

Mise en oeuvre d'une méthode Agile - Scrum



<http://www.sebastienchabal.fr/wp-content/uploads/2010/08/meleeBlack-Bleu.jpg>

(Ce cours suppose les principes d'XP connus)

Merci à tous ceux qui ont rendu leurs cours et exposés disponibles sur le web & dans les livres, voir Biblio. & refs dans les slides

M. Blay-Fornarino

Bibliographie

- ❧ Scrum et l'agilité des équipes de développement, NormandyJUG, Par Dimitri Baeli & Nicolas Giard
- ❧ The Zen of Scrum, Boris Gloger, David Koontz
- ❧ Scrum, état de l'art, François Potentier.
- ❧ eXtreme Programming & Scrum Practices, Embrace Change, Naresh Jain

Contrôler le Chaos ...

SCRUM

- ☛ Met l'accent sur la gestion de projet
- ☛ Met l'accent sur les caractéristiques livrées et l'ajustement selon les résultats
- ☛ L'objectif est de trouver un équilibre entre permettre au métier de changer d'approche et à l'équipe de développement de faire un travail de qualité à sa portée.

Backlog

TODO

DOING

DONE

valeurs de
SCRUM

organisations

Rôles

Réunions

Artefacts

Environnement

Conclusion

Backlog

TODO

DOING

DONE

valeurs de Scrum

organisations

Rôles

Réunions

Artefacts

Environnement

Conclusion

Scrum en 100 mots

- ❧ Scrum est un processus agile qui vise à produire la plus grande valeur métier dans la durée la plus courte.
- ❧ Un logiciel qui fonctionne est produit à chaque sprint (toutes les 2 à 4 semaines).
- ❧ Le métier définit les priorités.
- ❧ L'équipe s'organise elle-même pour déterminer la meilleure façon de répondre aux exigences les plus prioritaires.
- ❧ A chaque fin de sprint, tout le monde peut voir fonctionner le produit courant et décider soit de le livrer dans l'état, soit de continuer à l'améliorer pendant un sprint supplémentaire.

Les valeurs de SCRUM

- ❧ **Engagement.** Soyez prêt à vous engager sur un objectif. Scrum assure aux développeurs l'autorité dont ils ont besoin pour remplir leurs engagements.
- ❧ **Focus.** Faites votre travail. Concentrer tous vos efforts et vos compétences à faire le travail que vous vous êtes engagé à faire. Ne vous inquiétez pas d'autre chose
- ❧ **Transparence.** Scrum laisse tous les éléments d'un projet visibles à tous.
- ❧ **Respect.** Les individus sont façonnés par leurs antécédents et leur expérience. Il est important de respecter les différentes personnes qui composent une équipe.
- ❧ **Courage.** Ayez le courage de vous engager, d'agir, d'être ouvert et d'attendre du respect

Backlog

TODO

DOING

DONE

valeurs de SCRUM

organisations

Rôles

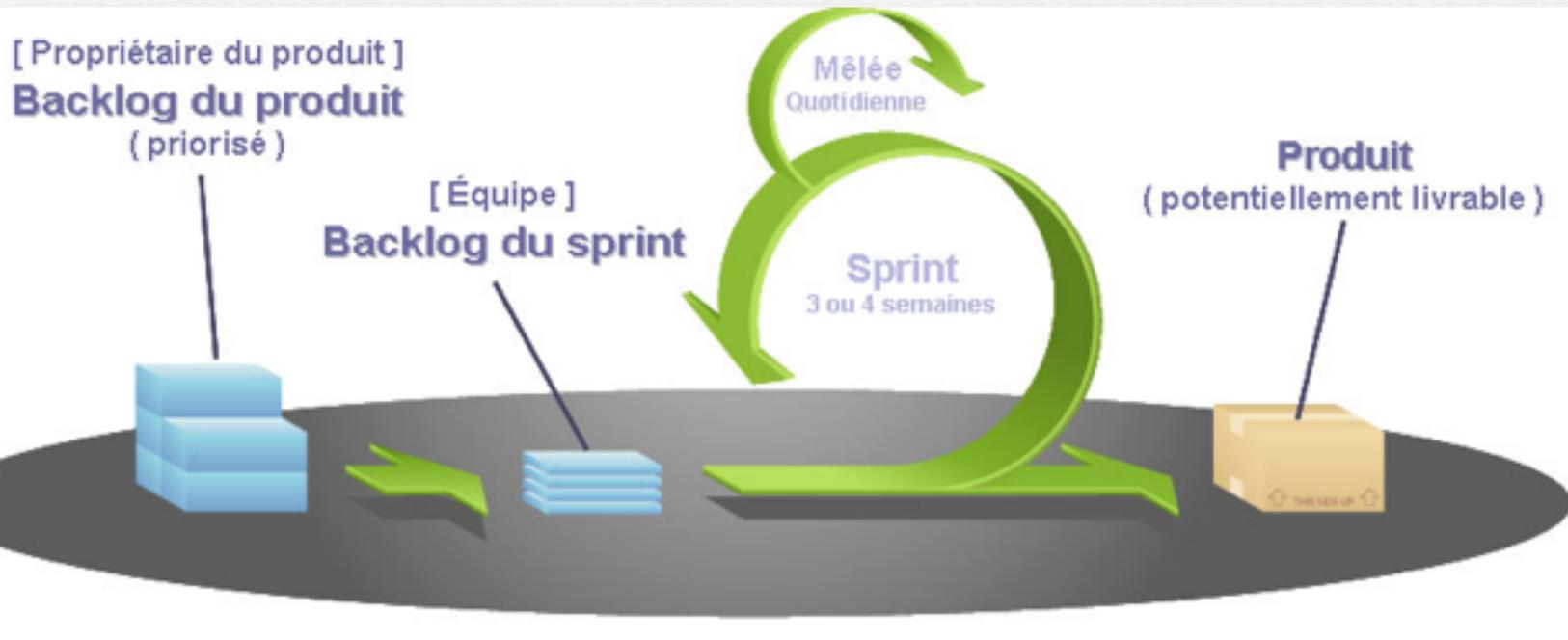
Réunions

Artefacts

Environnement

Conclusion

Scrum : processus général



<http://www.computure.net/fr/methodes/111-lesessentiel-de-scrum-suite>

Backlog

TODO

DOING

DONE

valeurs de SCRUM

organisations

Rôles

Réunions

Artefacts

Environnement

Conclusion

Caractéristiques de Scrum

Rôles

- Product Owner
- Scrum Master
- Equipe

Meetings

- Planification du sprint
- Scrum quotidien
- Revue de Sprint
- Rétrospective

Artefacts

- Backlog de produit
- Backlog de sprint
- Burndown Chart

Les Rôles : le Product Owner

- ❖ Définit les fonctionnalités du produit
- ❖ Choisit la date et le contenu de la livraison
- ❖ Responsable du retour sur investissement
- ❖ Définit les priorités dans le backlog en fonction de la valeur « métier »
- ❖ Ajuste les fonctionnalités et les priorités à chaque sprint si nécessaire
- ❖ Accepte ou rejette les résultats



PRODUCT OWNER

Les Rôles : le Scrum Master

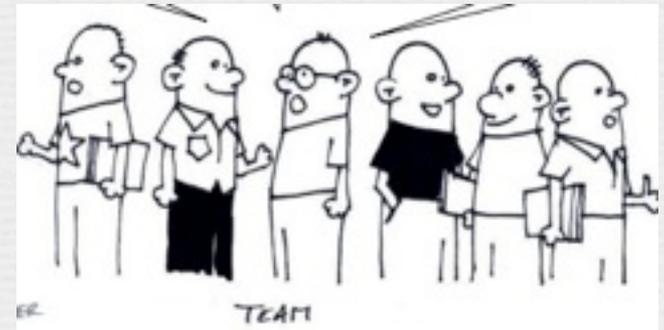
- ❧ Représente le management du projet
- ❧ Responsable de faire appliquer par l'équipe les valeurs et les pratiques de Scrum
- ❧ Résout des problèmes
- ❧ S'assure que l'équipe est complètement fonctionnelle et productive
- ❧ Facilite une coopération poussée entre tous les rôles et fonctions
- ❧ Protège l'équipe des interférences extérieures



Les Rôles : l'équipe

- De 5 à 10 personnes
- Regroupant tous les rôles
 - Architecte, concepteur, développeur, spécialiste IHM, testeur, etc.
- A plein temps sur le projet, de préférence
 - Exceptions possibles (administrateur, ...)
- L'équipe s'organise par elle-même
- La composition de l'équipe ne doit pas changer pendant un Sprint

Martin (2003): "The team is in it for the long term. They work hard, at a pace that can be sustained indefinitely. They conserve their energy, treating the project as a **marathon** rather than a **sprint**."



Backlog

TODO

DOING

DONE

Réunions

Artefacts

valeurs de scrum

organisations

Rôles

Environnement

Conclusion

Caractéristiques de Scrum

Rôles

- Product Owner
- Scrum Master
- Equipe

Meetings

- Planification du sprint
- Scrum quotidien
- Revue de sprint
- Rétrospective

Artefacts

- Backlog de produit
- Backlog de sprint
- Burndown Chart

Backlog

To Do

- Brands AMP webinar 7
- APM 3.0
- Big Weekly AMP Call
- Visit UrbanCode 3
- White paper 2.0 5
- Agile Cycle DEMO SCRIPT 5
- White paper 5
- Agile Cycle DEMO SET-UP 14
- TFS COMP All 14
- RTC COMP All 14
- Agile Cycle Cost 3
- BLOG #2 1
- BLOG #3 1
- BLOG #4 1
- BLOG #4 1

In Progress

Summer of Bob

- BLOG #1 1
- AMP Demo CA 14
- COMP MATRIX 7 3
- BLOG #1 1
- AMP Open SC 3
- White paper DEMO PREP 1.5
- White paper 4
- BLOG #2 1
- Texas 5
- BLOG #3 1
- Blog 2018 - Top 10 - why CI/CD? - Learn + build
- Blog 2018 - DevOps - the right way

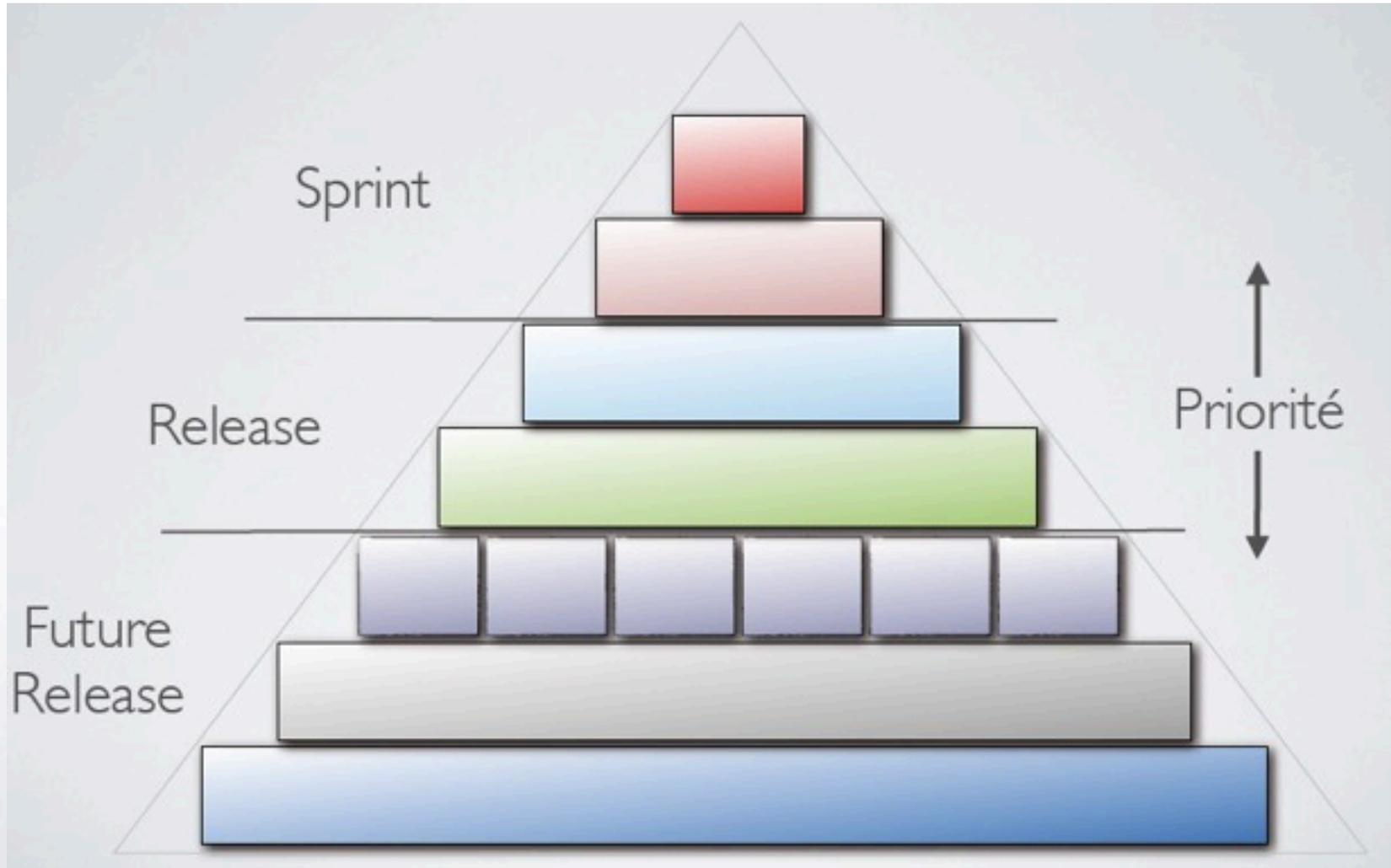
DONE

Academy 1	Training 6	White 2	White 8
HD Whiteboard 3	Agile Cycle Prep 1	Training 6	White 100
Kelly Demo 3	COMP MATRIX 3		

Q2

- Agile Cycle Training
- White paper on DevOps
- Sales Training
- White paper - / 100

Iceberg du product backlog



BACKLOG DE PRODUIT

- Les exigences du produit
- Toutes (Idées, fonctionnalités, Epic, Thème, etc.)
- Exprimées en User Stories
- Le PO le maintient organisé
- Toujours estimé et avec les priorités

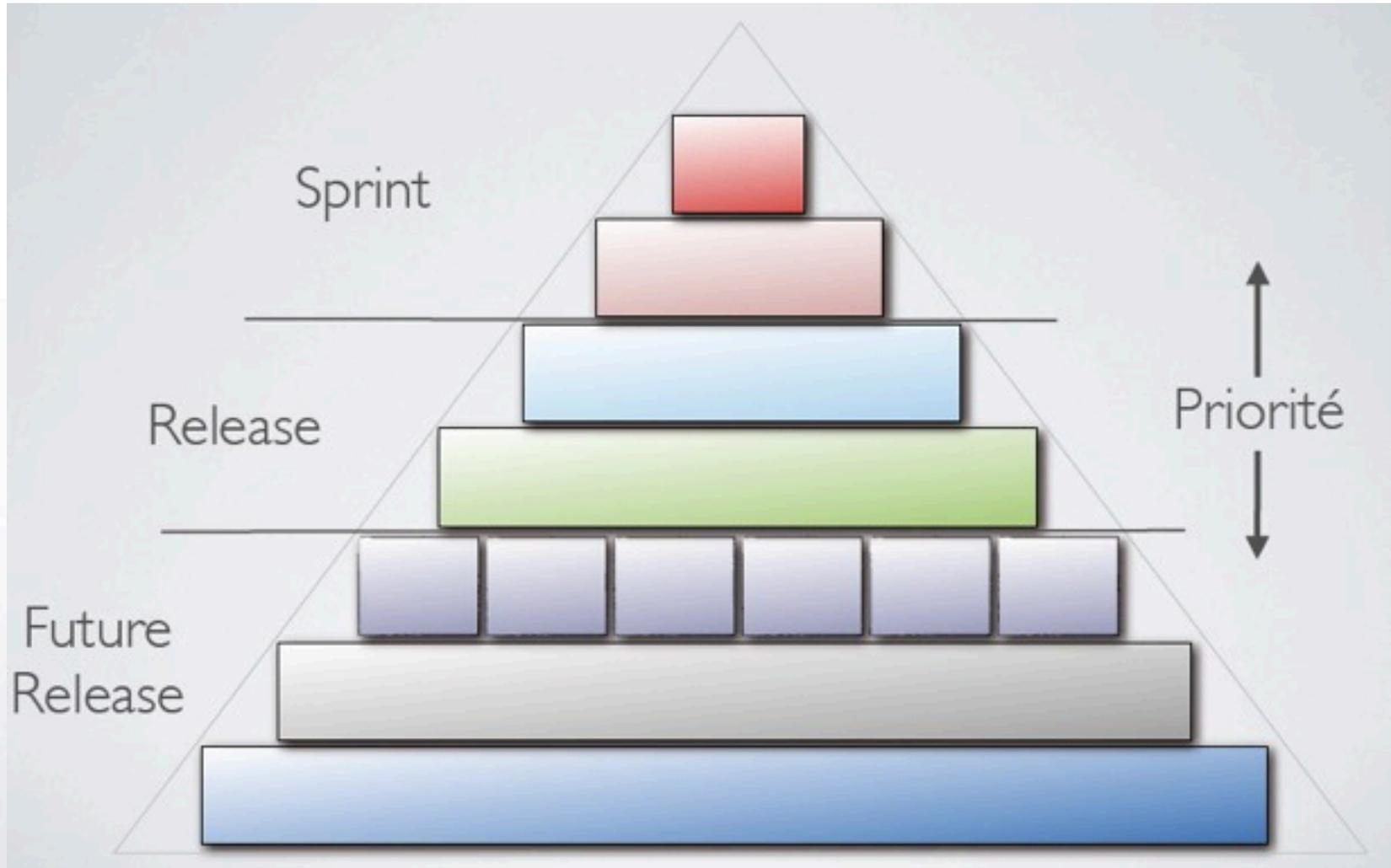
Les fonctionnalités sont décrites
sous forme de :
USER STORIES

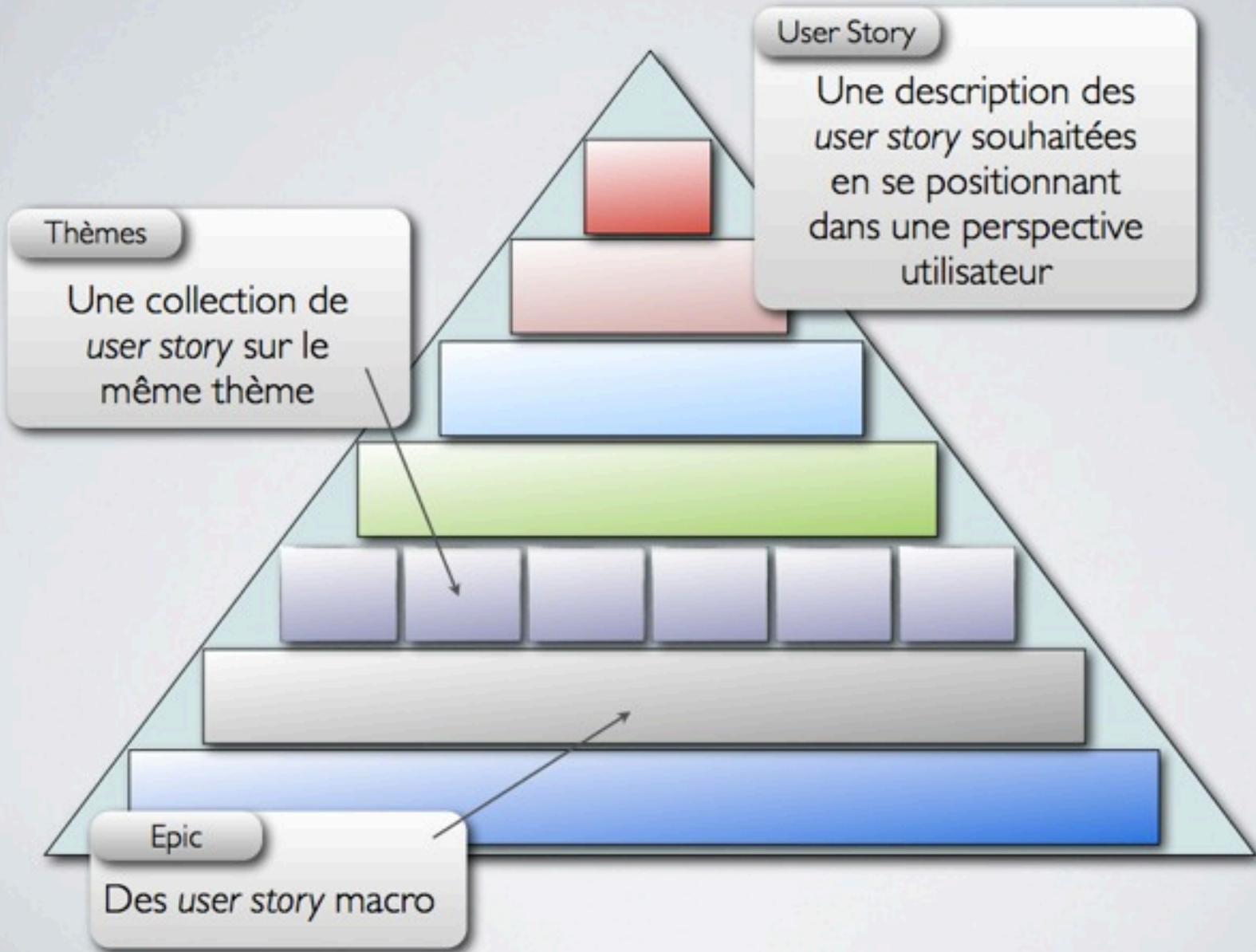
Je suis <rôle>,
Je veux <fonction>

Les fonctionnalités sont validées par des **Critères d'acceptation**

Je suis <rôle>,
Je veux <fonction>
et <un résultat>

Iceberg du product backlog





Thèmes
Une collection de *user story* sur le même thème

User Story
Une description des *user story* souhaitées en se positionnant dans une perspective utilisateur

Epic
Des *user story* macro

User stories / (Histoires d'utilisateur)

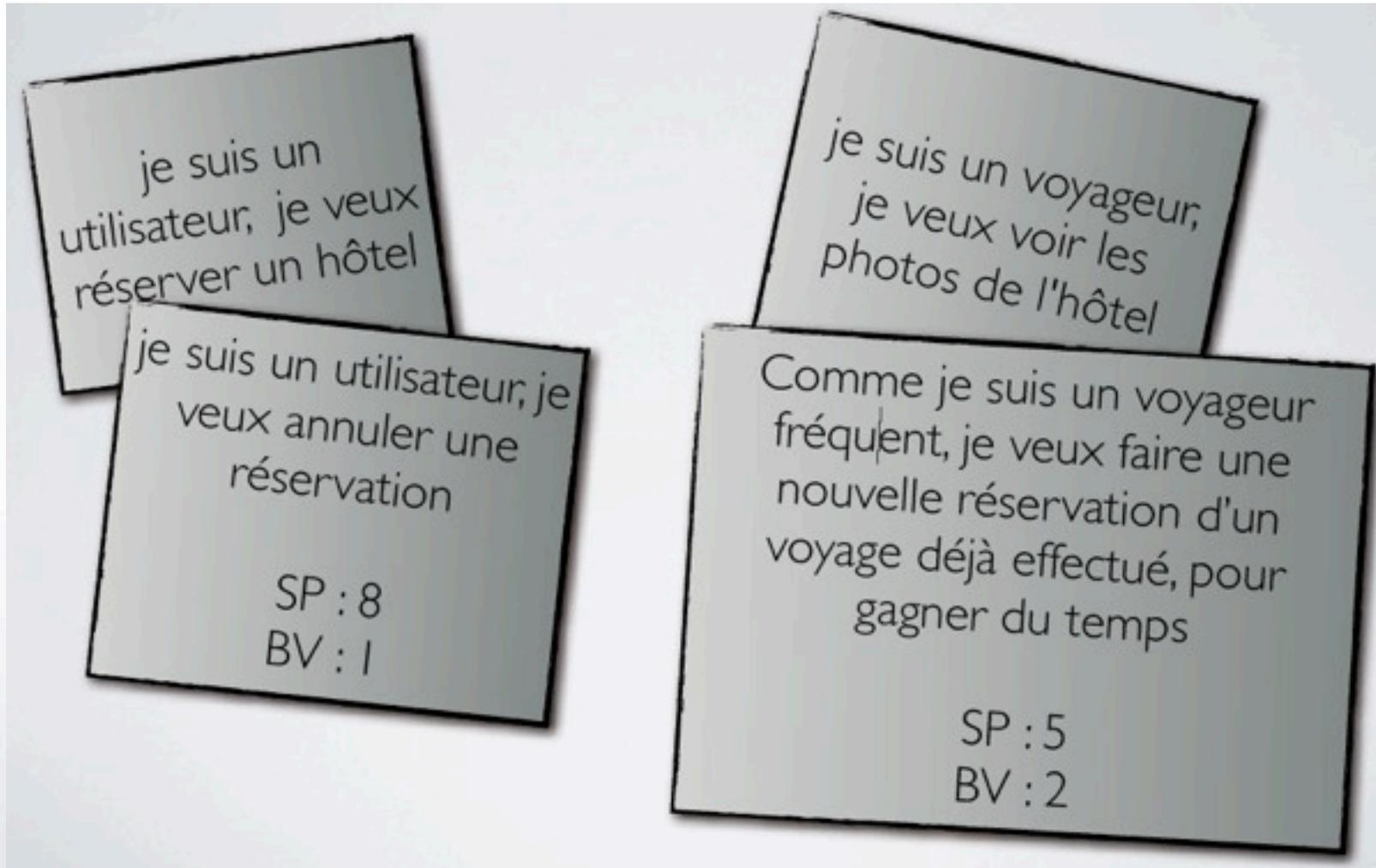
- Agile est associé à l'expérience utilisateur
- On raconte une histoire : User stories
- On montre notre histoire : Prototype

Recommandation

Utiliser ce masque

Je suis « Rôle de l'utilisateur »,
je veux « But »,
les « conditions » pour ...

EXEMPLE DEPUIS UN SITE DE VOYAGES

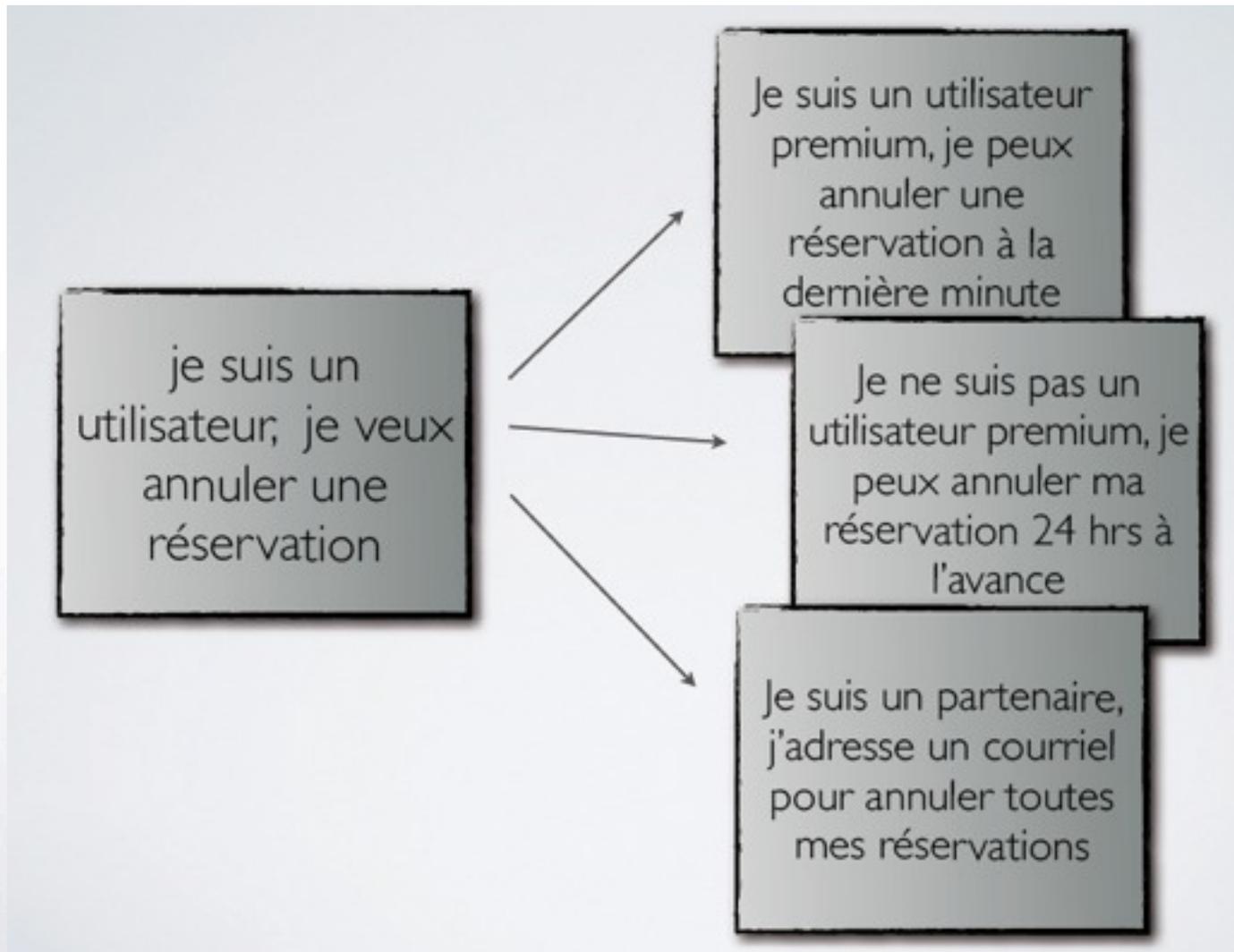


Détails

Je suis un utilisateur, je veux annuler une réservation

- Le client reçoit un remboursement complet ou partiel
 - Je rembourse directement sur son compte ou à l'intermédiaire
- Comment doit fonctionner l'annulation d'une réservation ?
 - C'est le même principe pour tous les hôtels ?
 - Un voyageur fréquent peut-il annuler après sa réservation
- Une confirmation est adressée à l'utilisateur ?
 - Comment ?

LES DÉTAILS SONT AJOUTÉS EN PETITES USER STORIES



LES DÉTAILS SONT AUSSI UNE CONDITION DE SATISFACTION

- Le product owner peut ajouter aux users stories des conditions de satisfaction
- Ce sont essentiellement des vérifications

je suis un utilisateur, je veux annuler une réservation

- Vérifier qu'un premium peut annuler le jour même sans charge supplémentaire
- Vérifier qu'un non-premium paye 10% du montant en cas d'annulation le jour de sa réservation
- Vérifier qu'il y a bien un courriel qui est adressé en cas d'annulation
- Vérifier que l'hôtel est bien notifié de l'ensemble des annulations

UN EXEMPLE DE JEUX DE DÉVELOPPEMENT

Je suis un joueur, je veux
que les ennemis aient une
réaction physique quand je
les touche

- Un ennemi bascule sur la gauche quand je le touche sur la gauche et sur la droite quand je le touche sur la droite
- Un ennemi bascule vers l'avant quand je le touche au centre
- Un ennemi tombe en arrière quand je le touche à la tête

EPIC

Je suis un musicien, je veux
gérer mes morceaux de
musique

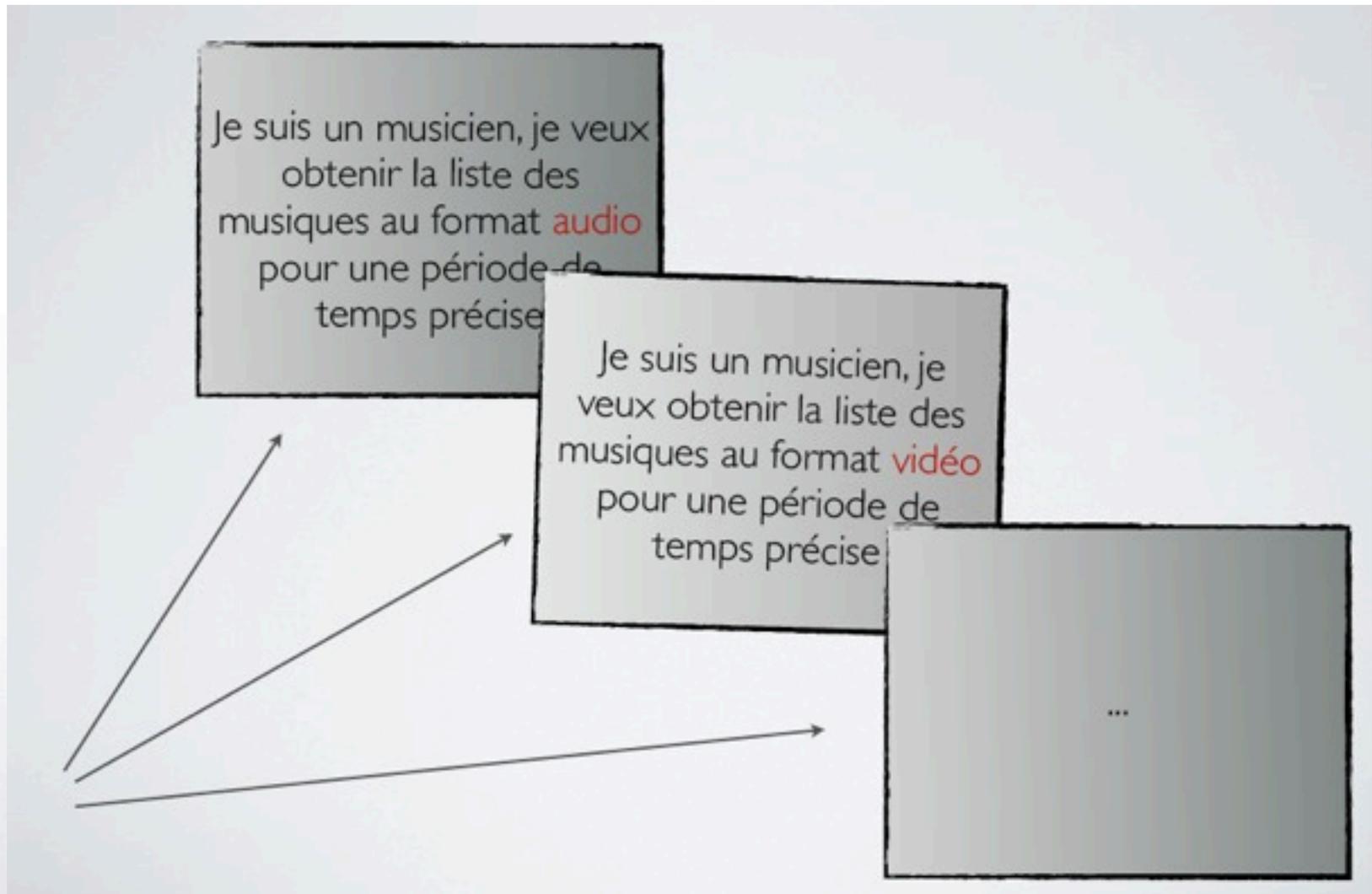
Clairement une Epic

Je suis un musicien, je veux
afficher les musiques par
catégorie qui ont le mieux
marché dans une période
de temps précise

Epic ?

Je suis un musicien, je veux
sélectionner une catégorie
de musique selon un type
de format (audio, vidéo,
etc.) sur une période de
temps précise.

EPIC vers User stories

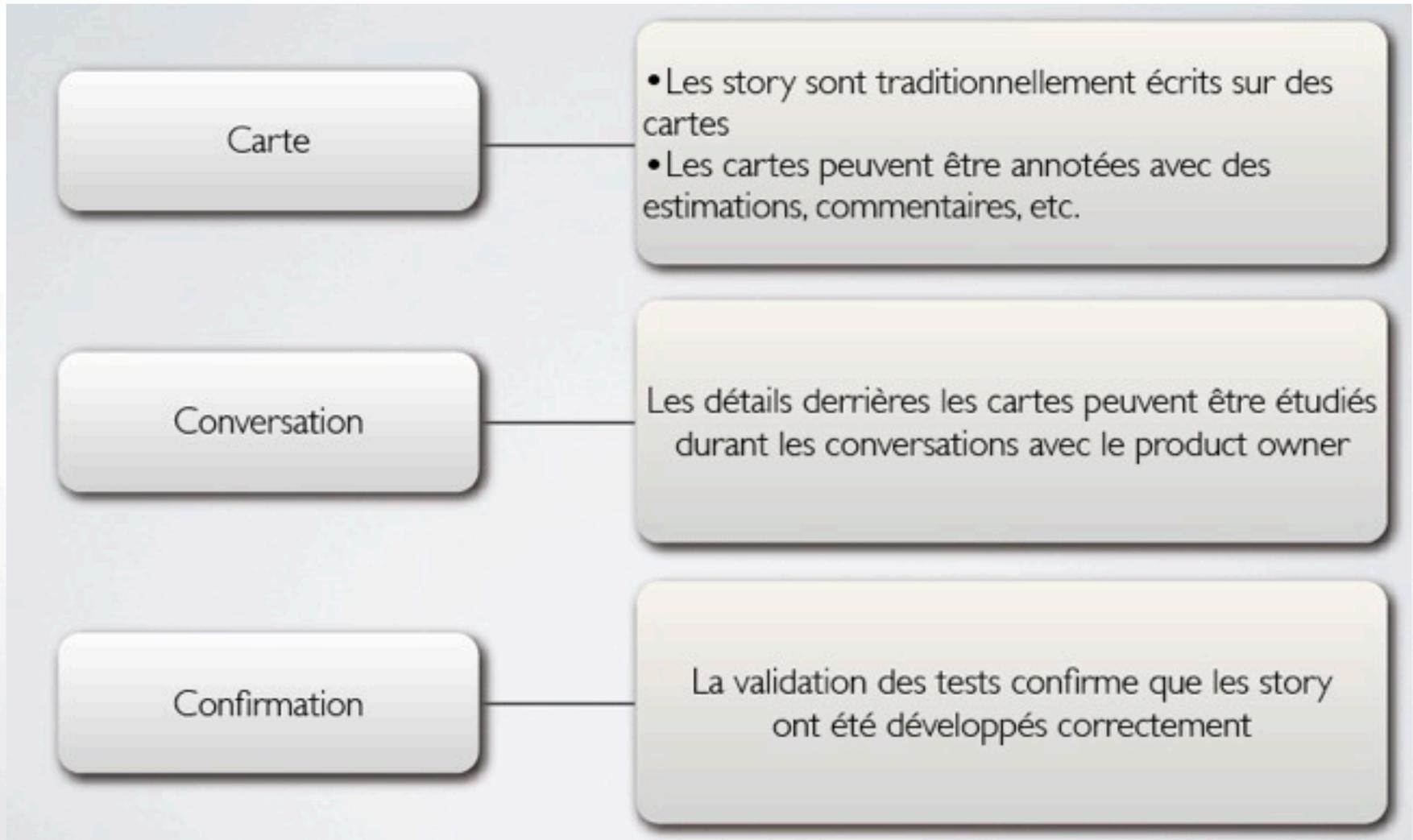


TERMINER, EN AJOUTANT LES CONDITIONS DE SATISFACTION

Je suis un musicien, je veux afficher les musiques les plus populaires par catégorie dans une période de temps précise

- Afficher le nombre d'albums vendus par musique
- Afficher le nombre d'écoutes par musique

Bonnes User stories & les 3 C



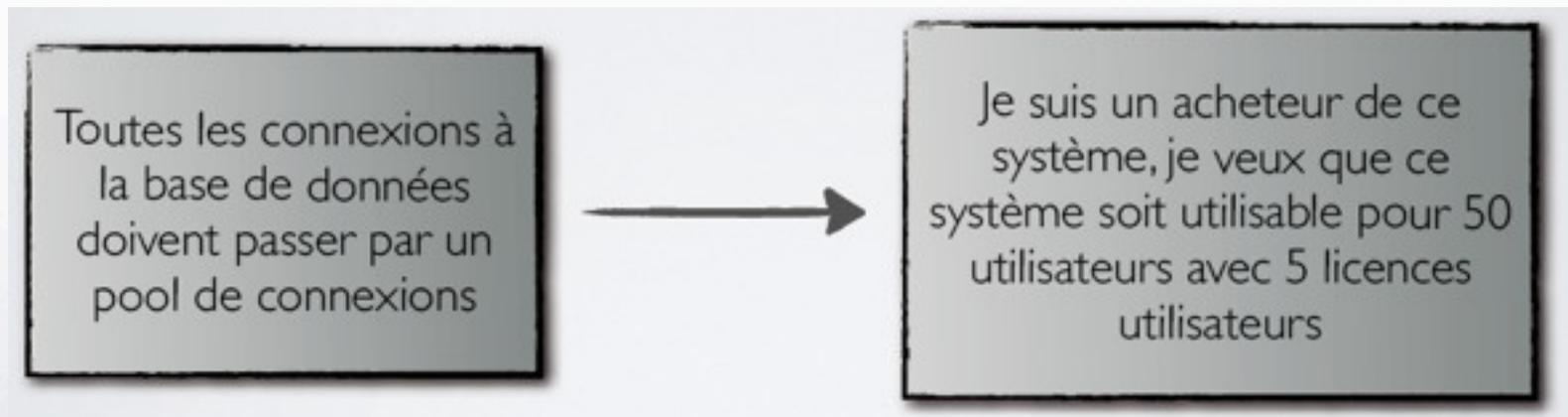
Une bonne «story»

INVEST

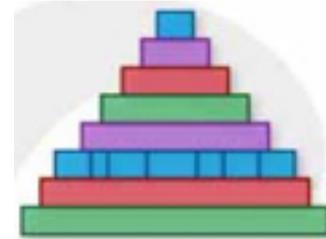
- I - Indépendant
- N - Négociable
- V - Valeur
- E - Estimable
- S - Size (Taille)
- T - Testable

Une bonne «story» : Valeur

Elle doit être réécrite pour montrer le bénéfice pour les utilisateurs et les clients



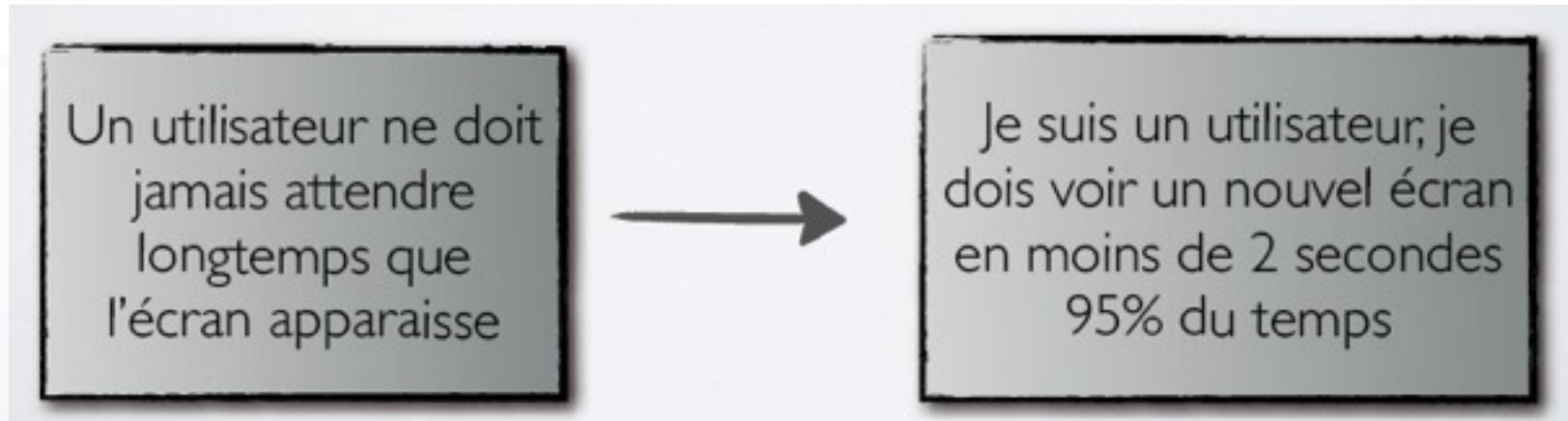
Une bonne «story» : Taille



- Des petites user stories pour un futur proche
- Macro (Epic) pour les prochaines
- Les user stories sont progressivement affinées dans le temps, plus elles s'approchent de la fin,
- Deux types de grandes user stories
 - Les user stories complexes : intrinsèquement grande et sans possibilités de les réduire
 - Les user stories combinées : Plusieurs user stories combinées en une seule

Une bonne «story» : Testable

- Un test démontre que la user stories correspond aux besoins du client



Une bonne «story» : Testable

■ SMART

- Spécifique - défini et explicite
- Mesurable - quantifiable et mesurable
- Atteignable - qui peut être réalisé et validé
- Relevant - pertinent pour la story
- Temporaire - limité dans le temps

Facilite la rédaction des critères d'acceptation

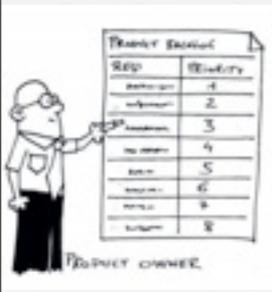
User Stories versus Use cases

- Une “User story” : un **but**, **pas** une **séquence d’actions**.
- Une “User story” souvent seulement **un des scénarios** du “Use case”.
- Une “User story” émerge plus **rapidement** => **moins** d’analyse
- Les “User stories” plus **faciles à lire** (Mouais...)
- Les “User stories” sont rédigées (en principe) sur des cartes: leur durée de vie est limitée et elles n’ont pas pour vocation à être conservées (contrairement aux “Use cases”). Or, ce besoin de traçabilité peut être réel.
- Les “User stories” reposent sur un **mode oral, collaboratif**, de proximité : elles sont discutées (Customer / Developer) et tout n’est pas rédigé (contrairement aux “Use cases”) => Pbme dans le cas de gros projets, équipe importante, offshore...
- Une “User story” doit être implémentée et testée en une itération. Un “Use case” peut être traité sur plusieurs itérations (scénario nominal sur une, scénarios alternatifs sur une autre) en fonction des risques à lever.
-

«Product Backlog»

SPRINT BACKLOG (TASKS)					
Product Backlog	Not Started	In Progress	In Tests	Rejected	Accepted
	  				
	 	 			
		 			
	 				 
					

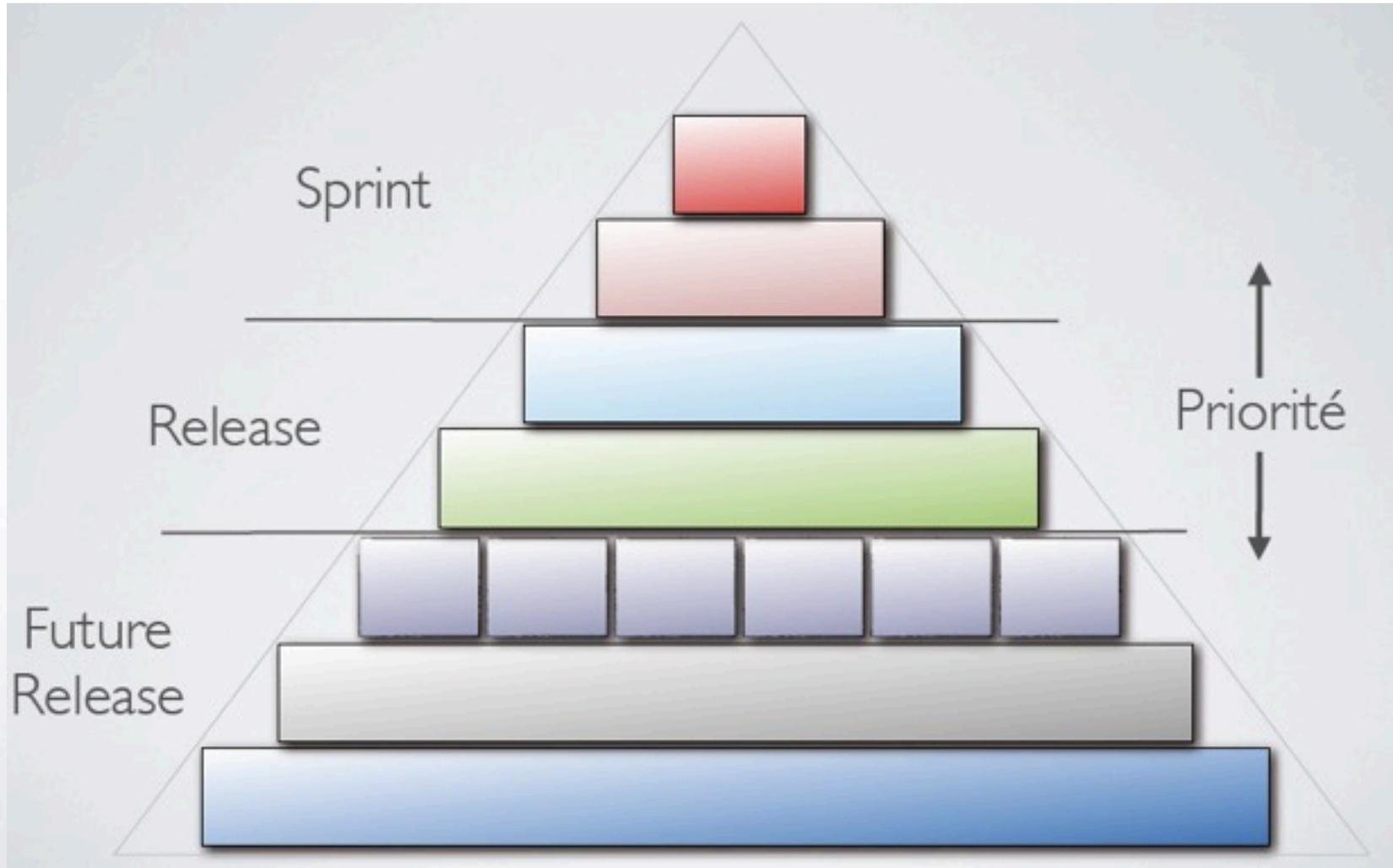
<http://blog.valkir.net/2011/02/24/core-ingredients-of-scrum/>



Artefacts : le Product Backlog ou PDL (Plan de Développement Logiciel)

- ❧ Recueil des exigences et des “User Stories”
« En tant que <rôle>, je veux pouvoir <besoin>, afin de <bénéfice, plus-value>. »
- ❧ Etabli et maintenu par le Product Owner
- ❧ Chaque élément doit apporter une valeur ajoutée au produit final (pas de superflu)
- ❧ Les “importances” sont définies par le Product Owner
- ❧ Les “importances” sont revues à chaque Sprint
- ❧ Les plus importantes sont les plus détaillées.

Iceberg du product backlog



BACKLOG DE PRODUIT

- Les exigences du produit
- Toutes (Idées, fonctionnalités, Epic, Thème, etc.)
- Exprimées en User Stories
- Le PO le maintient organisé
- Toujours estimé et avec les priorités

User stories / (Récits ou histoires d'utilisateur)

- ❧ Une ou deux phrases résumant ce que veut l'utilisateur
- ❧ Décrit comment le système est sensé travailler
- ❧ Ecrite sur une carte
- ❧ Contient suffisamment de détails pour pouvoir être estimée

Martin (2003): “As part of selecting each desired feature, the customers define automated **acceptance tests** to show that the feature is working.”

Lorsqu'une expression du besoin existe en UML, elle peut être utilisée

User stories

Initial Story List



Release planning

As a ____, I
want to be able
to ____ so that

Might have an initial estimate (perhaps for both analysis and development), and an expression of technical and business confidence that this is real and achievable

Au Recto :

- Fonctionnalité exigée sous la forme d'un «récit utilisateur»,
- Sa priorité attribuée par l'utilisateur et
- Le temps estimé par l'équipe pour sa réalisation.

(voir planning pocker)

**Je suis <rôle>,
Je veux <fonction>**

User stories

Au Verso :

- conditions d'acceptation définies par l'utilisateur
- niveau de risque (optionnel)
- modifications de la demande (optionnel)
- ...

Initial Story List

Release Story List



Release planning

As a ____, I
want to be able
to ____ so that

As a ____, I
want to be able
to ____ so that

I will know this is
done when

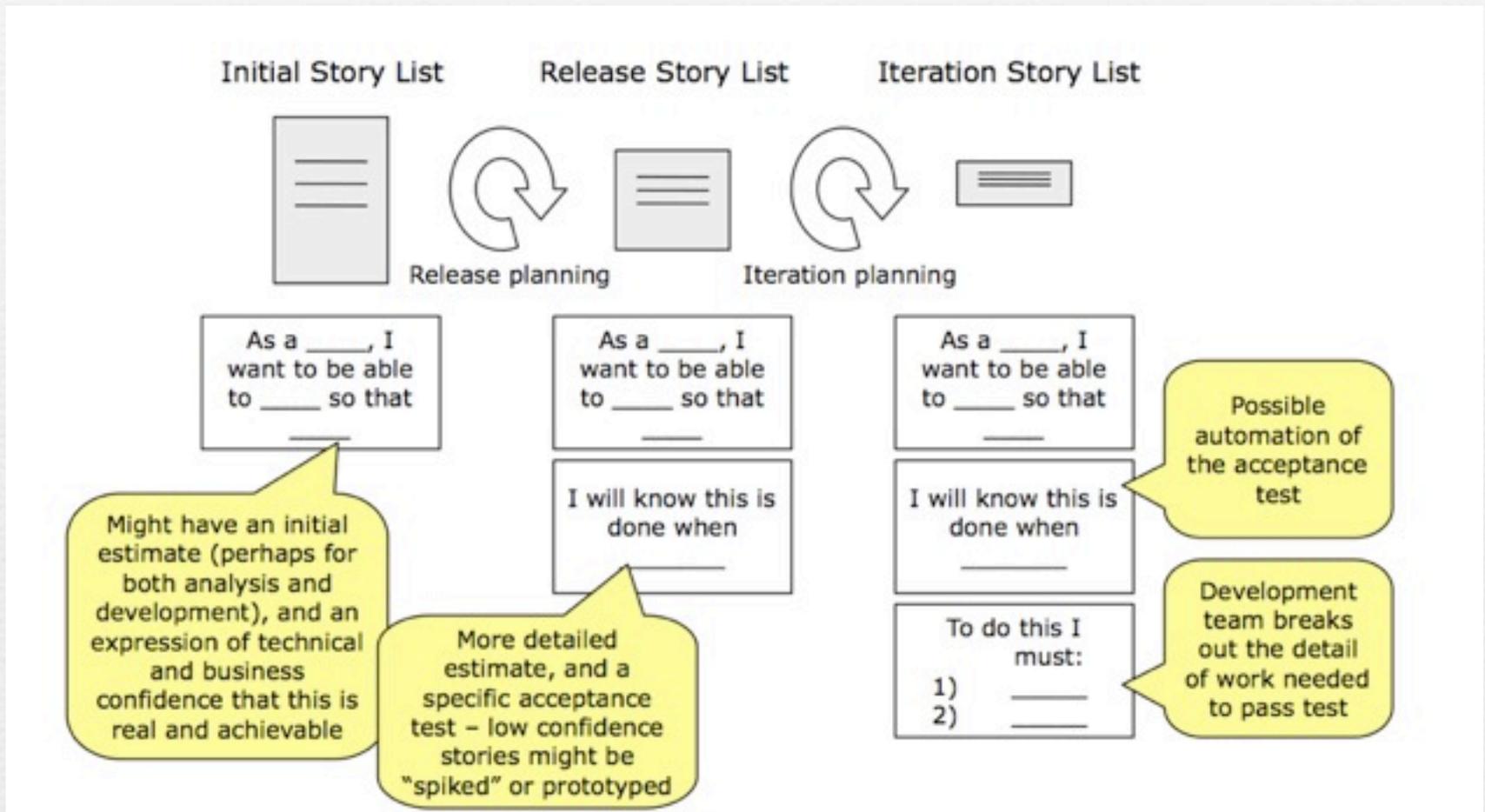
Might have an initial estimate (perhaps for both analysis and development), and an expression of technical and business confidence that this is real and achievable

More detailed estimate, and a specific acceptance test – low confidence stories might be "spiked" or prototyped

**Je suis <rôle>,
Je veux <fonction>
et <un résultat>**

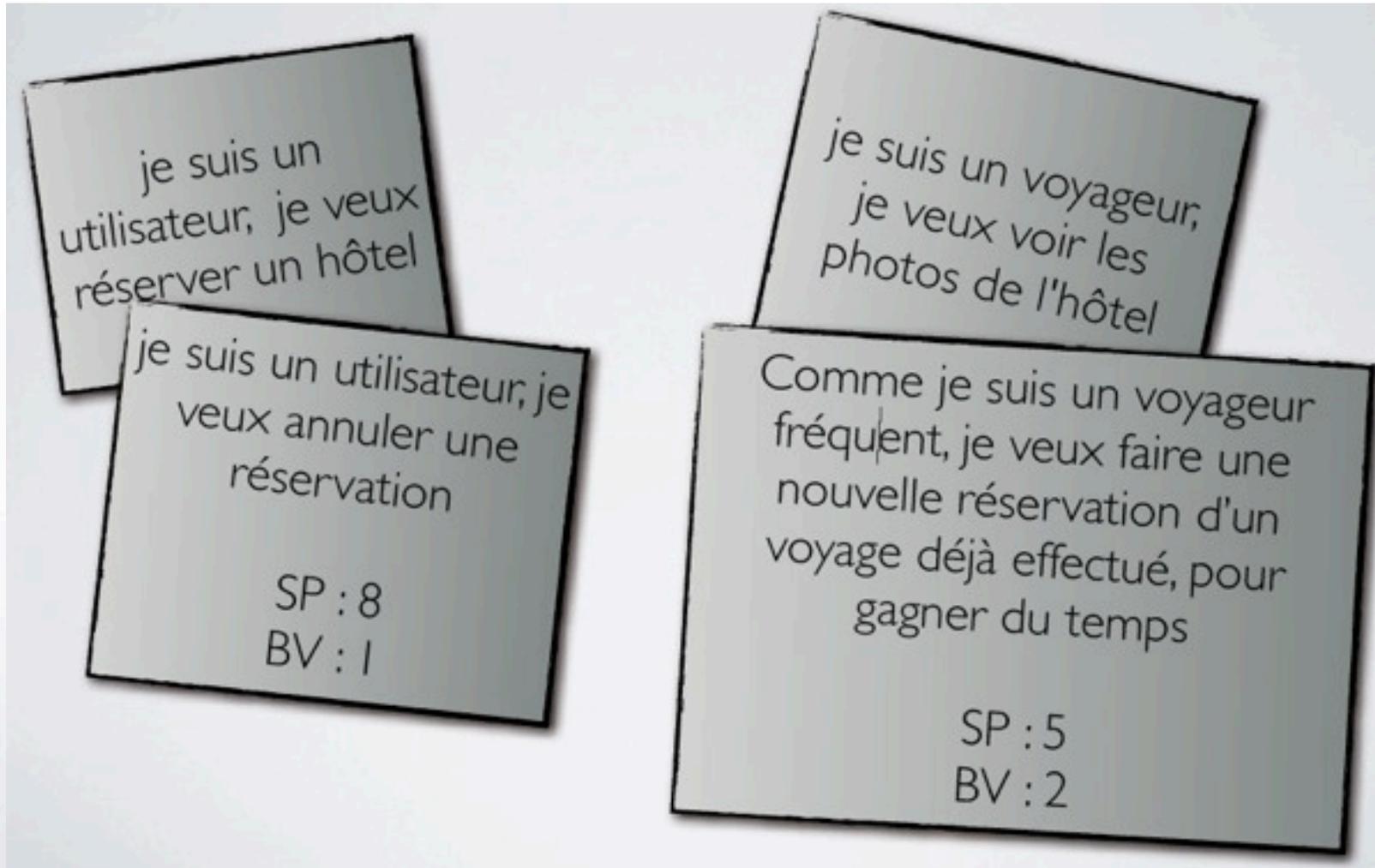
Estimation plus détaillée, et un test d'acceptation spécifique- des histoires imprécises peuvent être «enrichies» ou prototypées

User stories



L'équipe de développement détaille les travaux nécessaires pour passer le test

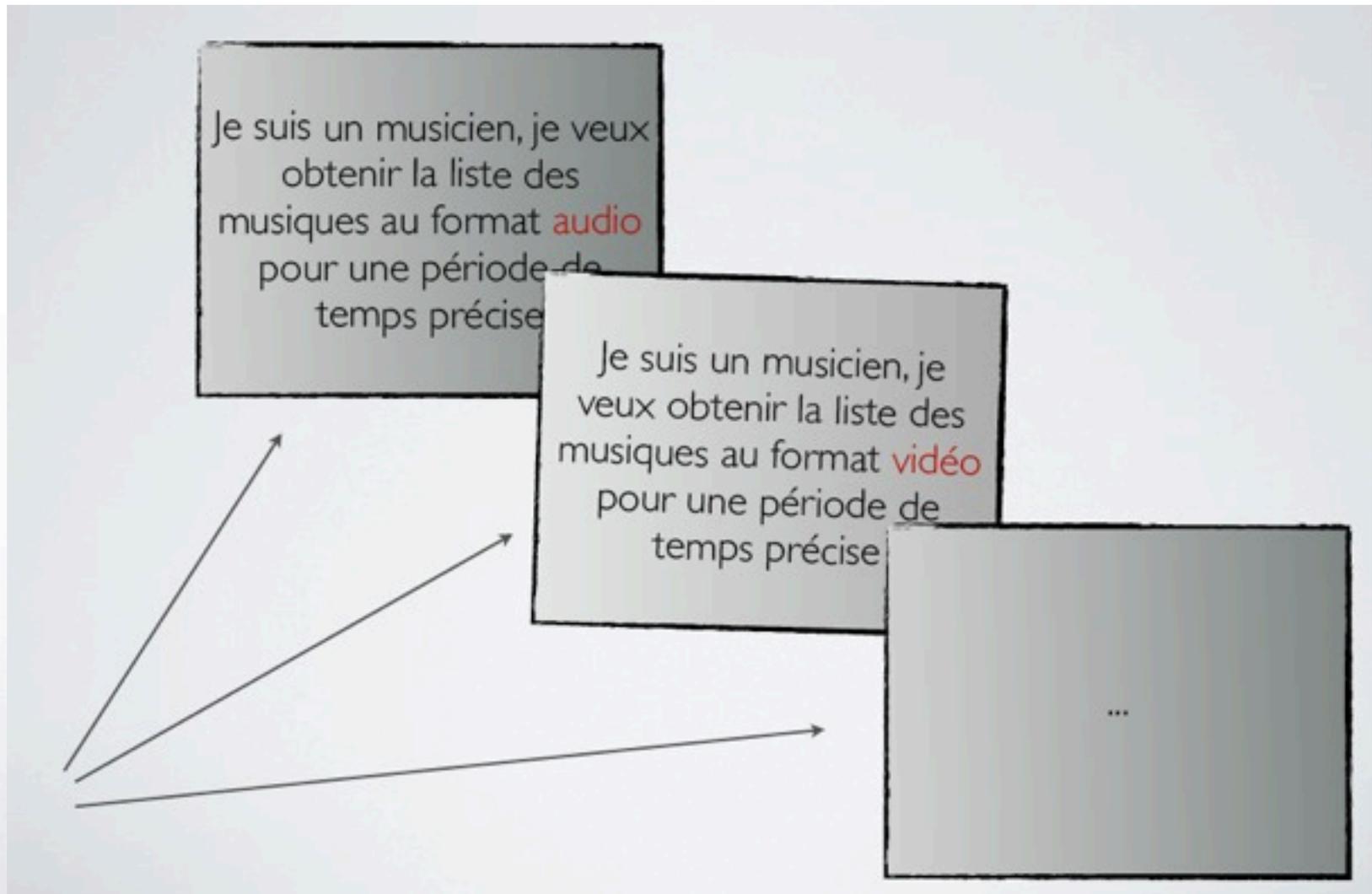
EXEMPLE DEPUIS UN SITE DE VOYAGES



User Stories versus Use cases

- Une “User story” : un **but**, **pas** une séquence d’actions.
- Une “User story” souvent seulement **un des scénarios** du “Use case”.
- Une “User story” émerge plus **rapidement** => **moins** d’analyse
- Les “User stories” plus **faciles à lire** (Mouais...)
- Les “User stories” sont rédigées (en principe) sur des cartes: leur durée de vie est limitée et elles n’ont pas pour vocation à être conservées (contrairement aux “Use cases”). Or, ce besoin de traçabilité peut être réel.
- Les “User stories” reposent sur un **mode oral, collaboratif**, de proximité : elles sont discutées (Customer / Developer) et tout n’est pas rédigé (contrairement aux “Use cases”) => Pbme dans le cas de gros projets, équipe importante, offshore...
- Une “User story” doit être implémentée et testée en une itération. Un “Use case” peut être traité sur plusieurs itérations (scénario nominal sur une, scénarios alternatifs sur une autre) en fonction des risques à lever.
-

EPIC vers User stories



TERMINER, EN AJOUTANT LES CONDITIONS DE SATISFACTION

Je suis un musicien, je veux afficher les musiques les plus populaires par catégorie dans une période de temps précise

- Afficher le nombre d'albums vendus par musique
- Afficher le nombre d'écoutes par musique

Bonnes User stories & les 3 C



Une bonne «story»

INVEST 

- I - Indépendant
- N - Négociable
- V - Valeur
- E - Estimable
- S - Size (Taille)
- T - Testable

Une bonne «story» : Valeur

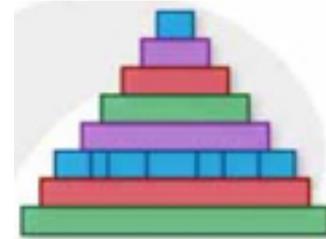
Elle doit être réécrite pour montrer le bénéfice pour les utilisateurs et les clients

Toutes les connexions à la base de données doivent passer par un pool de connexions



Je suis un acheteur de ce système, je veux que ce système soit utilisable pour 50 utilisateurs avec 5 licences utilisateurs

Une bonne «story» : Taille



- Des petites user stories pour un futur proche
- Macro (Epic) pour les prochaines
- Les user stories sont progressivement affinées dans le temps, plus elles s'approchent de la fin,
- Deux types de grandes user stories
 - Les user stories complexes : intrinsèquement grande et sans possibilités de les réduire
 - Les user stories combinées : Plusieurs user stories combinées en une seule

Une bonne «story» : Testable

- Un test démontre que la user stories correspond aux besoins du client



Une bonne «story» : Testable

■ SMART

- Spécifique - défini et explicite
- Mesurable - quantifiable et mesurable
- Atteignable - qui peut être réalisé et validé
- Relevant - pertinent pour la story
- Temporaire - limité dans le temps

Facilite la rédaction des critères d'acceptation

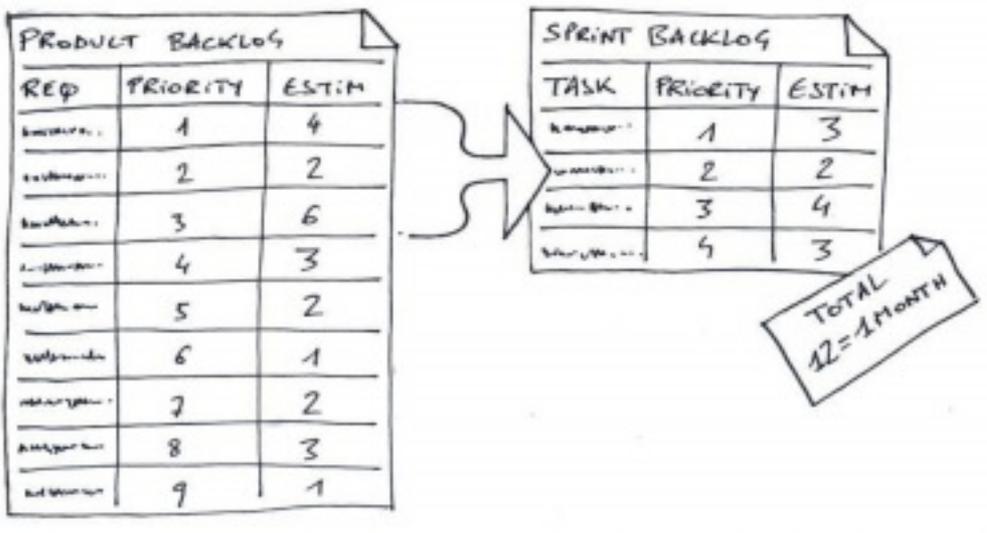
User Stories versus Use cases

- Une “User story” : un **but**, **pas** une **séquence d’actions**.
- Une “User story” souvent seulement **un des scénarios** du “Use case”.
- Une “User story” émerge plus **rapidement** => **moins** d’analyse
- Les “User stories” plus **faciles à lire** (Mouais...)
- Les “User stories” sont rédigées (en principe) sur des cartes: leur durée de vie est limitée et elles n’ont pas pour vocation à être conservées (contrairement aux “Use cases”). Or, ce besoin de traçabilité peut être réel.
- Les “User stories” reposent sur un **mode oral, collaboratif**, de proximité : elles sont discutées (Customer / Developer) et tout n’est pas rédigé (contrairement aux “Use cases”) => Pbme dans le cas de gros projets, équipe importante, offshore...
- Une “User story” doit être implémentée et testée en une itération. Un “Use case” peut être traité sur plusieurs itérations (scénario nominal sur une, scénarios alternatifs sur une autre) en fonction des risques à lever.
-

«Product Backlog»

SPRINT BACKLOG (TASKS)					
Product Backlog	Not Started	In Progress	In Tests	Rejected	Accepted
	  				
	 	 			
		 			
	 				 

Artefacts : le sprint Backlog



- ❧ Recueil des différentes tâches, extraites du Product Backlog, que l'équipe s'engage à réaliser lors du Sprint.
 - ❧ Le travail n'est jamais assigné par un autre
- ❧ L'estimation du reste à faire est ajustée chaque jour
- ❧ Si une tâche n'est pas claire, ou trop volumineuse, la décomposer en tâches plus petites.

Exemple de Product Backlog

ID	Nom	Importance	Estimation	Démo	Notes
1	Dépôt	30	5	Authentification, ouvrir la page de dépôt, déposer 10€, aller sur la page du solde et vérifier que ça a bien augmenté de 10€.	Nécessite un diagramme de séquences UML. Ne pas se soucier du cryptage pour l'instant.
2	Voir l'historique de ses transactions	10	8	Authentification, cliquez sur « transactions ». Faire un dépôt. Revenir aux transactions, vérifier que le nouveau dépôt apparaît.	Utiliser la pagination pour éviter des requêtes volumineuses. Conception semblable à la page des utilisateurs.

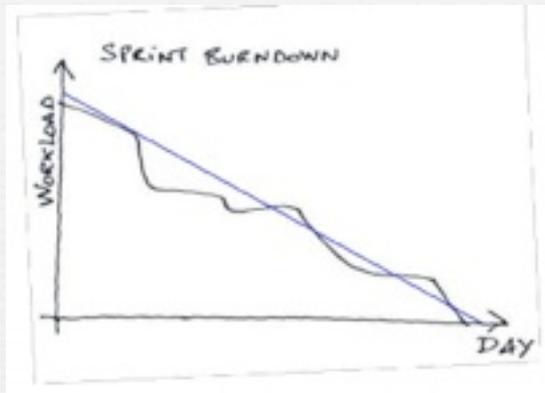


Base du diagramme de suivi des tâches

Version adaptée par les étudiants IUT 2013

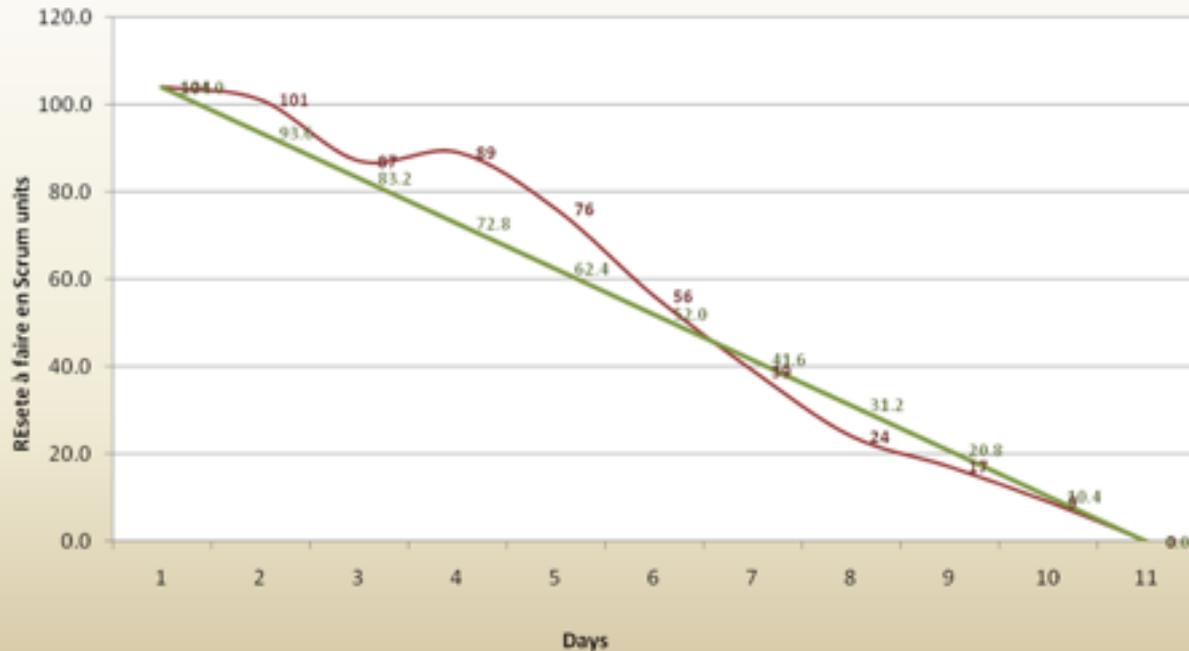


Artefacts : le Burndown Chart

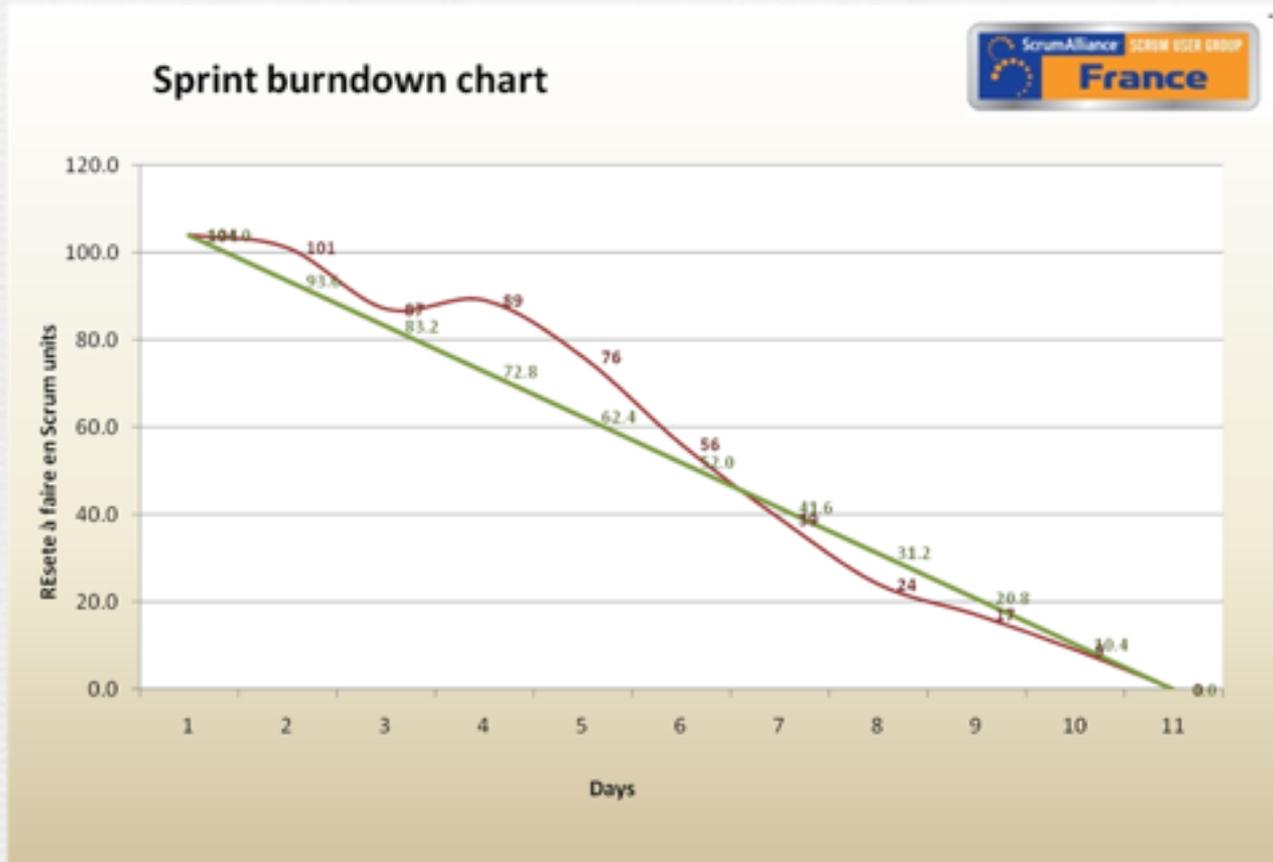


- Graphique permettant de voir le reste à faire sur un Sprint
 - En abscisse : le nombre de jours du Sprint
 - En ordonnée : la quantité de travail à réaliser
- La ligne droite (en bleu) représente la “Vélocité” idéale de l’équipe.
- La ligne courbe (en noir) représente la “Vélocité” véritable de l’équipe.
- Après chaque Daily Scrum Meeting, en fonction des travaux de la veille de chacun, le Burndown Chart est mis à jour

Sprint burndown chart



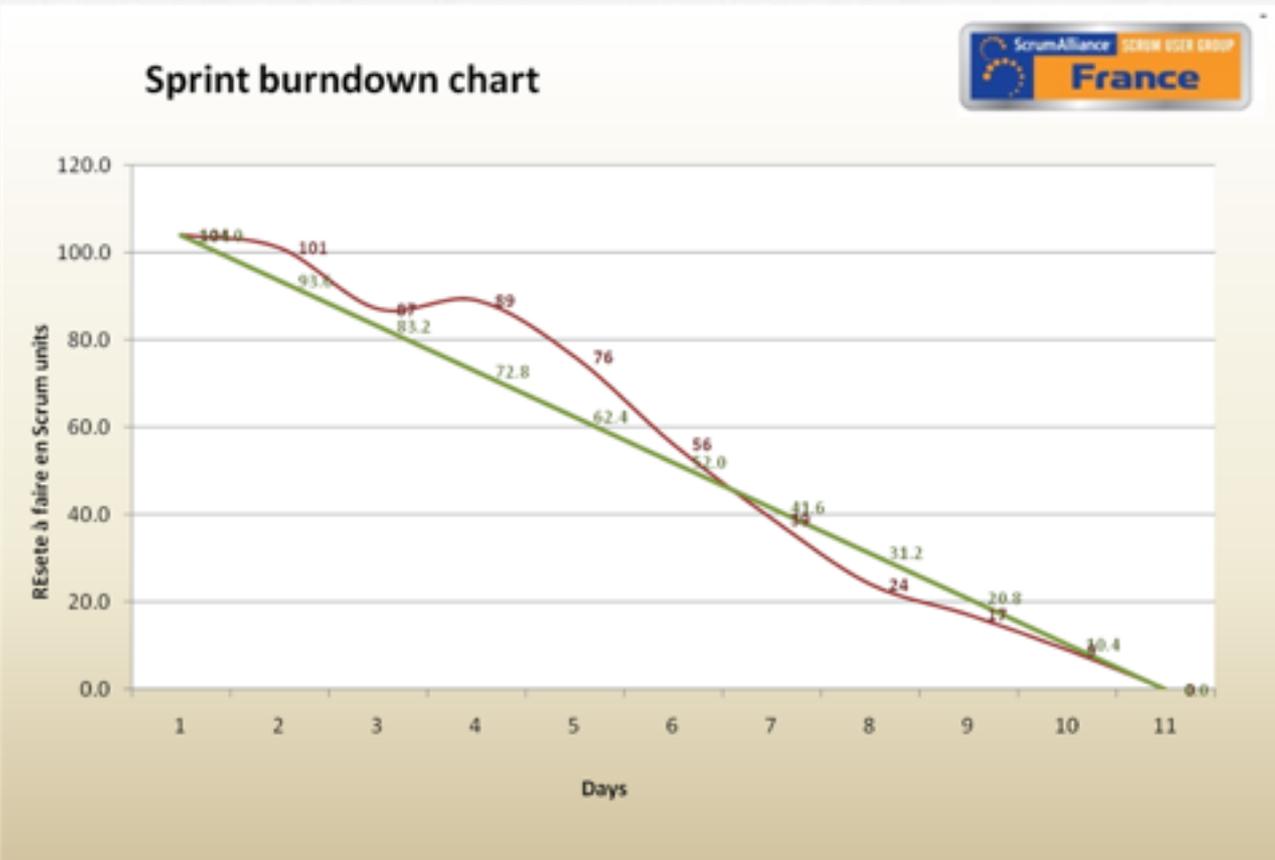
<http://www.frenchsug.org/display/FRSUG/Burndown+charts>



Durée du sprint fixée à 10 jours

<http://www.frenchsug.org/display/FRSUG/Burndown+charts>

104
points
scrum



Durée du
sprint fixée à
10 jours

<http://www.frenchsug.org/display/FRSUG/Burndown+charts>

104
points
scrum

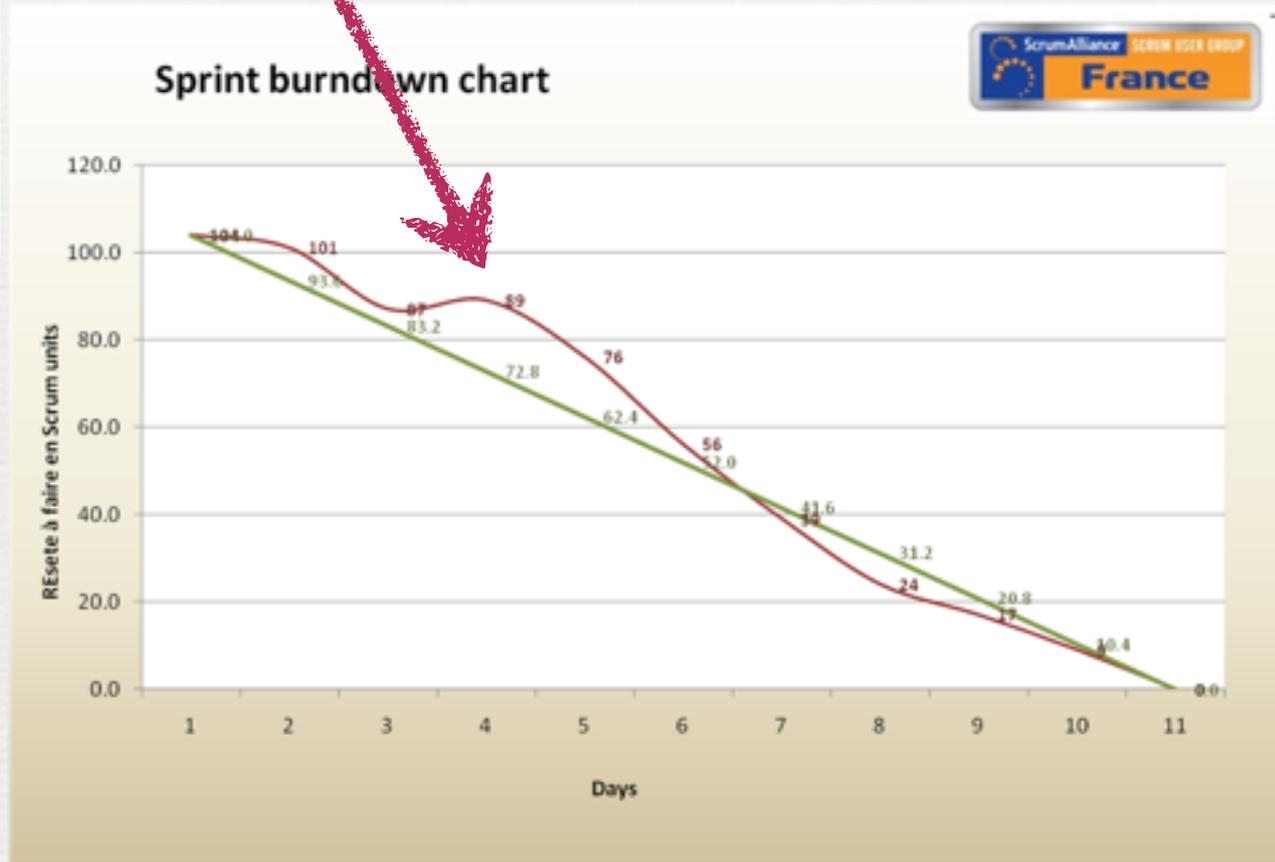


Durée du
sprint fixée à
10 jours

<http://www.frenchsug.org/display/FRSUG/Burndown+charts>

Un pbme est détecté au 4è
jour; la charge est
réévaluée

104
points
scrum



Durée du
sprint fixée à
10 jours

<http://www.frenchsug.org/display/FRSUG/Burndown+charts>

Accélération

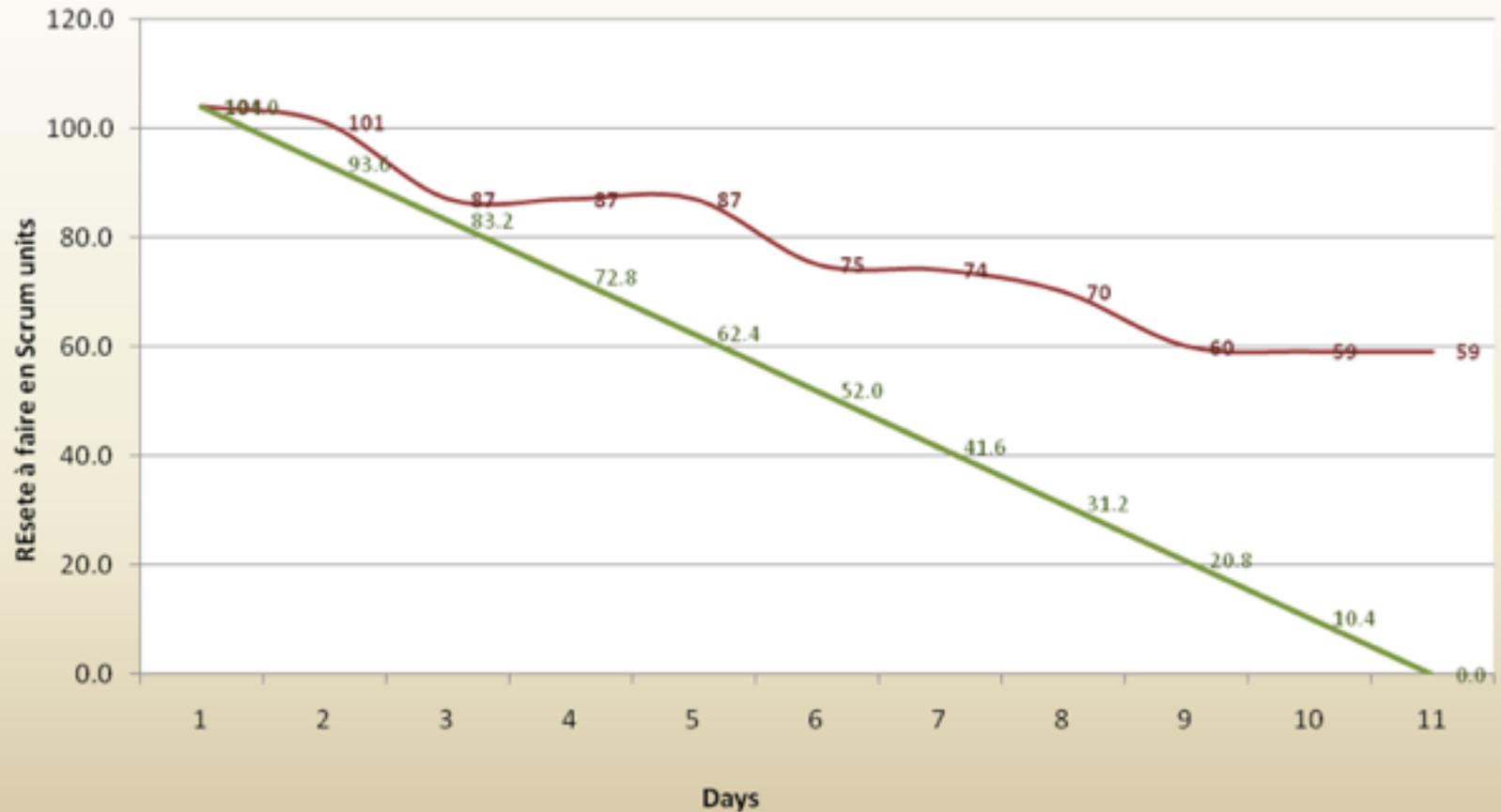
104
points
scrum



Durée du
sprint fixée à
10 jours

<http://www.frenchsug.org/display/FRSUG/Burndown+charts>

Sprint burndown chart - sans avancement



<http://www.frenchsug.org/display/FRSUG/Burndown+charts>

Backlog

TODO

DOING

DONE

valeurs de scrum

organisations

Rôles

Artefacts

Réunions

Environnement

Conclusion

Caractéristiques de Scrum

Rôles

- Product Owner
- Scrum Master
- Equipe

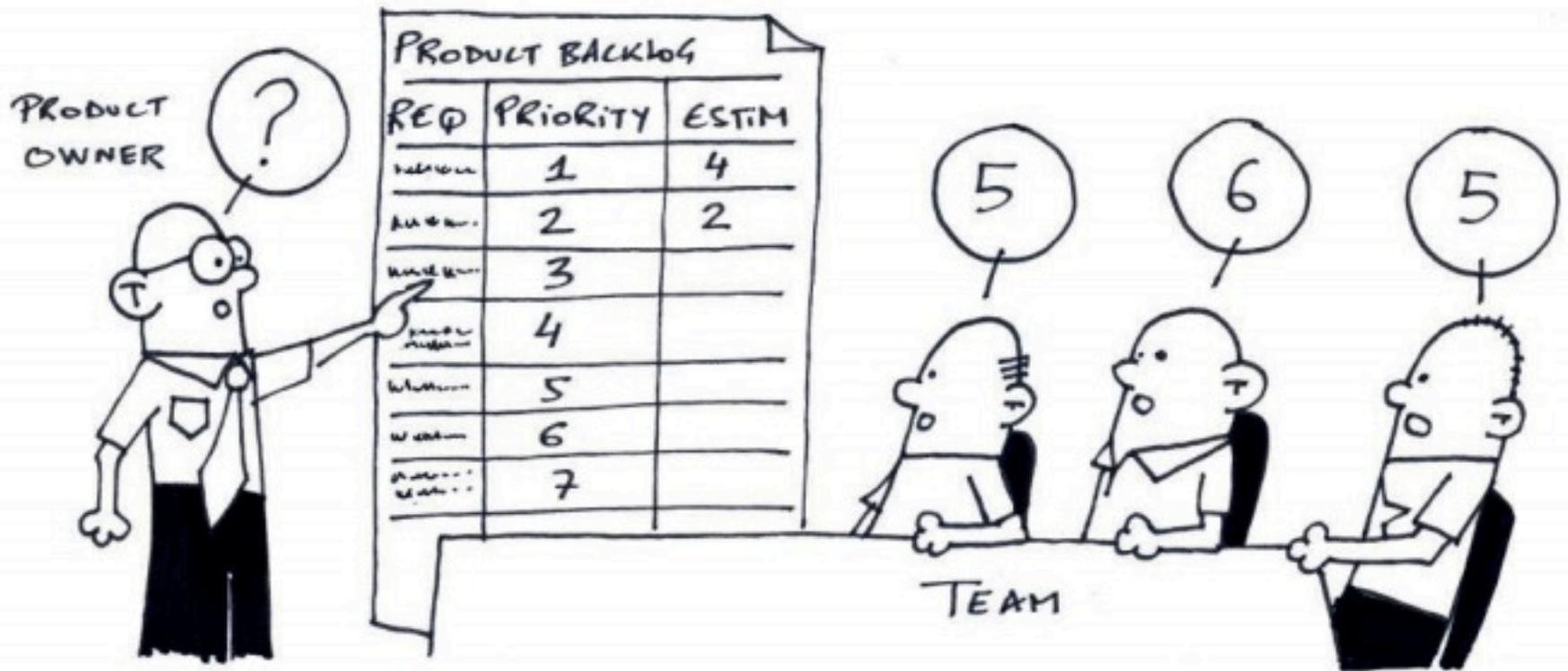
Meetings

- Planification du sprint
- Scrum quotidien
- Revue de sprint
- Rétrospective

Artefacts

- Backlog de produit
- Backlog de sprint
- Burndown Chart

Meetings : Planification du sprint



Meetings : Planification du sprint

L'équipe choisit, à partir du backlog de produits, les éléments qu'elle s'engage à finir.

- Le backlog de sprint est créé.
 - Les tâches sont identifiées et estimées (1-16 heures)
 - Collectivement, pas seulement par le ScrumMaster
- La conception de haut niveau est abordée

En tant que touriste
potentiel dans la région,
je veux voir les photos
des hôtels

Coder la couche de persistance (8 heures)
Coder l'IHM (4)
Ecrire les test fixtures (4)
Coder la classe foo (6)
Maj les test de performance (4)

Sprint Planning 2 : une approche

- Sprint Planning 2 est ici limité à 2 hours (6 histoires) ...
- Agenda for Sprint Planning 2
 - 14.00 - 14.05 Formation des paires et distribution des histoires
 - 14.05 - 14.35 **Concept** - Réflexion sur les aspects techniques -> Production de documents à présenter à l'équipe
 - 14.35 - 15.05 **Présentation** des paires limitée à 5 Minutes par histoire (Q/R comprises)
 - 15.05 - 15.35 **Tâches** - Chaque paire découpe les histoires en ensemble de tâches d'au plus un jour.
 - 15.35 - 16.00 Présentation des tâches en 4 Minutes maximum (25 Minutes / 6 Stories).

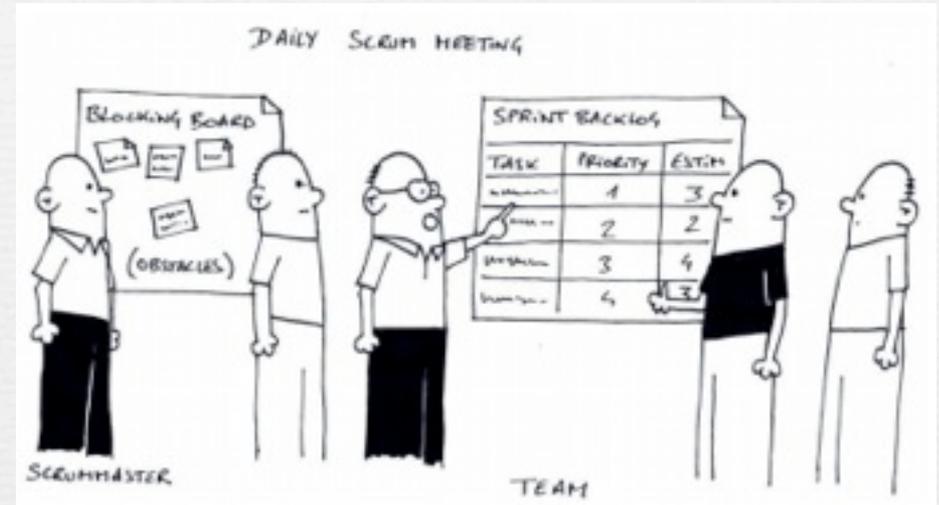
Sprint Planning 2 : une approche

- ➔ Au moins 2 personnes ont travaillé chaque tâche
- ➔ La réunion est structurée.
- ➔ Les solutions sont présentées et discutées avec l'ensemble de l'équipe qui peut alors s'entraider.

<http://www.scrum-breakfast.com/2011/02/how-we-do-sprint-planning-2.html>

Meetings : Scrum quotidien

- ☛ Tous les Jours
- ☛ 15 minutes (time boxed)
- ☛ Debout



- ☛ Pas fait pour résoudre les problèmes
 - ☛ Tout le monde est invité
 - ☛ Seuls les membres de l'équipe peuvent parler
- ☛ Permet d'éviter l'organisation d'autres réunion

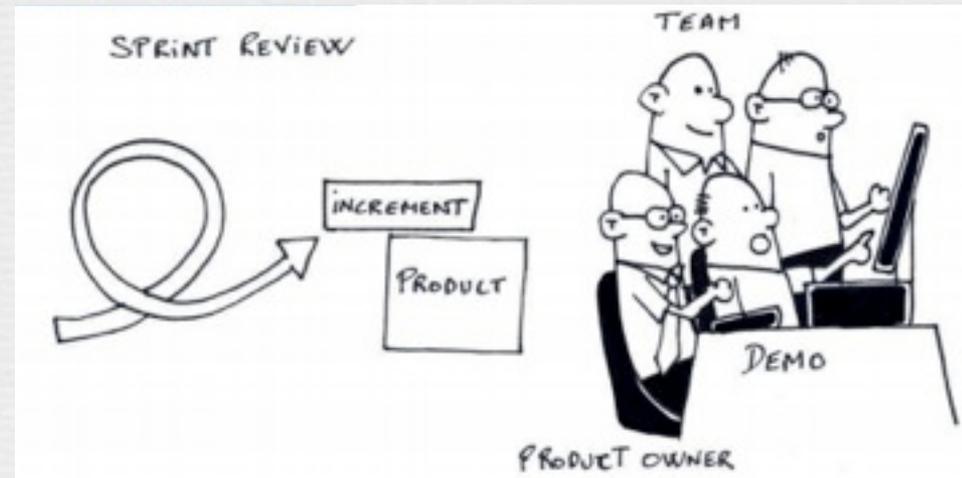
Meetings : Scrum quotidien



Il ne s'agit pas de compte-rendus au Scrum Master.
=> Ce sont des engagements devant les pairs

Meetings : Revue de Sprint

- ❧ L'équipe présente ce qu'elle a fait pendant le Sprint.
- ❧ L'équipe effectue une démo des nouvelles fonctionnalités incluses dans le livrable de ce Sprint.
- ❧ La revue de Sprint est "Informel".
 - ❧ Le temps de préparation doit être minimisé.
 - ❧ Pas de slides.
- ❧ Toute l'équipe participe.
- ❧ Tout le monde est invité.



Mais que signifie TERMINE ? DONE?

- Il est interdit de livrer un item inachevé, même avec l'intention de « le terminer plus tard ».
 - ➔ Maintenir la confiance avec le client de ne pas « cacher » le travail non « terminé ».
- Functionality has been code reviewed, functionality has been integrated and built, acceptance tests have been run, and documentation has been created.
- Code adheres to standards, is clean, has been refactored, has been unit tested, has been checked in, has been built, and has had a suite of unit tests applied to it

Meetings : Rétrospective

- A la fin de chaque sprint
- Permet de réfléchir régulièrement à ce qui marche et ce qui ne marche pas.
- Dure en général de 15 à 30 minutes.
- Fait à la fin de chaque Sprint.
- Toute l'équipe participe.
 - Scrum Master
 - Product Owner
 - Equipe
 - Eventuellement d'autres intervenants



Backlog

TODO

DOING

DONE

valeurs de scrum

organisations

Rôles

Artefacts

Réunions

Environnement

Conclusion

Environnement de collaboration

❧ Organisation en «war room»



Backlog

TODO

DOING

DONE

valeurs de scrum

organisations

Rôles

Artefacts

Réunions

Environnement

Conclusion

Conclusion (1)

- ❧ Méthode de gestion de projet – développement logiciel
- ❧ A compléter avec des techniques d'ingénierie logicielle (XP est un support intéressant)
- ❧ Conditions propices nécessaires
- ❧ Expérimentations prometteuses
- ❧ Principal bénéfice : des équipes motivées

Perspectives

- ❧ Défauts à palier
 - ❧ Absence de dépendance entre les tâches
 - ❧ Polyvalence des programmeurs
 - ❧ Productivité équivalente supposée
- ⇒ Grande **maturité** nécessaire
- ❧ Contrats à adapter
- ❧ Stratégie d'introduction de Scrum en entreprise

Agilité et Systèmes d'information

LeMondelInformatique
Toute l'info et les tendances du monde IT

19/9/11 : Agilité des SI, la révolution de l'adaptabilité permanente

- ❧ **Face aux évolutions permanentes de leur business, les entreprises misent sur l'agilité.**
- ❧ L'adoption des méthodes agiles itératives reste plus que jamais d'actualité. En s'inspirant de Scrum, XP ou Unified Process, on accélère la délivrance de chaque projet en accord avec les priorités des métiers. Parallèlement, une démarche de Lean Management implique plus fortement les développeurs vis-à-vis de la qualité au meilleur coût.

Article à lire (beaucoup d'autres informations)

Backlog

TODO

DOING

DONE

valeurs de scrum

organisations

Rôles

Artefacts

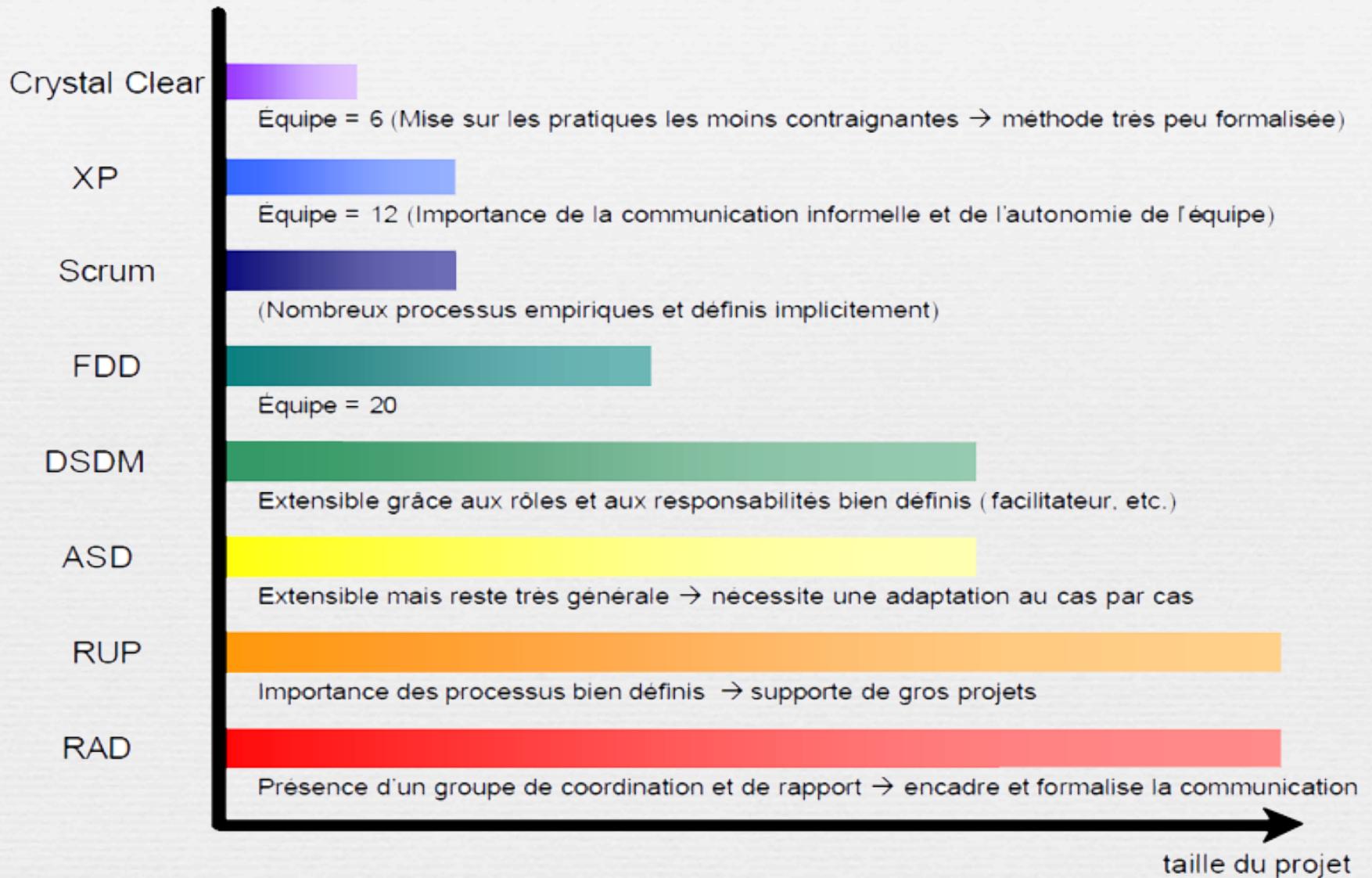
Réunions

Environnement

Conclusion

Bonus

Comparaison des méthodes agiles



Retour sur la méthodologie agile (1)

Ce stage a été l'occasion de pratiquer la méthodologie agile. Cette méthodologie comporte de très nombreux avantages. Elle **limite les risques pour les clients** de ne pas être satisfaits parfaitement par l'application, elle fait des miracles pour la **compréhension client-développeurs** en permettant de décrire les besoins du client en se basant sur une application existante développée lors de l'itération précédente. Elle limite la nécessité d'avoir un client expérimenté dans le développement. Elle fait **gagner beaucoup de temps** lors de la capture des exigences et **améliore la qualité** des exigences exprimées.

Par
Pierre Neu,
Stage Ingénieur
2011, Amadeus

Retour sur la méthodologie agile (2)

Le développement agile ne fonctionne que si les développeurs sont **capables de modifier et de refactoriser** l'application existante. Cela doit rester une préoccupation majeure des

développeurs et il ne faut pas se laisser déborder par les exigences du client, en sachant préserver du temps pour réfléchir à l'amélioration du code existant. Cela peut parfois nécessiter d'avoir une capacité du côté des développeurs à faire face et à savoir convaincre les clients de faire passer cette refactorisation avant le développement de nouvelles fonctionnalités, car le client ne perçoit pas les bénéfices de cette étape, malgré qu'elle soit cruciale pour la maintenabilité et la qualité du produit sur le long terme en général.

Par
Pierre Neu,
Stage Ingénieur
sept. 2011,
Amadeus

Retour sur la méthodologie agile (3)

Cette méthodologie ne doit cependant pas faire disparaître la documentation et les spécifications. Car ces documents permettent de garder un cap et des objectifs pour l'application. Ils constituent malgré tout un guide primordial pour les développeurs. Les interactions entre le client et les développeurs sont très fréquentes et nécessaires. Il ne faut cependant pas perdre de vue les objectifs de ces réunions qui doivent rester courtes. A mon avis il était très positif dans mon cas qu'elles soient organisées par quelqu'un d'extérieur, c'est-à-dire ni un client ni un développeur, pour garder une vision plus neutre.

De plus ces échanges fréquents ont parfois comme résultat un trop grand nombre de demandes de modifications du client, ou de besoins très volatils. L'expérience des développeurs est alors très importante pour évaluer ces besoins et échanger avec le client sur leur réelle nécessité, et prioriser ensemble ces nouveaux besoins, pour garder un cap dans le développement.

Les développeurs doivent aussi être capables de fournir une estimation du surcoût induit par un changement de besoin du client.

Retour sur la méthodologie agile (4)

Les limites de la méthodologie agile existent. Les projets de grande ampleur ne sont par exemple pas des candidats idéaux pour la méthodologie agile. Je pense cependant qu'il est possible d'adapter la méthodologie agile pour conserver certains de ses atouts dans des projets de grande taille.

Cette méthodologie ne doit également pas être un va-tout lorsque les clients ont du mal à décider de leurs besoins et n'ont pas une vision claire de l'application voulue. Même si l'Agile demande moins de documentation au départ, elle requiert un réel et continu investissement de la part des clients pour fonctionner. Les autres limites sont liées justement à l'absence de contrat. Comment arbitrer un conflit entre les clients et les développeurs ? De plus la planification du développement et l'estimation des coûts sont très difficiles, surtout lorsque les clients modifient les besoins.

Par
Pierre Neu, Stage Ingénieur sept. 2011,
Amadeus