Cours 4
Développement agile avec Scrum

Priorisation, Estimation, Planification de Sprint

V. Deslandres © – IUT de LYON

Sommaire

Les **étapes et cérémonies** du développement agile avec SCRUM

Backlog Produit : 3 types d'éléments

- **User story (US)**
  - Besoin exprimé par le Product Owner (PO)
- **Besoin technique**
  - Ex.: *tester le framework Guava pour la gestion du cache, tester l'algorithme de cryptage des mots de passe, configurer le conteneur Docker pour le projet*
  - Rarement exprimée par le PO ! Mais déduite du projet ou des US
- **Default Story**
  - soit un défaut constaté
  - soit l'évolution d'une *story* initiale

Notion de Fini (DoD)

C'est là qu'on discute des **détails !**

- En Agile, le **DoD** est fondamental
  - ***Definition of Done***
- On va se mettre d'accord **AVANT** sur ce qu'est une US **terminée** avec le Product Owner
  - Quand les Test Unitaires passent
  - … en plus, la doc est faite
  - … en plus, la version anglaise est terminée

Avec l'Agile, on va éviter **le syndrome des 90%**, où « tout est commencé mais pas encore complètement terminé »

Eviter la ***dette technique***

« Il *suffira* de faire appel à tel composant », « on duplique le code *en attendant* »…

DoD : Tests d'Acceptation

Product Owner

En tant que membre du site,
je peux annuler ma réservation d'hôtel
Afin de tenir compte d'imprévus dans mon voyage

- ❑ Vérifier qu'un email d'annulation est envoyé au voyageur et à l'hôtel
- ❑ Vérifier que si elle a lieu au moins 15 j. avant la date prévue, pas de charge imputée au voyageur
- ❑ Vérifier qu'un *premium* peut annuler n'importe quand sans charge suppl.
- ❑ Vérifier qu'un *non premium* paie 10% du montant de la résa si annulation moins de 15 j. avant la date prévue

LA CAPTURE des EXIGENCES

Product Owner

Classer les BESOINS

Comment identifier ce qui est important ?

Affectation de la **valeur** Client

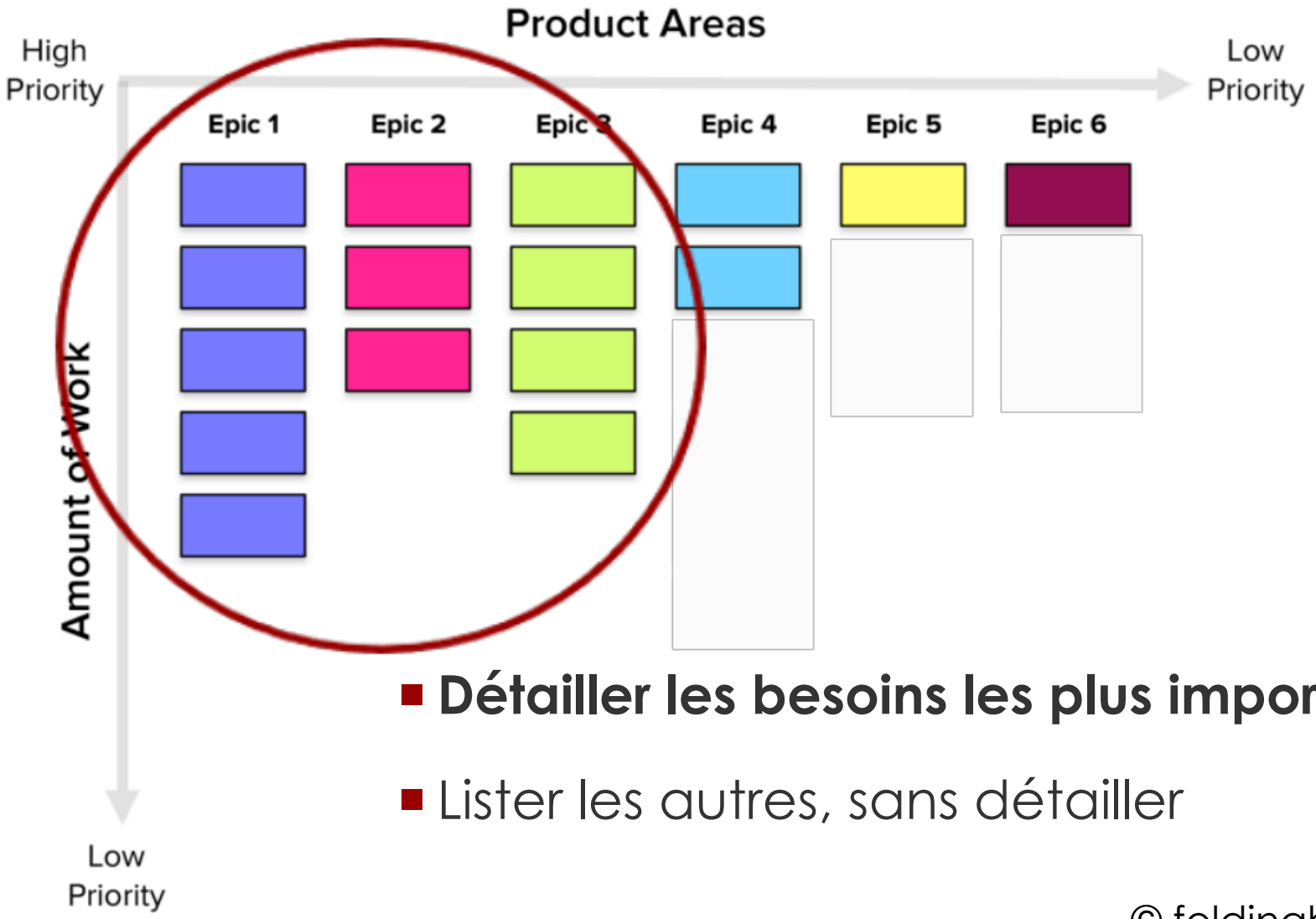
Objectif : Classer les US

- A l'issue du **brainstorming** avec les utilisateurs, de nombreuses idées ont été avancées. L'objectif est alors de *réduire le nombre d'éléments* : les fédérer et les regrouper en thèmes communs.

- Comment identifier dans le **Backlog Produit** ce qui est le **plus important** pour l'organisation ?

- **Valeur Métier** (business value)
  - C'est le BUT (« afin de …. ») des US
- Définie par le **PO + utilisateurs**

Ce que l'on souhaite obtenir :



- **Détailler les besoins les plus importants**
- Lister les autres, sans détailler

© foldingburritos.com

Classer les exigences : 2 méthodes

- **Matrice de Kano**
  - Méthode aussi utilisée en Marketing pour comprendre l'attitude des gens à l'égard d'une offre
- **MoSCoW**

  Must / Should / Could / Won't
  - Méthode la plus facile / rapide
  - (Il en existe d'autres : story mapping)

1- Matrice de Kano

3 types de fonctionnalités

- **Obligatoires**
- **Linéaires** : ajoutent un vrai plus au produit; sans elles, le produit ne sera pas aussi bon aux yeux des utilisateurs
- **Attractives** : si présentes, les clients adorent; pas d'impact si absentes, ils n'y pensent pas

Définies selon 2 axes :

présence / satisfaction

Attention : évalue la perception future des fonctionnalités **sur le client,** pas la satisfaction réelle

Matrice de KANO

Fonctionnalités **obligatoires**
Ex.: syst freinage véhicule

Fonctionnalités **linéaires**
Ex.: véhicule économe en carburant, performances moteur

Fonctionnalités **attractive**
Non indispensable, bonne surprise
Ex.: qualité syst audio d'un véhicule

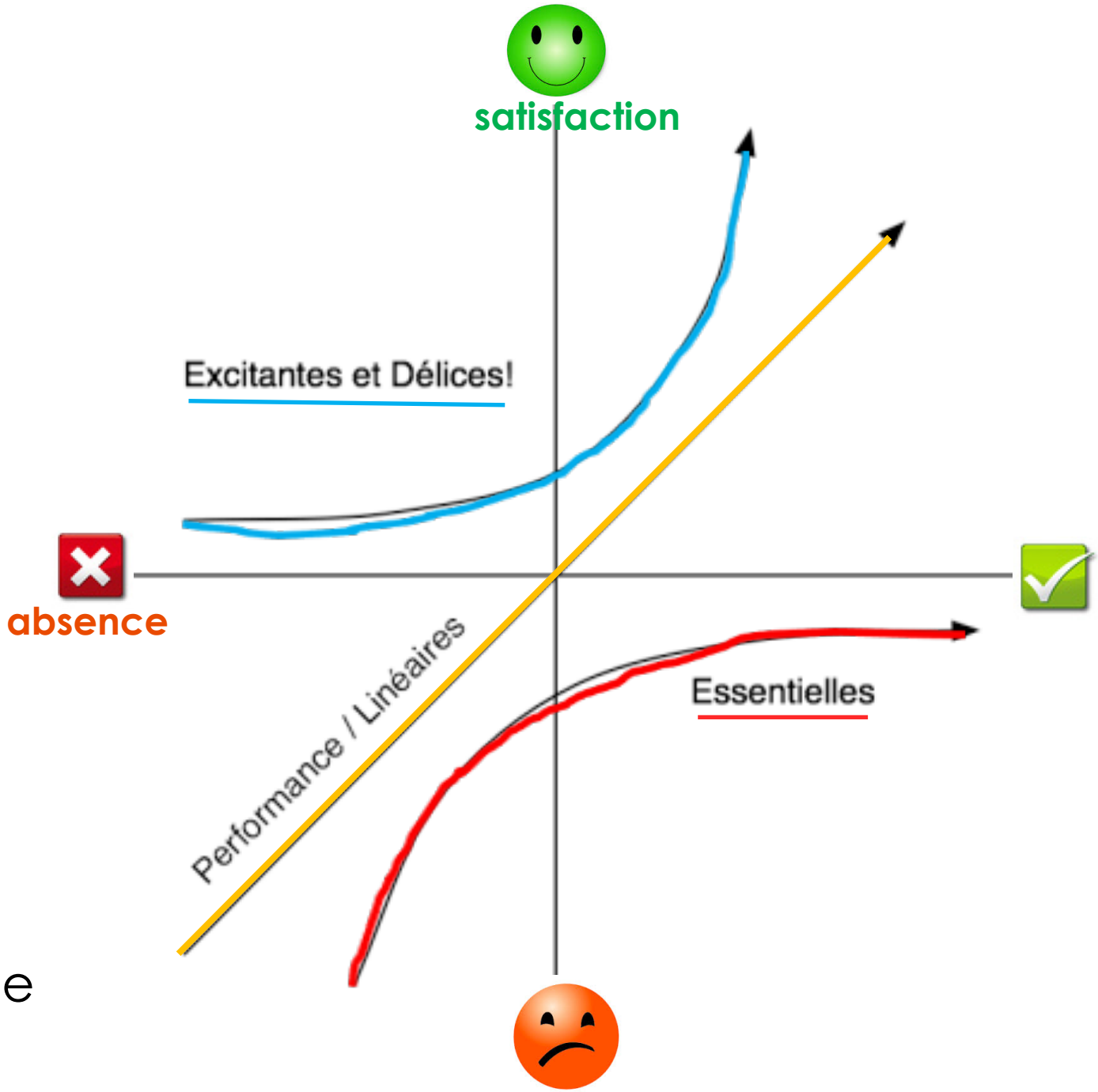


Illustration : café

- Vous entrez dans un bar et demandez un **expresso**

- Donner des illustrations pour :
  - Fonctionnalités **obligatoires**
  - Des raisons **d'insatisfaction**
  - Fonctionnalités **Linéaires**
  - Heureuse **surprise**

À quoi ça sert d'imaginer les causes d'insatisfaction ?

Illustration : café
par exemple

- **Fonctionnalités essentielles**
  - Bonne taille d'expresso
  - Café chaud
  - Mousse
  - Soucoupe
  - Cuillère
- **Insatisfaction forte**
  - Longue attente
  - Café froid
  - Soucoupe sale
  - Absence de sucre
- **Fonctionnalités linéaires**
  - Durée de l'attente
  - Empathie du serveur-euse
  - Qualité du café
- **Heureuse surprise**
  - Un gâteau / un chocolat
  - Un verre d'eau

Questionnaire de Kano : 2 parties

- Niveau de satisfaction des fonctionnalités **opérationnelles** (le fonctionnel).

Pour un produit 'Appareil Photo' (ici 9 personnes sur 15 ont trouvé qu'avoir un viseur était le min)

Si cet item est disponible :	1-Ça me fait plaisir	2- C'est le minimum	3- Ça m'est égal	4-Je l'accepte	5- Ça me dérange
Viseur	**4**	**9**		**2**	

- Ressenti des fonctionnalités **absentes** (le dysfonctionnel)

Si cet item est absent :	1-Ça me fait plaisir	2-C'est le minimum	3- Ça m'est égal	4- Je l'accepte	5- Ça me dérange
Viseur			**4**	**3**	**8**

- Ce **double questionnement** est la **clef de la méthode**
  - ***Satisfaction*** et ***insatisfaction*** ne sont pas ***symétriquement opposées***

http://christian.hohmann.free.fr/index.php/six-sigma/les-outils-de-la-qualite/476-le-diagramme-de-kano

Kano : grille

- Permet d'identifier les catégories de fonctionnalités parmi :
  - **O**bligatoires
  - **A**ttractives
  - **P**erformantes
  - **I**ndifférentes
  - **C**ontraires (hostiles)
  - ***D**éfaut (erreur)*

*Une fonctionnalité « Performante » est une fonctionnalité qui quand elle est **présente**, « ça fait plaisir » et quand elle est **absente**, « ça dérange » (par ex. du viseur : case P pour 12 personnes sur 15)*

		Absence de la fonction				
		1	2	3	4	5
Fonction présente	1	D	A	A	A	P
	2	C	I	I	I	O
	3	C	I	I	I	O
	4	C	I	I	I	O
	5	C	C	C	C	D

1: Ça fait plaisir
5: Ça dérange

CRITIQUES de KANO

- Long et fastidieux
- Utilisé en Agile pour prioriser quand beaucoup d'utilisateurs

2- Classer avec MoSCoW

(1) On prépare **4 fiches** :

(2) En équipe, on **place les User Stories** sur les fiches

- Soit l'un après l'autre, soit par vote : argumentation, débat, précisions

- Max **8** par fiche

REGLE SCRUM

Tous les items MUST devront être finis **sur les 2 premiers** sprints

Quantifier la **valeur** Client

- Une fois **la liste priorisée des user stories**, on attribue une valeur globale au projet (par ex. 100 ou 500 points) et on répartit les valeurs sur les US
- On vérifie qu'on est **cohérent** avec les priorités

265

230

70

On inscrit la valeur dans un angle de la User Story

Définir les besoins importants : Je retiens….

La somme des points Valeur définit la *business value* du projet

- Il s'agit de trier les US selon la **valeur Métier,** définie avec les utilisateurs ou le client
- Définir la valeur Métier n'est pas évident (notion d'utilité, de désir, de facilité et fréquence d'utilisation etc.)
- On hiérarchisera le *Backlog* en fonction de cette VALEUR, et de l'EFFORT (étape suivante)

Comment PRIORISER le BESOIN

- Différentes méthodes existent, ici on a vu :
  - La **méthode de KANO** (double questionnaire selon la présence / absence d'une fonctionnalité
  - **MoSCWo** (Must – Should – Could – Won't)

POUR EN SAVOIR PLUS…

Un exemple de priorisation Agile avec la matrice de KANO

https://blog.myagilepartner.fr/index.php/2017/01/09/quest-ce-que-le-modele-de-kano/

LA CAPTURE des EXIGENCES

Dév., testeurs, designers

ESTIMER
Définir l'effort nécessaire

Estimer le coût de réalisation

Estimer l'effort

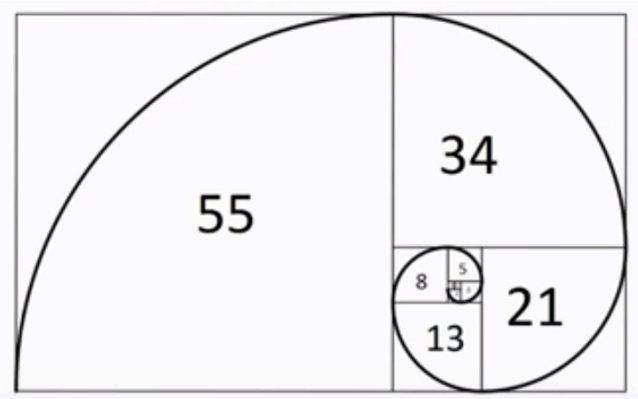
- Une fois les US affectées de valeur Métier avec le PO, l'équipe va **quantifier l'effort nécessaire pour les réaliser**

3 méthodes d'estimation

- Planning POKER
- T-Shirt Size
- Bucket System

1- Estimer Effort avec le Planning Poker

- Fonctionne avec le système de points de Fibonacci :
  - **1(facile), 2, 3, 5, 8, … (difficile)**
  - Chacun dispose des cartes de Fibonacci



- Fonctionnement
  - Les US sont au centre, on en retourne une
  - Chaque Dév retourne la valeur d'effort qu'il estime juste, tous en même temps
  - On discute des valeurs extrêmes
  - On se met d'accord sur l'effort de la story

2- Estimer l'effort avec la méthode T-SHIRT size

- Certains préfèrent attribuer un **niveau de difficulté** aux US et tâches plutôt que lui donner une *valeur d'effort*
  - Plus facile de comparer un tâche par rapport à une autre
- Méthode du T-Shirt Size : on choisit les catégories

  XS, S, M, L, XL
  - Soit définir une référence : M = 5 (par ex. 2 jours)
  - Soit choisir une **User Story étalon**, connue et maîtrisée, et lui affecter l'effort qui va bien. *Ex. : développer l'interface de saisie de commande Client estimé S*
  - Mesure de **l'effort « relatif »**
    - *l'interface de saisie des infos Client = 2x plus long, disons M*

Tshirt-size Méthode

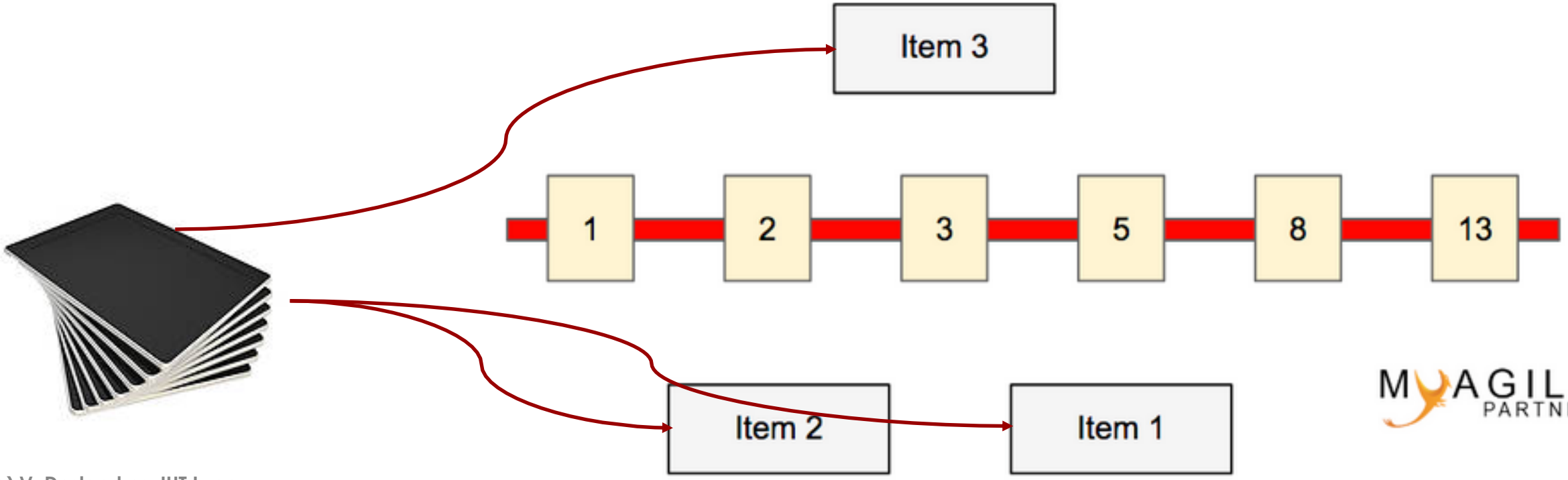
- Les tâches **XL** sont en général **floues**
  ➔ Les découper et repositionner
- L'objectif est de trier les *items* en complexité

- On attribue ensuite des **points** de Fibonacci aux catégories
  - Niveau difficulté *faible* → effort de développement *faible*
  - Par ex. :

XS : 1 S : 3 **M : 5** L : 8 XL : 13

3- Estimer l'effort avec le Bucket System

- Appelé aussi Extreme Quotation (vu en CVDA, s2)
- Très utilisé pour le **premier** chiffrage de tout un backlog en 1h max
- https://blog.myagilepartner.fr/index.php/2018/07/16/bucket-system/ ou https://www.youtube.com/watch?v=3kzJARYetFU

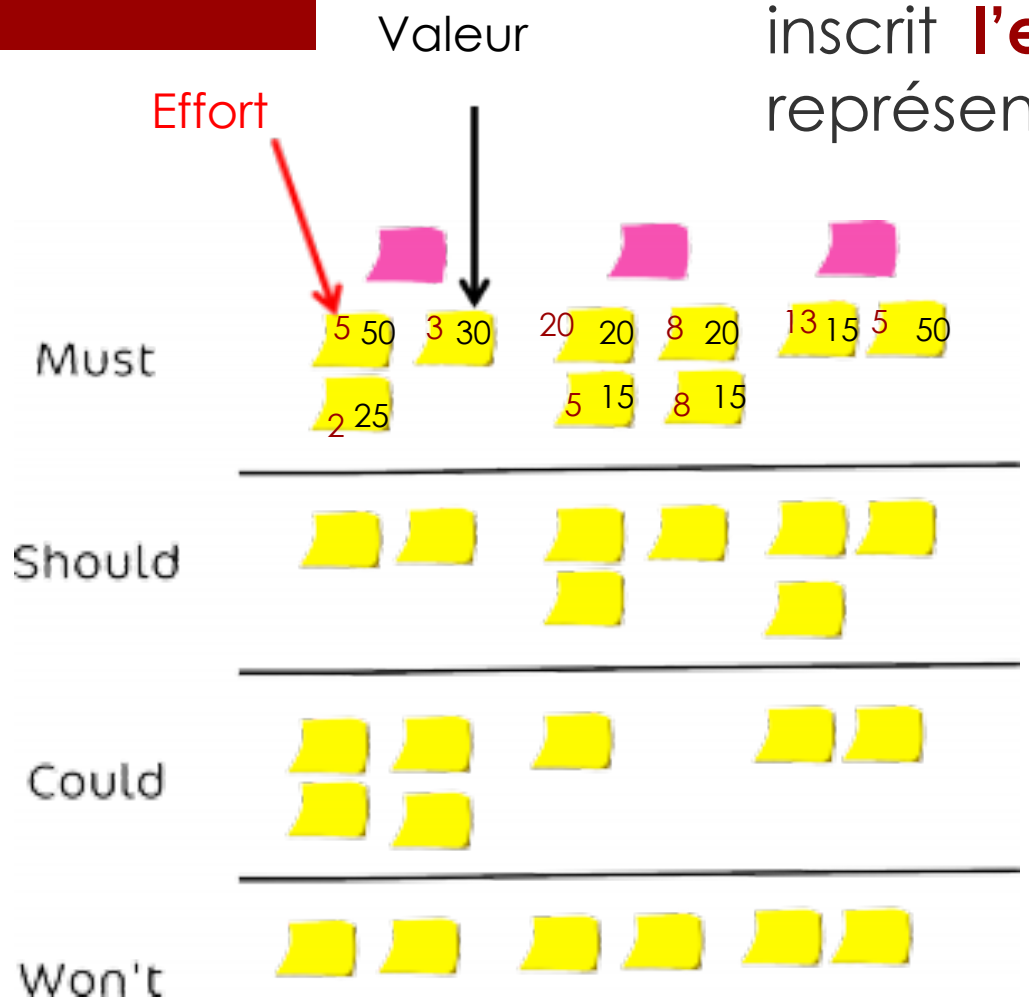


2 principes pour l'Estimation Effort : Faible précision / Marge collective

- **Faible précision**
  - Rester flou, inutile d'aller trop dans le détail de l'estimation (on verra + tard)
  - Ce qui compte c'est l'estimation **relative**
- **Marge collective**
  - Chaque développeur doit être **honnête** dans son estimation ; ne pas se donner « un peu de mou »
    - Risque de **surévaluation** de l'estimation
  - C'est au bénéfice de **l'équipe** qu'il faut donner de la marge
  - Il y **aura moins de gâchis**

Compléter les US

- Comme on l'a fait pour la VALEUR METIER, on inscrit **l'effort** sur un autre angle de la carte représentant la User Story



Productivité de l'équipe

- La **somme des points d'effort** = effort global nécessaire pour développer le produit
- En comptant **combien de points d'efforts sont réalisés par itération**, on a une idée de la **capacité à produire** du code de l'équipe
  - Une fois le projet fini, on connaît la **productivité** de l'équipe (nombre de points d'effort : indépendant du type de projet)

C'est différent de la *Business Value* (somme des points de Valeur Métier)

Un ex. de Backlog Produit

Backlog item	Acceptance Criteria	Effort	Value
US1 - En tant qu'internaute, je peux réserver l'hôtel en ligne	• Un email de confirmation est envoyé • Réservation doit être faite au min 24h avant date	8	**25**
US3 - Améliorer la gestion des exceptions	Pas de msg "Exception …" même en cas de pb	21	**10**
US4 - En tant que membre, je peux modifier les dates d'une réservation	…	8	**20**
US6 - En tant que membre, je peux annuler une réservation	• Un email de confirmation est envoyé • Ne peut être annulée qu'au moins 15 jours avant la date	11	**25**
US7 - En tant que Resp. Hotel, je peux voir les réservations à venir	…	21	**15**

Hiérarchisation du Backlog & Planifications

Release : lot

Sprint : itération

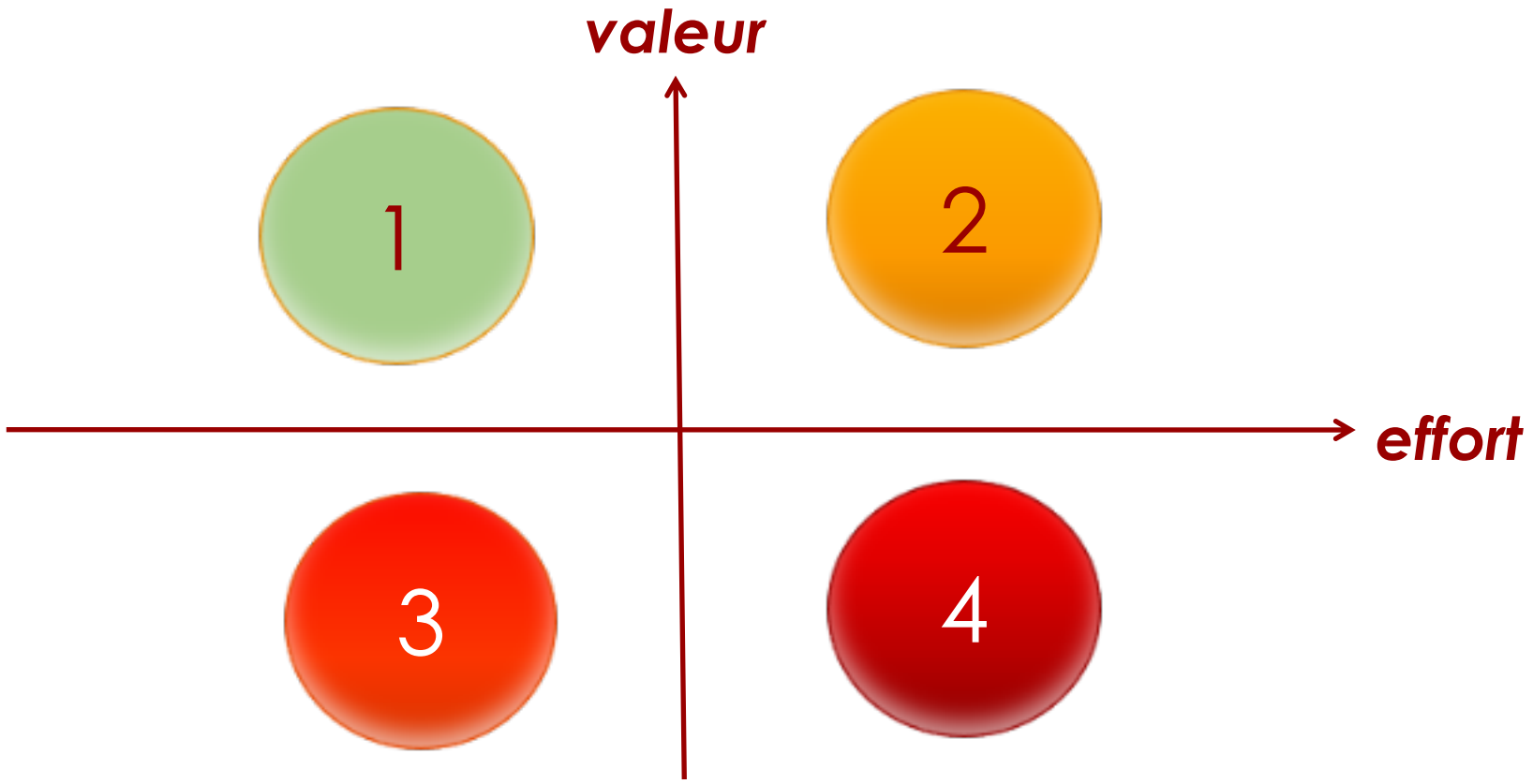
Hiérarchiser le backlog

- Pas en fonction de la VALEUR métier seule
- Mais du **couple (valeur, effort)**

- Objectif : faire ce qui est **le plus important** et **le plus facile** d'abord
- 2 méthodes pour hiérarchiser le backlog:
  - **Matrice Valeur / Effort**
  - **ROI** (*Return On Investment*)

1- Hiérarchisation du Backlog avec la **Matrice Valeur / Effort**

- Méthode la plus utilisée, la plus efficace



2- Hiérarchiser le Backlog Produit **avec calcul de ROI**

- Pour chaque User Story, on calcule le ratio :

Valeur Business de la US / Effort estimé

C'est le **ROI** (Return On Investment)

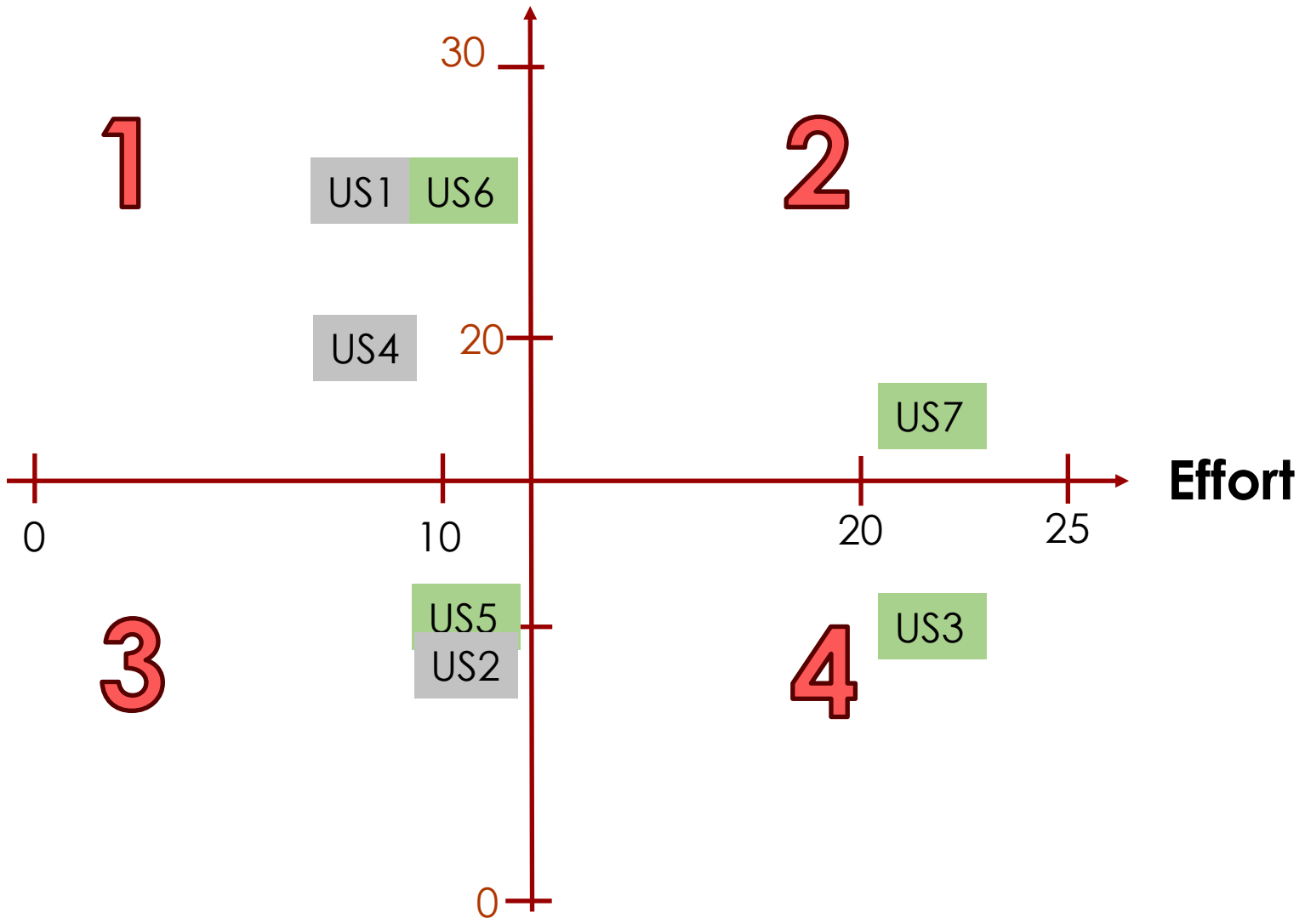
- Pour le premier sprint, on choisit les US qui ont le ROI le plus élevé

Avec les 2 méthodes, on obtient un **backlog de produit HIÉRARCHISÉ**

- On sait par quoi on va commencer à développer

Sur l'exemple

Centre du référentiel = valeur médiane de la Valeur Métier et de l'Effort



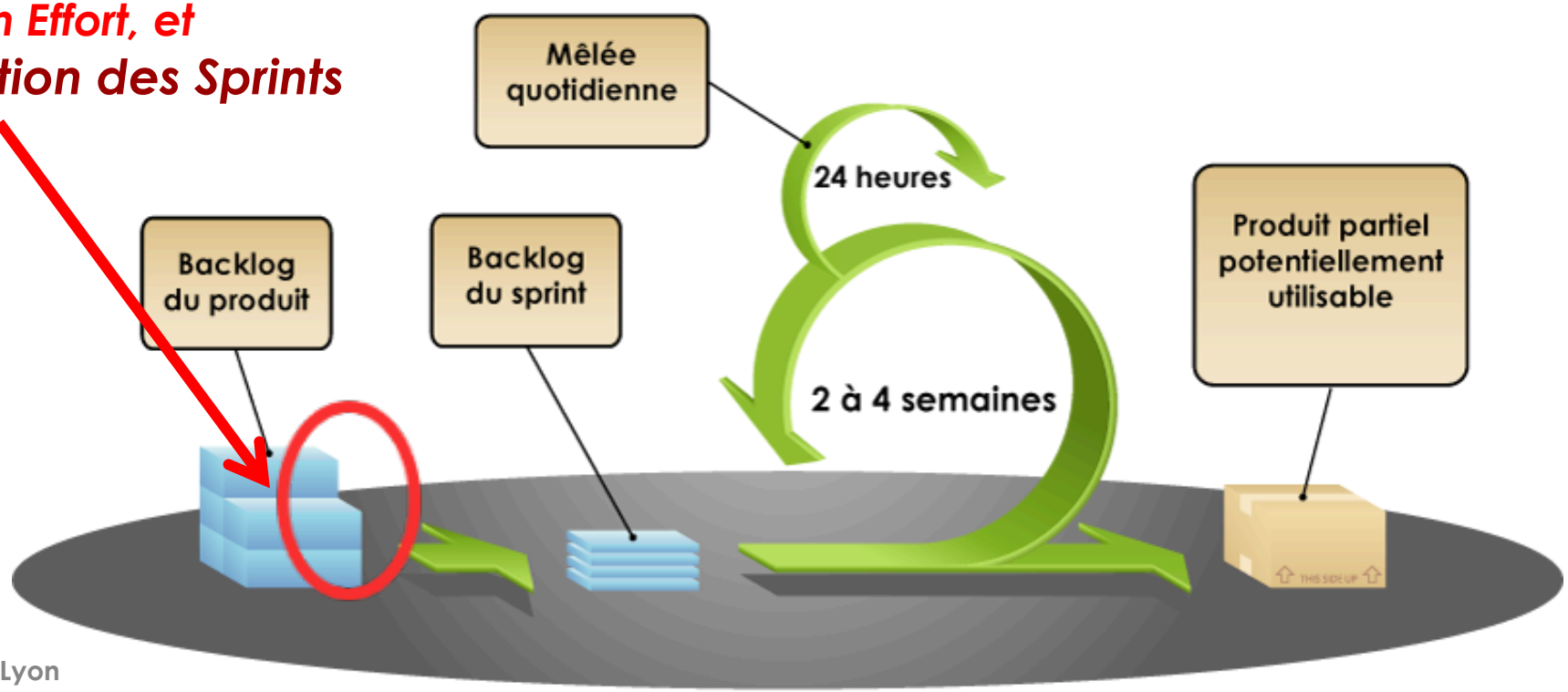
→ Backlog produit hiérarchisé

Priority	Backlog item	Acceptance Criteria	Estimate	Value
1	US1 - En tant qu'internaute, je peux réserver l'hôtel en ligne	• Un email de confirmation est envoyé • Réservation doit être faite au min 24h avant date	8	25
2	US6 - En tant que membre, je peux annuler une réservation	• Un email de confirmation est envoyé • Ne peut être annulée qu'au moins 15 jours avant la date	13	25
3	US4 - En tant que membre, je peux modifier les dates d'une réservation	…	8	20
4 …	US7 - En tant que Resp. Hotel, je peux voir les réservations à venir	…	21	15
7	US3 - Améliorer la gestion des exceptions	Pas de msg "Exception …" même en cas de pb	21	10

Planification des Sprints

- Quand ? Une fois le Backlog hiérarchisé
- En général : 4 à 5 User Stories par sprint

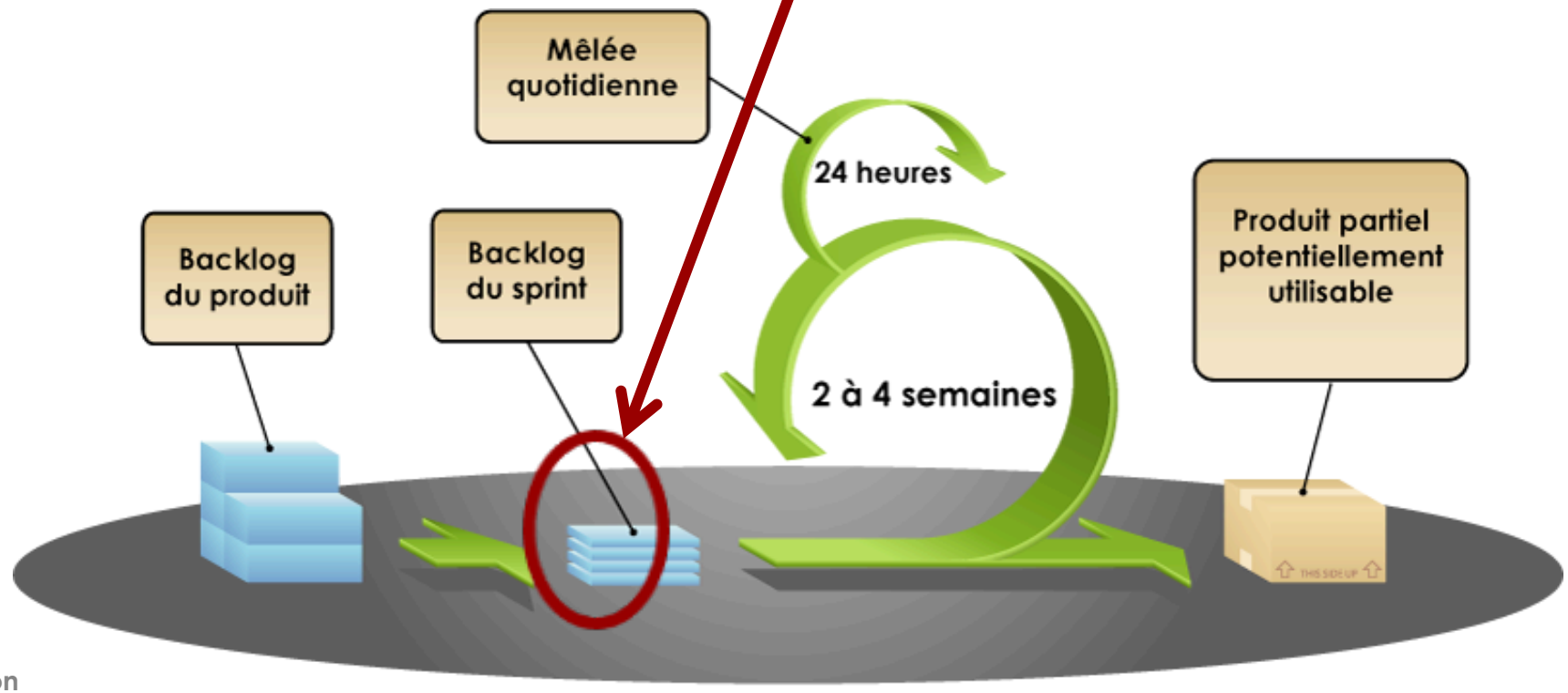
Priorisation Valeur + estimation Effort, et Planification des Sprints



Planning Game (Planification **du** Sprint)

- Quand ? En **début** de sprint
- Durée : max 1h pour un sprint de 1 semaine (2h pour 2 semaines)

Planning Game : découpage des US en tâches, répartition des tâches



Planning Game: un exemple

- Les tâches sont identifiées et **estimées**
  - (parfois en heures comme ici)

ETQ touriste dans la région, je veux **voir les photos** des hôtels afin de savoir si je vais m'y rendre

- Modéliser et implémener la couche de persistance (8h)
- Coder l'IHM (4h)
- Ecrire les tests (4h)
- Coder la classe *foo* (3h), etc.

Priorisation / Estimation : Je retiens

- On peut **prioriser** le besoin (les User Stories) avec 2 méthodes : Kano et MosCow
- **Estimer** l'effort de réalisation de chaque US avec 3 méthodes : Planning Poker, T-shirt Size et Bucket system
- **Hiérarchiser** le backlog avec la Matrice Valeur/ Effort ou le ROI
- **Planifier les sprints** (et les releases)
- Faire le **Planning Game** en découpant les US en tâches dont on évalue aussi l'effort