et compare cela à la dépense idéale



- **Le tableau donne une tendance qui indique si l'équipe est susceptible de respecter son engagement**

MEETINGS


Sprint Release
Sprint Planning
Daly Meetings
SPRINT
Sprint review
Retrospective
Sprint Release
.....

SPRINT RELEASE MEETING

- **LE QUOI**

Sprint Release Meeting

Objectif

- Definir le quoi

Participants

- Le SDT (astimer l'effort PBI),
- Le SM (présider)
- PO (prioriser)

Durée approximative

- 2 H pour un Sprint de 2 semaines

Objectif

Présenter en détail ce que le PO souhaite faire construire durant le prochain sprint. Il permettre à l'équipe de développement d'avoir une idée claire de ce qui est attendu durant ce sprint.
A la fin du cérémonial, l'équipe sera en mesure de dire ce qu'elle peut accomplir durant les prochaines semaines compte tenu de sa vélocité.

En sortie :

✓ Un backlog de sprint estimé (en Story Point)
✓ Des US INVEST

Ne pas faire :

- Le PO n'estime pas

Remarque : Aussi appelé Backlog raffinement meeting ou Sprint planning meeting 1

COMPLEXITE DES US

- Un effort doit être affecté à chaque PBI (T-shirt sizing, planning pocker (***Estimation grossière***), etc.)
- Une Story Point (SP) est attribuée à chaque User Story
- Permet de mesurer la vélocité de l'équipe
- Permet de bien estimer les Sprints

COMPLEXITE DES US

Estimation de charge selon le planning pocker :

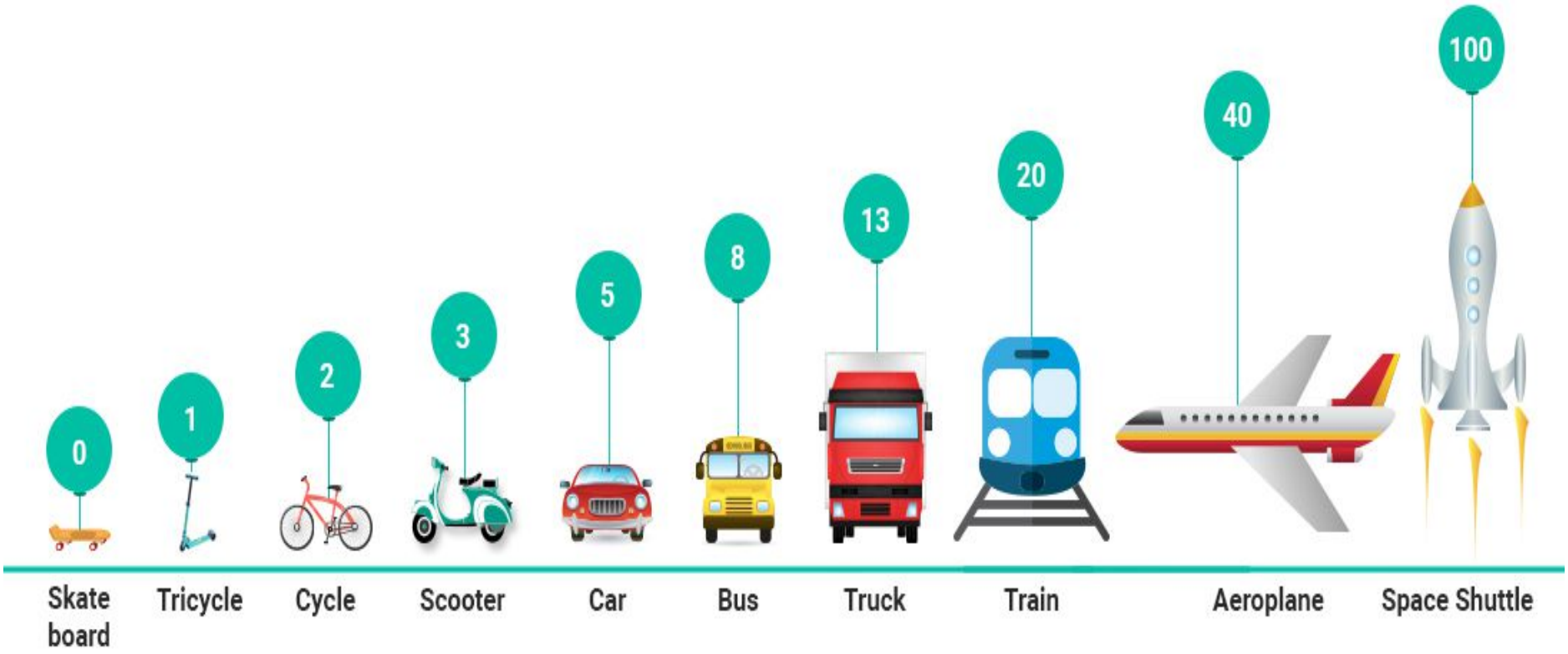
- Une valeur relative
- Les points n'ont pas d'unité
  - ✓ Suite de fibonacci
    - ➢ **Trivial, simple, moyen, difficile, complexe**
  - ✓ Taille de T-shirt
  - ✓ Etc.

$$u_0 = 1\ ,\ u_1 = 1\ ,\ u_n = u_{n-1} + u_{n-2}\ \forall n \in N,\ n > 1$$

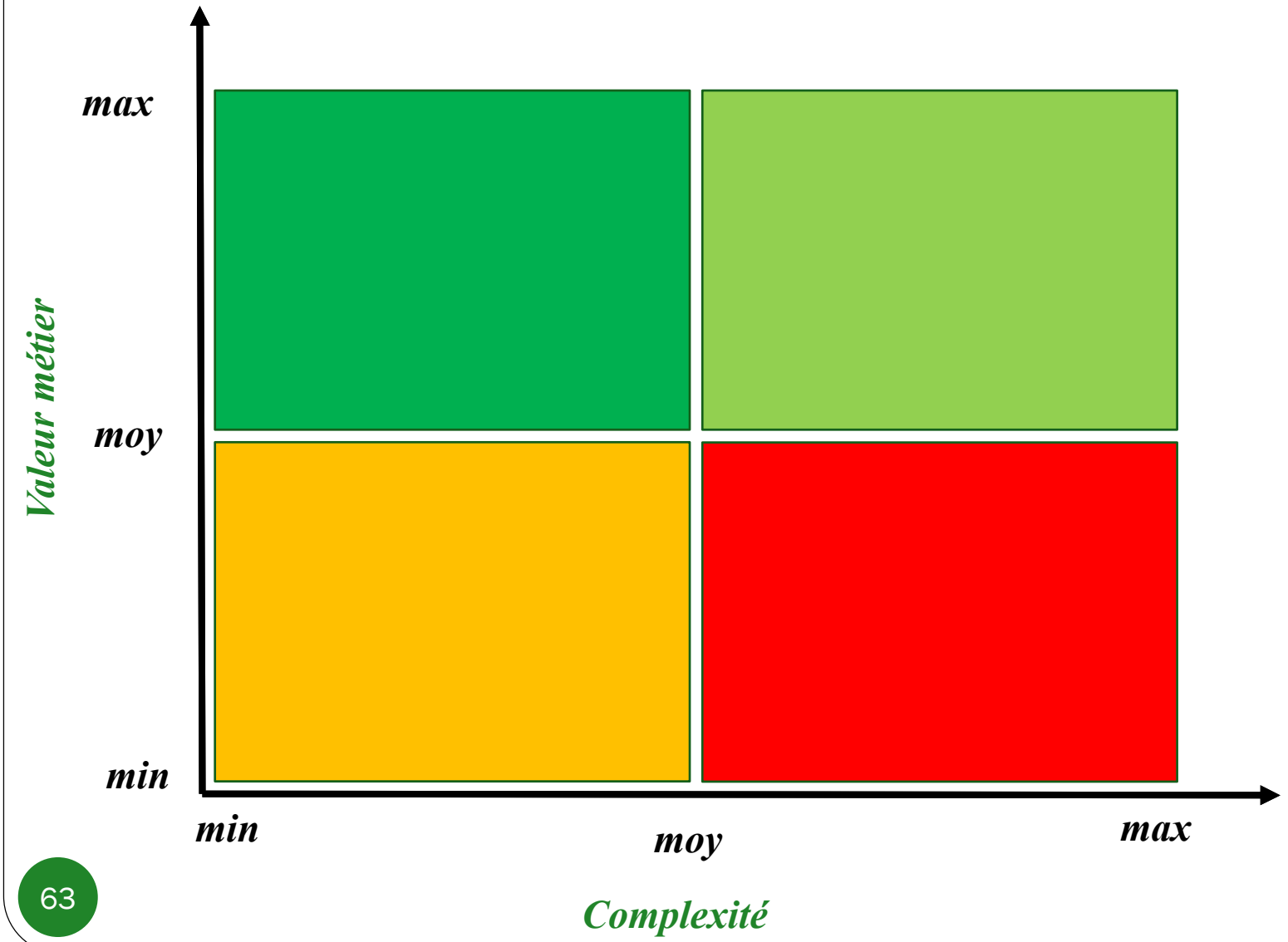
Seule la SDT estime

COMPLEXITE DES US

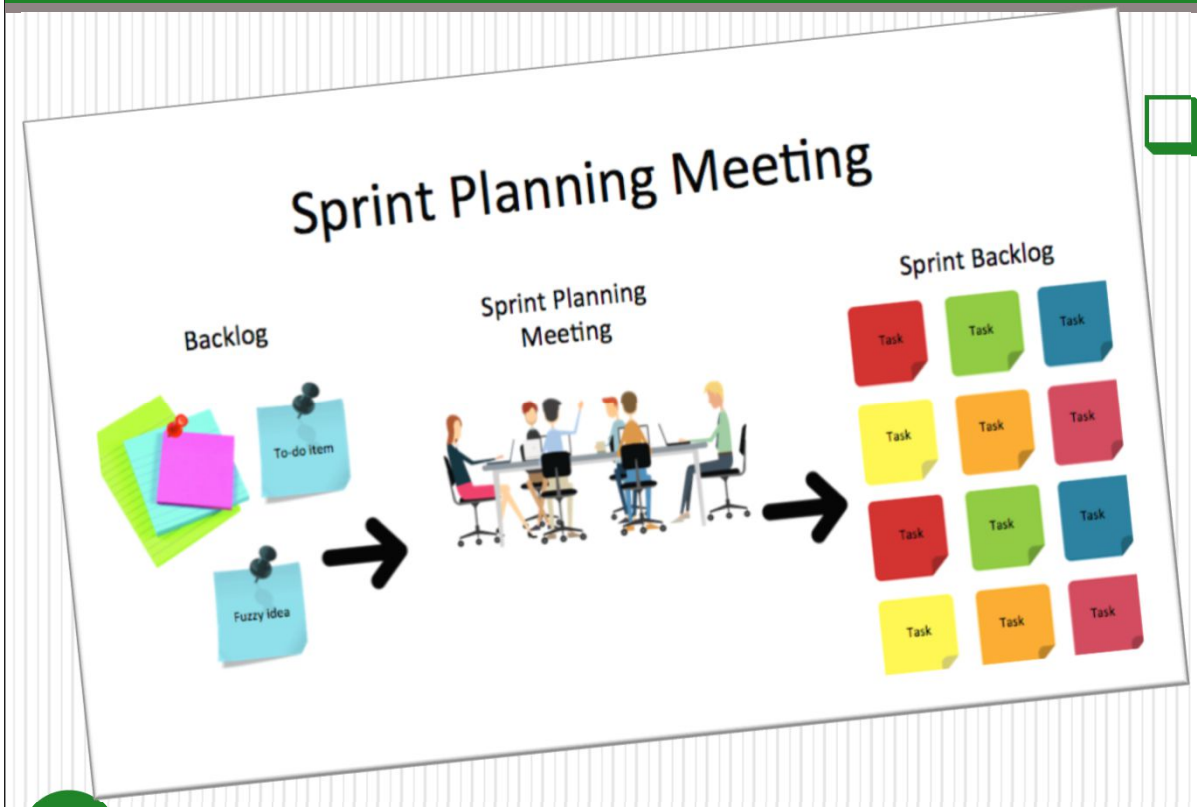
Une idée de comment s'y prendre



PRIORISATION DES US



SPRINT PLANNING MEETING



- LE COMMENT

SPRINT PLANNING MEETING

Objectif:

Savoir comment implémenter les US. A la fin de ce cérémonial, l'équipe doit savoir comment développer les US. La présence du PO n'est plus nécessaire.

Base :

Seuls les membres de l'équipe de réalisation conçoivent la solution. Les architectes et les autres personnes hors de l'équipe de développement sont seulement invités à aider l'équipe. L'équipe décide des personnes nécessaires durant ce cérémonial

En sortie :

- Des US INVEST
- Un tableau des taches

Durée:

- 60 min par semaine de sprint.

Ne pas faire :

- Ne pas assigner les tâches
- Ne pas estimer les tâches

SPRINT PLANNING MEETING

Procédure :

- Prendre le Backlog du sprint
- Faites confirmer par l'équipe qu'elle comprend bien le contenu
- Lancer la session de conception avec des questions telles que :
  - Quelles interfaces devons-nous développer ?
  - Quelle architecture avons-nous besoin de construire ?
  - Quelles tables devons-nous mettre à jour ?
  - Quels composants devons-nous mettre à jour ou écrire ?

Quand l'équipe comprends comment elle va développer cette US , elle peut prendre la US suivante. Tout au long de ce cérémonial, les membres de l'équipe utilisent les post-its pour écrire les tâches de bases. Les tâches associées aux US sont positionnées sur le tableau des tâches. **Il ne faut pas estimer les tâches.**

DEFINIR CORRECTEMENT SES TACHES

Une tâche correcte doit être SMART

S PECFIQUE

MESURABLE

A TTEIGNABLE

R EALISTE (RAISONNABLE)

T EMPOREL

DEFINITION DE TERMINE (DONE)

- Une Story/Item est “done” lorsque l’équipe a atteint son Level Done
- Le Sprint/Iteration est “done” lorsque
  - Tous les items sont “done”
  - Le sprint atteint son objectif
  - les critères d’acceptation sont adressés.
- La Release est “done”
    - “done” pour l’intégration
    - “done” pour la production

DAILY SCRUM MEETING

- **Synchronisation**
- **Engagement sur les tâches quotidiennes**

DAILY MEETING

Participants

- l'équipe (actif),
- le ScrumMaster (passif),
- Product Owner (passif)

Durée

- 15 min

C'est l'***inspect-and-adapt*** de l'équipe: synchronisation et engagement

Les 3 questions:

- *Qu'est-ce que j'ai fait hier?*
- *Qu'est-ce qui me bloque ?*
- *Qu'est-ce que j'ai prévu aujourd'hui?*

Moment pour mettre à jour le sprint burn-down chart

SPRINT REVIEW MEETING

- Analyse des résultats

SPRINT REVIEW MEETING

L'équipe démontre le résultat de son travail à l'équipe client et au PO. Elle peut inviter toutes les personnes qui sont intéressées pour découvrir ce qui a été réalisé durant le dernier sprint.

Ne pas faire :

- Ne pas présenter un produit qui n'est pas potentiellement utilisable
- Le Scrum Master ne fait pas la présentation
- L'équipe ne restreint pas la présentation au seul Product Owner
- Etc.

SPRINT REVIEW MEETING

Procédure :

- Le Scrum Master accueille les participants à la revue de sprint.
- Le Scrum Master rappelle l'objectif du sprint.
- L'équipe de réalisation fait une démonstration des nouvelles fonctionnalités sous la forme de scénario de bout en bout.
- Elle laisse par la suite au PO et son équipe utiliser ces fonctionnalités.
- Le Product Owner prend des notes concernant le retour de l'équipe sur ce sprint, pour proposer des éventuelles évolutions lors des prochains sprints

SRINT RETROSPECTIVE MEETING

- Amelioration continue

RETROSPECTIVE

Participants:

- l'équipe (actif),
- le SM(actif),
- le PO (actif en sa qualité de membre de l'équipe)

Durée:

- 3 heures pour un Sprint de 4 semaines

Analyse du Process Scrum:

- *Comment cela c'est passé pendant le Sprint*
- *Comment s'améliorer*

Points principaux à vérifier:

- *La communication dans l'équipe*
- *Les relations entre les membres*
- *Les besoins en formation*
- *Etc.*

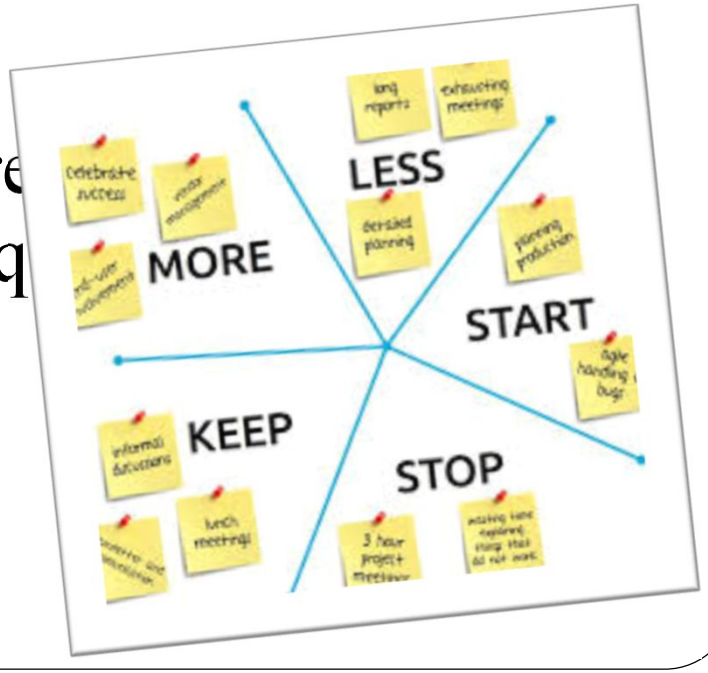
RETROSPECTIVE

Nous faisons un point après l'action en nous posant deux questions:

- Que devons-nous améliorer?
- Qu'est-ce qui a bien fonctionné?
- Etc.

Objectifs:

- Apprendre du passé pour prépare
- Améliorer la productivité de l'éq



CODE REVIEW

- D’aucun pense qu’elle n’est pas necessaire dans une demarche agile
- **Ken Schwaber** et **Jeff Sutherland** (cofondateurs de Scrum) la considèrent comme indispensable.
- Selon eux, le critère "done“ ne peut être attaint que si le code est revu.
- Le but est d’éviter les bugs
- Il est necessaire de faire la revue de code avec une frequence precise (après le sprint, une fois par semaine, etc.)
- De manière croisée ou par pair programming(pour des fonctionnalités complexes)
- Pas d’outils pour faire la revue de code, il faut regarder le code en vue de l’ameliorer
- Celui qui revoie le code n’est pas nécessairement plus experimenté que celui qui l’a ecrit

AVANTAGES

Les avantages clés des méthodes Agile sont les suivants :

- Satisfaction du client mise en avant, meilleur ROI
- Mise à jour des priorités
- Plus grande autonomie des développeurs;
- Meilleure collaboration entre les membres de l'équipe;
- Meilleure vue d'ensemble du projet
- Amélioration continue
- Minimisation des risques
- Eliminer la documentation inutile
- Indiqué pour l'imprévu
- Eviter les efforts croissants du waterfall
- Etc.

PROBLÈME FRÉQUENTS

- L'agilité préconise que tout le monde puisse se saisir d'une tâche quelconque du backlog. Mais, en réalité, d'obscures technologies ne sont comprises que par certaines personnes
- Le Scrum master est censé déplacer des montagnes pour aider l'équipe à avancer sans être dérangée. En réalité, il manque de pouvoir pour faire cela. Ce pouvoir est généralement entre les mains de managers. Il est addictif et il est difficile de s'en dessaisir.
- Scrum master agissant comme un chef de projet
- Client n’est pas prêt pour travailler avec l’équipe en temps réel
- Débordement probable des réunions
- Les erreurs sont visibles par tout le monde
- Etc.

QUESTIONS

Bibliographie et webographie

- Abdel Lahidel, Intiation au management de projet
- http://www.aubryconseil.com
- http://igm.univ-mlv.fr
- https://www.slideshare.net/PierreNeis/scrum-pmbok-pmi
- www.agilemanifesto.org
- Mustapha BOUBEKRI, Méthodes agiles Scrum, Tunis, 2016
- Arnaud Pasquiers et Chris Ozanne, Decouvrir l'agilité
- Maria Belkina, Jennifer Schiller, Maxim Masunov, Vycheslav Filippov, April 2006,SCRUM – Agile Project Management

A faire

On se propose de mettre en place un système d'annonce et d'abonnement entre professeurs et élèves dans le cadre d'un service de cours particuliers. Proposer un backlog du produit pour ce système.

https://www.youtube.com/watch?v=UJ-NnDficnE