Méthodes Agiles

Objectifs

- Introduire les participants aux ***méthodologies*** de développement dites "Agiles" et plus particulièrement à **SCRUM**

AGILE = Méthodologie

AGILE >< Langage

Tables des matières

Phase initiale

...ou l'on compare les méthodes Agile aux méthodes traditionnelles

Méthodes de développement de software

Qu'est ce qu'une méthode de développement?

> *Cadre établi afin de structurer, planifier et contrôler le processus de développement d'un système d'information*

Méthodes de développement de software

Quelles sont les disciplines d'un processus de développement?

Les disciplines du développement de software

- **Analyse des exigences**
  - Comprendre les besoins du client
- **Conception (Design)**
  - Définir la solution technique
- **Développement**
  - Implémenter la solution
- **Validation (Testing)**
  - S'assurer que la solution répond adéquatement aux besoins
- **Déploiement**
  - Intégration globale et mise en production
- **Maintenance**

Les activités du développement de software

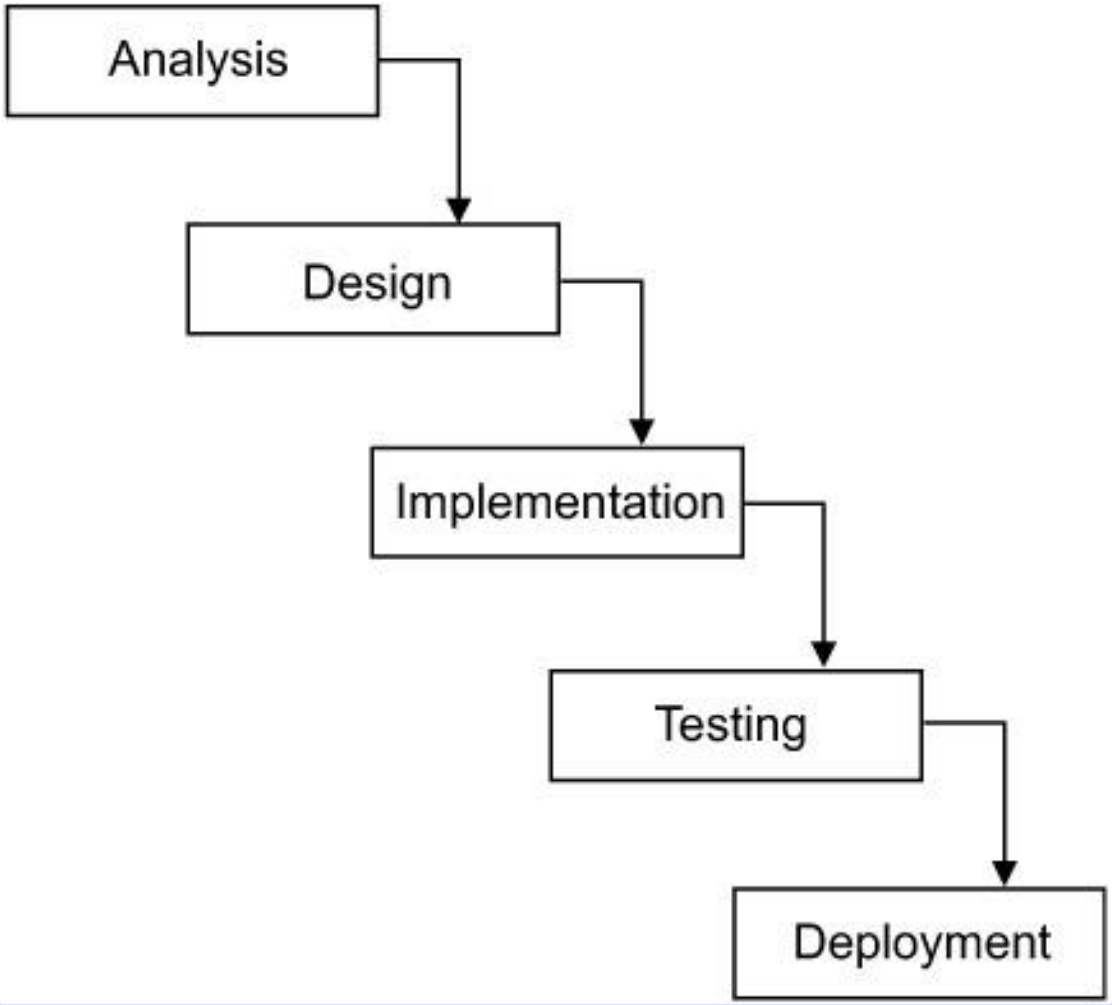
Les différences entre différentes méthodologies de développement résident essentiellement dans:

- l'importance donnée à chaque activité
- la séquence permise entre chaque activité

Large variété de méthodologie: en cascade, en spirale, incrémentale, agile...

Méthode en cascade

Méthode en cascade



Réflexion

Quels sont les **avantages** et les **inconvénients** d'une telle méthode?

Quels sont les critères favorable/défavorable pour cette méthode?

Avantages

1. Le temps alloué dans les premières phases des projets permet d'atteindre des **économies d'échelle**

2. La **documentation** est aussi importante que le code. Cela permet de pérenniser la connaissance au sein du projet

3. **Approche simple**, disciplinée et facile à comprendre

Inconvénients

1. La rigidité de l’approche
2. L’effet tunnel ou "black box"
3. Une mauvaise communication
4. La levée tardive des facteurs à risques
5. Une documentation pléthorique

Exigences pour la méthode en cascade

1. L'environnement et les exigences sont **stables**
2. La technologie est bien **connue** et **mature**
3. Il n'y a rien de nouveau ou d'**inconnu** dans le projet (**prévisibilité**)
4. (De nombreux projets semblables ont déjà été exécutés avant)

Méthode en cascade

Qui utilise ce genre de méthode?

Méthode en cascade

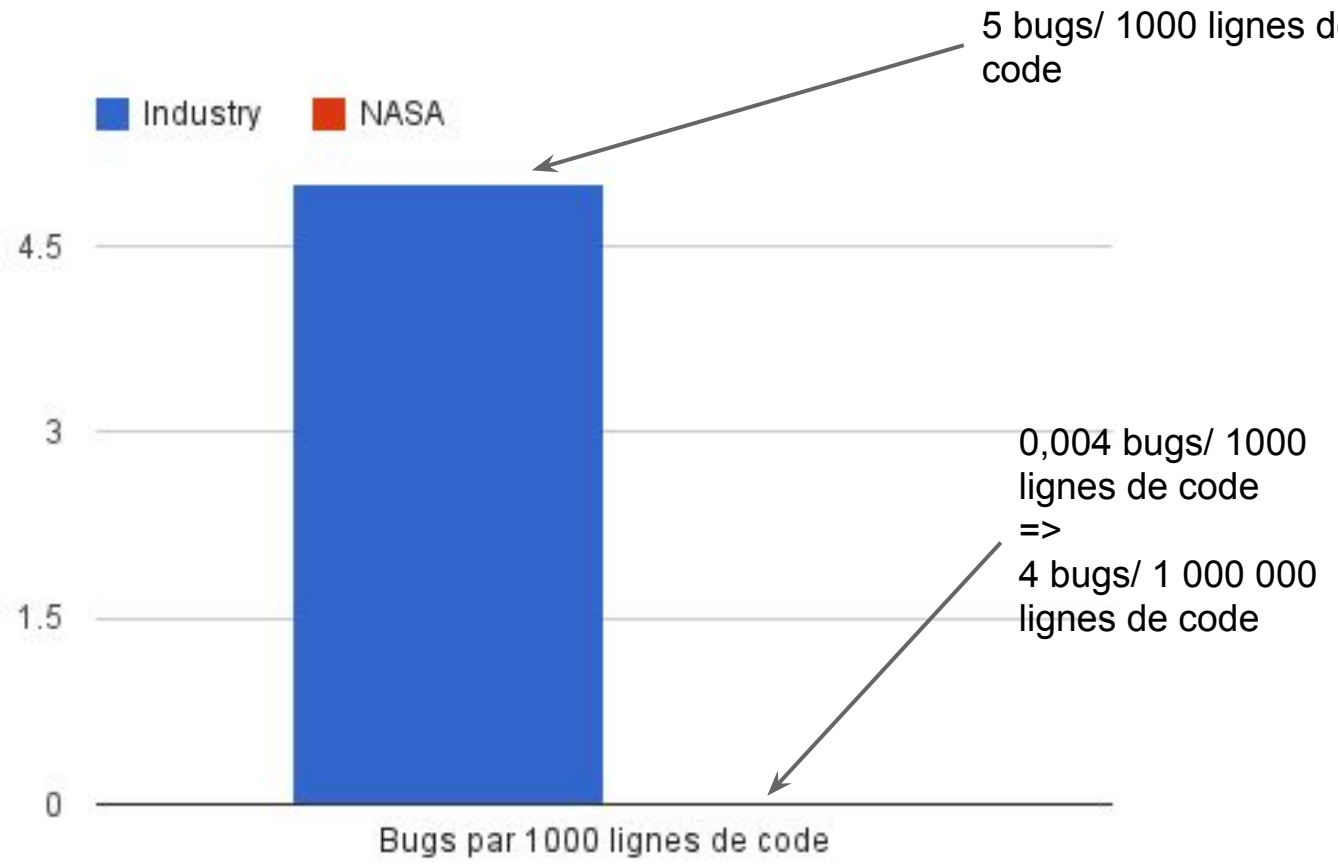
Qui utilise ce genre de méthode?

- Institutionnel
- Banque/Assurance
- Aéronautique
- ….

Méthode en cascade

Qui utilise ce genre de méthode?

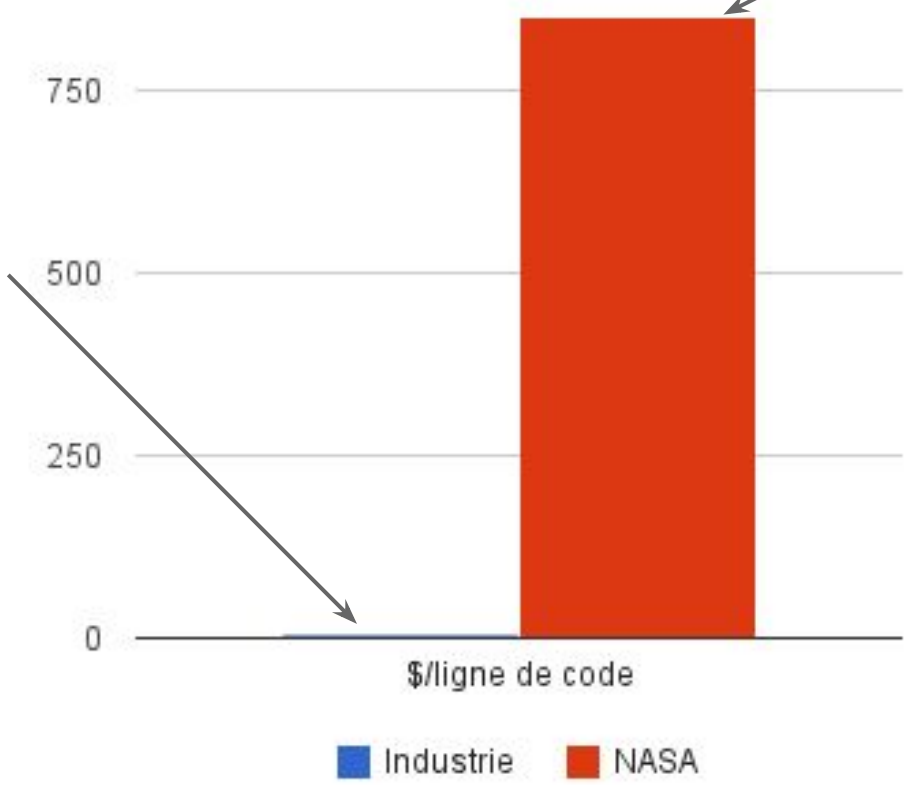
Statistiques NASA-Soft. Eng.



Statistiques NASA-Soft. Eng.

850$ / ligne de code

5$ / ligne de code



Réflexion

Comment pourrait-on modifier le processus waterfall afin de l'améliorer ?

Processus Incrémental

Sources: Jeff Patton

Processus Itératif

Sources: Jeff Patton

Rapid Application Development

Origines

Double constat:

- Manque de concertation entre informaticiens et utilisateurs
  => *inadéquation du produit aux besoins*

- Durée des méthodes classiques inadaptées à la vitesse d'évolution des technologies

Principes

- Fondement de base: **Communication**
- Répartition des rôles très structurée:
  - *Maîtrise d'ouvrage*: représente l'utilisateur et détermine les fonctionnalités à développer
  - *Maîtrise d'oeuvre*: apporte les solutions techniques aux problèmes posés par la maîtrise d'ouvrage
  - *Groupe d'animation* et *Rapporteur*: organise la communication du projet

Maîtrise d'ouvrage

- Responsabilités:
  - Définition des objectifs et des exigences
  - Valider les solutions proposées et élaborées
  - Préparer et piloter le changement induit

- Acteurs:
  - **Maître d'ouvrage**

    Fixe les objectifs
  - **Coordinateur de Projet Utilisateurs ou Maître d'Ouvrage délégué**

    Assure le suivi des objectifs
  - **Responsable de la cohérence et de la qualité fonctionnelle**

    Contrôle la cohérence des décisions

    dans les domaines fonctionnels

Maîtrise d'oeuvre

- Responsabilités:
  - Proposer et réaliser la solution
  - Livrer des "fonctionnalités"
  - Respecter les directives de qualité

- Acteurs:
  - Maître d'œuvre
  - Pilote de Projet Informatique
  - Responsable par domaine

Interactions RAD

Groupe d'animation et de Rapport RAD

Animateur
- Provoque et conduit les réunions
- N'émet pas d'avis personnel
- Recadre les discussions
- Synthétise et formalise

Rapporteur
- Produit une documentation automatisée
- Modélise le système en direct lors des réunions de conception
- Rédige les comptes-rendus

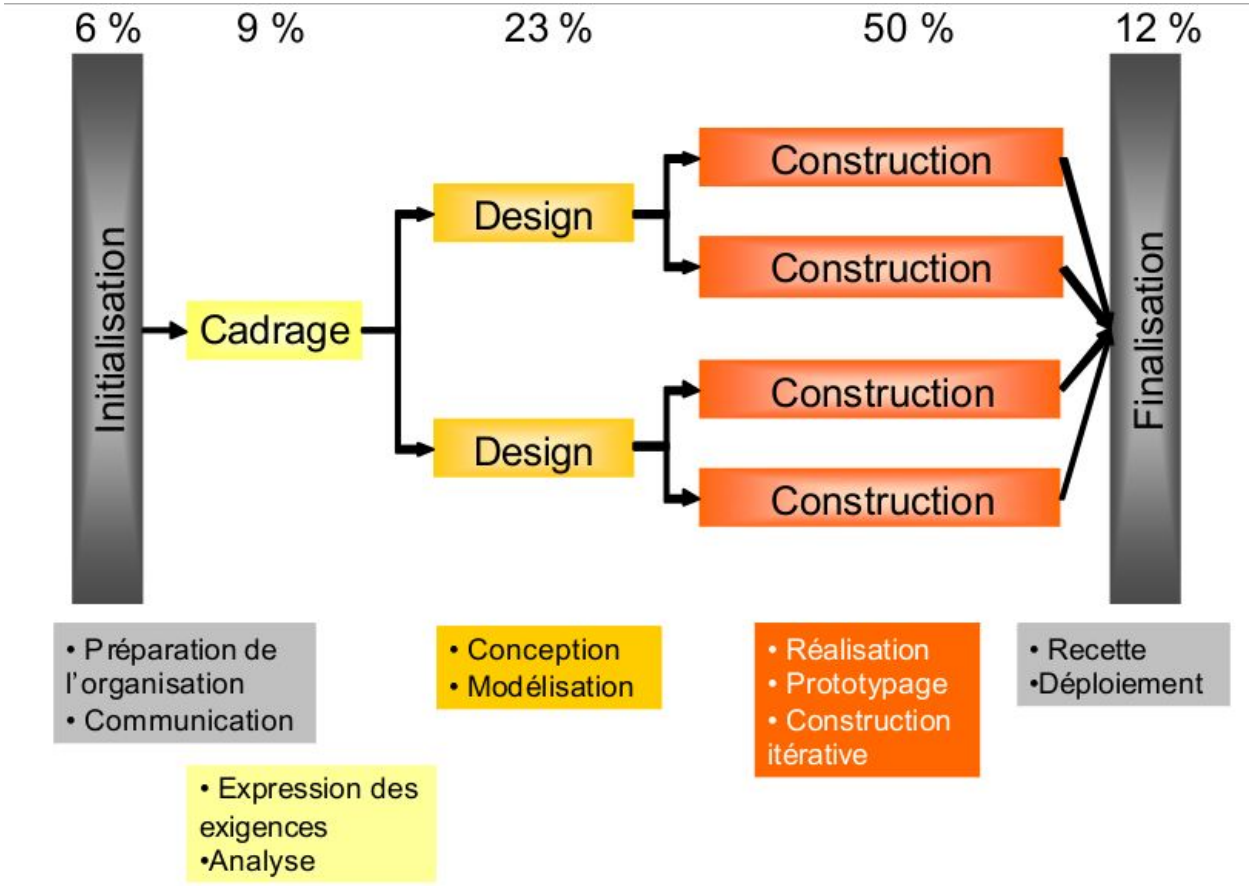
Maîtrise d'ouvrage
- Définit les objectifs et exigences du système
- Valide le solutions proposées et élaborées
- Prépare et pilote le changement

Maîtrise d'œuvre
- Propose et réalise la solution
- Livre des « fonctionnalités »
- respecte les directives du Plan d'Assurance Qualité

Les 5 phases d'un RAD

- Initialisation
  - *Préparation de l'organisation et communication*
- Cadrage
  - *Analyse et expression des exigences*
- Design
  - *Conception et modélisation*
- Construction
  - *Réalisation, prototypage*
- Finalisation
  - *Recette et déploiement*

Cycle de vie d'un projet RAD



Cycle de vie d'un projet RAD

Initialisation

Préparation de l'organisation et communication

- définit le périmètre général du projet
- établit la structure du travail par thèmes
- recense les acteurs pertinents
- amorce la dynamique du projet

5% du projet

Cadrage

Analyse et expression des exigences

- Spécification des exigences par les utilisateurs lors d'entretiens de groupe
- 2 à 5 jours de sessions par commission

10% du projet

Design

Conception et modélisation

- Validation des modèles organisationnels par les utilisateurs: flux, traitements, données
- Validation d'un premier prototype par les utilisateurs
- 4 à 8 jours de sessions sont prévus par commission

25% du projet

Construction

Réalisation, prototypage

- Construction au cours de plusieurs sessions itératives de l’application module par module
- Validation des prototypes par l'utilisateur

50% du projet

Finalisation

Livraison globale et le transfert du système en exploitation et maintenance.

10% du projet

Unified Process

Le Processus Unifié

- Processus de développement pour les systèmes orientés objets
- Importance de la **modélisation** et de l'utilisation d'**outils** de support (particulièrement UML)

Les 4 phases du UP

Chaque projet UP comprend les 4 phases suivantes:

- Inception
- Elaboration
- Construction
- Transition

Inception

- Vision approximative de la finalité du projet
- Etude d'opportunité
- Définition de périmètre
- Estimations globales

Elaboration

- Ebauche plus élaborée
- Implémentation de l'architecture du noyau
- Résolutions des risques élevés
- Identification de la plupart des besoins
- Estimations plus réalistes

Construction

- Implémentation des éléments à moindre risque
- Préparation du déploiement

Transition

- Bêta test
- Déploiement final

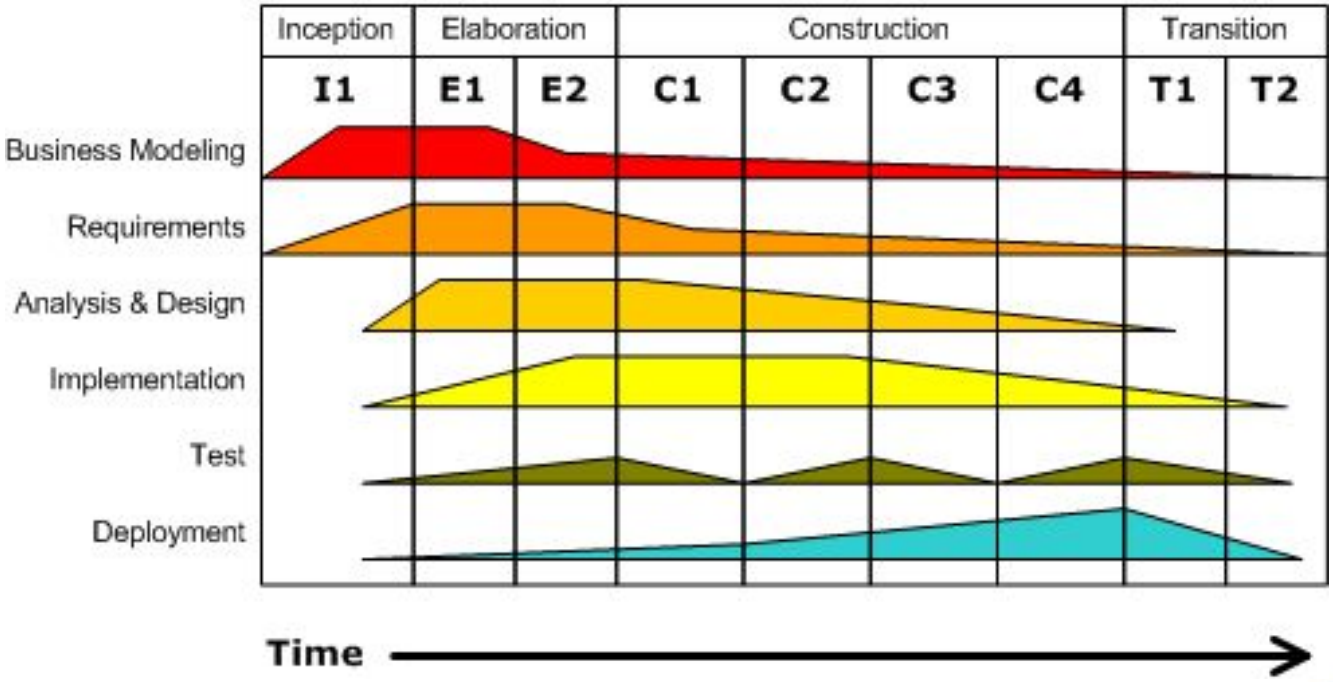
Les disciplines du UP

Les disciplines sont les ensembles d'activités dans un domaine donné

- Modélisation métier
- Exigences
- Conception
- Implémentation
- Validation
- Déploiement

Iterative Development

Business value is delivered incrementally in time-boxed cross-discipline iterations.



Réflexion

Quels sont les avantages de l'UP

Avantages du développement itératif

- La communication est de meilleure qualité
- La visibilité est meilleure
- La qualité est évaluée en continu
- Les risques sont détectés très tôt
- L’équipe prend confiance
- Les coûts sont contrôlés
- Possibilité d'exploiter méthodiquement les leçons tirées d'une itération

Méthode Agile

Le manifeste Agile

Valeurs et Principes

Valeurs

- Priorité aux **personnes** et aux **interactions** par rapport aux *procédures* et aux *outils*
  - Travail en groupe, communication

- Priorité aux **applications opérationnelles** par rapport à une *documentation pléthorique*
  - Documentations succinctes à jour, documentation permanente du code

Valeurs

- Priorité de la **collaboration avec le client** par rapport à la *négociation de contrat*
  - Feedback régulier du client, solution répondant réellement aux attentes
  - Grande maturité du client, relation de confiance

- Priorité de **l'acceptation du changement** par rapport à la *planification*
  - Planning flexible, modifications possibles après 1ère version du système

Principes

1. La plus grande priorité est de **satisfaire le client** en lui livrant très tôt et régulièrement des versions fonctionnelles de l'application source de valeur
   - *Le client peut décider de la mise en production de l'application*

2. Accueillir les **demandes de changement** à bras ouverts, même tard dans le processus de développement. Les méthodologies agiles exploitent les changements pour apporter au client un avantage concurrentiel
   - *Produire des systèmes flexibles*

Principes

3. Livrer le plus souvent possible des versions opérationnelles de l'application, avec une fréquence comprise entre **deux semaines et deux mois**, avec une préférence pour l'échelle de temps la plus courte
   - *Objectif : livrer une application qui satisfasse aux besoins du client*
4. Clients et développeurs doivent **coopérer quotidiennement** tout au long du projet
5. Construire des projets autour **d'individus motivés**. Leur donner l'environnement et le support dont ils ont besoin et leur faire confiance pour remplir leur mission

Principes

6. La méthode la plus efficace pour communiquer des informations à une équipe et à l'intérieur de celle-ci reste la **conversation en face à face**
7. Le **fonctionnement** de l'application est le premier indicateur d'avancement du projet
8. Les méthodes agiles recommandent que le projet avance à un **rythme soutenable** : développeurs et utilisateurs devraient pouvoir maintenir un rythme constant indéfiniment
   - *Adapter le rythme pour préserver la qualité du travail sur la durée du projet*

Principes

9. Porter une attention continue à **l'excellence technique** et à la conception améliore l'agilité
   - *Maintenir le code source propre, clair et robuste*
10. La **simplicité**, art de maximiser la quantité de travail à ne pas faire, est essentielle
    - *Répondre le + simplement aux besoins actuels pour que celui ci soit adaptable*
11. Les meilleures architectures, spécifications et conceptions sont le fruit d'équipes qui **s'auto-organisent**
    - *Partage des responsabilités par volontariat*

Principes

12. A intervalles de temps réguliers, l'ensemble de l'équipe **s'interroge** sur la manière de devenir encore plus efficace, puis ajuste son comportement en conséquence
    - *Environnement en perpétuelle évolution*

Comparaison

Cycle de vie	Phases séquentielles	Agiles
Planification	Prédictive	Adaptative
Documentation	Produite en quantité	Réduite au strict nécessaire
Équipe	Ressources spécialisées	Responsabilisation, initiative et communication
Qualité	Contrôle à la fin du cycle	Contrôle qualité précoce et permanent
Changement	Opposition au changement.	Intégré dans le processus
Suivi de l'avancement	Mesure de la conformité aux plans initiaux	Travail restant à faire
Gestion des risques	Processus distinct	Intégré dans le processus
Mesure du succès	Respect des engagements initiaux	Satisfaction client

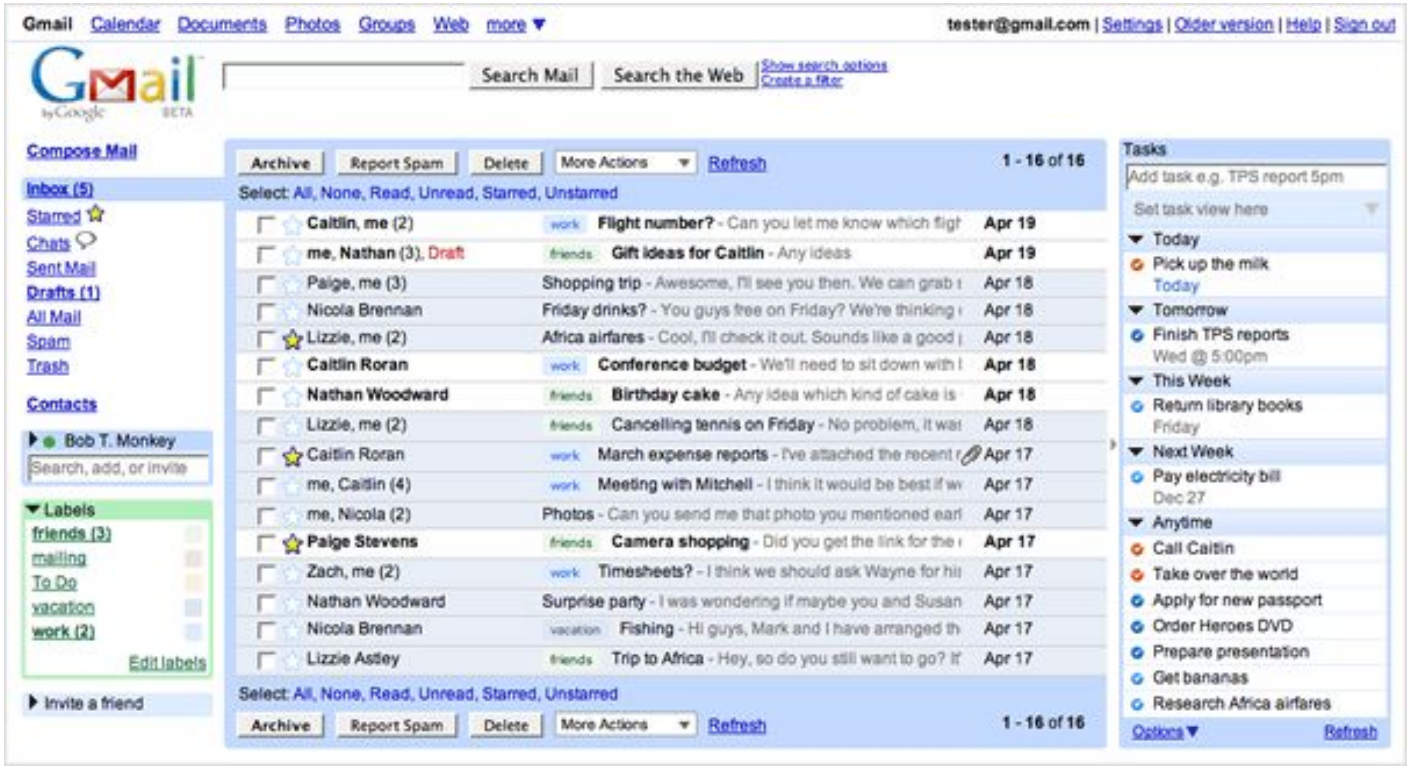
Exercice

Par groupe de 2, déterminez quelle serait la meilleure méthode entre waterfall et agile.

Justifiez votre réponse.

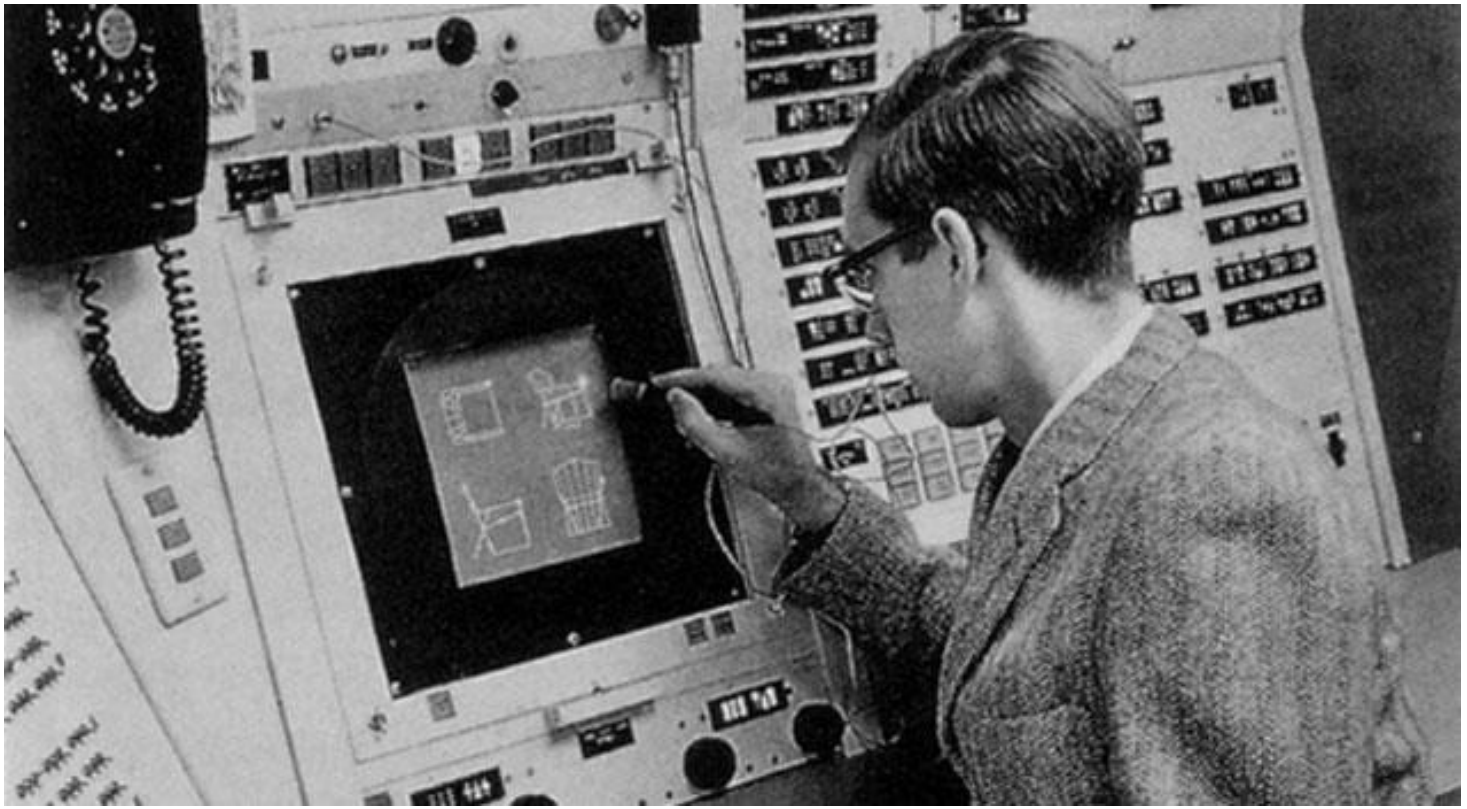
Système de navigation A380

Gmail


Gmail Calendar Documents Photos Groups Web more ▼
tester@gmail.com | Settings | Older version | Help | Sign out
Search Mail
Search the Web
Show search options
Create a filter
Compose Mail
Inbox (5)
Starred
Chats
Sent Mail
Drafts (1)
All Mail
Spam
Trash
Contacts
Bob T. Monkey
Search, add, or invite
Labels
friends (3)
mailing
To Do
vacation
work (2)
Edit labels
Invite a friend
Archive
Report Spam
Delete
More Actions
Refresh
1 - 16 of 16
Select: All, None, Read, Unread, Starred, Unstarred
Caitlin, me (2) work Flight number? - Can you let me know which flig Apr 19
me, Nathan (3), Draft friends Gift ideas for Caitlin - Any ideas Apr 19
Paige, me (3) Shopping trip - Awesome, I'll see you then. We can grab Apr 18
Nicola Brennan Friday drinks? - You guys free on Friday? We're thinking Apr 18
Lizzie, me (2) Africa airfares - Cool, I'll check it out. Sounds like a good Apr 18
Caitlin Roran work Conference budget - We'll need to sit down with Apr 18
Nathan Woodward friends Birthday cake - Any idea which kind of cake is Apr 18
Lizzie, me (2) friends Cancelling tennis on Friday - No problem, it wa Apr 18
Caitlin Roran work March expense reports - I've attached the recent r Apr 17
me, Caitlin (4) work Meeting with Mitchell - I think it would be best if w Apr 17
me, Nicola (2) Photos - Can you send me that photo you mentioned earl Apr 17
Paige Stevens friends Camera shopping - Did you get the link for the Apr 17
Zach, me (2) work Timesheets? - I think we should ask Wayne for hi Apr 17
Nathan Woodward Surprise party - I was wondering if maybe you and Susan Apr 17
Nicola Brennan vacation Fishing - Hi guys, Mark and I have arranged th Apr 17
Lizzie Astley friends Trip to Africa - Hey, so do you still want to go? It Apr 17
Tasks
Add task e.g. TPS report 5pm
Set task view here
Today
Pick up the milk
Today
Tomorrow
Finish TPS reports
Wed @ 5:00pm
This Week
Return library books
Friday
Next Week
Pay electricity bill
Dec 27
Anytime
Call Caitlin
Take over the world
Apply for new passport
Order Heroes DVD
Prepare presentation
Get bananas
Research Africa airfares
Options ▼
Refresh

Système de gestion intégrée du stock

ARPANET (Ancêtre d'Internet)



Facebook


www.facebook.com/zuck?sk=wall
facebook
Search
Mark Zuckerberg
Works at Facebook Studied Computer Science at Harvard University Lives in Palo Alto, California Knows English, Mandarin Chinese From Dobbs Ferry, New York Born on May 14, 1984
Wall
RECENT ACTIVITY
"I like dangerous thoughts." on Samuel W. Lessin's status.
Wall
Info
Share Profile
Report/Block This Person
Mark Zuckerberg
Steve, you've done so much good for the world already. I hope you get better soon.
January 17 at 11:43am via iPhone
150 people like this.

Scanner médical

Phase de Sprint

6 itérations pour se plonger dans SCRUM

Itération 1

... où l'on survole SCRUM

Introduction

Lecture d'un texte:

- Identifier des éléments, événements et concepts de SCRUM et mettre ces derniers en relation avec les valeurs agiles

Les origines

SCRUM a été formalisé en 1995 par:

- Jeff Sutherland
  - http://jeffsutherland.com/

- Ken Schwaber
  - http://kenschwaber.wordpress.com/

SCRUM en quelques mots

Scrum:

- A framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value [*Scrum Guide*]

Scrum est:

- Léger (*Lightweight*)
- Facile à comprendre (*Simple to understand*)
- Difficile à maîtriser (*Extremely difficult to master*)

SCRUM en quelques mots

- Scrum est un **processus agile** qui produit toutes les 4 semaines (= 1 **SPRINT**) du logiciel qui fonctionne
- Les exigences et les priorités sont définies dans le **Product Backlog**
- L'**équipe Scrum** s'auto-organise
- A chaque fin de Sprint, la décision de **livrer le produit** dans l'état ou de continuer à l'améliorer est prise

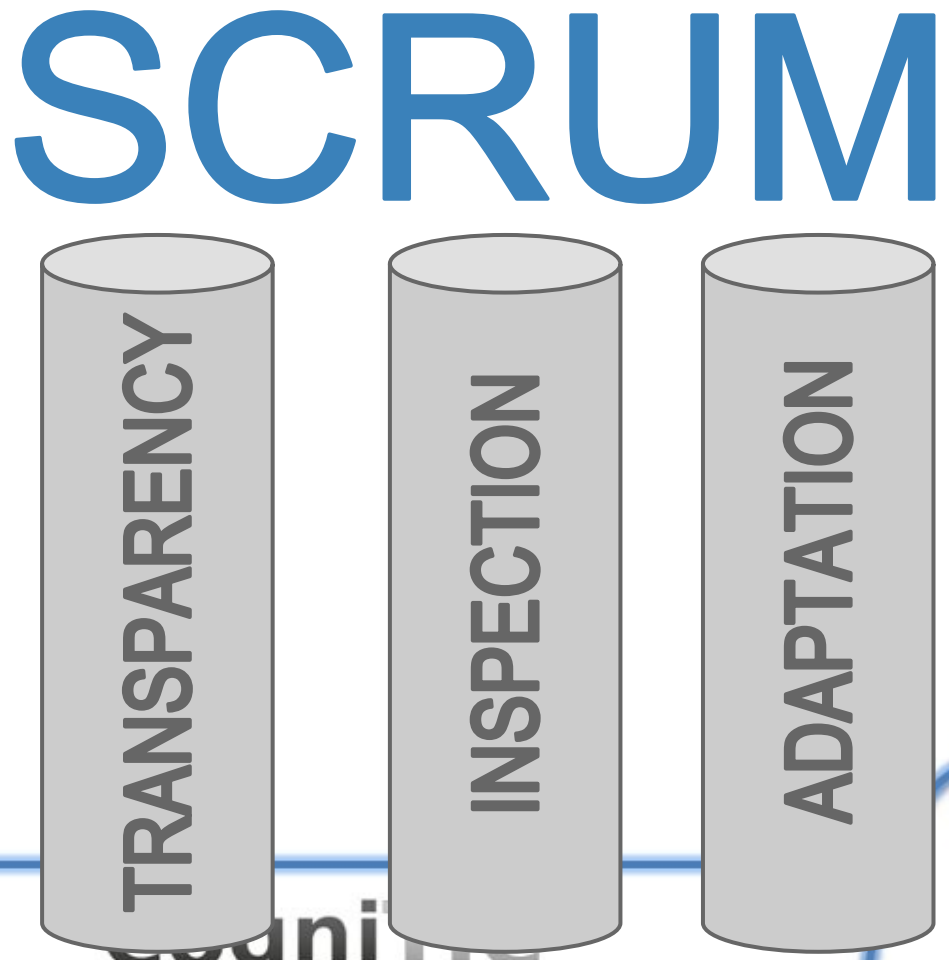
SCRUM en quelques mots

Un framework **Scrum** est constitué:

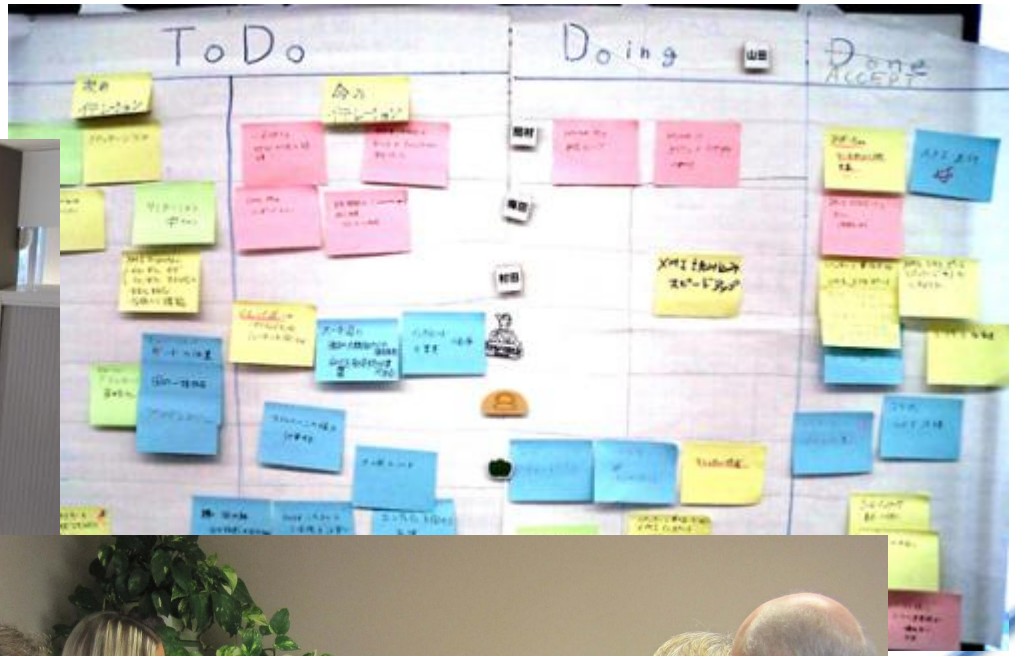
- d'une équipe SCRUM
- de rôles
- d'événements (avec Time-box)
- d'artefacts

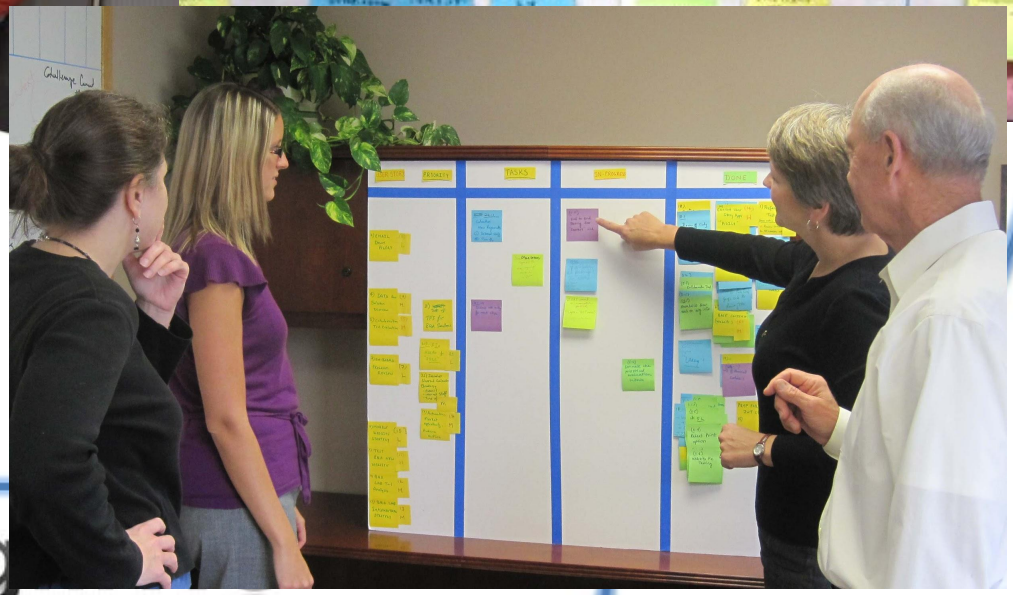
SCRUM en quelques mots

Les 3 piliers de **Scrum** sont:

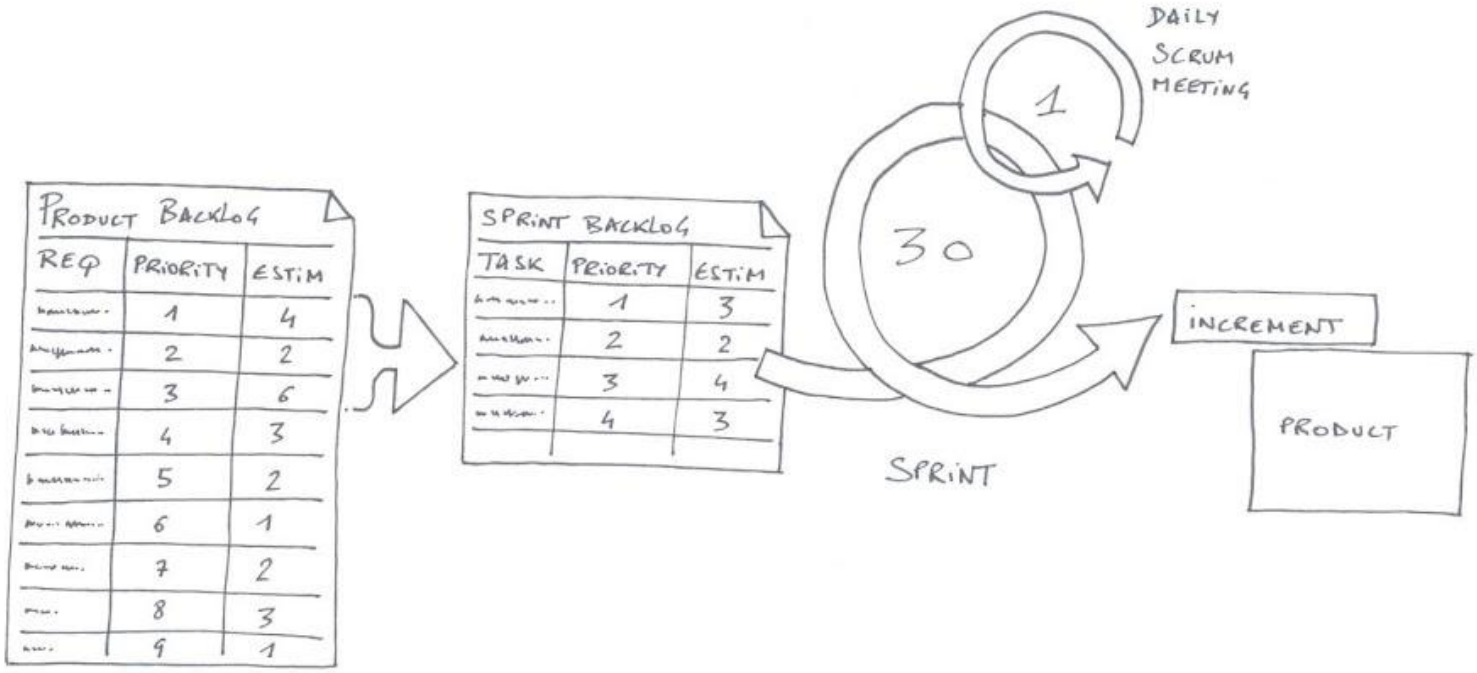


SCRUM en images





SCRUM en images



Le Processus

Processus SCRUM

- Un processus SCRUM consiste en 3 phases:
  - La phase Initiale
  - La phase de Sprints
  - La phase de Clôture

Phase Initiale

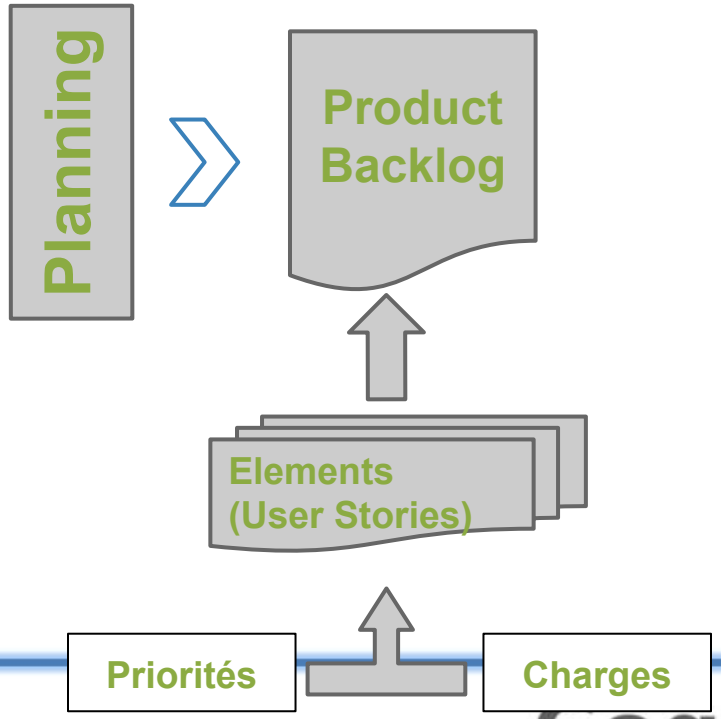
La phase initiale aboutit à:

- La conceptualisation et l'analyse du système
- La mise en place d'un "**Product Backlog**", c'est à dire une liste de tâches restant à effectuer (=La définition des **fonctionnalités** à livrer)
- La définition approximative de la **date de livraison**
- La formation et la constitution de l'**équipe de développement**
- L'analyse du **risque** et des **coûts**

Phase **courte** mais qui requiert des compétences variées: connaissance du marché, des utilisateurs potentiels, à des ressources techniques issues d'autres développement similaires...

Processus

Phase Initiale



Phase de sprint

- Phase où le développement est, à proprement parlé, réalisé (analyse, conception, implémentation... de chaque élément)
- Les sprints sont guidés par une liste de tâches provenant du Product Backlog formant le **Sprint Backlog**
- Durée de +/- 30 jours (4 semaines)
- **Isolation de l'équipe de toute influence extérieure**

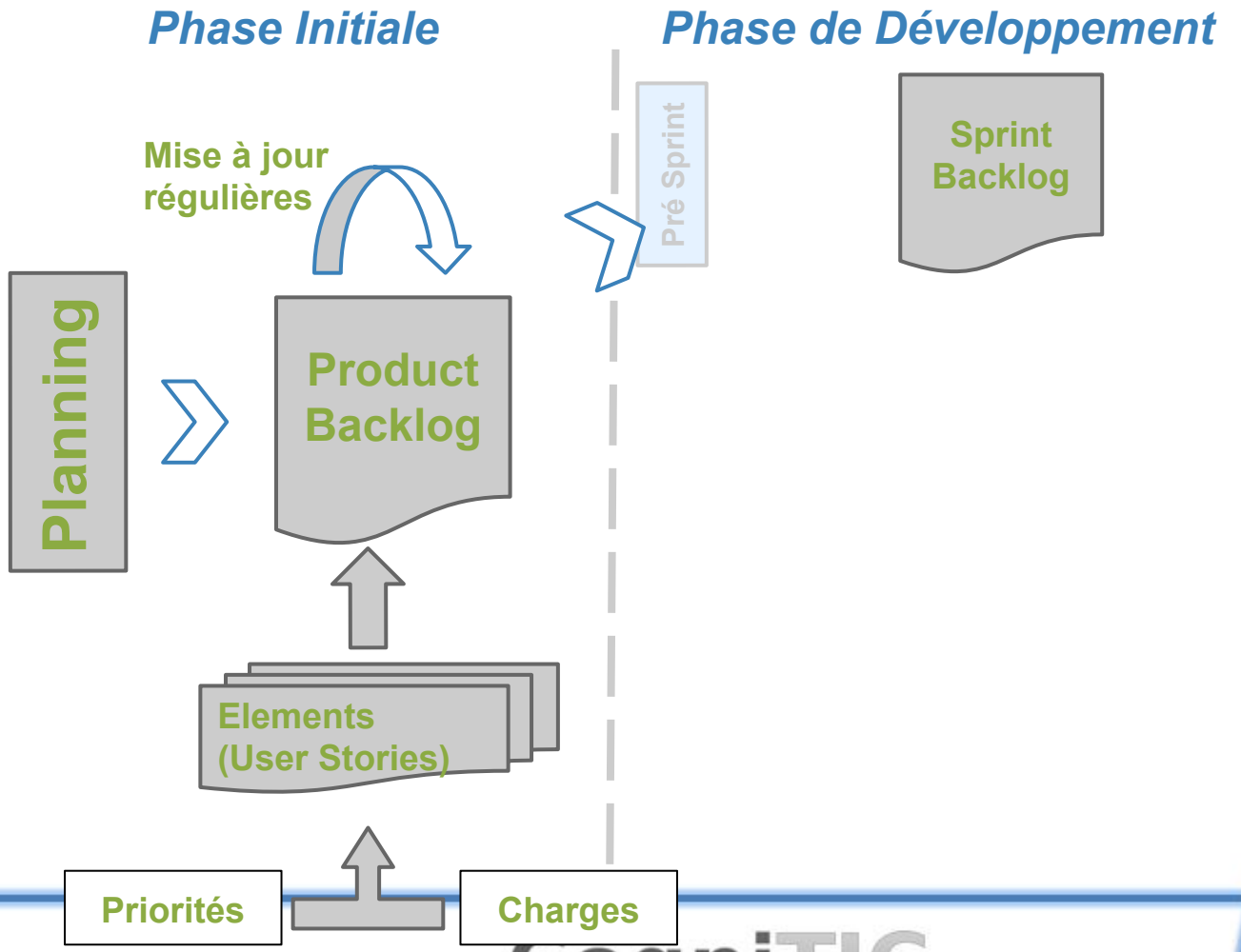
Phase de sprint
Sprint Planning Meeting

Réunion de pré-Sprint organisée par le SCRUM Master en 2-temps:

- **1ère Phase:** Clients, utilisateurs, management, product owner et Scrum Team établissent le Sprint Backlog
- **2ème Phase:** Le Scrum Master et la Scrum Team organisent le déroulement du Sprint

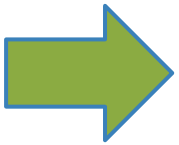
TIMEBOX: 2 heures par semaine de SPRINT

Processus


Phase Initiale
Phase de Développement
Mise à jour régulières
Pré Sprint
Sprint Backlog
Planning
Product Backlog
Elements (User Stories)
Priorités
Charges

Phase de sprint
Daily Scrum

- Participation quotidienne aux réunions Scrum
  - partager les connaissances acquises
  - faire un point sur l'avancement
  - donner au management une certaine visibilité sur la progression du projet
- Contrôle continu et empirique via 3 questions quotidiennes:
  - *qu'est ce qui a été fait pendant la journée ?*
  - *que reste-t-il à faire ?*
  - *quels sont les obstacles qui gênent l'avancement du projet ?*



Phase de sprint
Daily Scrum

- Réunion quotidienne informelle de toute l'équipe
- Toujours à la même heure et au même endroit
- N'interviennent que les personnes qui travaillent effectivement sur le développement
- Les extérieurs y sont invités pour suivre l'avancement du projet

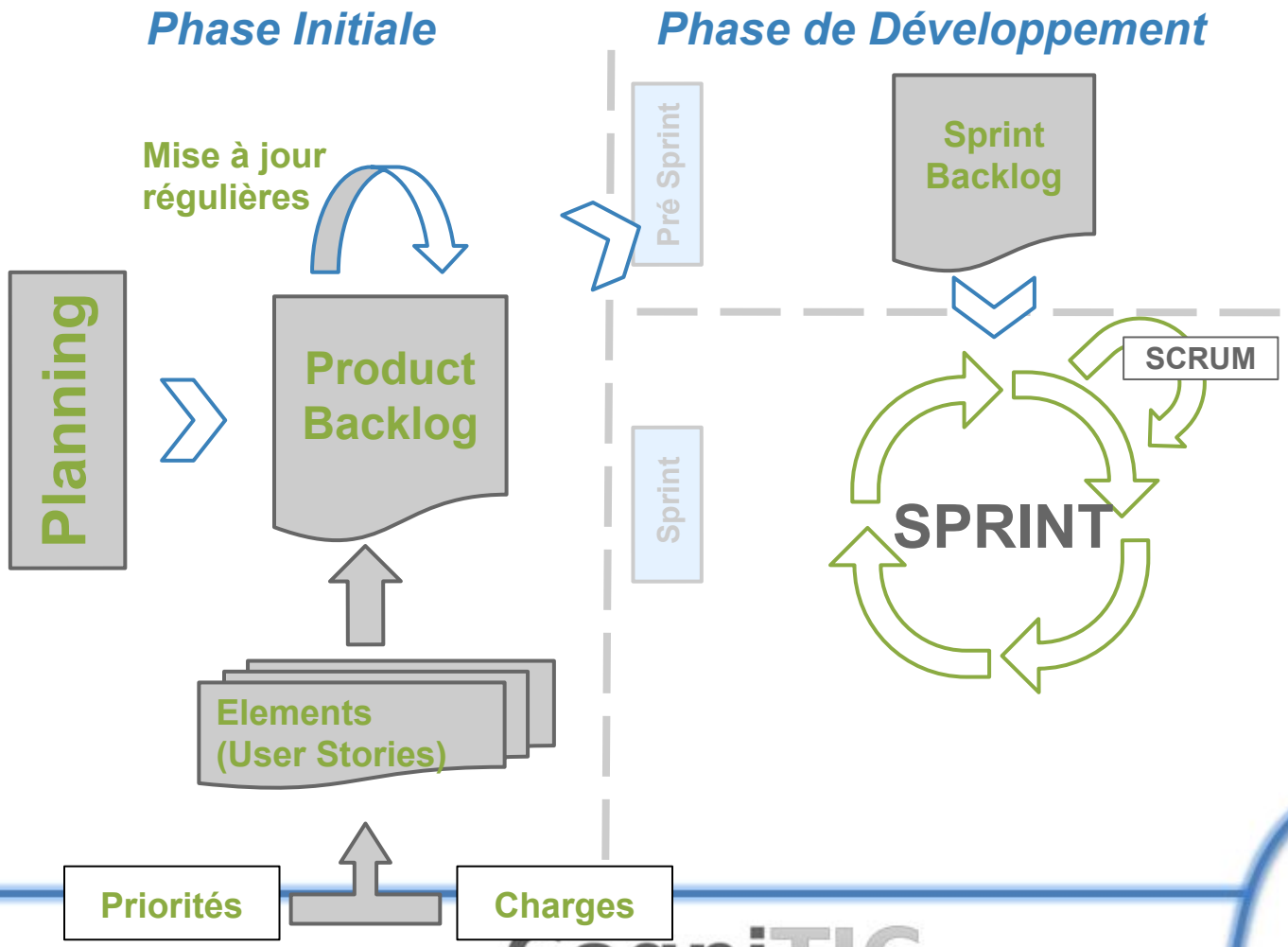
Phase de sprint
Daily Scrum

- **TIMEBOX**: 15 à 30 minutes max

- Le Scrum Master prend les décisions que l'équipe est incapable de prendre par elle même et s'engage à apporter une solution à tout ce qui entrave le développement du projet

- Permet la synchronisation quotidienne de l'équipe et le partage des connaissances

Processus


Phase Initiale
Phase de Développement
Mise à jour régulières
Planning
Product Backlog
Elements (User Stories)
Priorités
Charges
Pré Sprint
Sprint
Sprint Backlog
SCRUM
SPRINT

Phase de sprint
Sprint Review

- Réunion informelle
- L'équipe présente au client ce qui a été développé pendant les 30 jours précédents via une démonstration
- Confronter les résultats du travail de l'équipe avec la complexité et le chaos de l'environnement dans lequel l'application sera utilisée
  - Décision de release ou non

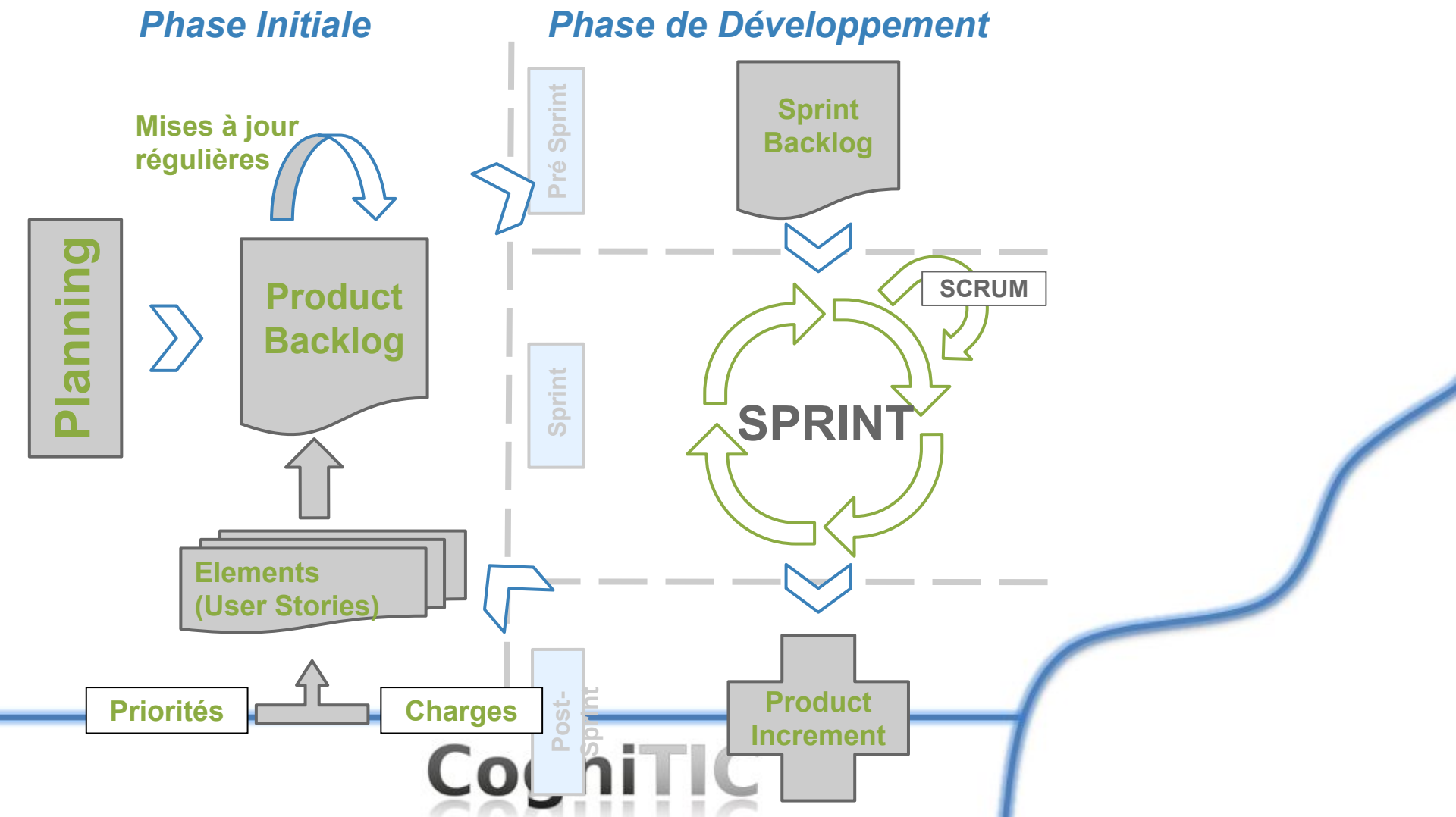
TIMEBOX: 1 heure par semaine de Sprint

Phase de sprint
Sprint Retrospective

- Réunion ayant lieu après le Sprint Review et avant le Sprint Planning Meeting
- Le but du Sprint Rétrospective est de:
  - Inspecter la manière dont s'est déroulée le Sprint précédent quant aux personnes, relations, processus et outils utilisés
  - Identifier et ordonner les éléments majeurs qui se sont déroulés ainsi que les améliorations potentielles
  - Créer un plan pour implémenter des améliorations quant à la manière de travailler de l'équipe Scrum

TIMEBOX: 3 heures pour 4 semaines de Sprint

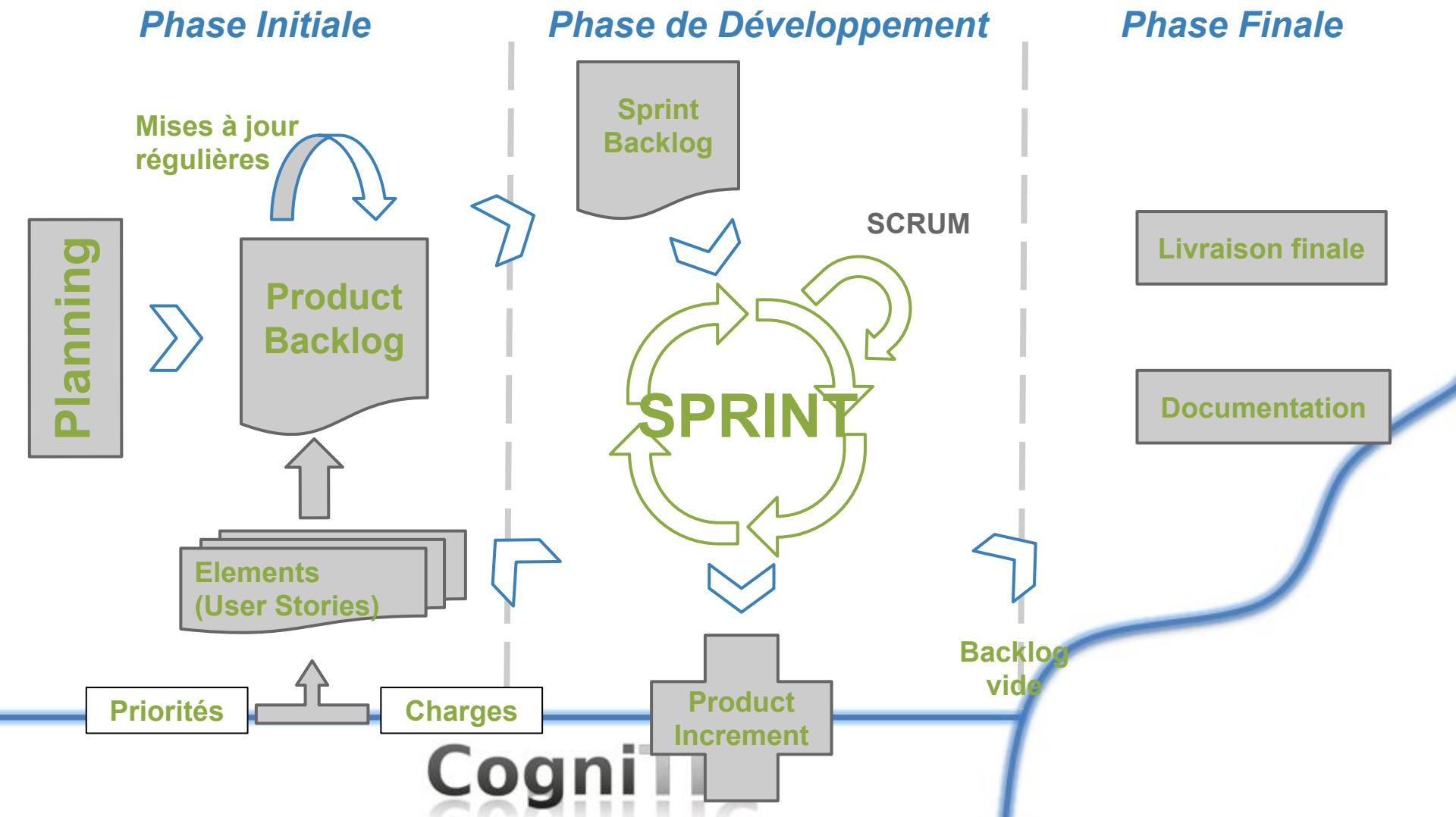
Processus


Phase Initiale
Phase de Développement
Mises à jour régulières
Planning
Product Backlog
Elements (User Stories)
Priorités
Charges
Pré Sprint
Sprint
Post-Sprint
Sprint Backlog
SCRUM
SPRINT
Product Increment

Phase de clôture

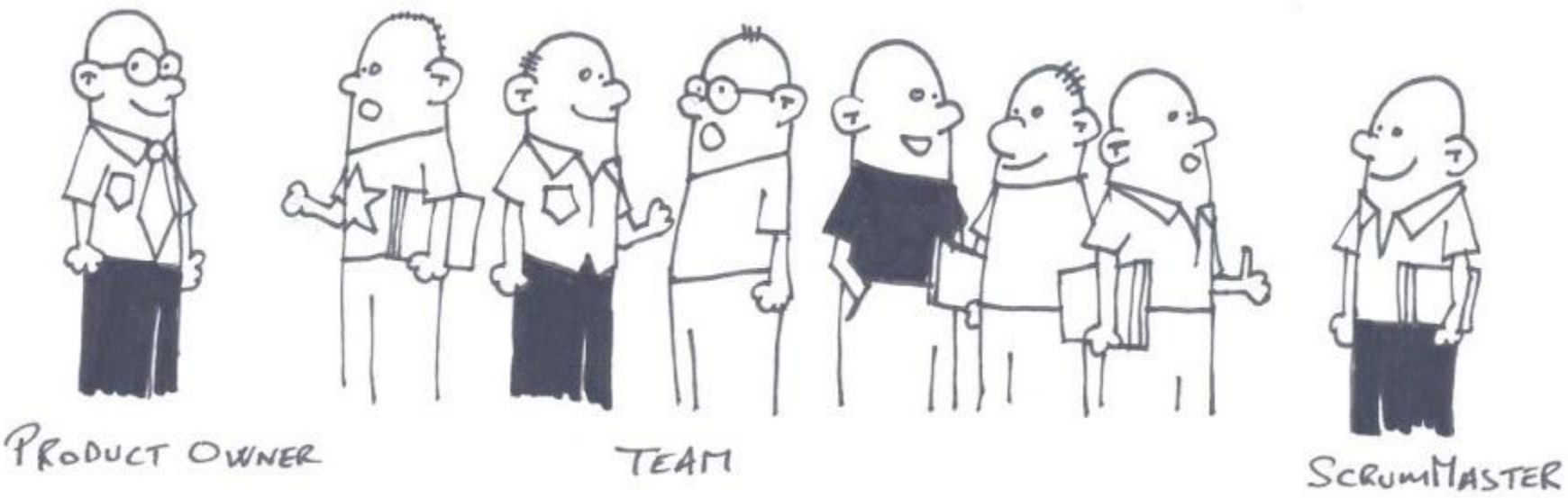
- Préparation du produit pour une livraison:
  - intégration,
  - tests systèmes,
  - documentation utilisateur,
  - préparation de supports de formation,
  - préparation de supports marketing,
  - ...

Processus

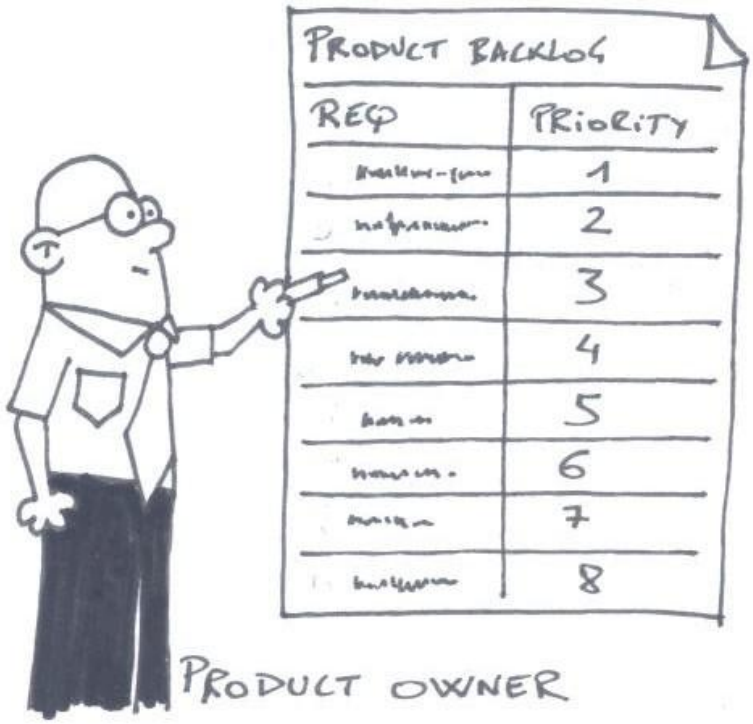

Phase Initiale
Phase de Développement
Phase Finale
Mises à jour régulières
Planning
Product Backlog
Elements (User Stories)
Priorités
Charges
Sprint Backlog
SCRUM
SPRINT
Product Increment
Backlog vide
Livraison finale
Documentation

Itération 2

...où l'on se focalise sur les rôles SCRUM



Les Rôles



Le Product Owner

CogniTIC

Product Owner
Le rôle du Product Owner

Le Product Owner:

- Est responsable de la gestion du Product Backlog
- S'assure que le Product Backlog soit visible pour chacun
- S'assure de la valeur du travail fourni par l'équipe

Product Owner
Les caractéristiques

- Visionnaire
- Leader et Team Player
- Communicateur et Négociateur
- Concerné et Engagé
- Disponible et Qualifié

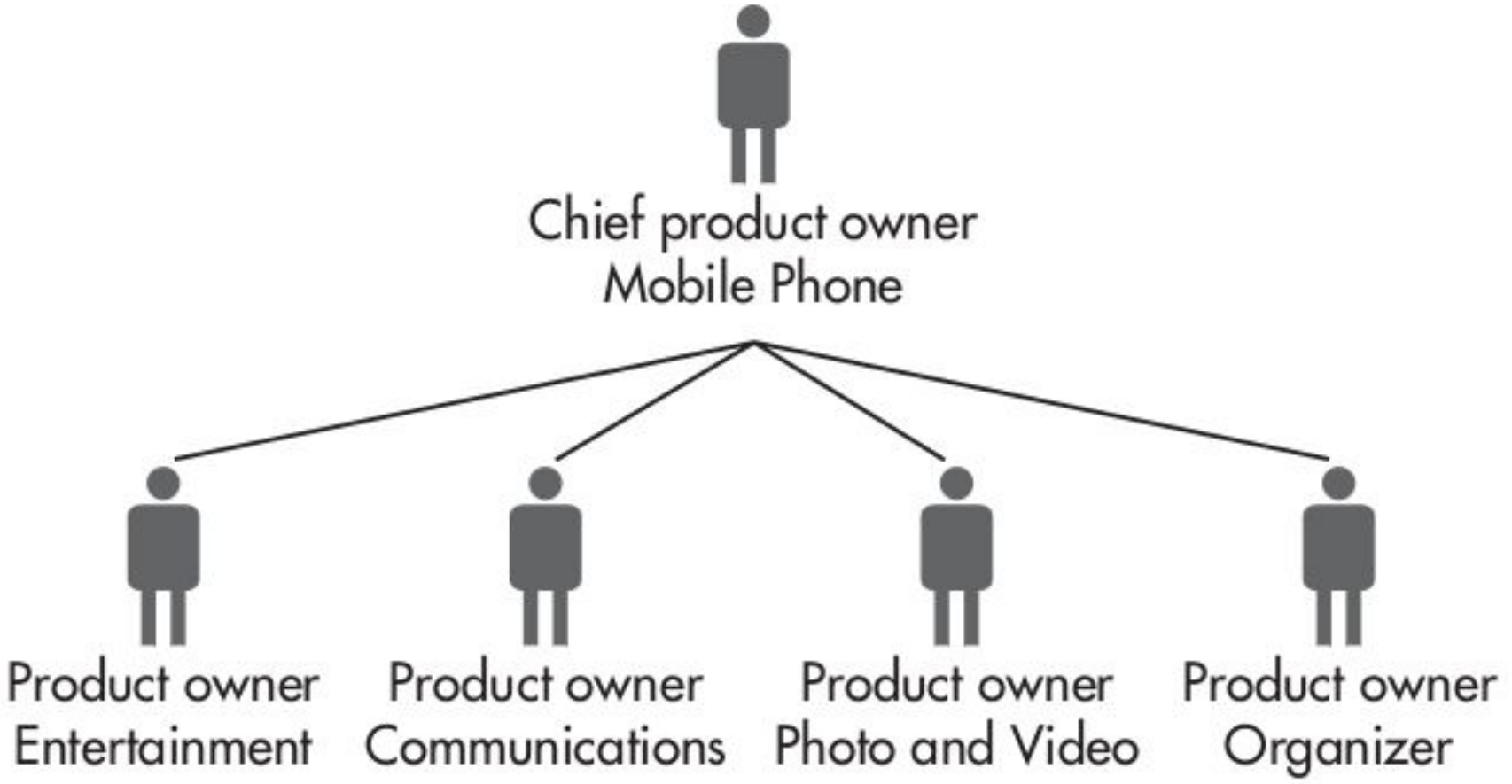
Product Owner Collaboration

- Collaboration avec l'équipe SCRUM
  - Lui même membre de la SCRUM Team
  - Le Product Owner doit se trouver sur le même site que la SCRUM Team
- Collaboration avec le SCRUM Master
  - Rôles complémentaires
  - Product Owner est responsable du "QUOI"
  - SCRUM Master est responsable du "COMMENT"
  - Une personne ne doit pas cumuler les 2 rôles

Product Owner
Ajustement du rôle

- Pour les grands projets SCRUM
  - Difficile pour un Product Owner de s'occuper de plus de 2 SCRUM Team
  - Quid de projets comprenant plus de 2 SCRUM Team alors que le Product Owner est unique?
- Introduction du rôle du Chief Product Owner
  - Responsable de coordonner différents Product Owner
  - Responsable de la vision du produit

Chief Product Owner



Product Owner
Erreurs courantes

- Ne pas donner assez de pouvoir au Product Owner
  - Il ne dispose pas assez d'autonomie
- Surcharger le Product Owner
  - Pas soutenable
  - Devient le bottleneck du projet
- Ne pas séparer (transversalement) les tâches du Product Owner entre différentes personnes

Le SCRUM Master

CogniTIC

SCRUM Master
Le rôle du SCRUM Master

Le SCRUM Master:

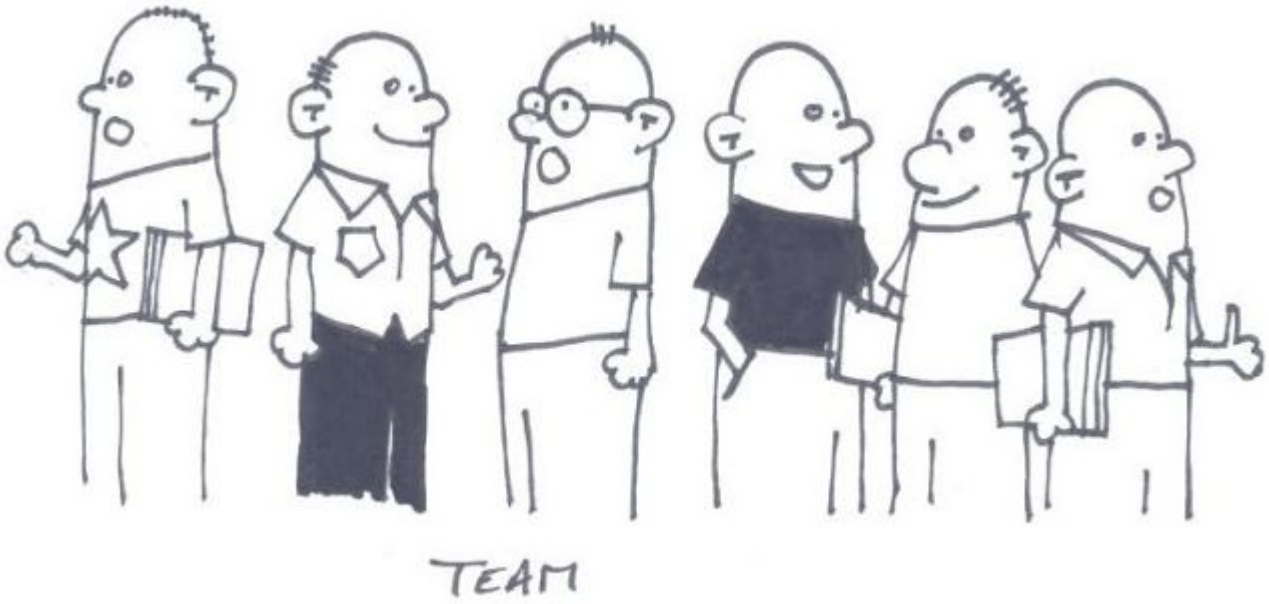
- est le garant de l'application du processus Scrum
- aide l'équipe à travailler de façon autonome et à s'améliorer constamment
- anime les réunions de Pré-Sprint, Scrum, Post-Sprint
- élimine les obstacles qui ralentissent l'équipe
- communique avec le management (ex. rapports d'avancement)

SCRUM Master
Les caractéristiques

- Bien connaître SCRUM
- Talent de communication
- Pouvoir résoudre les conflits
- Etre capable de guider sans imposer

SCRUM Master
Erreurs courantes

- Le SCRUM Master n'est pas un chef de projet
  - Il ne dirige pas, il n'impose pas, il ne contraint pas
  - Il fait partie de l'équipe



L’équipe

L’équipe de développement

- Responsable du déroulement de chaque Sprint, c-à-d., ce qui est mis en oeuvre pour atteindre les objectifs du Sprint

- Impliqué dans:
  - l'estimation de la charge de travail
  - la création du Sprint Backlog
  - l'identification des freins à l'avancement du projet

L’équipe de développement

Programmeurs, testeurs, concepteurs…

Responsabilités	Caractéristiques
Créer le produit	Apprécie participer à la création du produit Dispose d’au moins une compétence nécessaire à la création du produit
S’auto-organise et s’auto-gère	Fait preuve d’initiative et d'indépendance
Est pluri-disciplinaire	A de la curiosité Désire contribuer au delà de sa zone de maîtrise Apprécie apprendre de nouvelles compétences Partage volontier sa connaissance
Est dédiée et rassemblée	

L’équipe de développement Pluridisciplinaire

La volonté et l’aptitude à travailler sur différent type de tâches

- Mettre de côté les titres qui inhibent les compétences
  - Testeurs, développeur .NET, développeur Java…
- Travailler à étendre ses compétences
  - Apprendre quelque chose de nouveau chaque sprint
- Se proposer pour aider un membre qui fait face à un obstacle
- Être flexible

L’équipe de développement S’auto-organise

Afin de tirer avantage de la connaissance et l’expérience des membres de l’équipe

- S’engage vis-à-vis des objectifs du sprint
- Identifie ses tâches
- Estime l’effort nécessaire à chaque tâche
- Se focalise sur la communication
- Collabore
- Prend des décisions sur base de consensus
- Participe activement

L’équipe de développement S’auto-gère

Contrôle la manière dont elle travaille afin d’assurer le succès du projet

- Autorise la fluctuation du leadership
- S’appuie sur les outils et processus agile
- Rapport régulièrement et de manière transparente l’avancement du projet
- Règle les problèmes au sein de l’équipe
- Crée une convention d’équipe
- Inspect and Adapt

L’équipe de développement
Taille réduite

Idéalement est composé de 5 à 9 membres

- Facilite la bonne communication
- Maintien l’unicité de l’équipe
  - Évite la création de sous-équipes
- Encourage la pluridisciplinarité, la communication en face-à-face…
- Prend des initiatives
- Réussi ou échoue en tant qu’équipe

Autres: Management et Client

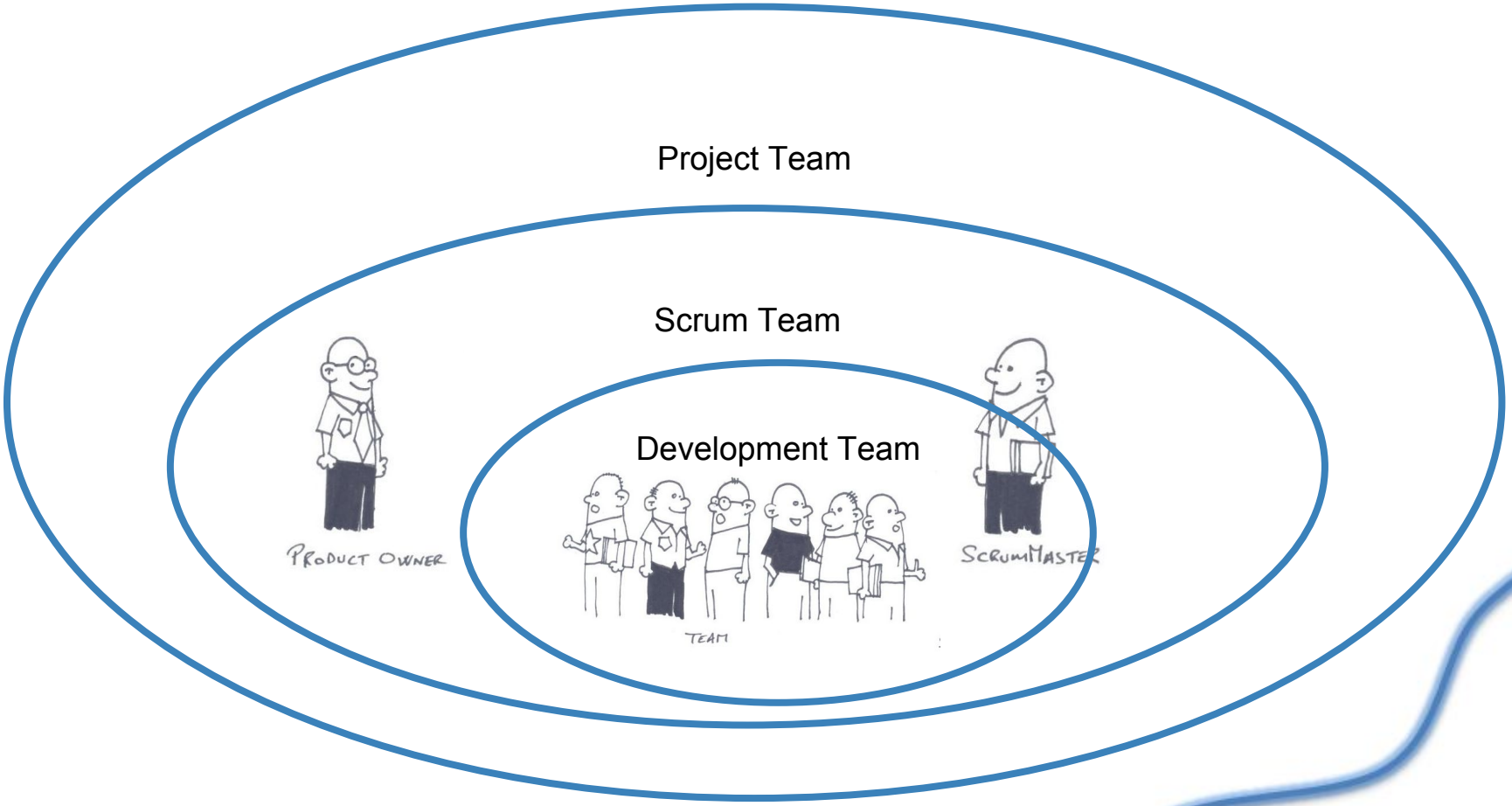
Management

- Responsable de la décision finale
- Impliqué dans le choix du Product Owner
- Surveille l'avancement du projet
  - Peut réduire le Backlog en concertation avec le Scrum Master

Client

- Participe à l'élaboration du Product Backlog

Les différentes équipes



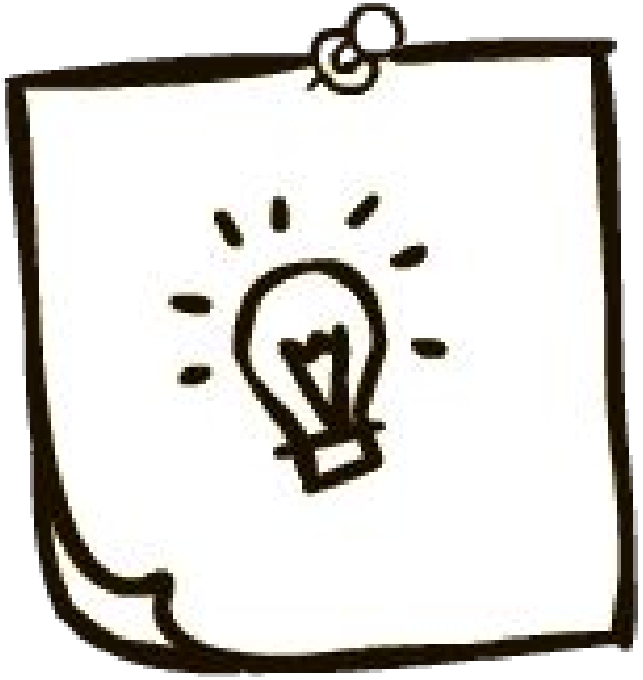
Exercice

Par deux, classifier les "commandements" entre les rôles du Scrum Master et du Product Owner

(seulement 1 ou 2 commandements s’appliquent aux deux rôles)

Itération 3

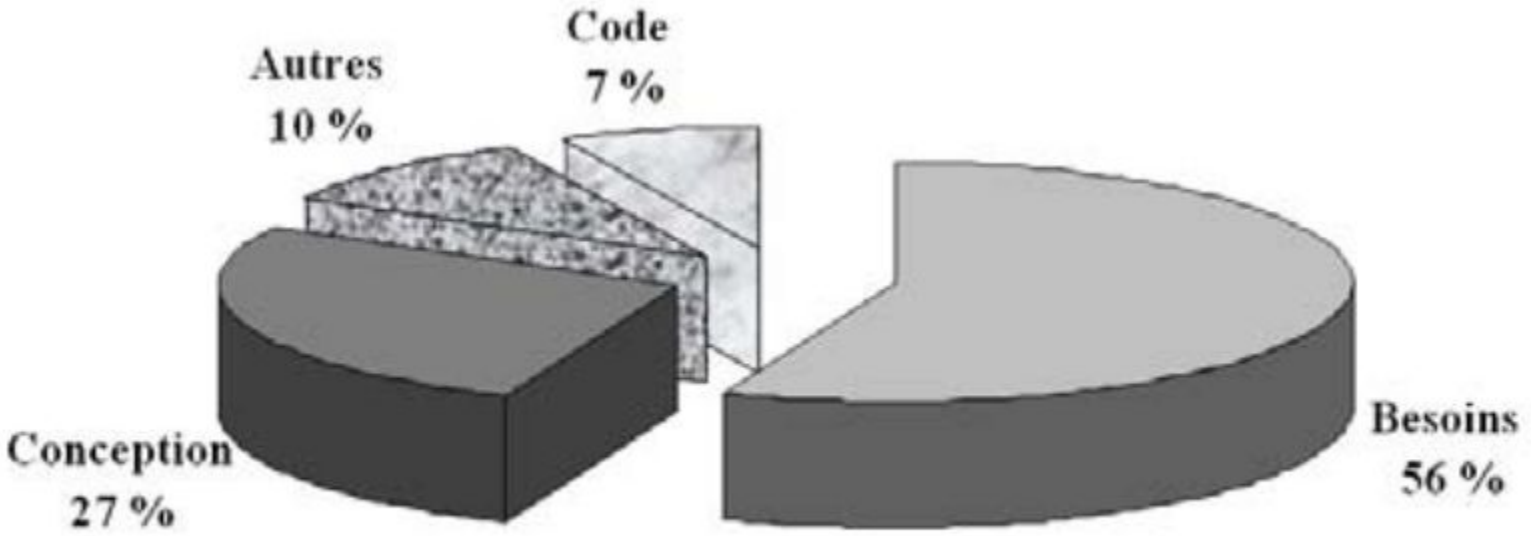
...ou comment imaginer le Produit



Le produit

Les besoins
Quelques chiffres

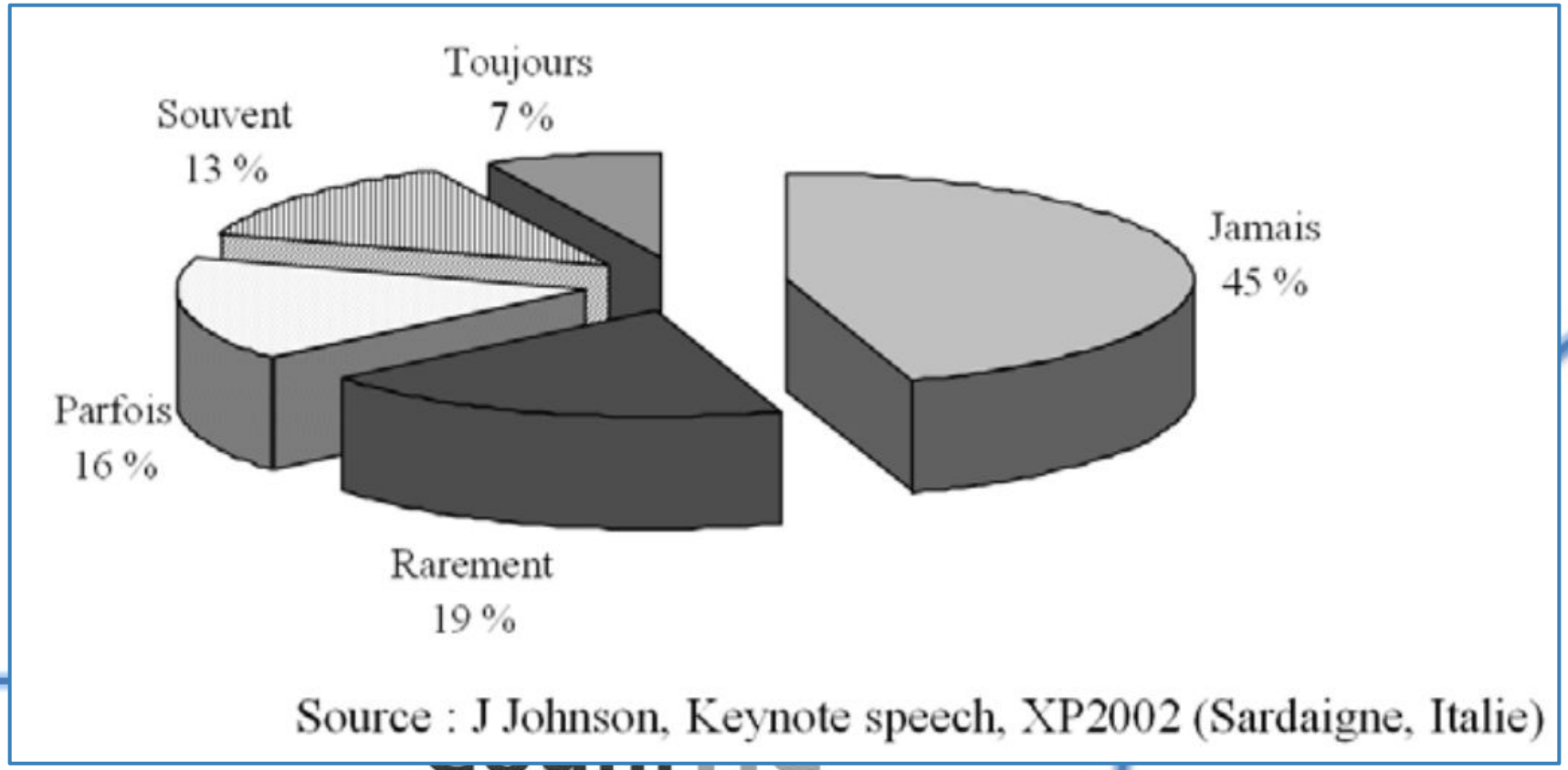
Origines des défauts logiciels



Source : J Johnson, Keynote speech, XP2002 (Sardaigne, Italie)

Les besoins
Quelques chiffres

Pourcentage de fonctionnalités implémentées réellement utilisées



Source : J Johnson, Keynote speech, XP2002 (Sardaigne, Italie)

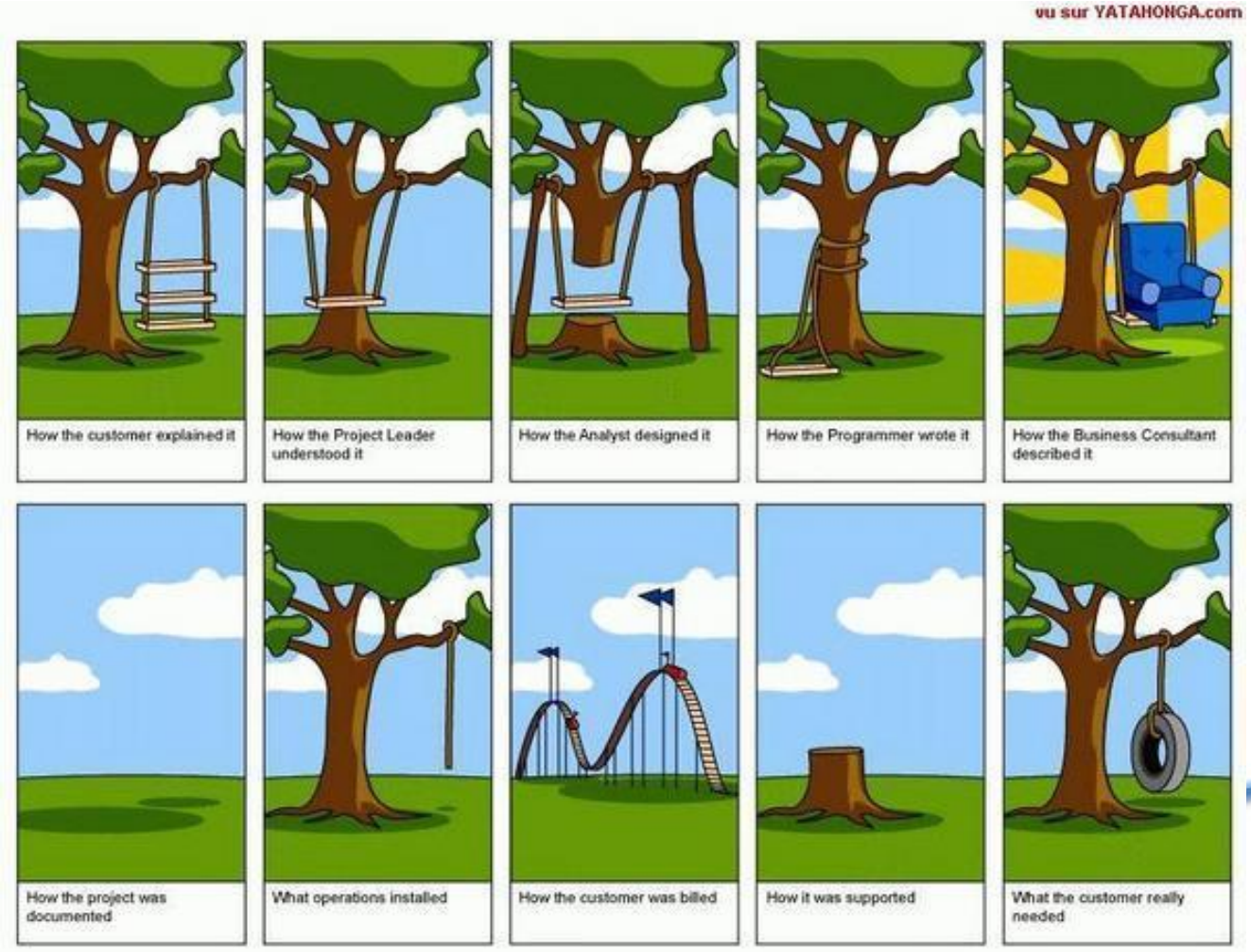
Les besoins
Difficultés du recueil

- Recueillir les besoins présentent 3 problèmes majeurs:
  - Une mauvaise communication
  - L'illusoire exhaustivité
  - La défaillance du client

Les besoins
Une mauvaise communication

- **Divergences** possibles entre:
  - la représentation qu'a le client de son futur produit
  - la difficulté éventuelle qu'il rencontre à le décrire
  - l'interprétation possible de cette description par l'équipe de réalisation
  - l'outil qui lui est livré au final

Les besoins
Une mauvaise communication



Les besoins
Une mauvaise communication

Dessiner une pizza qui a 8 parts avec trois traits

Les besoins
L'illusoire exhaustivité

- Le marché est **instable** et l'organisation doit être réactive face à ses fluctuations
- **Priorités** différentes entre les besoins
- **Après analyse** une idée peut s'avérer inutile, trop coûteuse ou secondaire
- L'introduction d'une nouvelle demande peut impliquer la **renonciation** d'un autre besoin

Le produit
La vision du produit

Comment se prémunir de ces problèmes?

La Vision du Produit

Le produit
La vision du produit

Les questions à se poser:

- Qui va acheter le produit? Qui va utiliser le produit?
- Quels besoins vont être satisfaits par le produit? Qu'est ce que le produit ajoute de plus?
- Quels sont les attributs critiques pour répondre à ces besoins?
- Comment se comporte le produit par rapport à la concurrence?
- Quel est le business model?
- Le produit est-il réalisable?

Exemple: iPod

Le produit
Les qualités de la vision

- Partagée et unifiée
  - Permet d'aligner les efforts fournis par la Team
- Large et engageante
  - Guide le développement mais laisse la place à la créativité
- Brève et concise
  - Ne contient que l'information critique au succès du produit

Le produit
Une version minimale du produit

- Créer une vision = Projection du futur
- Capacité de prédiction limitée
- Nécessité de prévoir une version minimale du produit
- Exemple:
  - Newton **vs** iPhone

-> SIMPLICITE !

Le produit
Une version minimale du produit

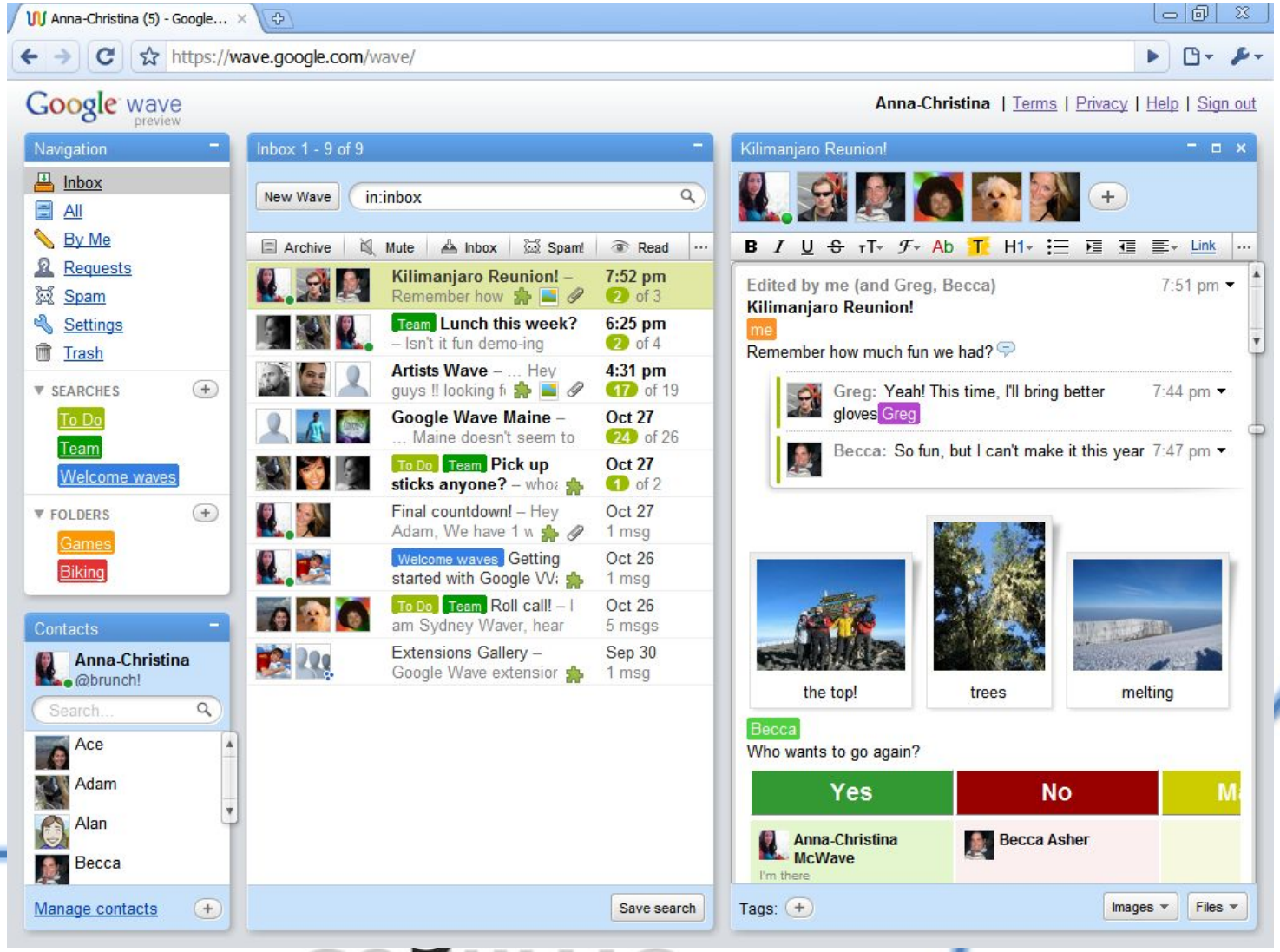
- Concentration sur les besoins cruciaux du consommateur
- N'a pas tenté d'offrir toutes les fonctionnalités offertes par les concurrents
  - Pas de copier-coller
  - Pas de possibilité d'envoyer un message à plusieurs destinataires
  - Pas de kit de développement

Le produit
Simplicité

Google Chrome

google.com

Le produit
Simplicité - Contre-exemple



Le produit
Besoins et Attributs

- Le produit est un **moyen** de satisfaire ces **besoins**, pas une fin en soi

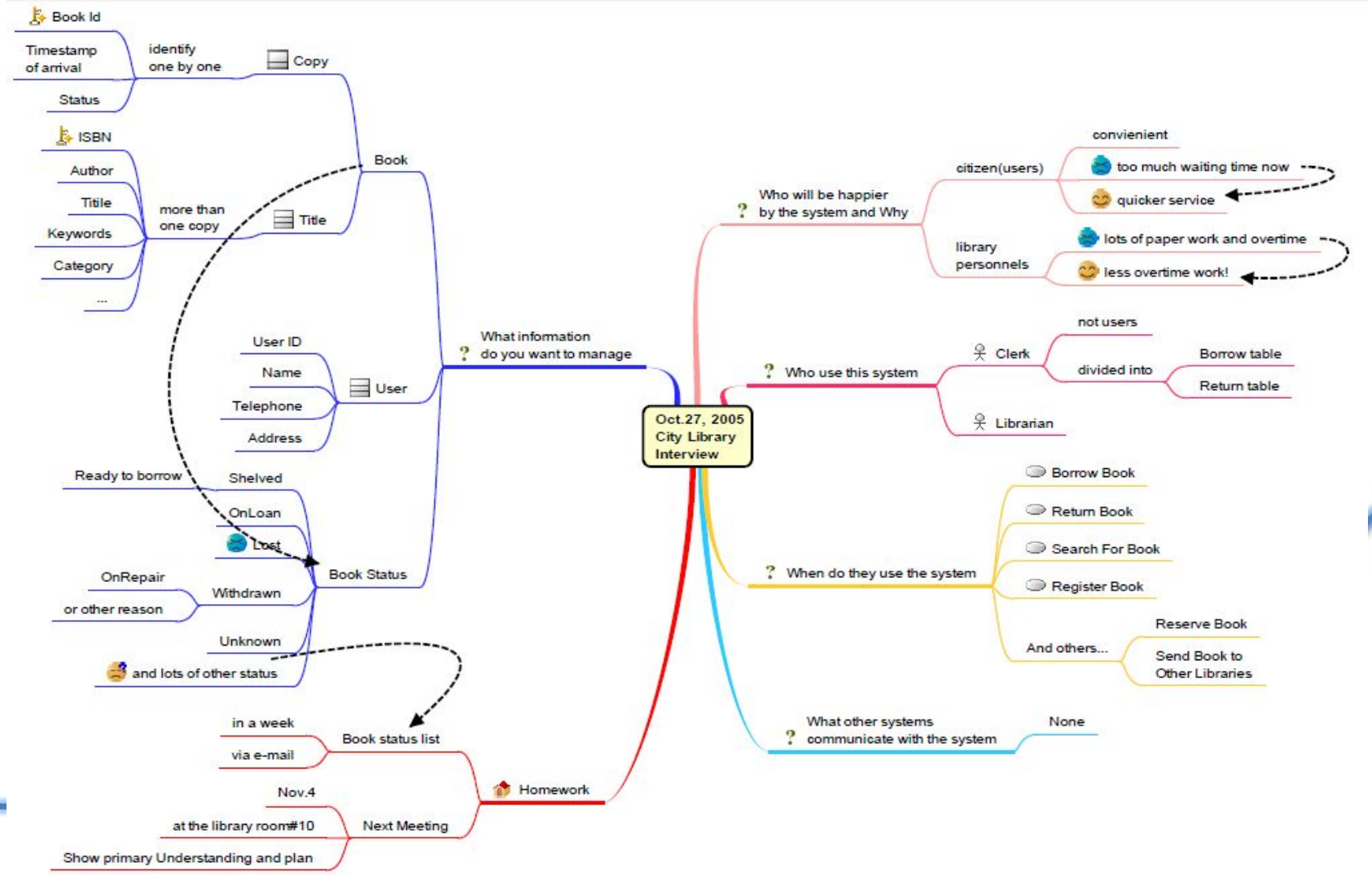
- Attributs du produit = propriétés critiques du produit permettant de satisfaire ces besoins

- Explorer les différents attributs possibles et trouver la meilleure alternative

Les besoins
Comment recueillir les besoins

1. En amont du projet:
   a. Le brainstorming
      (Instancié à l'aide d'une *mind map)*
   b. L'analyse de la concurrence (*benchmarking*)
2. Affiner les besoins:
   a. Interviews
      (Questionnaires, croisement des résultats)
   b. Workshops
3. La description détaillée:
   a. L'analyse de l'existant
      (Forces/faiblesses de l'application)
      (Quelles fonctionalités améliorer/remplacer)
   b. L'observation du comportement de l'utilisateur
      « en situation »

Les besoins
Comment recueillir les besoins



Les besoins
Comment recueillir les besoins

1. En amont du projet:
   a. Le brainstorming
      (Instancié à l'aide d'une *mind map)*
   b. L'analyse de la concurrence (*benchmarking*)
2. Affiner les besoins:
   a. Interviews
      (Questionnaires, croisement des résultats)
   b. Workshops
3. La description détaillée:
   a. L'analyse de l'existant
      (Forces/faiblesses de l'application)
      (Quelles fonctionalités améliorer/remplacer)
   b. L'observation du comportement de l'utilisateur
      « en situation »

Techniques d'interview
Les neuf cases de Solution Selling

	Quel est le problème?	**Qui est impacté?**	**Visualiser la solution**
Questions ouvertes "Dites moi..." "Racontez moi..." "Et puis..."	**1**	**4**	**7**
Contrôle Combien? Quand? Où?	**2**	**5**	**8**
Validation "Si je comprends bien..." Si non, revenir au questions ouvertes Si oui, passer à la question suivante	**3**	**6**	**9**

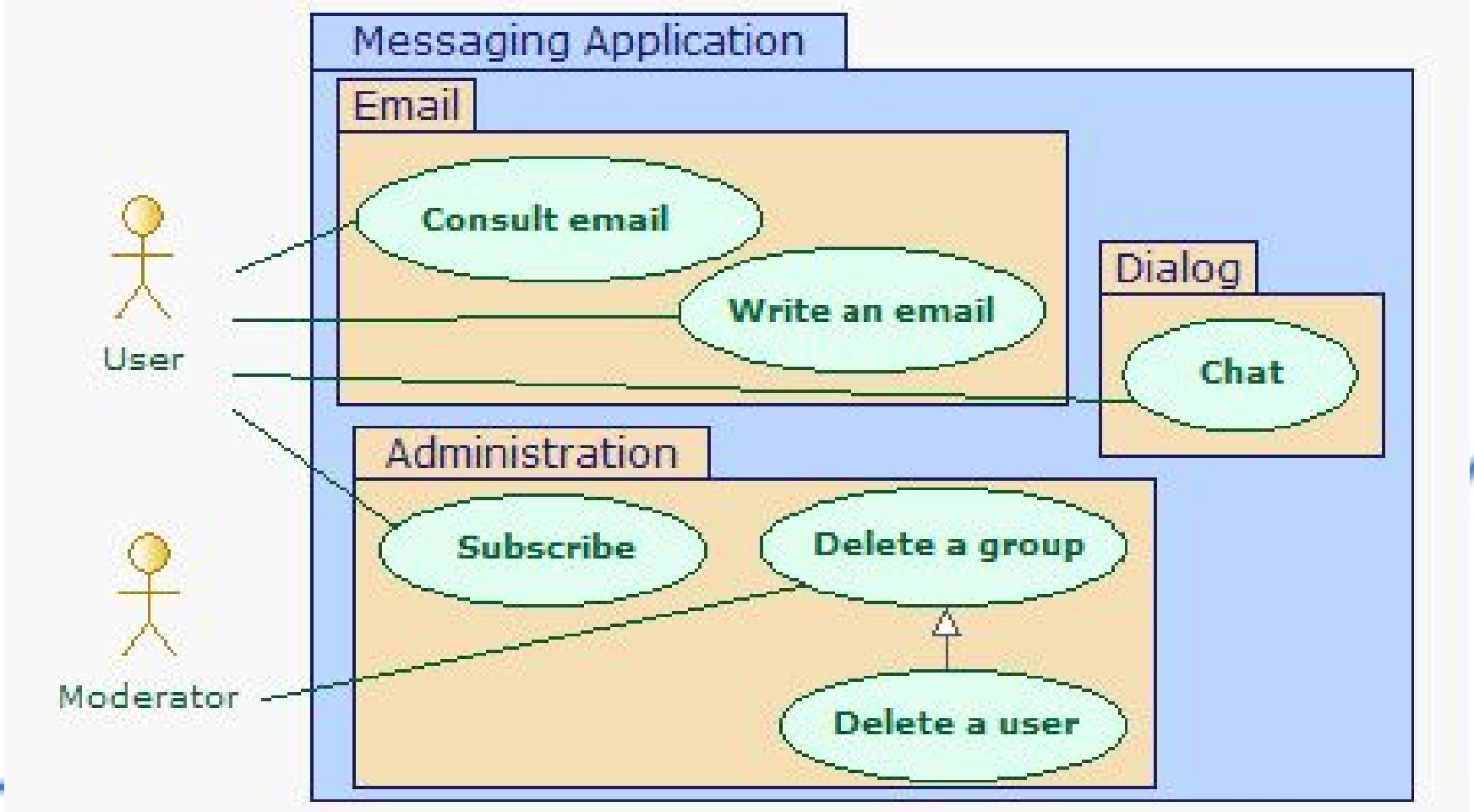
Les besoins
Comment recueillir les besoins

1. En amont du projet:
   a. Le brainstorming
      (Instancié à l'aide d'une *mind map)*
   b. L'analyse de la concurrence (*benchmarking*)
2. Affiner les besoins:
   a. Interviews
      (Questionnaires, croisement des résultats)
   b. Workshops
3. La description détaillée:
   a. L'analyse de l'existant (prototypage)
      (Forces/faiblesses de l'application)
      (Quelles fonctionnalités améliorer/remplacer)
   b. L'observation du comportement de l'utilisateur « en situation »

Le produit
Formaliser et exprimer les besoins

- Use Cases (Cas d'utilisations)
  - Modélisation UML
  - Décrit les échanges entre le système et les acteurs externes (utilisateurs ou autres systèmes) pour un contexte particulier
  - Décrit les différentes étapes pour les séquences:
    - nominales
    - d'exception
- User Stories
  - En tant que ..., je veux ... afin de ...

Le produit
Diagramme de cas d'utilisation



Le produit
Cas d'utilisation textuel

USE CASE # 1	*Forme verbale (infinitif)*		*(ex : Passer une commande)*
Résumé			
Acteur principal	X		
Intervenants & Intérêts	*Intervenants*	*Intérêt*	
	X Y Z		
Préconditions			
Garanties de succès	*(Conditions d'arrivée)*		
Déclencheur	*(Evènement)*		
Scénario nominal	Etape	Action	*(Le cas où tout se passe bien)*
	1	L'utilisateur	
	2	X lance	
	3	Le système	
	4	X lance le processus (Cas d'utilisation 1.1)	
	5	Le système.	
Extensions	Etape	Action	*(Variations au scénario et cas d'erreurs)*
	2a	X reçoit ...	
		2a.1 X fait ceci	
		2a.2 Y fait cela	

Le produit
Use Case vs. User Story

User Story	Use Case
Brève description vue par l'utilisateur	Séquence d'action du système et ses interactions avec les autres acteurs
Format court, stimule la discussion orale	Format riche en information, peu de place à l'oral
Peut être une partie d'un Use Case	Somme d'un scénario
Utilisé pour la spécification et/ou la planification	Uniquement utilisé pour la spécification
Emergence rapide au travers d'ateliers collaboratifs	Long travail d'analyse et de formalisation
Grande visibilité	Visibilité réduite (due au volume)
Difficile à lier les unes aux autres	Liaison et vision globale

Le produit
Les erreurs courantes

- Pas de vision
  - Produit limité à un ensemble de features
- Vision prophétique
  - Capacité limité de projection
- Analyse sans fin des besoins
  - Peur de l'échec
- "On sait ce qui est bon pour le client"
- "Big is beautiful"

Exercice:
La vision du produit

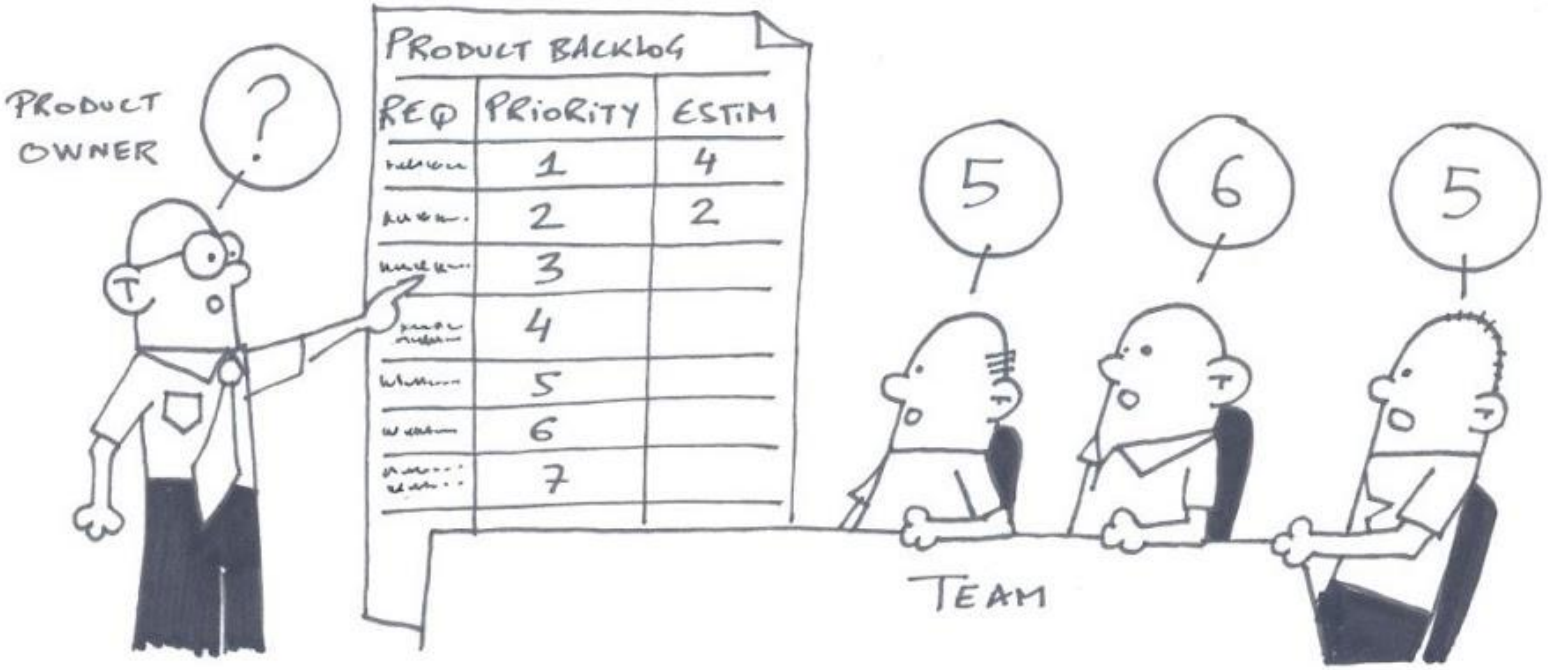
Les questions à se poser:

- Qui va acheter le produit? Qui va utiliser le produit?
- Quels besoins vont être satisfaits par le produit? Qu'est ce que le produit ajoute de plus?
- Quels sont les attributs critiques pour répondre à ces besoins?
- Comment se comporte le produit par rapport à la concurrence?
- Quel est le business model?
- Le produit est-il réalisable?

Exercice:
La vision du produit

Itération 4

... ou comment travailler avec le Product Backlog



Le Product Backlog

Entretenir le product backlog

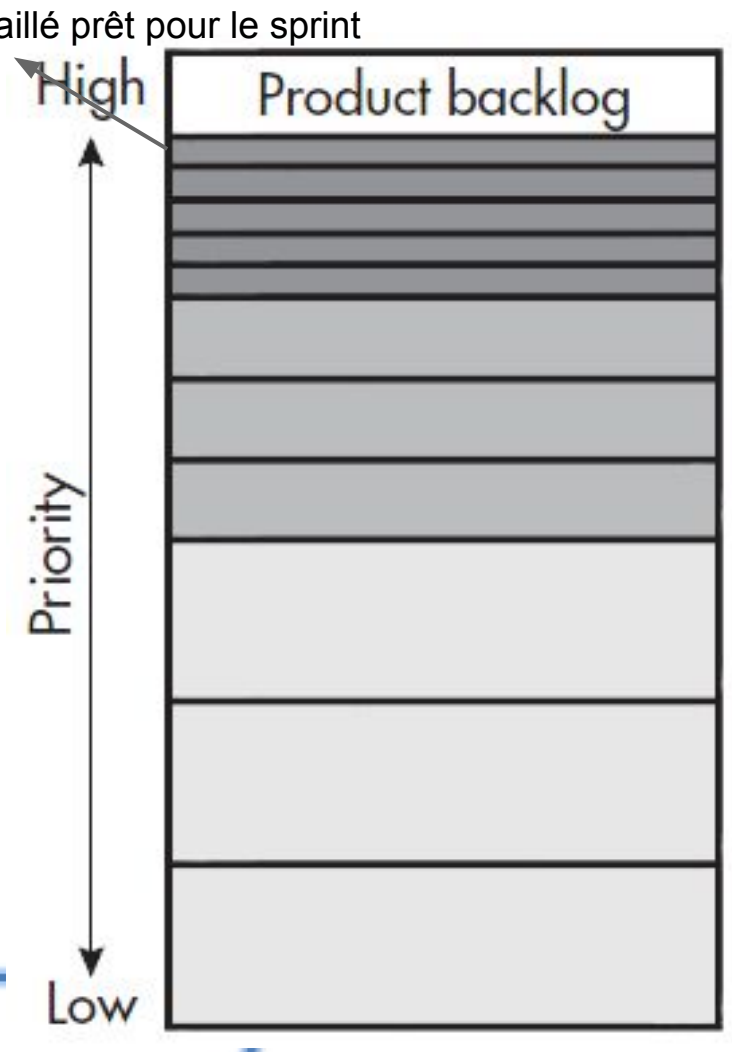
- Processus **continu** qui assure les 4 qualités du Product Backlog comprenant:
  - Les éléments hautement prioritaires sont décomposés et raffinés en vue de la réunion du planning de Sprint (**D**etailed Appropriately)
  - **E**stimation des éléments via le planning poker
  - Ajout, modification, suppression d'éléments (**E**mergent)
  - **P**rioritisation (plus important en haut)
- Processus collaboratif:
  - Toute l'équipe Scrum y **participe**
  - Communication en face-à-face plutôt qu'au travers de documents
- Responsabilité du **Product Owner**

Les qualités du Product Backlog

- 4 Qualités: **DEEP**
  - **D**etailed Appropriately
  - **E**stimated
  - **E**mergent
  - **P**rioritized

Détaillé de manière appropriée

- Les éléments les plus prioritaires sont les éléments les plus détaillés
- Implique que les exigences soient décomposées et raffinées durant l'entièreté du projet
- Les éléments du PB sont des **User Stories**



Détaillé de manière appropriée

- Les éléments du product backlog sont des **user story**
  - *En tant que* **[rôle]**, *je veux* **[action]** *afin de* **[but]**
- Exemple:
  - As a user closing the application, I want to be prompted to save anything that has changed since the last save so that I can preserve useful work and discard erroneous work.

Workshop d'écriture des User Stories

- Participants: le **Product Owner**, partenaires additionnels (ex. utilisateurs), (la SCRUM Team)
- Pas forcément à chaque Sprint
- Génération sur base de Brainstorming
  - On commence avec des "**epics**" et on découpe
- Ecrire un maximum de stories possible
  - Certaines directement implémentable
  - Certaines resteront temporairement des "epics"

Bonnes pratiques

- Ecrire pour un seul utilisateur
- Ecrire à la forme active
- Ne pas numéroter les cartes
- Ecrire des user stories fermées

Détaillé de manière appropriée

Pourquoi utiliser des user stories?

- Suscite la communication verbale
- Compréhensible pour chacun
- Taille appropriée pour la planification
- Fonctionne bien dans des processus itératifs

DEEP
INVEST

Détaillé de manière appropriée

User Story:

- Independent
- Negotiable
- Valuable
- Estimable
- Small
- Testable

Détaillé de manière appropriée

Independent

- The user story should be self-contained, in a way that there is no inherent dependency on another user story.

Détaillé de manière appropriée

Negociable

- Stimule la conversation

The user can pay with a credit card.	The user can pay with a credit card.
Note: Accept Visa & MasterCard. Consider American Express. On purchases over $100 ask for card ID from the back of the card. The system can tell what type of card it is from the first two digits of the card number. Collect the expiration month and year.	Note: Accept Visa & MasterCard. Consider American Express. UI: no need for a separate field for the card type.
TOO MUCH DETAIL	**JUST ENOUGH**

DEEP
INVEST

Détaillé de manière appropriée

Valuable

- To users and purchasers NOT to developers.
  - The user can pay with a credit card.
    - **Valuable to users**
  - For the authentication stories, produce documentation for ISO 9001 audit.
    - **Valuable to purchaser**
  - All error handling is done through a set of common classes
    - **No tangible value for the user or the purchaser**
  - All errors are presented to the user and logged in a consistent manner.
    - **Tangible value is clearer**

Détaillé de manière appropriée

Qui pour écrire les user stories?

- Utilisateurs
- Si accès limité aux utilisateurs
  - Proxy Users
    - Users' Manager
    - Salespersons
    - Domain Experts
    - Former Users
    - Customers
    - Trainers and Technical Support
    - Business or Systems Analysts
    - ~~Development Manager~~

Détaillé de manière appropriée

Estimable

- Unités utilisées:
  - Story Points (relatif)
  - Jour-hommes (absolut)

- What makes a story difficult to estimate?
  - Lack of domain knowledge
  - Lack of technical knowledge
  - Size is too big

Détaillé de manière appropriée

Small

- A user can plan a vacation.
  - **Too big**
- A user can post her resume.
  - **Too big**

- A user can add her resume.
- A user can update her resume.
- A user can mark resumes as inactive.

Détaillé de manière appropriée

Stratégies de décomposition:

1. Par étape d'un workflow
2. Par scénario (happy scenario / extends)
3. Par séquence d'un scénario
4. Par opérations (CRUD)
5. Par format ou type de données
6. Par rôle
7. Par niveau de connaissance
8. Par niveau de complexité
9. Par niveau de qualité attendu

DEEP
INVEST

Détaillé de manière appropriée

Testable

- Concrete testing criteria.
- Usually written on the back of the card.
- Most tests should be automated.

- The startup screen disappears shortly after the running the program.
  - **Not concrete enough**
- The startup screen disappears within 4 seconds.
  - **Concrete & Testable**

DEEP
INVEST

Détaillé de manière appropriée

- Les tests sont utilisés pour:
  - La spécification
  - Déterminer qu'une user story est terminée
- Ecrire les tests avant de coder
  - Lors des discussions Product Owner/Développeurs
  - Au début du Sprint
- Les tests sont principalement spécifiés par le Product Owner
- Les tests font partie du processus
- Les types de test
- Testing for bugs, not coverage

Détaillé de manière appropriée

Exemple:

- A company can pay for a job posting with a credit card
  - Test with Visa, MasterCard, American Express
  - Test with good, bad and missing card ID numbers
  - Test with expired cards
  - Test with different purchase amounts (including one over card's limit)

Eléments Estimés

- Connaître la charge de travail aide à donner des priorités aux éléments

- Estimation grossière exprimée en nombre *Story Points*
  - Jour idéal de travail
  - Effort à fournir - NUT (Nebulous Units of Time)
    - Effort vs Complexité

Estimer les éléments

- Les estimations sont réalisées par les membres de la Scrum Team
- Le Product Owner et le Scrum Master ne doivent ni participer, ni influencer l'estimation

Estimer les éléments

- Story points
  - Mesure **relative** de l'effort à fournir

- Planning Poker
  - On affecte la plus petite valeur au plus petit item
  - Pour l'item suivant, chaque membre dépose une carte avec une valeur de l'échelle face cachée
  - Si il y a des divergences, les deux extrêmes présentent leur motivations
  - On réitère l'opération jusqu'à convergence
  - Les éléments de même taille sont groupés ensemble pour valider leur valeur relative ***(triangulation)***

Planning Poker

Planning Poker

Quelques bonnes pratiques:

- Spécifier un intervalle de temps par tour de table (ex.: 2min max) et le chronométrer
- Éviter d'influencer l'estimation avec des "Je pense que ce sera vite fait" ou "Ca risque de prendre des semaines… si pas des mois"
- Ne jamais prononcer son estimation tant que tous les autres membres n'ont pas estimé

Planning Poker

Quelques bonnes pratiques:

- Toujours obtenir un consensus plutôt qu'une moyenne
- Ceux qui votent sont ceux qui feront le boulot (le Product Owner ne vote pas)

Estimer les éléments

Emergent

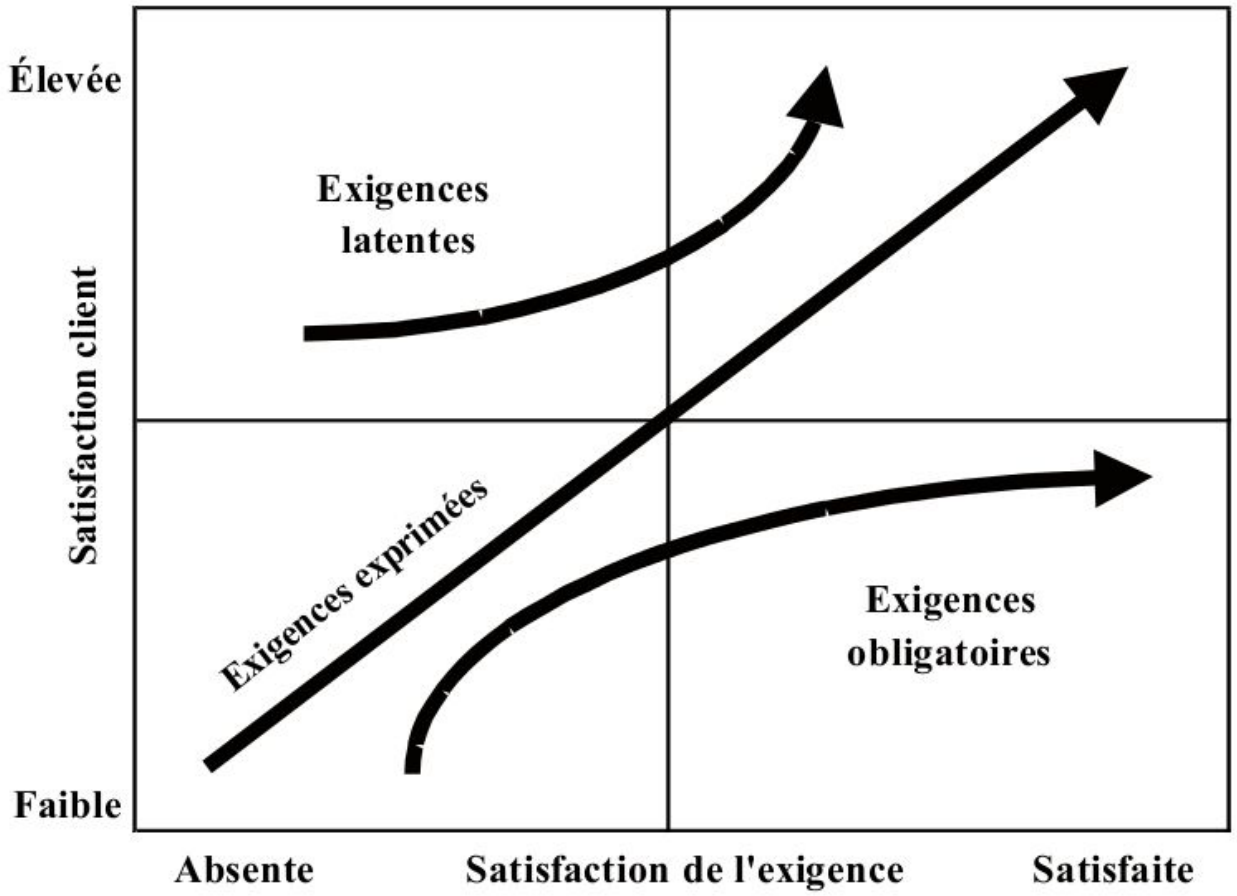
- Le contenu du product backlog change continuellement:
  - De nouveaux éléments sont ajoutés
  - Des éléments sont modifiés
  - Des éléments sont supprimés

Priorisés

- Valeur apportée aux clients/utilisateurs
  - « Quel est le bénéfice financier à développer cette fonctionnalité ? »
- Coût de développement estimé
  - « Quel est le coût de développement ? De maintenance ? De formation des utilisateurs ? »
  - « Quel est le coût d'un changement ? »
- Opportunité d'apprentissage pour l'équipe
  - « L'implémentation de cette fonctionnalité nous permet-elle d'apprendre ou de développer de nouvelles compétences ? »
  - « Cette fonctionnalité va-t-elle améliorer la productivité de mon personnel ? »

Priorisés

- Satisfaction du client: Modèle Kano



Priorisés

- Satisfaction du client: Modèle Kano

Fonctionnalité n

Si cette fonctionnalité était présente, que ressentiriez-vous ?	*« Ça me ferait plaisir »*	
	« Ce serait un minimum »	X
	« Je n'ai pas d'avis »	
	« Je l'accepterais »	
	« Ca me dérangerait »	

Si cette fonctionnalité n'était pas présente, que ressentiriez-vous ?	*« Ça me ferait plaisir »*	
	« Ce serait un minimum »	
	« Je n'ai pas d'avis »	
	« Je l'accepterais »	X
	« Ca me dérangerait »	

		Question dysfonctionnelle				
		Plaisir	Minimum	Neutre	Acceptation	Dérangement
Question fonctionnelle	Plaisir	?	L	L	L	E
	Minimum	R	I	I	I	O
	Neutre	R	I	I	I	O
	Acceptation	R	I	I	I	O
	Dérangement	R	R	R	R	?

O Obligatoire **R** "Reverse" (Inversée)

E Exprimée **?** Incertaine

L Latente **I** Indifférente

Priorisés

- Satisfaction du client: Modèle MoSCoW
  - **MUST**
    - Describes a requirement that must be satisfied in the final solution for the solution to be considered a success.
  - **SHOULD**
    - Represents a high-priority item that should be included in the solution if it is possible. This is often a critical requirement but one which can be satisfied in other ways if strictly necessary.
  - **COULD**
    - Describes a requirement which is considered desirable but not necessary. This will be included if time and resources permit.
  - **WON'T**
    - Represents a requirement that stakeholders have agreed will not be implemented in a given release, but may be considered for the future. (note: occasionally the word "**Would**" is substituted for "Won't" to give a clearer understanding of this choice).

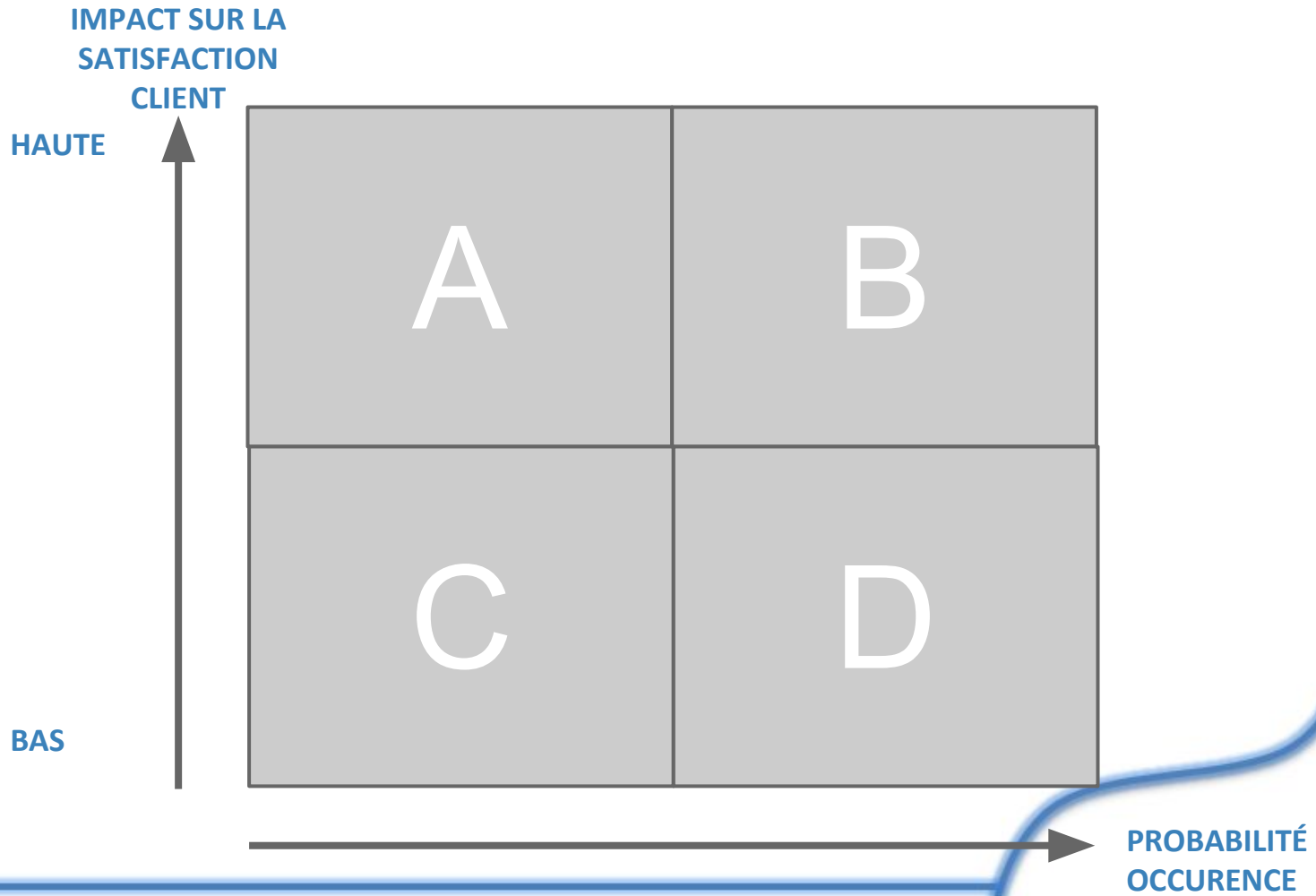
Priorisés

- Valeur apportée aux clients/utilisateurs
  - « Quel est le bénéfice financier à développer cette fonctionnalité ? »
- Coût de développement estimé
  - « Quel est le coût de développement ? De maintenance ? De formation des utilisateurs ? »
  - « Quel est le coût d'un changement ? »
- Opportunité d'apprentissage pour l'équipe
  - « L'implémentation de cette fonctionnalité nous permet-elle d'apprendre ou de développer de nouvelles compétences ? »
  - « Cette fonctionnalité va-t-elle améliorer la productivité de mon personnel ? »

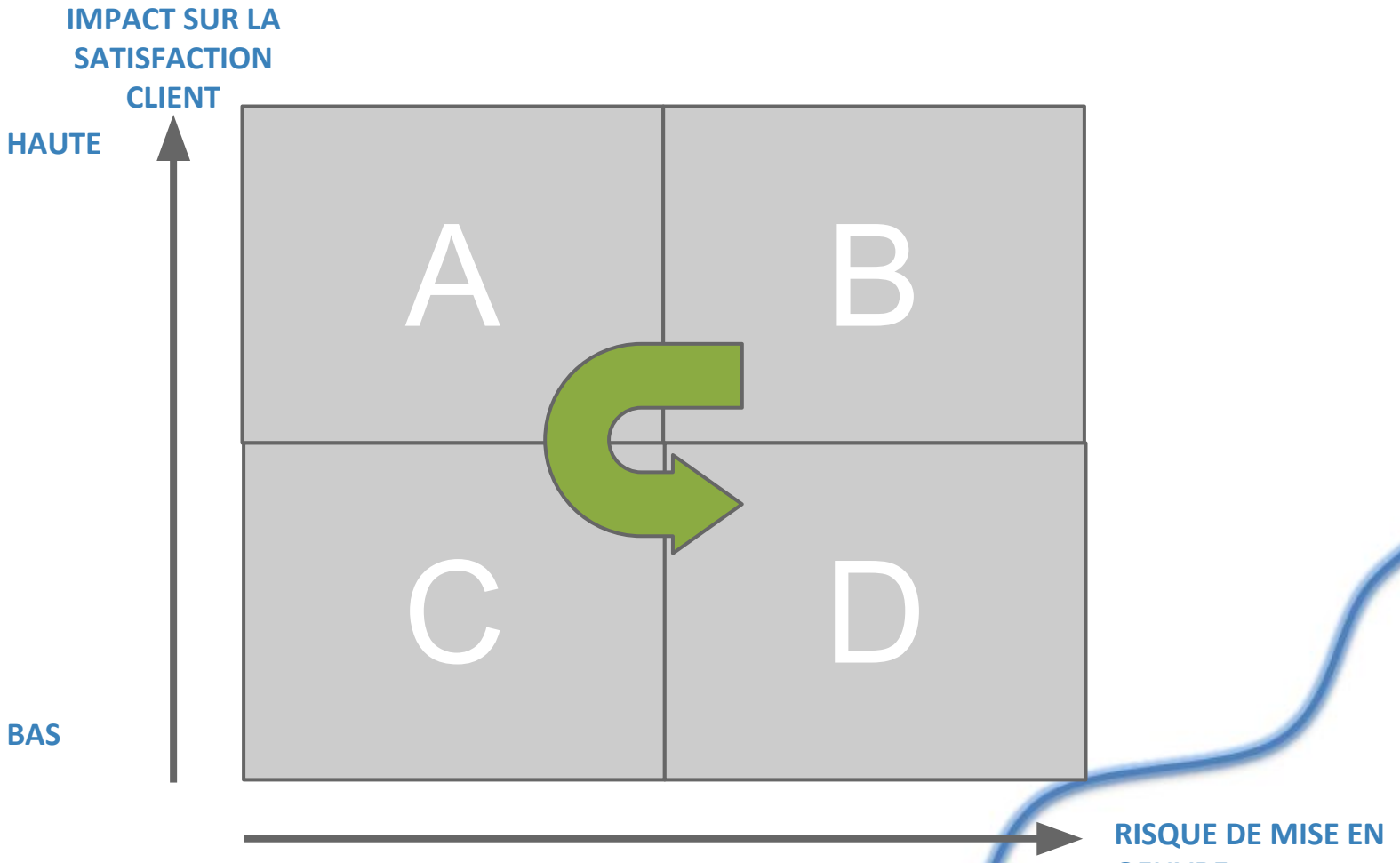
Priorisés

- Le risque de développement
  - « Cette fonctionnalité nous expose-t-elle à davantage de risques ? Ou, au contraire, nous permet-elle de nous confronter au risque ? »
  - « Cette fonctionnalité va-t-elle impacter les processus métier ? »
  - « Cette fonctionnalité va-t-elle susciter de la frustration ou de la résistance ? »
  - « Quel est le préjudice si cette fonctionnalité n'est pas implémentée ? »

Priorisés - Analyse du risque



Priorisés - Analyse du risque



Exigences non-fonctionnelles

- Ex.: performance, robustesse, utilisability...

- Influence:
  - Interface utilisateur
  - Architecture
  - Choix technologiques
  - ...

Exigences non-fonctionnelles

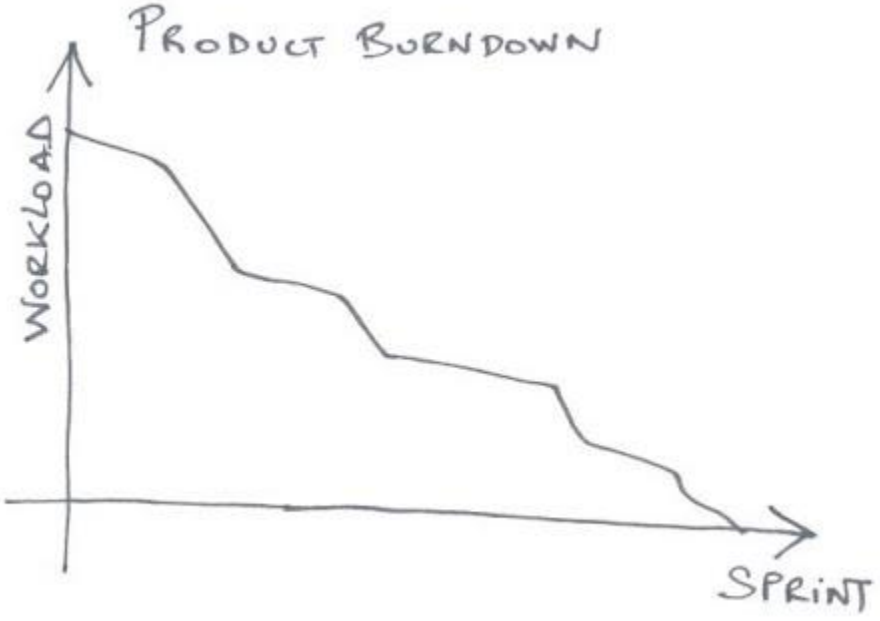
- Description sous forme de contraintes:
  - Ex.: Le système doit pouvoir répondre à n'importe quelle requête en moins d'une seconde
- Distinction entre les exigences globales et locales
  - Globales: doivent être définie avec la vision du produit
  - Locales: peut être définie et attachée avec la user story affectée

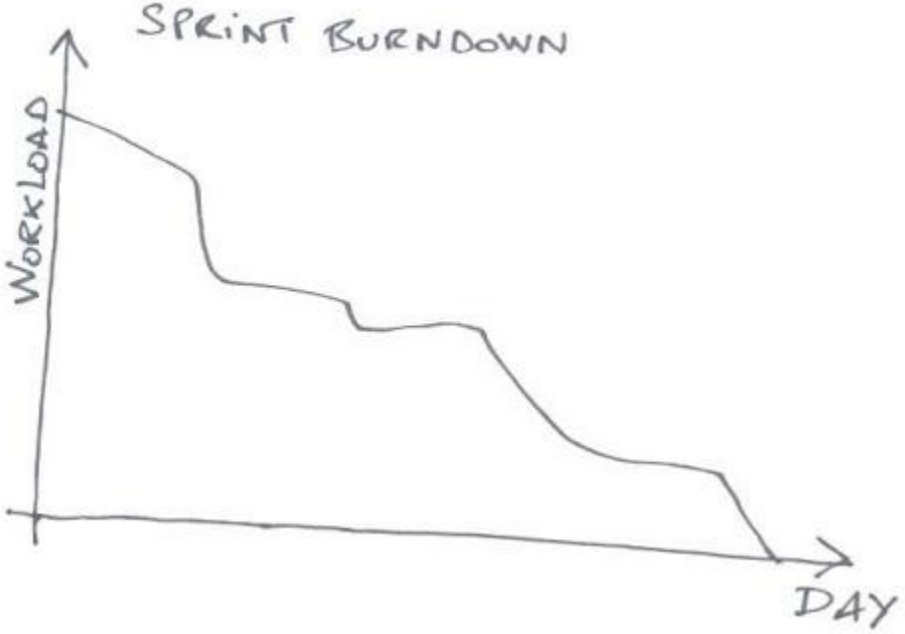
Etendue du Product Backlog dans de larges projets

- Utilisé un seul Product Backlog
  - Permet d'éviter les problèmes de synchronisation
- Préparer les éléments pour les 2 à 3 Sprints suivants
  - Les nombres d'éléments détaillés est donc plus important
- Proposer différentes vues pour les différentes teams SCRUM

Itération 5

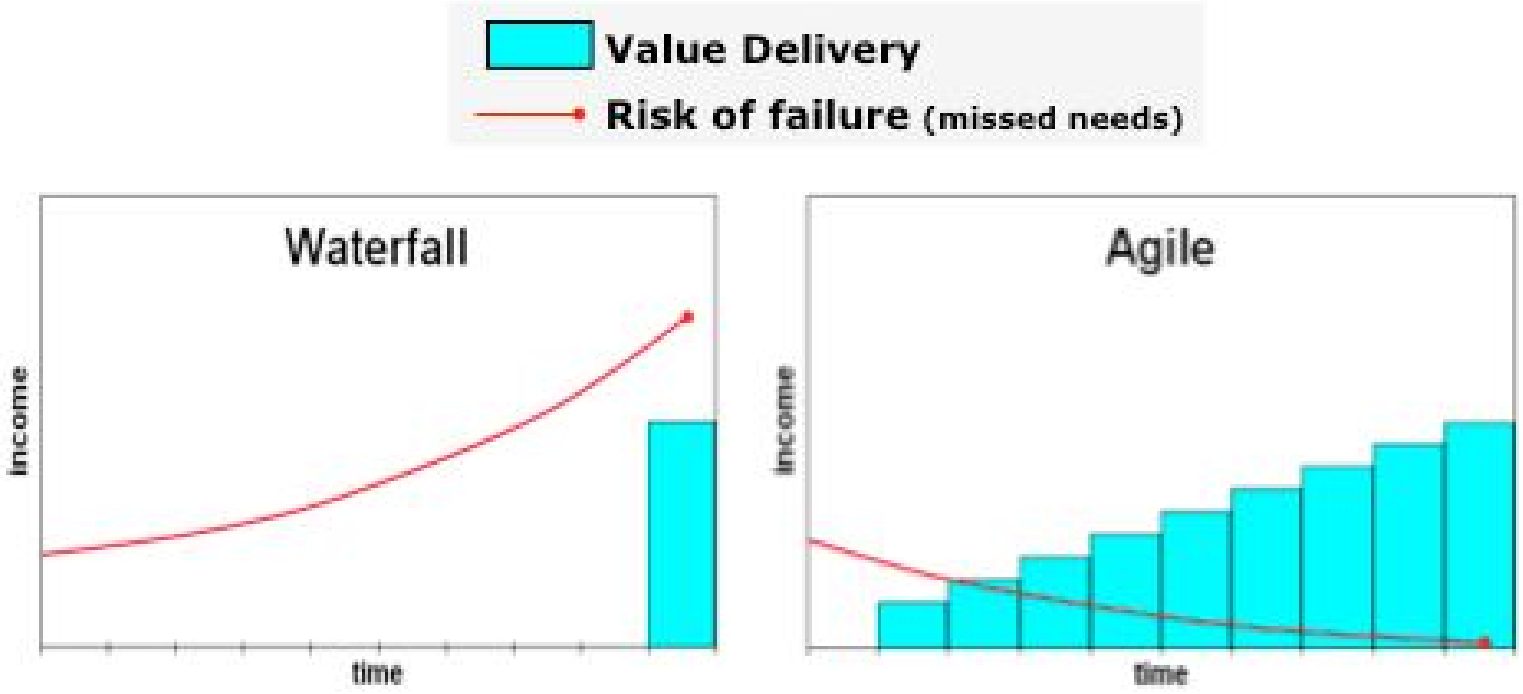
… ou comment planifier les Releases et les Sprints



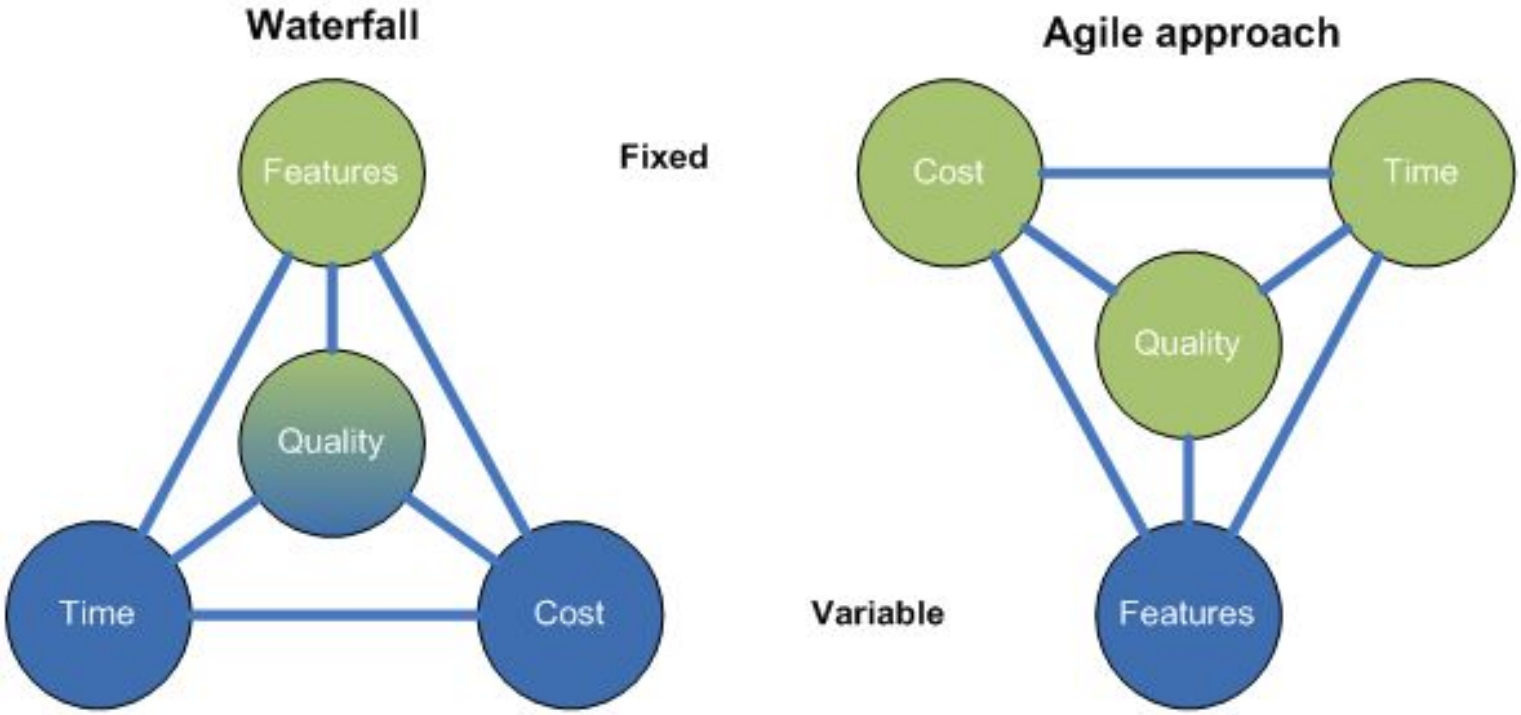


Les Releases

Avantages de releases fréquentes



Les Releases



La problématique de la planification

Pourquoi planifier:

- Outil d'aide à la décision
- Outil de pilotage
- Support de communication

- **Démarche traditionnelle:** élaboration d'un plan et recherche perpétuelle de la conformité
- **Démarche Agile:** adaptation au changement

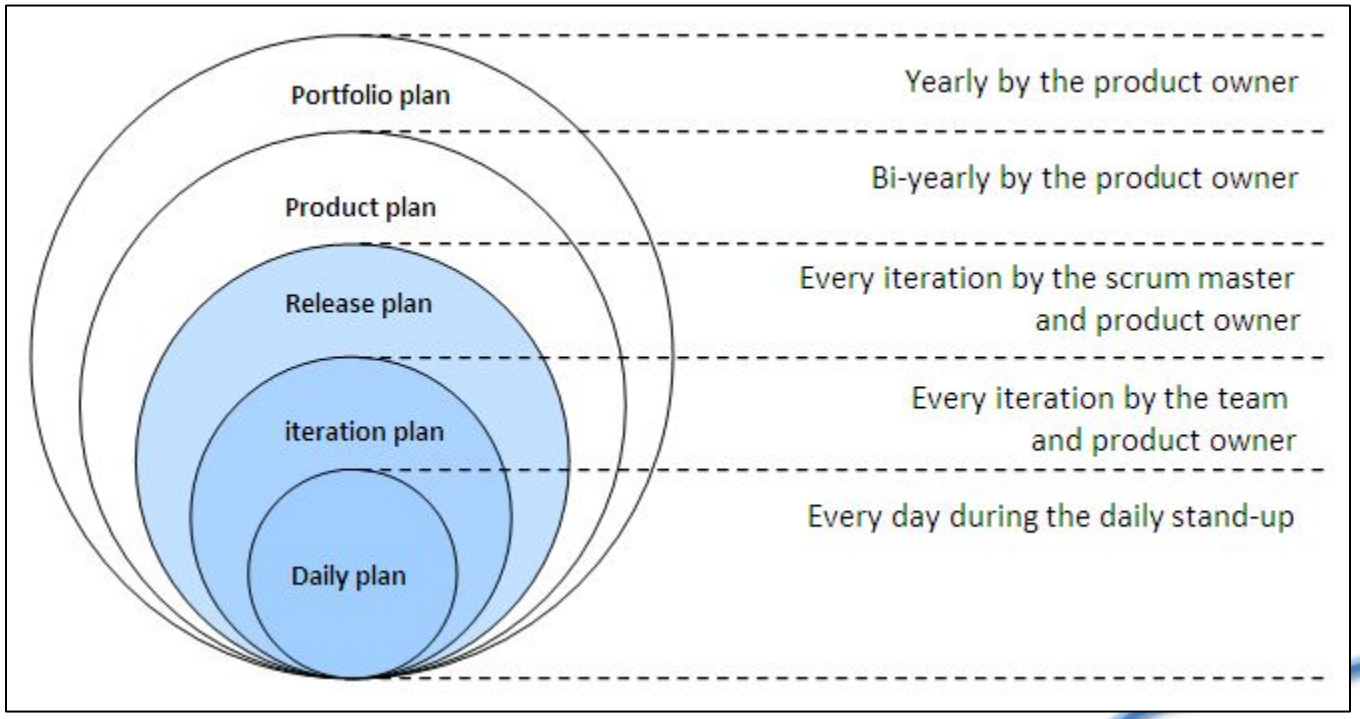
La problématique de la planification

- Planification au fil de l'eau
  - Acceptation de l'incertitude
  - La fiabilité des estimations augmente avec les enseignements des itérations précédentes

5 niveaux de planification

- Etablissement de la vision
  - Livraison finale
- Etablissement d'une roadmap
  - Les différentes releases
- Planification d'une release
  - Les différents Sprint aboutissant à la release
- Planification d'un SPRINT
  - Les différentes user stories à réaliser durant cette itération
- SCRUM
  - Réajustement du planning

Planning Onion



Release Plan

Major Feature	Sprint 1	Sprint 2	Sprint 3	Sprint 4 to 6	Quarter 2	Quarter 3	Year 2
Authen-tication	Login Screen		Security Questions		SSO Integration with Partner Sites		
	SSL Encryption						
Master	Product Master						RFID Implemen-tation
	Rate Master						
Order Entry		Product Selection				Recom-mended Products	Sponsored Look-ups
		Product Preview	Product Comparison				
				Product Review			
Checkout		Checkout		Coupons			Reward Points
				Order Track			

Estimer les durées

- Principe de **vélocité** *v*
  - La somme des story points qu'une équipe est capable de développer durant une itération

- Valeur hypothétique au début du projet et s'affine au fur et à mesure des itérations

- Après une première itération, on peut estimer une durée réaliste du projet
  - Durée du projet = (Somme SP) / *v*

- Estimer la vélocité initiale
  - Données historiques
  - Après la première itération

Les Releases

- Trois variables de décision à considérer pour planifier une release
  - **Temps**

    La date de lancement est-elle fixe?
  - **Coûts**

    Le budget de développement est-il fixe?
  - **Fonctionnalités**

    Un certain niveau de fonctionnalité doit-il être assuré pour une release?
- Fixer les trois variables est impossible

Les Releases

- Fixer les **fonctionnalités**
  - Ne prend pas en compte l'émergence des exigences
  - **Limite la capacité d'adaptation** de l'équipe aux besoins du client

Les Releases

- Fixer un **prix** (enveloppe budgétaire)
  - Arrêt du développement lorsque:
    - Toutes les fonctionnalités sont implémentées ou,
    - Le budget est consommé

Les Releases

- Fixer une **date** de lancement
  - Déterminer une *fenêtre d'opportunité*
  - Déterminer une "timebox" identique pour toutes les releases
    - Améliore l'innovation
    - Améliore les estimations (coûts, planning...)
    - La détermination du budget est direct si la composition de l'équipe reste stable (Hypothèse: travail est coût principal)

Les Releases

- Fixer un **prix** et un ensemble de **fonctionnalités**
  - A éviter
  - Séparer le contrat en deux projets consécutifs
    1. Créer la vision du produit et partiellement l'implémenter en 2 ou 3 sprints
    2. Créer un planning de releases sur la base du Product Backlog avec un budget réaliste

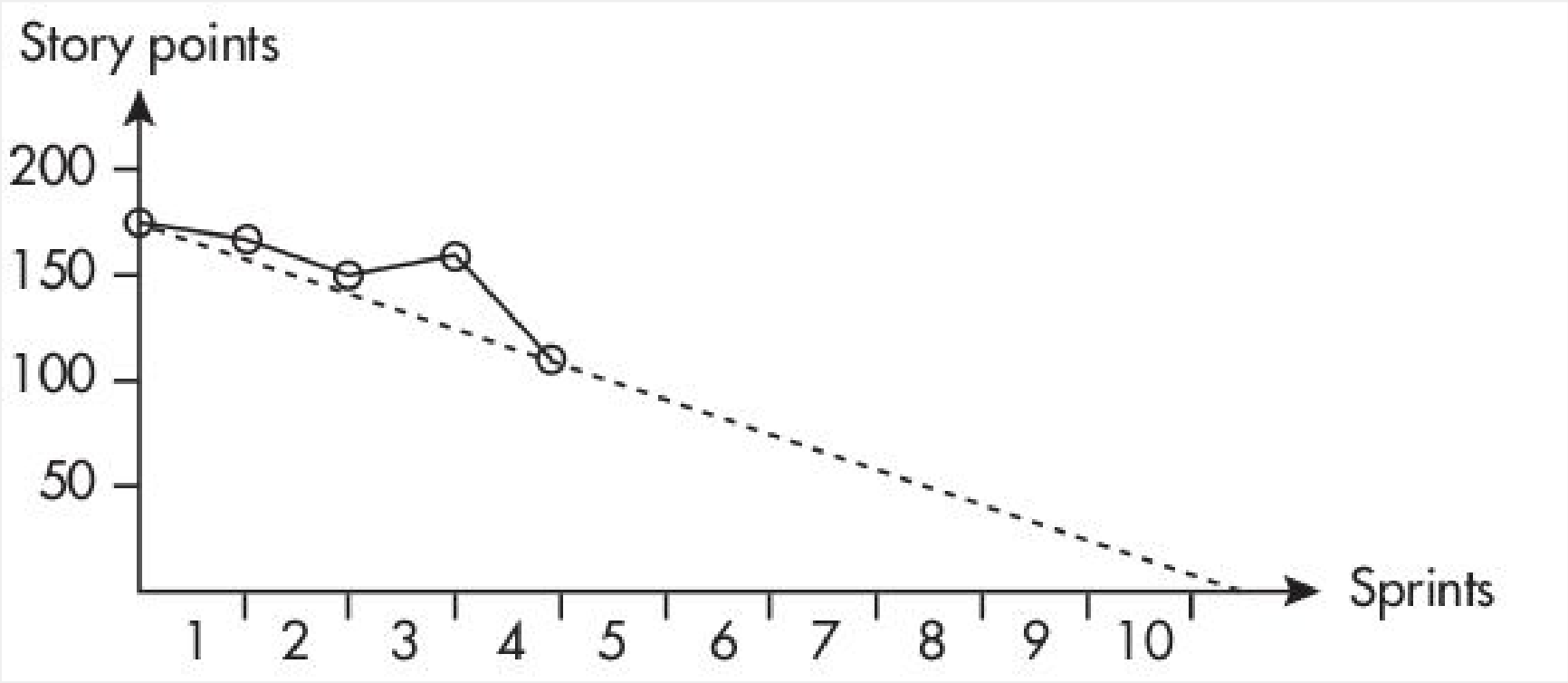
Les Releases

Release Plan:

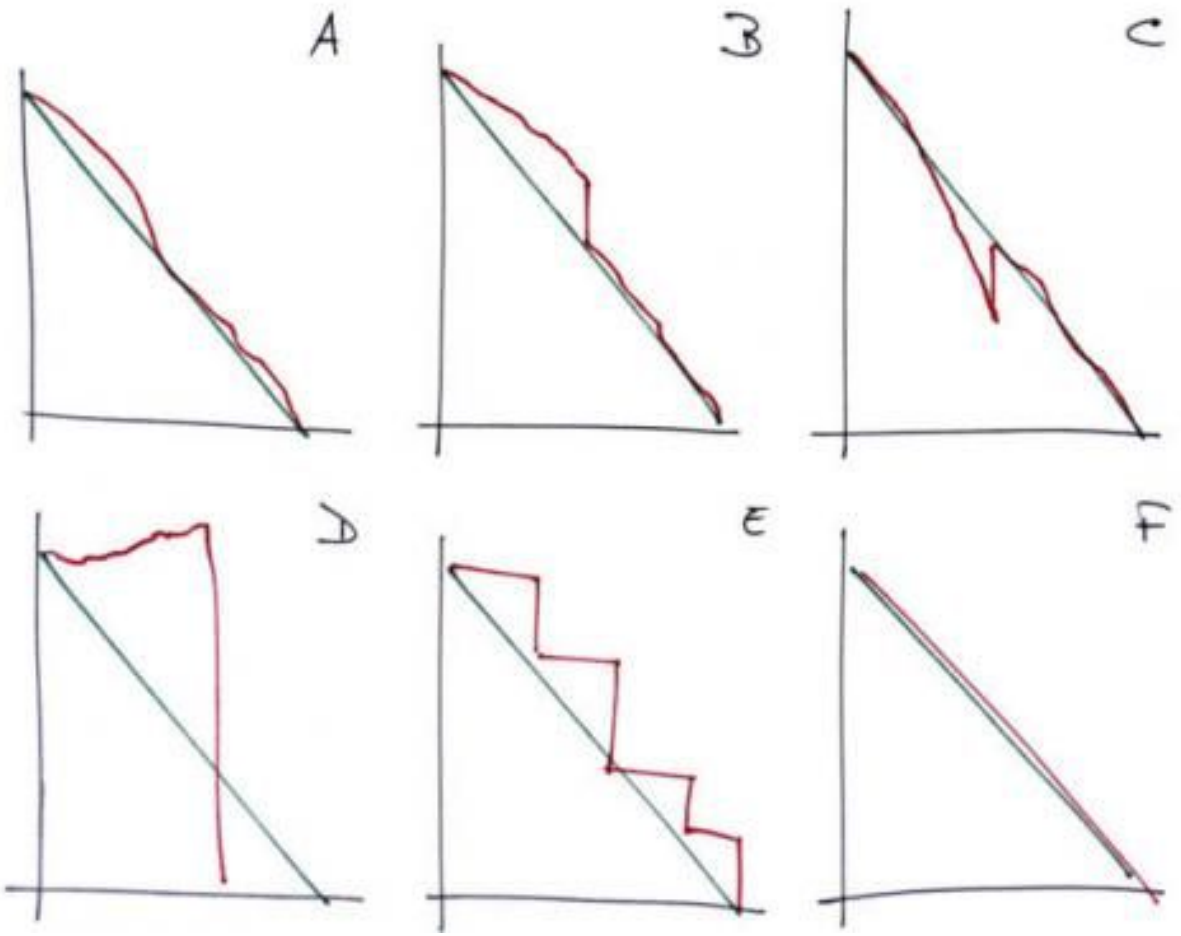
Two Tips to Help Product Owners with Release Planning

http://www.scrumalliance.org/community/articles/2008/june/two-tips-to-help-product-owners-with-release-plann

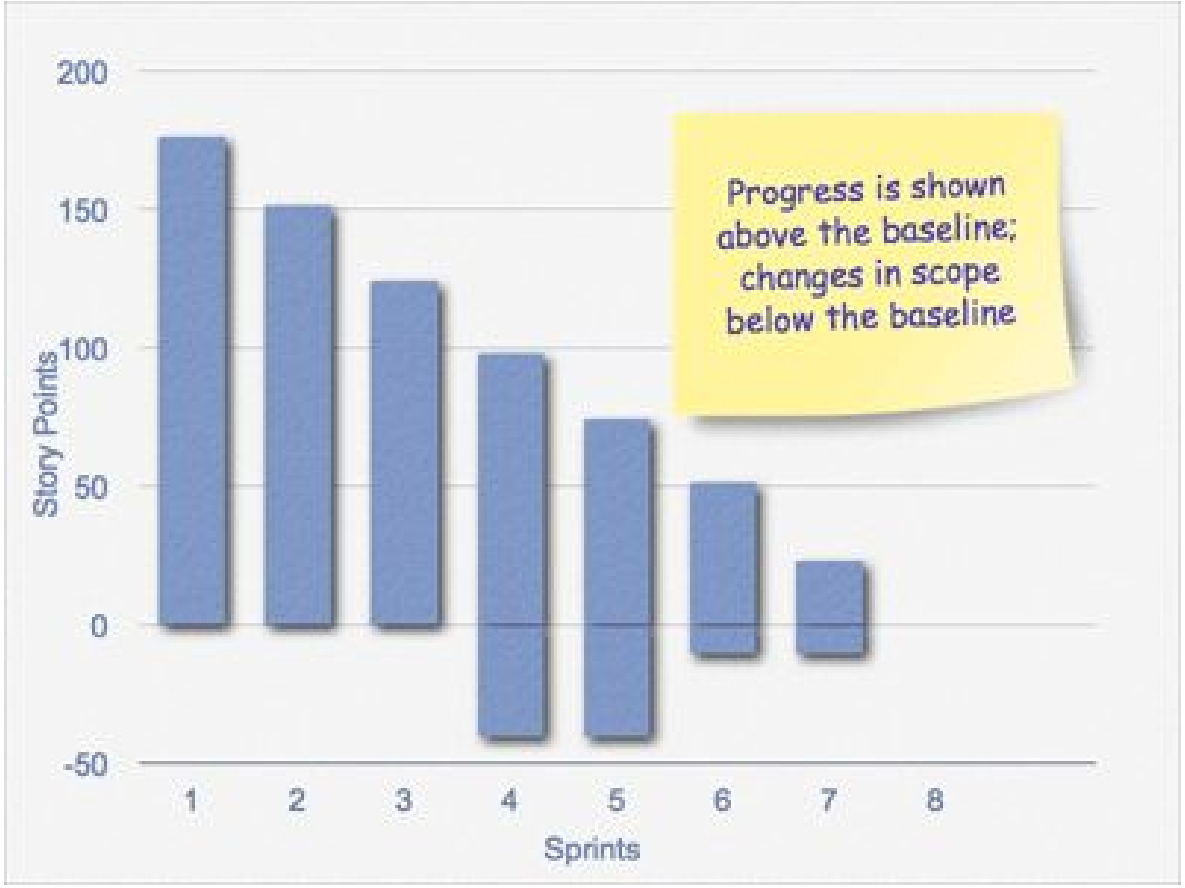
Les Releases
Burndown chart



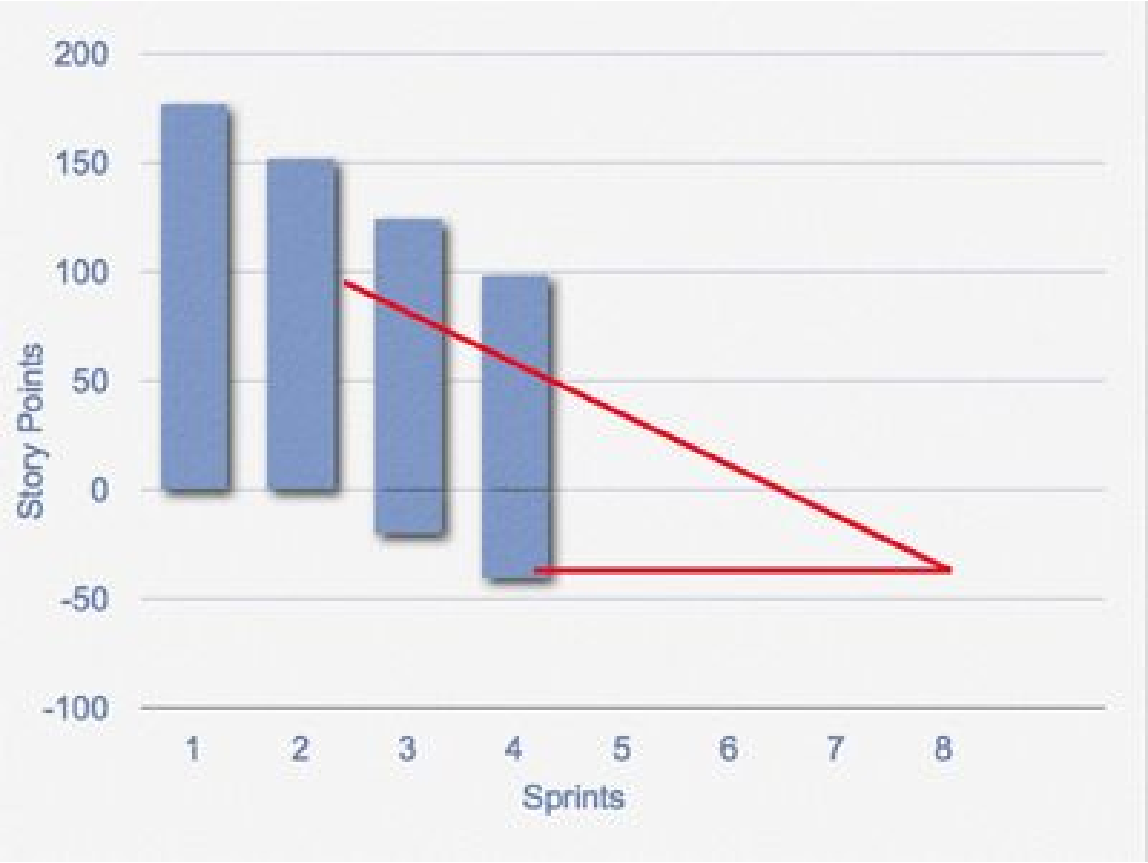
Les Releases
Burndown chart



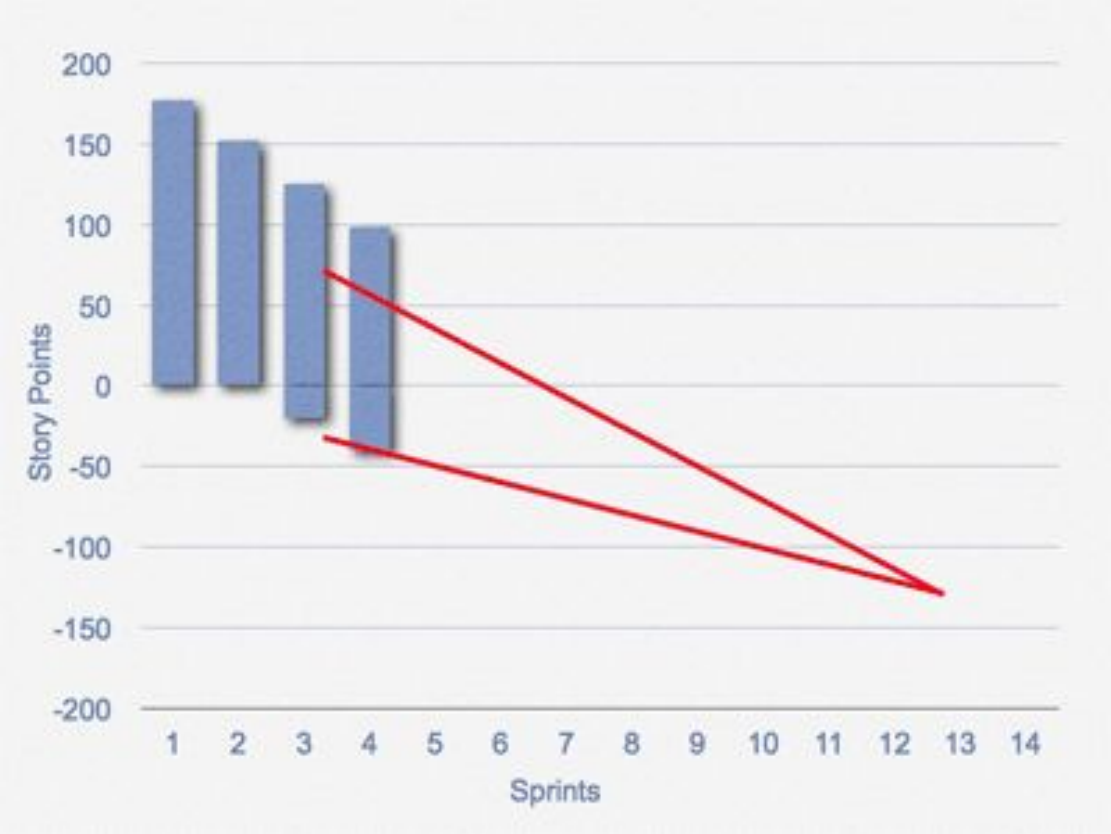
Les Releases
Bar Style Release Burndown chart



Les Releases
Bar Style Release Burndown chart

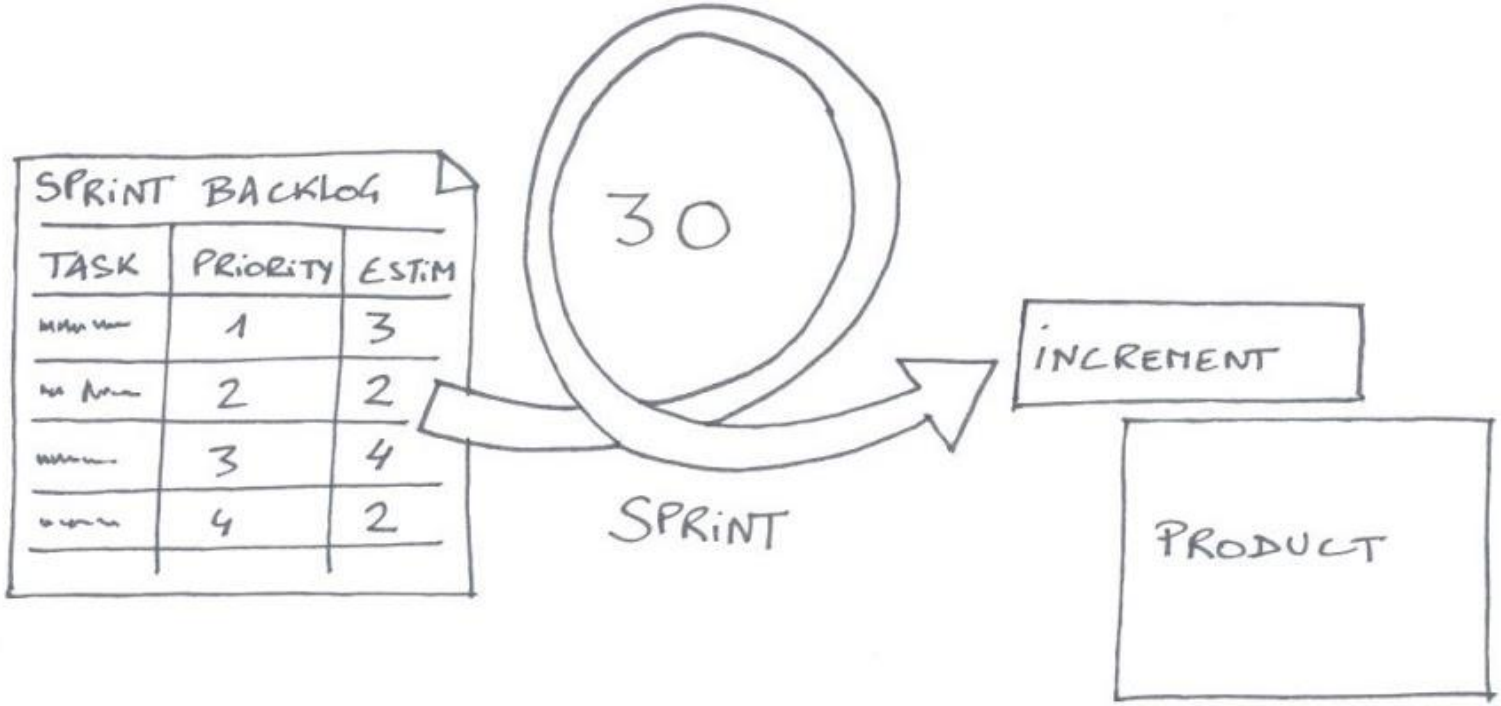


Les Releases
Bar Style Release Burndown chart



Les Releases
Erreurs courantes

- Pas d'utilisation du Release Burndown
- Big-Bang Release
- Délivrer un maximum de fonctionnalités au détriment de la qualité


SPRINT BACKLOG
TASK
PRIORITY
ESTIM
1
3
2
2
3
4
4
2
30
SPRINT
INCREMENT
PRODUCT

Les Sprints

Les Sprints
Définition du done

- Les critères sur la définition du *done* sont définis avant le premier Sprint par:
  - Le Product Owner
  - Le Scrum Master
  - L'équipe de développement

- Typiquement, un élément du product backlog est considéré comme achevé lorsque:
  - il est implémenté
  - il est testé
  - il est intégré
  - il est documenté

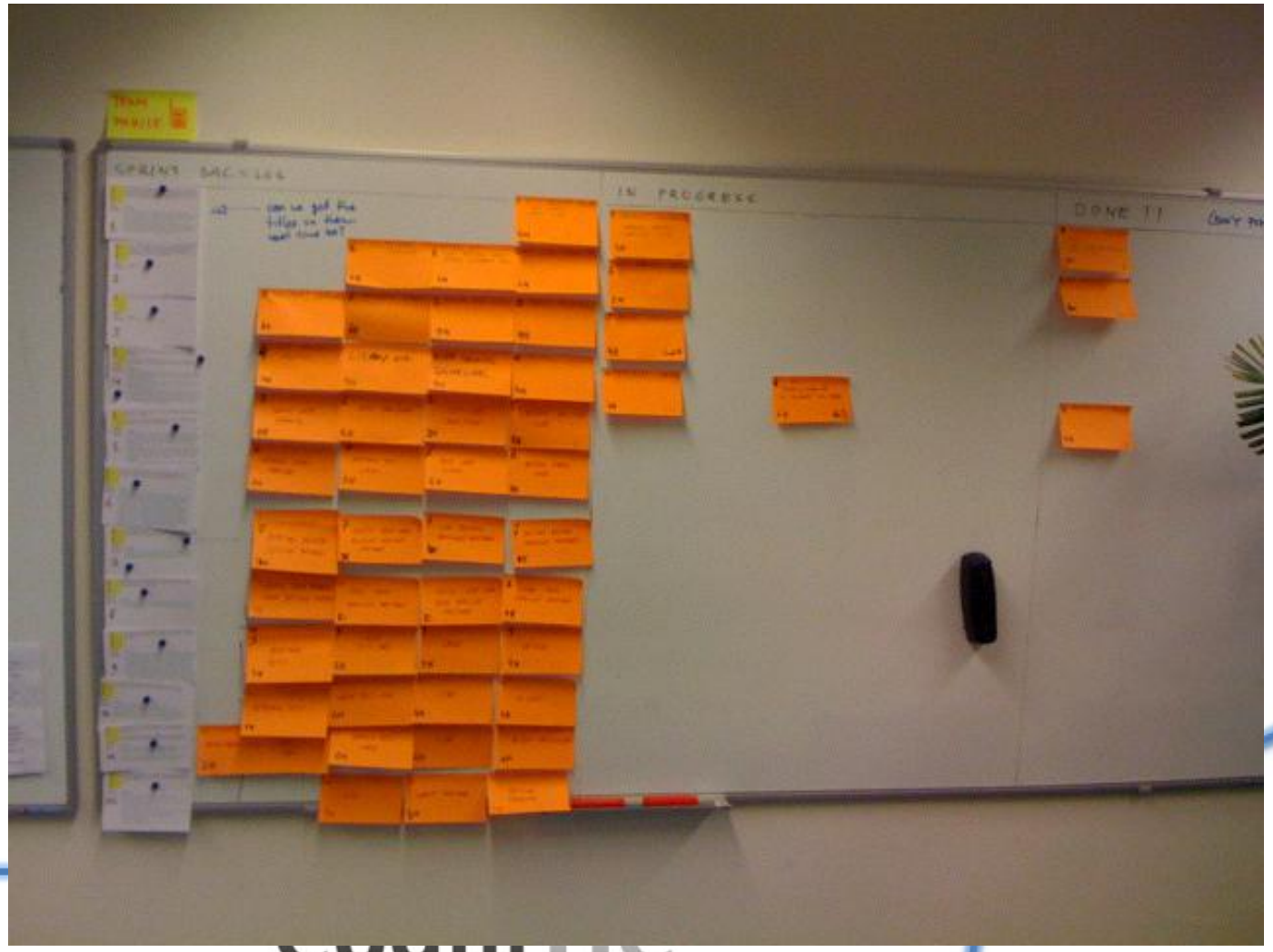
Les Sprints
Les SCRUM

- Permet à la Scrum Team de:
  - gérer son travail
  - soulever les freins à l'avancement du projet
- Le Product Owner peut répondre à certaines interrogations de l'équipe

Les Sprints
Les Sprints Backlog et Burndown

- Sprint Backlog
  - Contient les différents éléments à implémenter
- Sprint Burndown
  - Visualise la progression du Sprint
  - Visualise la probabilité d'atteindre les objectifs

Les Sprints
Sprint Backlog



Itération 6

http://www.contrat-agile.org/index.html

Pourquoi un contrat **agile**?

- Les deux parties ont chacunes des droits et des devoirs
- Contrat équilibré qui prend en compte les problématique des directions achats sans pervertir la philosophie agile

Structure du contrat

- Le corpus juridique
- Annexe 1 : Description de la méthodologie Scrum et des pratiques XP
- Annexe 2 : La vision partagée client/prestataire
- Annexe 3 : Estimations initiales, délais et charges du prestataire
- Annexe 4 : Plan de qualité de service V0 (PQS)
- Annexe 5 : Conditions particulières
- Annexe 6 : Annexe financière
- Annexe 7 : Product backlog V0

Corpus Juridique

- pose les bases contractuelles de la collaboration client/fournisseur
- spécifie l'esprit du contrat et de la collaboration agile
- spécifie également:
  - les définitions des termes utilisés,
  - liste les documents contractuels utilisés,
  - les acteurs nécessaires au bon fonctionnement du projet,
  - les garanties, obligations, conditions financières,
  - la propriété intellectuelle
  - etc.

Annexe 1

Description de la méthodologie Scrum et des pratiques XP

- décrit les principes agiles tels qu'ils doivent être respectés
- les parties peuvent s'y référer à tout moment

Annexe 2

La vision partagée client/prestataire

- Ce document fait foi pour les estimations préalablement faites en Story Points.

Annexe 3

Estimations initiales, délais et charges du prestataire

- En se basant sur la reformulation du cahier des charges du client, le prestataire fera une estimation globale du projet en termes de charge et de délai de réalisation.
- Celle ci, à l'issue du Sprint 0, pourront éventuellement être revues à la hausse comme à la baisse dans la limite prévue en Annexe 5 dans les conditions particulières.

Annexe 4

Plan de qualité de service V0 (PQS)

Pièce maîtresse du dispositif contractuel, permet de régler le fonctionnement opérationnel du projet. Il définit :

- L'organisation projet, les équipes impliquées avec les rôles et responsabilités de chacun
- Les mécanismes de pilotage du projet
- Les indicateurs clés de qualité et les éventuelles pénalités afférentes (maintien de la vélocité, véracité des estimations, qualité logicielle,etc).

Annexe 5

Conditions particulières

- Tout ce qui doit surseoir aux conditions générales ou les compléter doit figurer dans cette Annexe. Cette partie est à constituer en fonction du contexte projet.

Annexe 6

Annexe financière

- L'annexe financière donne les tarifs de la prestation ainsi que les modalités de facturation. On y trouve également le montant et la méthode de calcul de la pénalité forfaitaire définie dans le contrat.

Annexe 7

Product backlog V0

- Le Product Backlog évolue au fur et à mesure du projet. La dernière version, arrêtée lors des Comités de Pilotage après accord entre les parties, fait partie intégrante du Contrat en tant que Pièce Annexe 7.
- Ce mécanisme permet ainsi d'intégrer le changement sans nouvelle signature du contrat.

Itération 8

Considérations complémentaires

Environnement adéquat

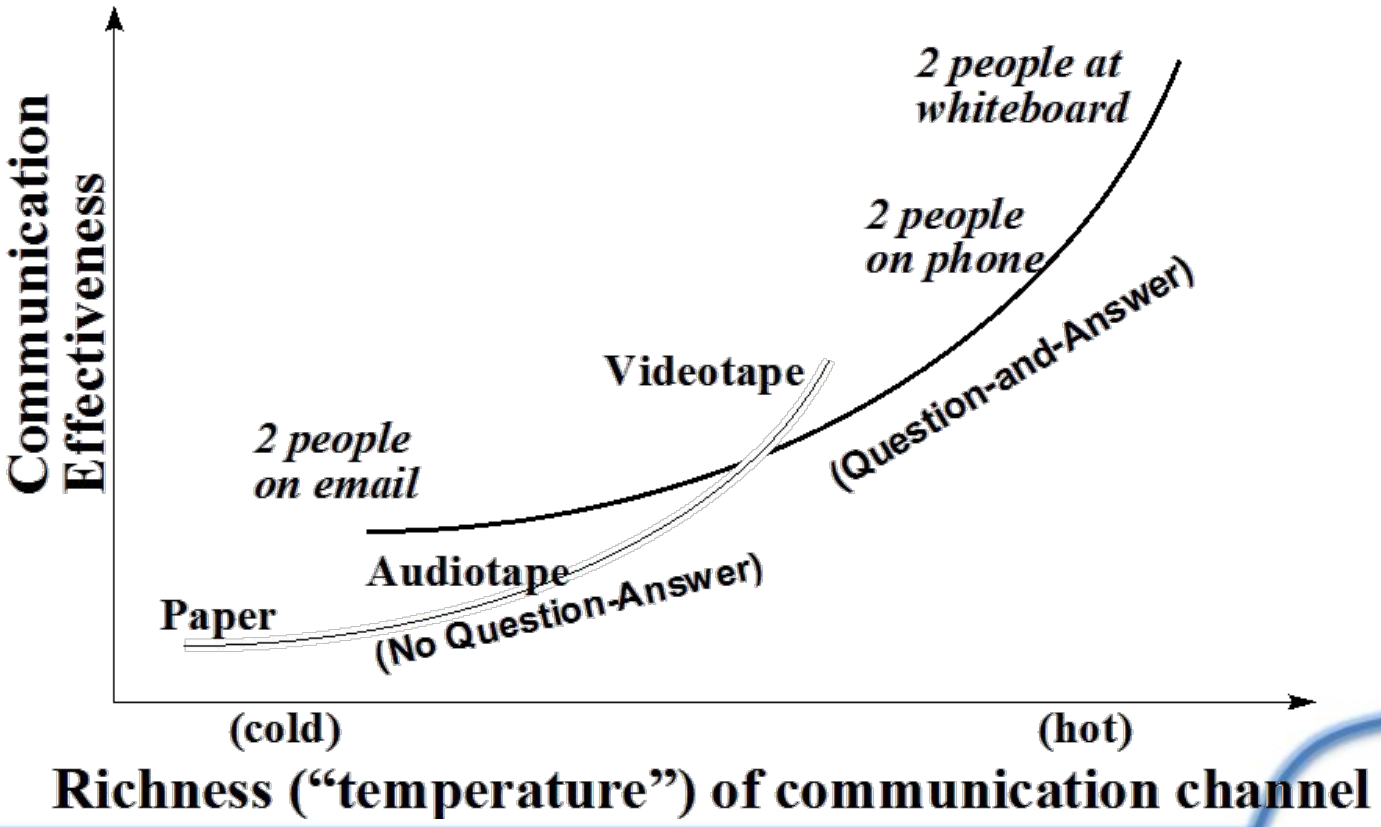
- Les membres de l'équipe Scrum sont les éléments centraux au succès d'un projet Agile

- Un environnement adéquat est nécessaire pour supporter leurs efforts

Rassembler l'équipe

- Autant que faire se peut, tous les membres de l'équipe devrait être localisé au même endroit
  - Communication en face-à-face
  - Daily scrum debout
  - Usage d'outils simples pour la communication
  - Obtention de clarifications
  - Partage de l'information
  - Demande d'aide pour une tâche
  - Supporter les autres membres pour une tâche

Rassembler l'équipe



Supprimer les distractions

Distraction	A faire	A ne pas faire
Multiple projets	S'assurer que l'équipe est allouée à 100% sur 1 seul projet	Ne pas fragmenter l'équipe entre différents projets, opérations de supports et autres
Multi-tâches	Garder l'équipe focalisée sur 1 seule tâche à la fois	
Sur-supervision	Laisser l'équipe s'auto-organiser	Ne pas interférer avec l'équipe de développement. Profiter du daily scrum pour contrôler l'avancement
Influences extérieures		Ne pas accepter les nouvelles demandes du clients quand le sprint a démarrer

Gérer des projets avec des équipes séparées

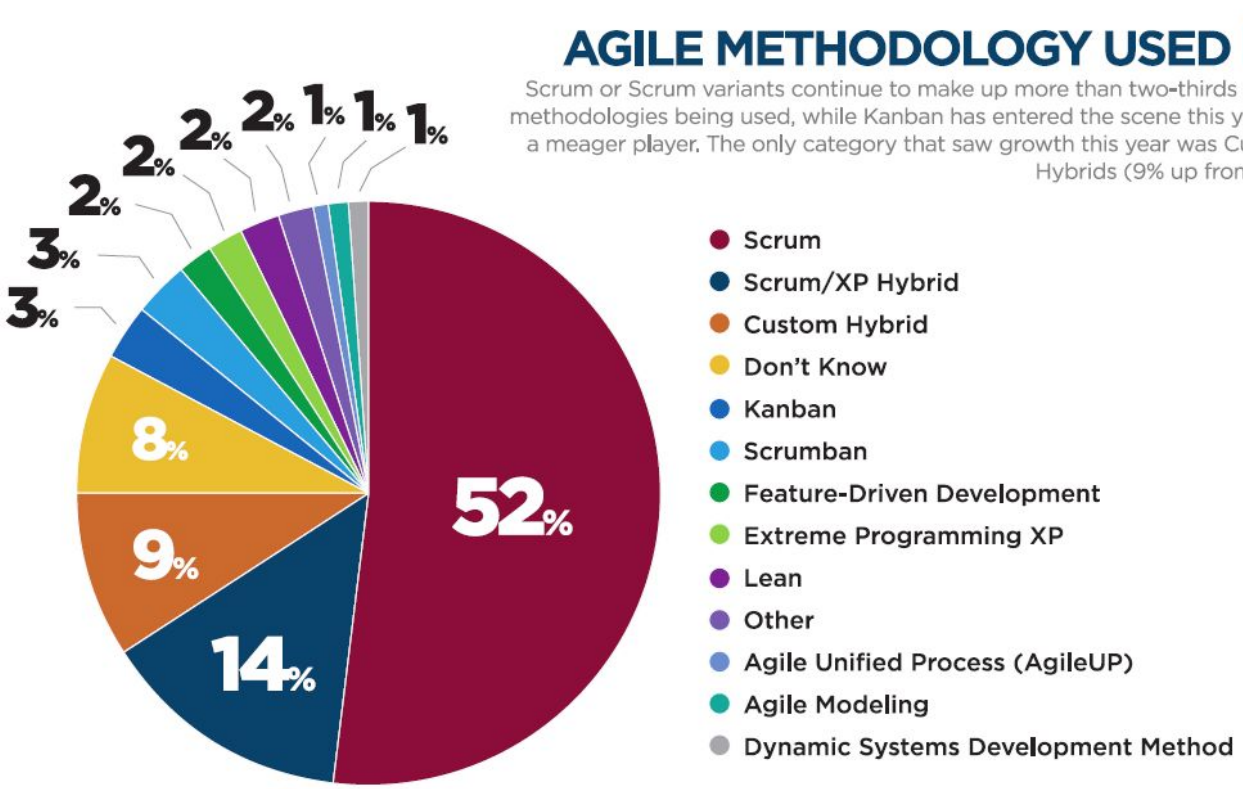
- Scrums isolés
  - Géographiquement séparées
  - Pas de communication/collaboration
  - Seulement intégration du code
- Distributed Scrum of Scrums
  - Géographiquement séparées
  - Relativement autonome
  - Daily scrum entre les Scrum Masters
- Integrated scrums
  - Chaque équipe dispose de membres dans de multiples lieux
  - Daily scrum entre les Scrum Masters

Distributed Scrum: Agile Project Management with Outsourced Development Teams, Jeff Sutherland, Anton Viktorov, Jack Blount

Gérer des projets avec des équipes séparées

- Utiliser de la vidéoconférence
  - Afin de simuler la conversation en face à face
- Si possible organiser une rencontre en personne au moins une fois avant de démarrer le projet
- Utiliser un outils de collaboration online
  - Afin de simuler le tableau
- Etre conscient des fuseaux
- Être flexible
- En cas de doute, “double check”

Scrum et les méthodes Agile



References

- *Agile Product Management with Scrum, Creating Products that Customers Love*, **Roman Pichler**, Addison-Wesley, 2010
- *Gestion de projet, Vers les méthodes Agiles*, **Véronique Messager Rota**, Eyrolles, 2007
- *The Scrum Guide, The definitive Guide to Scrum: The Rules of the Game*, **Ken Schwaber**, **Jeff Sutherland**, Scrum.org, 2011

Sources

- Illustrations - Emmanuel Chenu:
  - http://emmanuelchenu.blogspot.com/