



Conduite de Projets

Ingénieurs 2^o et 3^o année

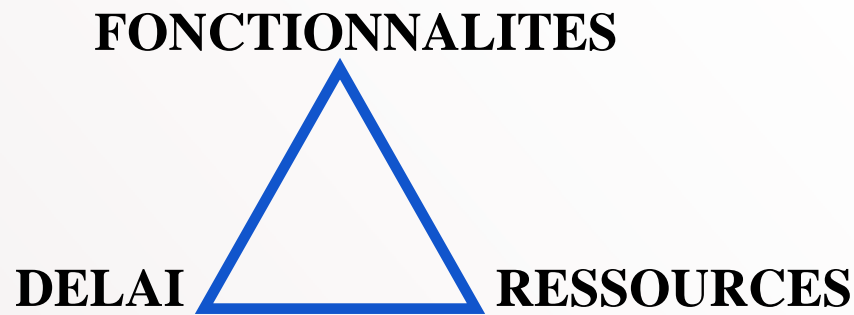
III a – Planification agile

1 – Pourquoi Planifier ?



❑ Planifier = Rechercher un optimum

- Quel produit va-t-on construire à travers le projet ? Quelle est la combinaison optimale de ces 3 éléments ?



- A partir d'un point d'équilibre, toute modification d'un des 3 éléments affecte au moins un des deux autres.

Objectif de la planification

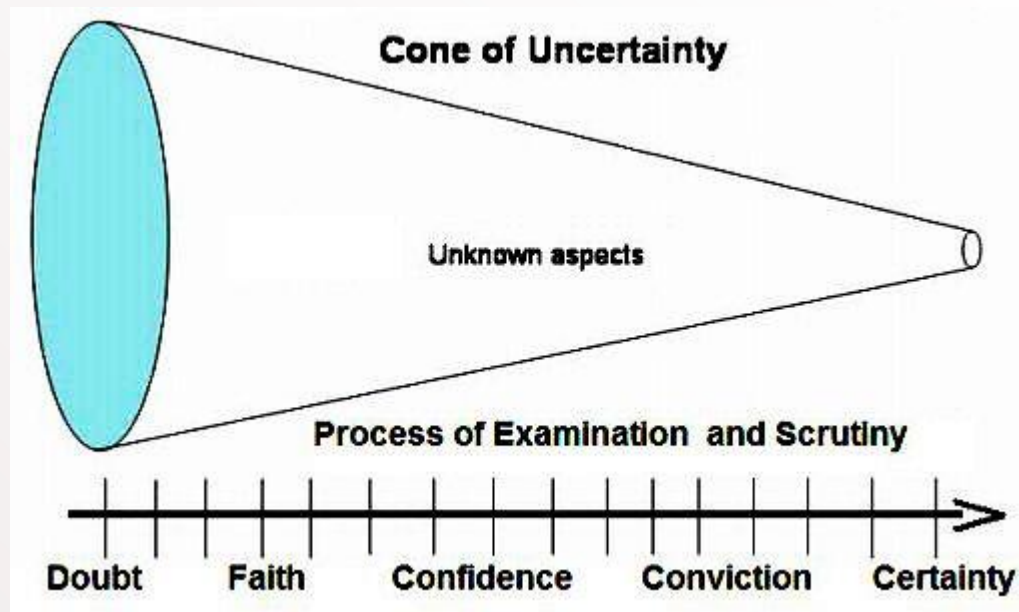
❑ Réduire les risques et l'incertitude

- estimer les charges = réflexion sur la complexité → découverte des risques.
- bon ordonnancement → découvrir progressivement les besoins réels et ses propres capacités à réaliser le produit.

❑ Communiquer et prendre des décisions

- informer de façon transparente tous les décideurs de l'avancement réel → relation de confiance.
- décider de lancer ou poursuivre le projet, adapter le besoin, changer de technologie,....

2 – Le cône d'incertitude



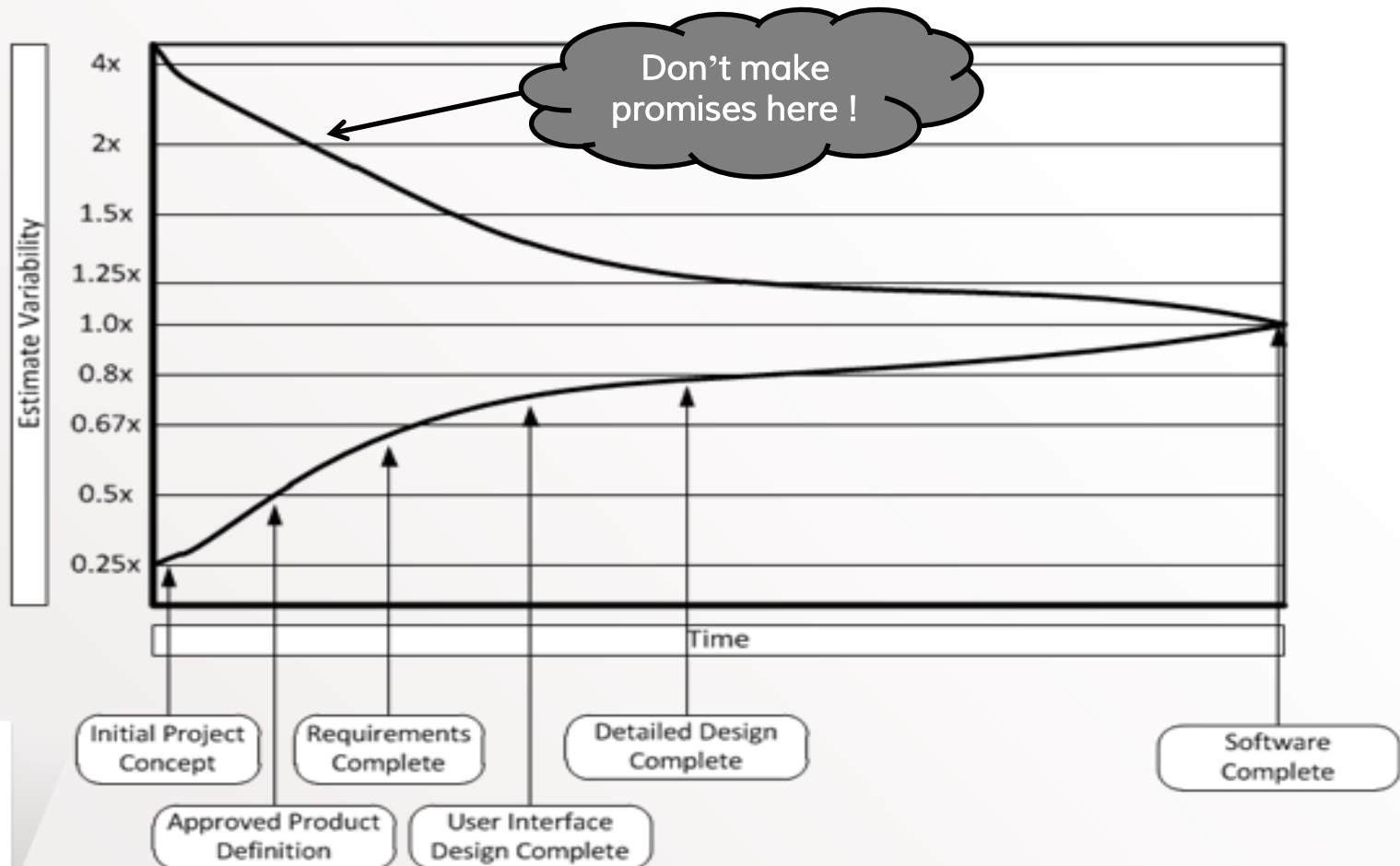
Planification traditionnelle: le tout prédictif

- ❑ Un plan détaillé, à long terme, établi en amont
 - Pas ou peu remis en cause par la suite.
 - Les fonctionnalités, le délai et les ressources sont fixes.

- ❑ Cette approche échoue car:
 - On ne peut pas prévoir avec précision à LT (cf. cône)
 - Les inévitables changements de besoin sont exclus
 - Le CP est jugé sur le respect du plan initial et s'y accroche
➔ aucune souplesse

Le cone d'incertitude (étude empirique)

- ❑ Phases amont: estimations très largement fausses.
- ❑ Spécifications nécessaires pour réduire l'incertitude.



3 – Estimation Agile



Story Points

“

« Mieux vaut être
grossièrement juste que
précisément faux. »

(John Keynes)




❑ Story Points: mesure relative

- Unité de mesure exprimant la taille globale d'un WI.
- Valeur relative : une US à 2 SP \approx en gros deux fois plus d'effort qu'une US à 1 SP (**inutile d'être précis !**).
- Dépend de la quantité de travail, de la complexité, des risques, etc...
- Partir d'une US de taille moyenne, puis en déduire la taille des autres.
- Exemples...

❑ Vitesse

- Vitesse d'avancement de l'équipe = somme des SP des US complétées dans une itération.
- Exemple:
 - ✓ Vitesse de l'équipe = 30
 - ✓ Chaque TM a la même vitesse (10)

Itération i

TM 1	
TM 2	
TM 3	

□ Vitesse → durée

- Ex: \sum de toutes les US d'un projet = 150 SP
- vitesse de l'équipe = 15 SP par itération de 2 semaines
- durée projet = 10 itérations de 2 semaines = 20 semaines

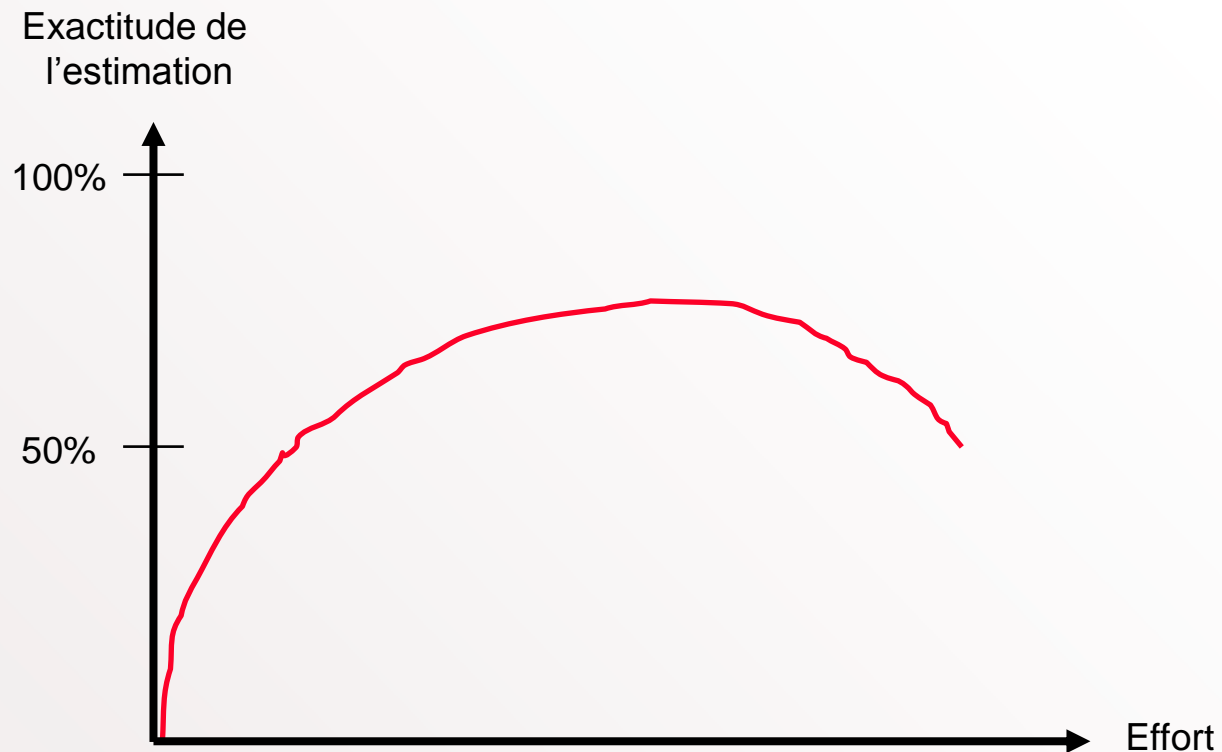
□ La vitesse « corrige » les erreurs

- Ex: on s'aperçoit après 2 itérations que la vitesse est en fait de 10 SP
- → recalculer la durée totale du projet (15 itérations) sans avoir à refaire les estimations de taille.

□ Exemple...

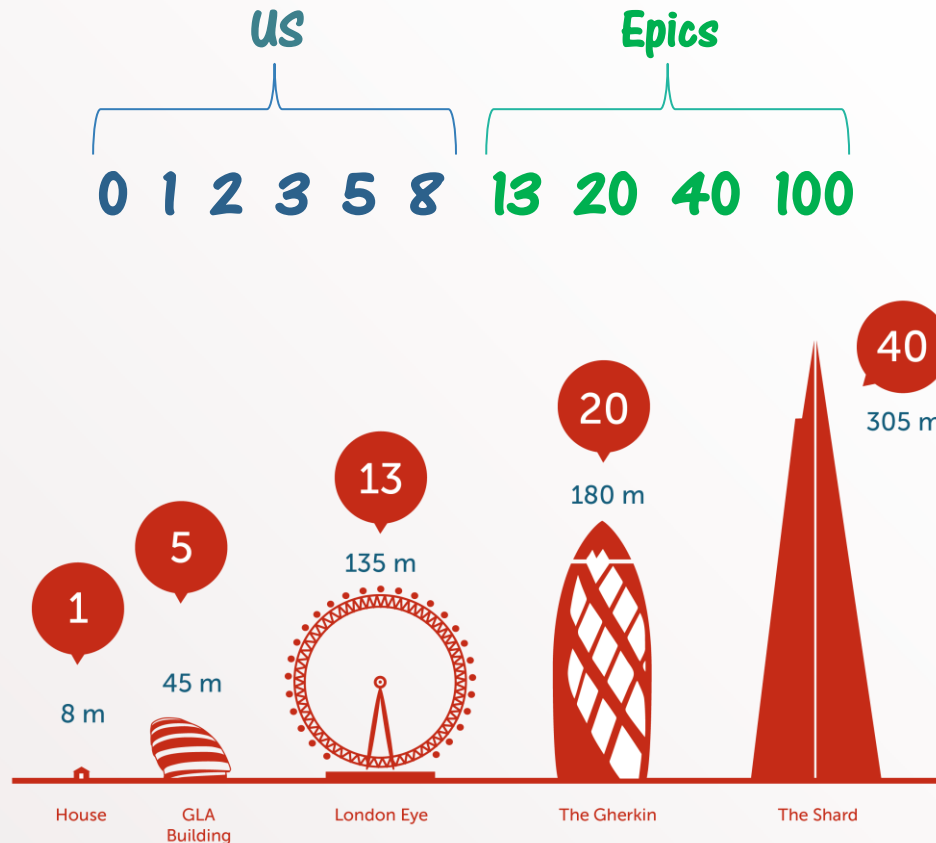
Effort d'estimation

- ❑ Consacrer l'effort juste nécessaire à l'estimation:
 - quel que soit l'effort on n'atteindra pas 100% d'exactitude
 - effort minimal → obtenir une estimation moyenne
 - pousser trop loin l'effort dégrade le résultat



Echelles d'estimation

- ❑ Pour les US on peut utiliser la suite de Fibonacci (1, 2, 3, 5, 8)
- ❑ Pour les fonctionnalités de haut niveau qui ne seront estimées en détail que plus tard (Epics): 13, 20, 40, 100



❏ Techniques d'estimation

- Opinion d'expert
- Analogie avec des US réalisées antérieurement
- Planning poker : estimation collective

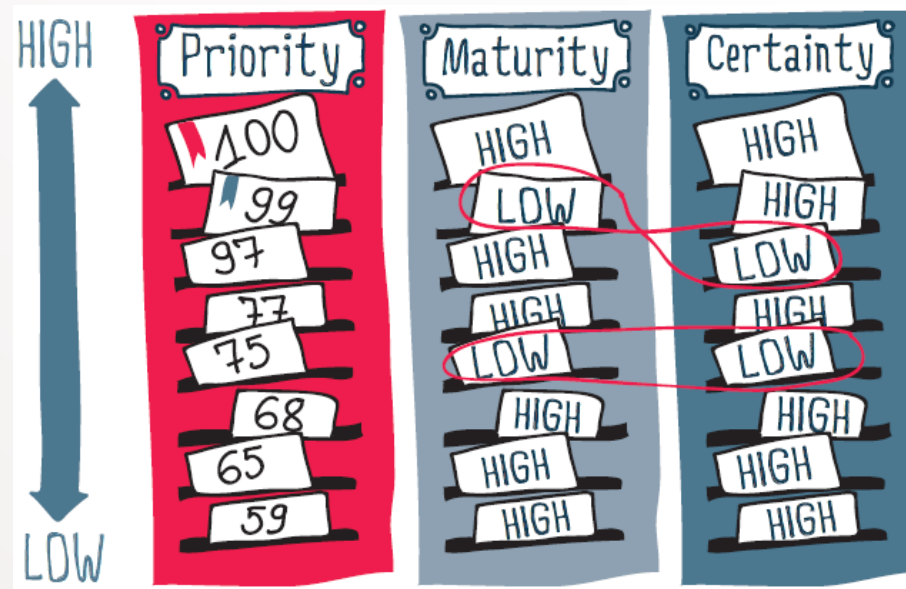


*Désagréger la US/Feature en ensembles plus petits
augmente largement l'effort*

❏ Ré-estimer

- Ré-estimer en cours de projet **uniquement** si la taille relative des US a changé.
- Sinon, ajuster **via la vélocité**.

4 – Prioritisation



- ❑ Priorisation des Epics/US = déterminer dans quel ordre on veut les réaliser.

- ❑ Repose principalement sur les 2 critères suivants:
 - **Risque:** non maîtrise technologique, ne pas savoir répondre à une contrainte forte (performance, volumétrie,...)
 - ✓ Prioriser = lever l'incertitude et éviter de modifier en fin de projet l'architecture (recoder des composants centraux,).

 - **Valeur Client:** valeur financière ou souhait
 - ✓ Prioriser = Satisfaire le client + s'assurer qu'on construit le produit attendu (besoins compris).

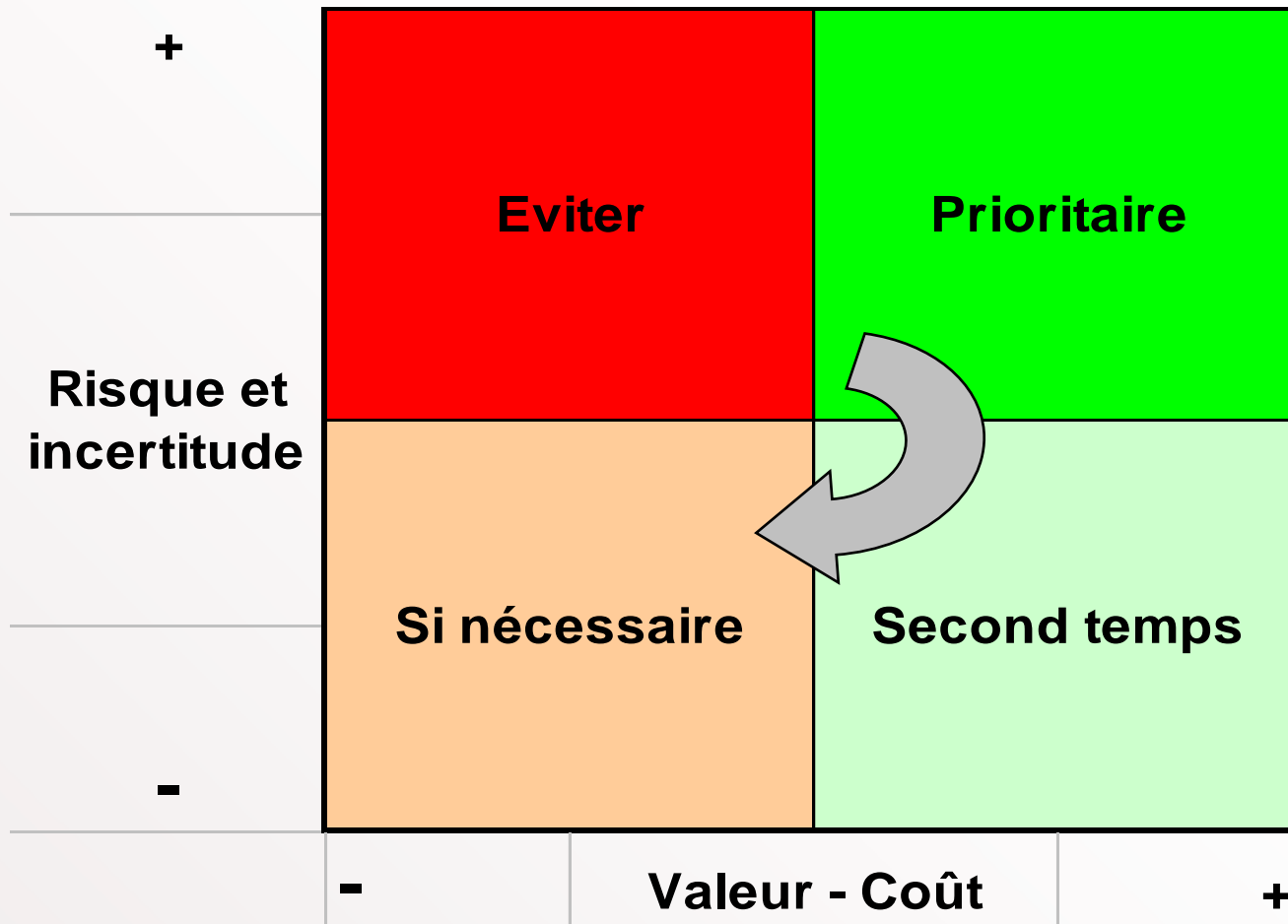
- ❑ Si la valeur des 2 critères est antinomique pour un même Item un arbitrage est nécessaire.

Epics	Risque	Valeur Client	Ordre de réalisation
Rechercher des Ouvrages	Moyen	Forte	2
Gérer son Panier	Faible	Forte	3
Effectuer une Commande	Fort	Moyenne	4
Consulter commandes en cours	Moyen	Faible	7
Consulter l'aide en ligne	Faible	Faible	8
Maintenir le Catalogue	Fort	Forte	1
Maintenir les informations éditoriales	Faible	Moyenne	5
Maintenir le Site	Faible	Moyenne	6



Modèle plus élaboré: combinaison de critères

- ❑ Combiner les 4 critères suivants
 - La valeur client (cf. section suivante)
 - Le coût du développement de la fonctionnalité (cf. estimation taille)
 - Les risques levés par le développement de la fonctionnalité
 - La connaissance acquise au cours du développement de la fonctionnalité qui permet de réduire les incertitudes
 - ✓ connaissance sur le besoin du client (via les revues et démonstrations)
 - ✓ connaissance sur la maîtrise de l'équipe (notamment la maîtrise de la technologie via les premiers développements)



NB: $\text{Valeur} - \text{Coût} = \text{Marge (ou profit)}$



Evaluer la Valeur Client des Epics/US

- ❑ Diverses modèles peuvent être utilisés:
 - MoSCoW
 - ✓ Must have, Should have, Could have ou Would have
 - Kano
 - ✓ Must be, Attractive, Indifferent, ...
 - Gain financier procuré par la fonctionnalité
 - Poids relatifs (Cf. page suivante)

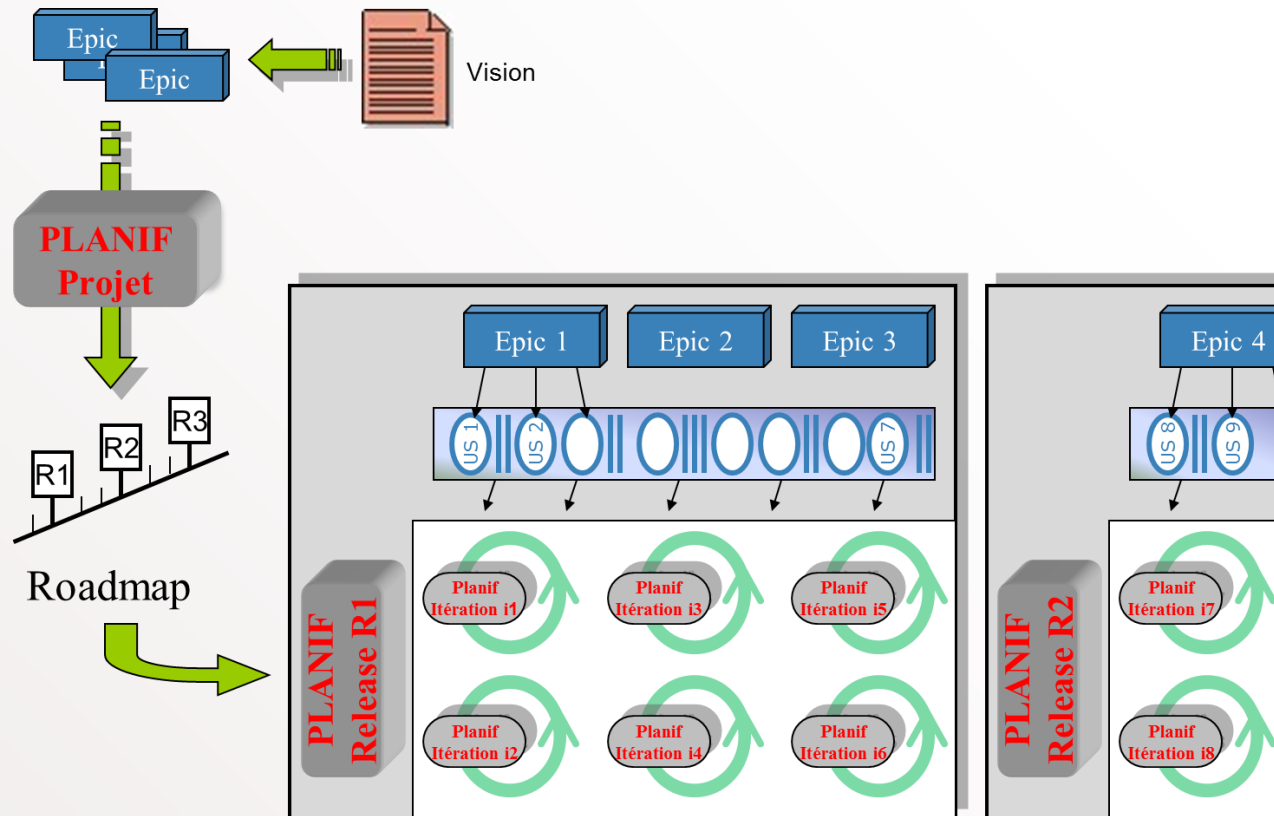


❑ Modèle des poids relatifs

- Il est basé sur des estimations (en points relatifs, de 0 à 9) de la désirabilité des fonctionnalités obtenues par questionnaire ou jugement d'expert
 - ✓ Bénéfice apporté par la fonctionnalité
 - ✓ Pénalité en cas d'absence de la fonctionnalité

Epic	Bénéfice	Pénalité	Valeur totale	% Valeur	Taille estimée	% Coût	Priorité
Epic 1	2	9	11	35%	13	28%	1,26
Epic 2	5	2	7	23%	13	28%	0,80
Epic 3	9	4	13	42%	20	43%	0,96
TOTAL	16	15	31	100%	46	100%	

5 – Planification agile



Démarche générale

Planification Multi-Niveaux

1 à 2 fois
par an

Macroscopiques



(mois)

Objectif/
Epic

1

Product
Roadmap

1 fois
par Release

Story Points



(jours)

Epic → US

2

Release

1 fois
par Itération



Heures

US → task

3

Itération

Fréquence

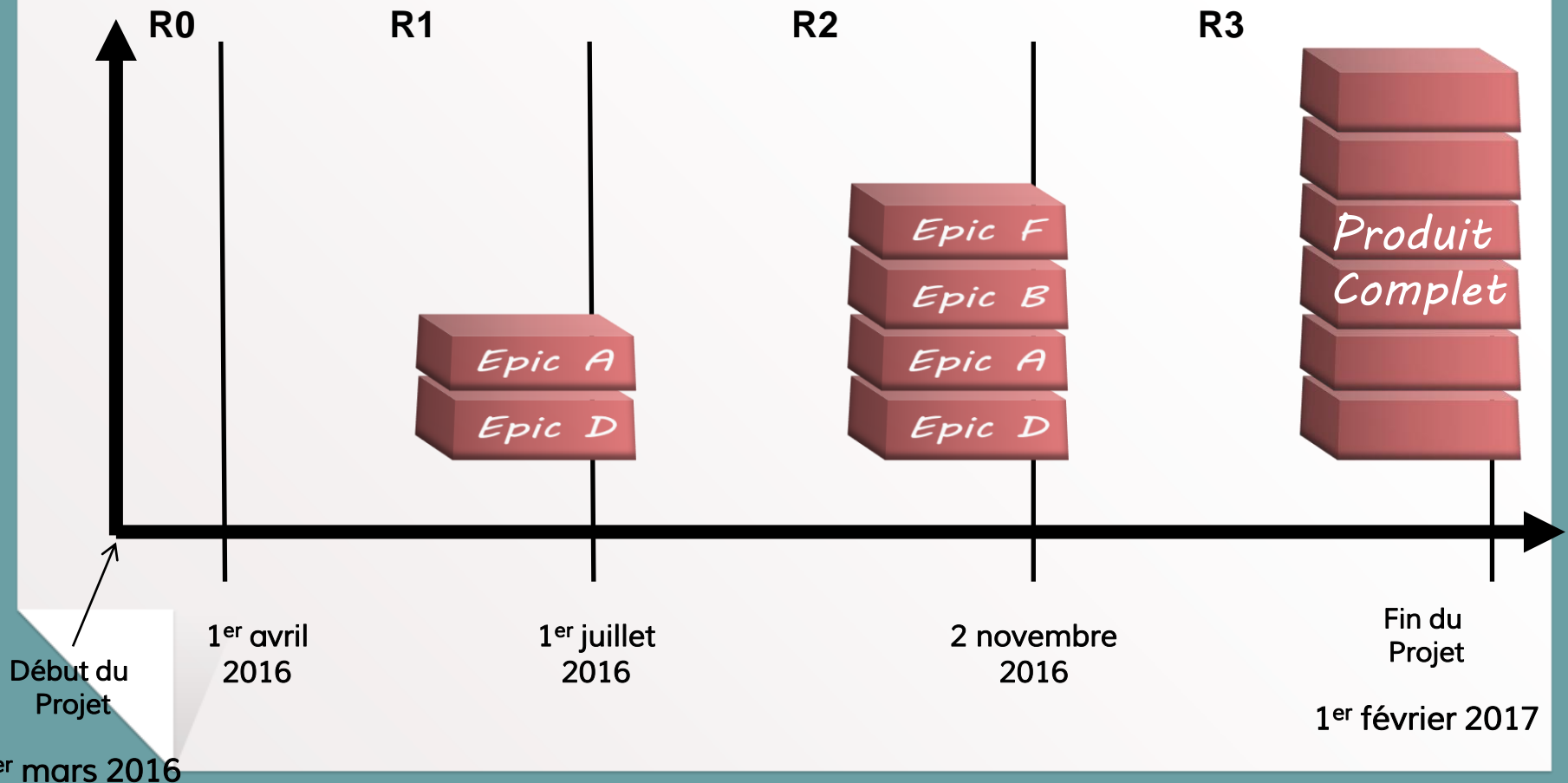
Estimations

Granularité

Niveau

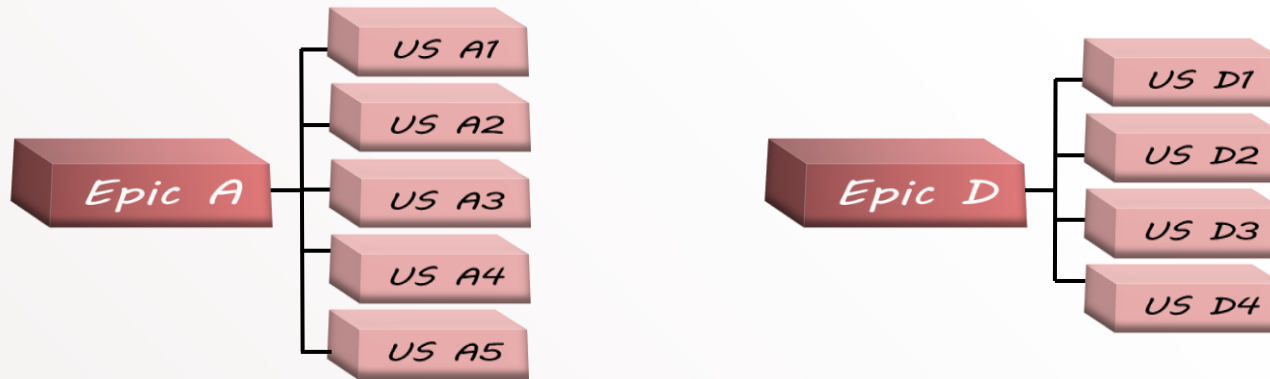
1. Planification niveau product (Roadmap)

- ❑ Macroscopique et imprécise mais donne des objectifs, un cadre.
- ❑ Roadmap = Cycle de vie macro du projet (Releases: contenu, date)

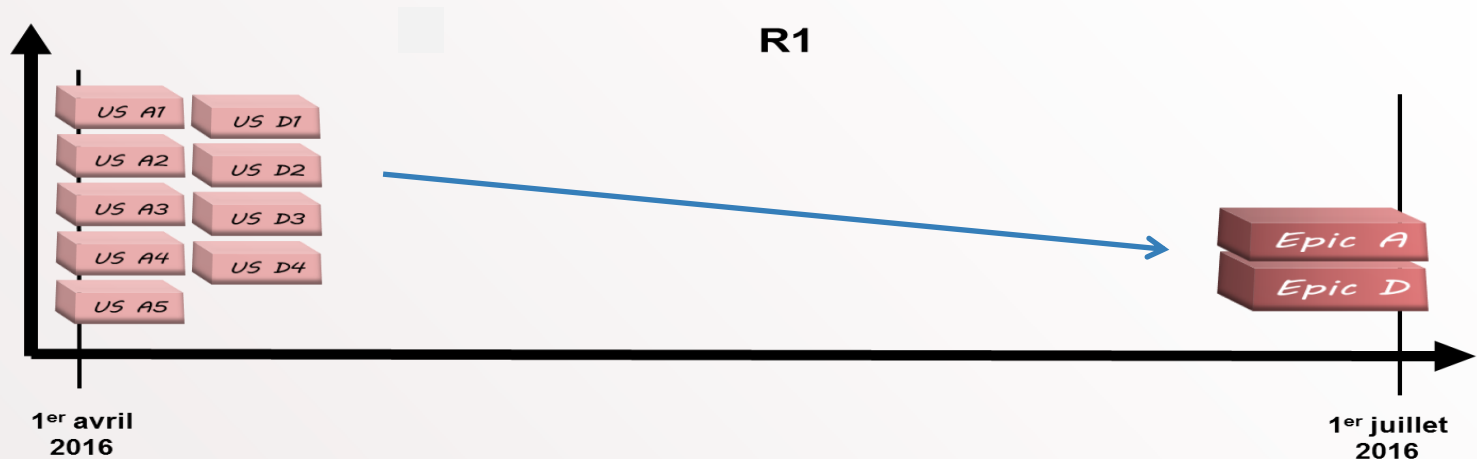


2. Planification niveau Release

- ❑ Splitter les WI macros en WI micros (pour la Release à venir)

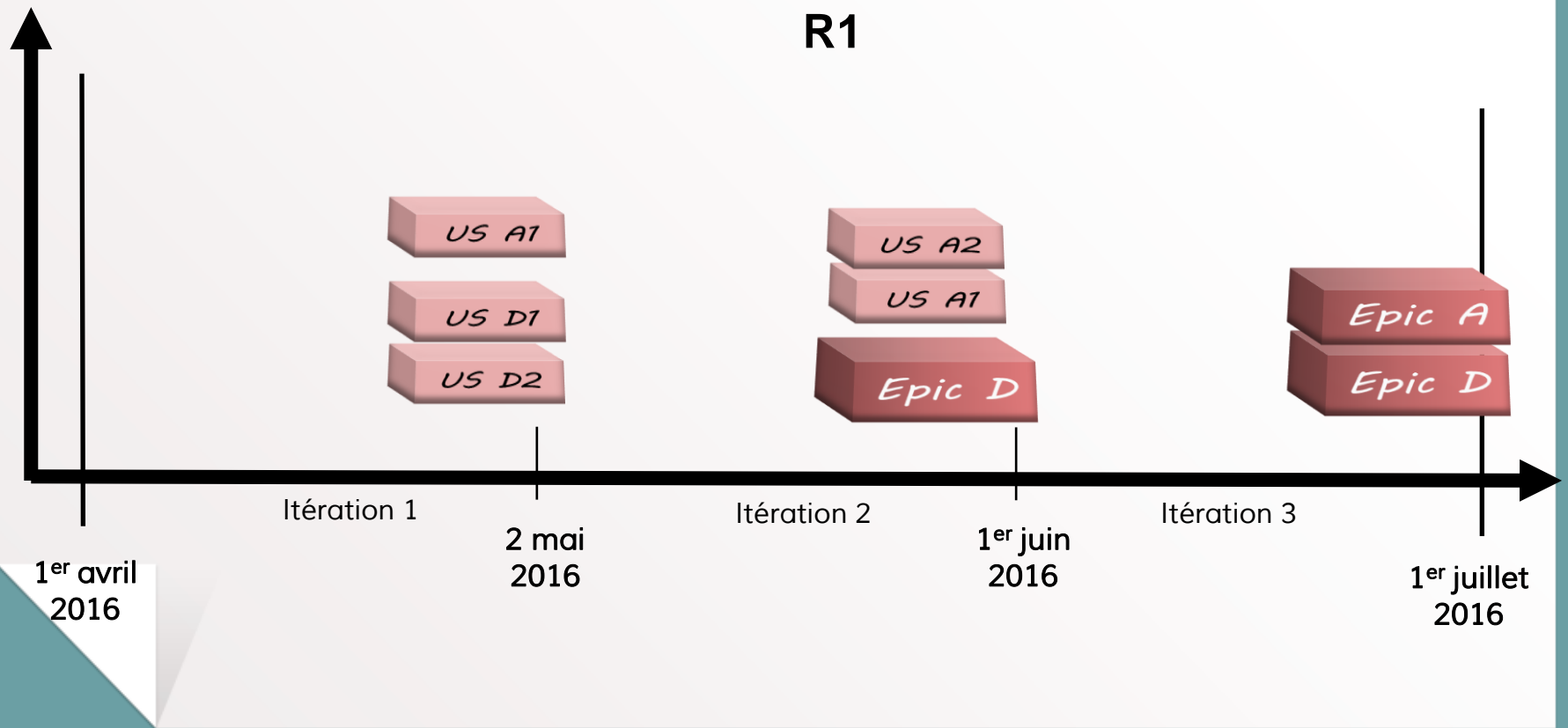


- ❑ On obtient un backlog pour la Release

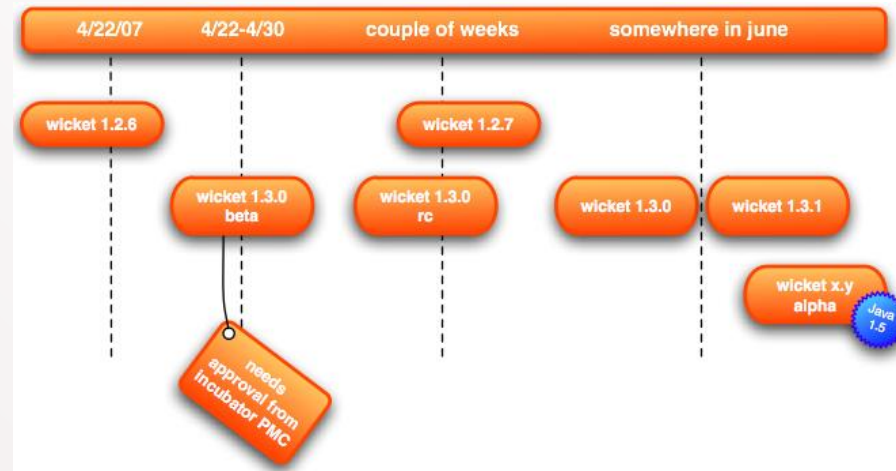


3. Planification niveau Itération

- ☐ Ne planifier que l'itération à venir (3 sessions sur l'exemple ci-dessous)
- ☐ Choisir les WI micros
- ☐ Détailler leurs tasks



6 – Planification au Niveau Projet/Produit



Le Roadmap

Roadmap

- ❑ Premier niveau de planification, établi **tout en amont** du projet et révisé très occasionnellement
- ❑ Etabli par des **décideurs de haut niveau** (direction), des PO et des team managers (experts projets)
 - La taille globale du projet est estimée en points
 - la durée totale est aussi estimée
- ❑ On y planifie des releases:
 - externes : livrées à l'utilisateur final
 - internes : livrées aux Key Users ➔ feedback

“

Roadmap

Planification imprécise,
repose fortement sur
l'expérience pure.

Mais fournit:

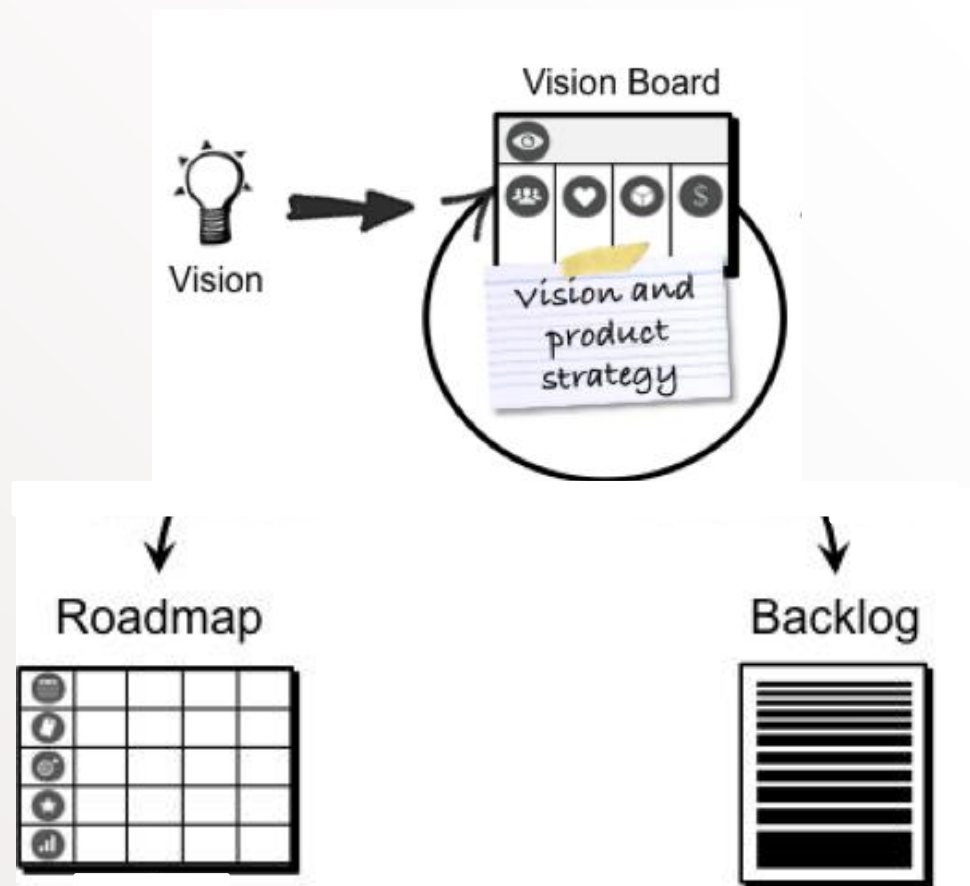
- des objectifs à atteindre
- des repères → articuler les planifs de niveau inférieur

Template de Roadmap

01/06/2016	01/10/2016	01/02/2017
Release 1	Release 2	Release 3
Thème Pour <tel client> cette release apporte <telle valeur métier> <pour telle raison>	Thème Pour <tel client> cette release apporte <telle valeur métier> <pour telle raison>	Thème Pour <tel client> cette release apporte <telle valeur métier> <pour telle raison>
Périmètre * Fonctionnalité 1 * Fonctionnalité 2 * Fonctionnalité 3	Périmètre * Fonctionnalité 4 * Fonctionnalité ...	Périmètre * Fonctionnalité ... * Fonctionnalité ...
Critères d'acceptation * tests acceptance * scope * budget	Critères d'acceptation * tests acceptance * scope * budget	Critères d'acceptation * tests acceptance * scope * budget
Ressources * team manager : * product owner : * team members :	Ressources * team manager : * product owner : * team members :	Ressources * team manager : * product owner : * team members :
Vélocité 50 points	Vélocité 50 points	Vélocité 50 points
Dépendances, Obstacles et Risques	Dépendances, Obstacles et Risques	Dépendances, Obstacles et Risques

Vision Doc / Product Backlog / Roadmap

- ❑ Périmètre = Fonctionnalités et contraintes de haut niveau (issues du Vision document).



- Les Fonctionnalités de haut niveau (Epics) sont:
 - ✓ éventuellement estimées séparément (20, 40, 100 SP)
 - ✓ priorisées
 - ✓ affectées aux différentes releases planifiées

- Thème des releases = Ensembles de fonctions cohérents, **exploitables** et significatifs pour l'utilisateur

Exemple: étude de cas WebBank

Fréquence des releases

☐ Les releases doivent être assez fréquentes →

- réactivité importante au changement de besoin
- réduction rapide du risque « compréhension du besoin »

→ déterminer la fréquence optimale en fonction du besoin de feedback, de la réactivité du marché, etc...

☐ Schedule wins: les dates sont fixées et le contenu de la release est estimé (parfois le contraire).

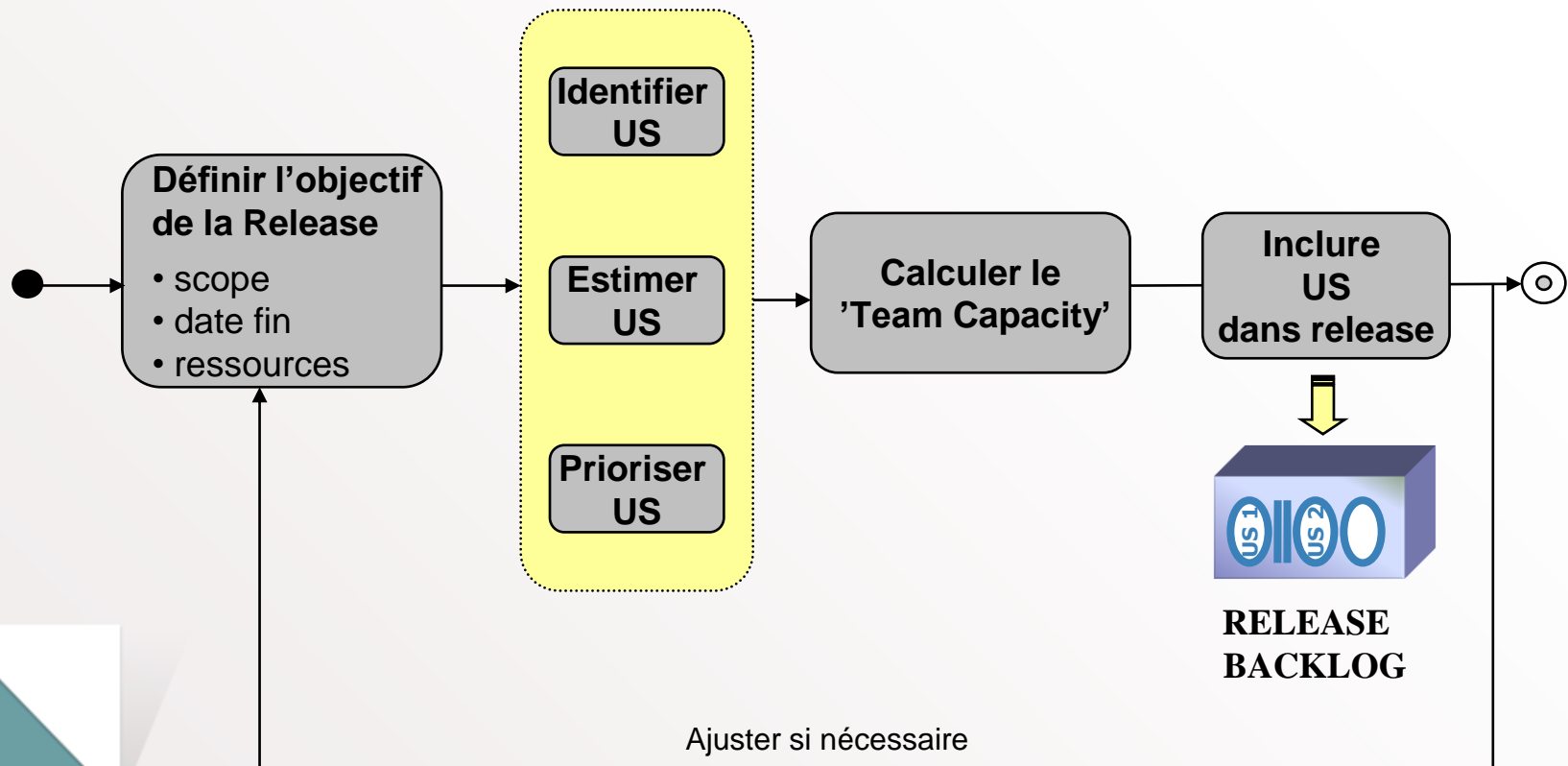
☐ Placer les releases à des **dates régulières** (ex: tous les 4 mois) ou bien selon le marché ou des **événements** précis (salon, ...).

7 – Planification au Niveau Release



Session de planification de Release

- ❑ Session de **2 jours**, en tout début de Release, impliquant toute l'équipe.
- ❑ La présence du **PO** est **indispensable**.



Définir l'objectif de la Release

- ❑ Roadmap: objectifs initiaux
- ❑ Session de planification: ajustement
 - Approfondir la compréhension du besoin et de la solution
 - ➔ Affiner les estimations
 - ➔ Confirmer ou **ajuster les objectifs**

❑ Privilégier :

- Date de fin : « schedule driven »
 - ➔ Date non négociable, périmètre le plus grand possible

OU

- Périmètre : « scope driven »
 - ➔ Périmètre non négociable, livré le plus vite possible

- ❑ Les ressources font l'objet d'hypothèses de départ qui peuvent être ajustées (jusqu'à un certain point).

Identifier les US

[seulement sur le scope de la release]

- ❑ Préparer la session (quelques jours avant):
 - Le PO recueille les besoins détaillés auprès des utilisateurs/clients
 - Les TM préparent leurs questions

- ❑ Session: Approfondir les connaissances
 - Le PO présente longuement les Epics
 - Questions/Réponses sur le besoin fonctionnel
 - Hypothèses/Débats sur les solutions techniques

Split: Epics -> User Stories



❑ Décomposer les Epics en US*

- Suffisamment **Petites**

- ✓ Tenir dans une seule itération
- ✓ Voire seulement 2 ou 3 jours de travail

- « **Valuable** »

- ✓ L'US doit apporter de la valeur au produit*
- ✓ Le PO doit pouvoir l'accepter*

** Plus généralement on parlera de Work Items (WI) de bas niveau. Le terme « US » est restrictif car il ne concerne que les Work Items fonctionnels.*

D'autres types de WI (technical work, ...) peuvent ne pas faire partie du produit et n'ont pas à être acceptés par le PO.

Techniques de Split

❑ Splitter les UC en scénarios

- Identifier tous les cas de figure possibles
 - ✓ Nominal: cas le plus courant
 - ✓ Alternatifs: autres façons d'atteindre le but
 - ✓ Exceptions: tous les cas d'erreurs, fausses manips, échecs



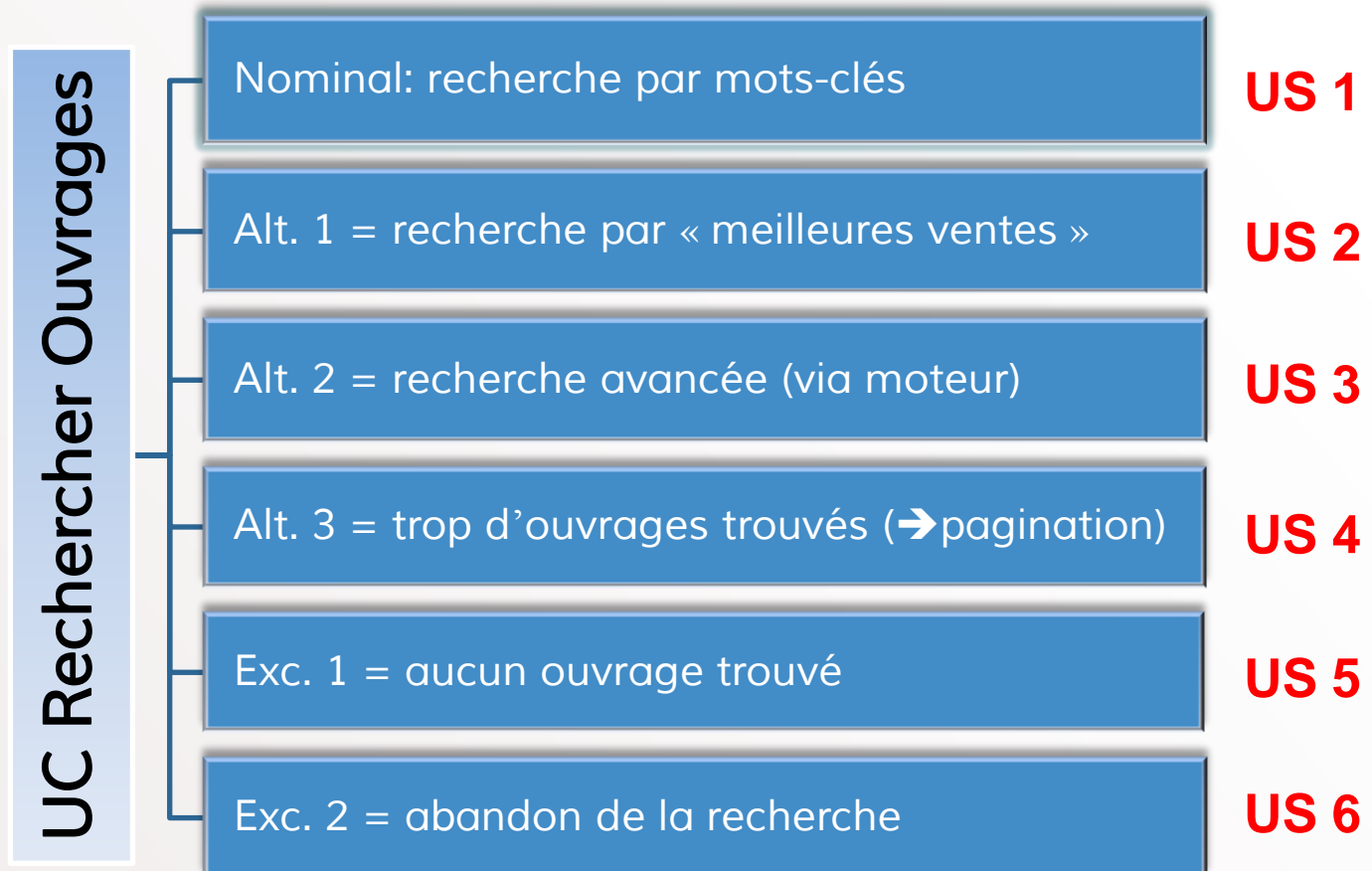
L'utilisateur va commettre des erreurs, ou faire des choix inattendus.

Si les exceptions ne sont pas gérées → plantage

❑ Exemple de Split de UC

Cf. chapitre IV c - « Modélisation agile »

- S.u.d = librairie en ligne



❑ Splitter un scénario

- scénario trop gros (souvent le cas du nominal) →
2 ou 3 steps = une US

Exemple: scénario nominal = recherche par mots-clés

1.	L'Internaute lance une recherche rapide à partir d'un ou deux mots-clés : un thème, un titre, le nom d'un auteur. Il peut également saisir directement un numéro ISBN.
2.	Le Système affiche une page de résultat. Les ouvrages sont classés par défaut par date de parution, le plus récent en premier.

US 1

3.	L'Internaute sélectionne un ouvrage.
4.	Le Système lui présente une fiche détaillée pour l'ouvrage sélectionné. On y trouvera en particulier une image, le code ISBN, la table des matières, un résumé,...
5.	L'Internaute met l'ouvrage de côté dans son panier virtuel.

US 2

❑ CRUD(S): split par opération

- Create = Nominal
- Read, Update, Delete, Search = Alternatifs

❑ Chaque « Supporting Requirement » d'un UC = un WI

- Contraintes
 - ✓ Performance
 - ✓ Accès simultanés
- Formats – Contrôles + Messages d'erreur
- Gui Need (IHM complexe)

❑ Ajouter des WI pour les Tâches Transverses (nécessaires en marge des UC)

- Technical works (installation de soft, refactoring code)
- Acquisitions de connaissances (formation, spike, proto)

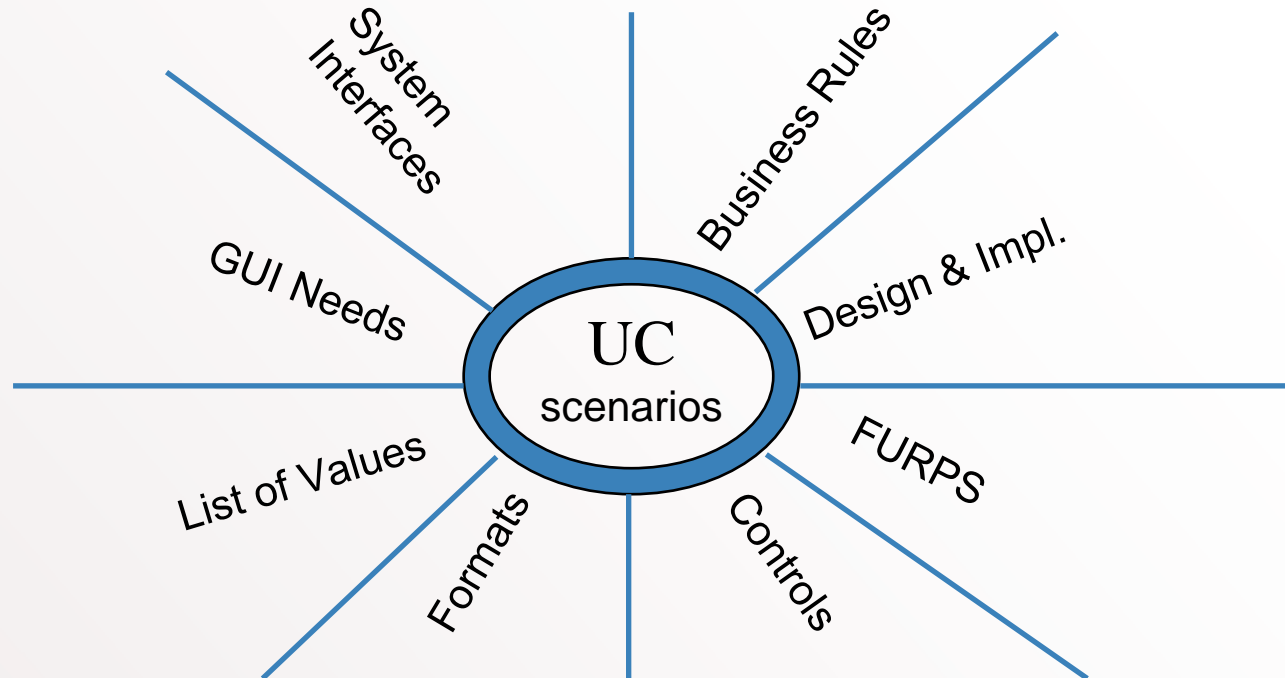
❑ Exemple: Split complet

UC Rechercher des Ouvrages

Type de WI	Description du WI
Sc. Nominal	recherche par mots-clés
Sc. Alt 1	recherche par « meilleures ventes »
Sc. Alt 2	recherche avancée (via moteur)
Sc. Alt 3	trop d'ouvrages trouvés (→ pagination)
Sc. Exc. 1	aucun ouvrage trouvé
Sc. Exc. 2	abandon de la recherche
Spike	recherche sur la codification ISBN
CT Perf	95 % des requêtes doivent aboutir en moins de 3 sec.
Gui Needs	ergonomie de la « fiche d'ouvrage », pagination
Format	zone de saisie du code ISBN
Contrôles	contrôle du format ISBN



❑ Splitter plus fin: Model « Hub of wheel »



- Chaque besoin additionnel associé à un UC peut faire l'objet de WI → split très fin



❑ Exemple: Business Rule = US

- Cas Web Bank: Le UC « Gérer les virements » est un CRUD qui a été splitté selon ses scénarios.
- Il inclut une Business Rule: « Le virement est autorisé seulement si le débit demandé est inférieur au plafond défini » → US additionnelle
- On pourra implémenter dans l'itération i1 le scénario de Création d'un virement sans la BR
 - le PO accepte l'US en ayant conscience que l'absence de plafond n'est pas une anomalie
- En i2 on implémente la BR en ajoutant quelques lignes de code (+ TNR)
 - TA du UC dans les conditions cibles



❑ Exemple: Interface Système = US

- ❑ Web Bank: Le UC « Visualiser titre du CAC40 » nécessite l'implémentation d'une **API** pour recevoir le flux des valeurs de titres issu du SI de la bourse
 - ➔ US additionnelle
- ❑ Pour tester rapidement cette API on pourra développer un **Mock** du flux de titres
 - Développer un petit moteur qui génère des valeurs de titres et simule leur transmission
 - ➔ US additionnelle
- ❑ Ce flux doit être traité en Temps Réel = **Contrainte de performance**
 - ➔ US additionnelle

Estimer et Prioriser les US

❑ Estimation

- Cf. section 3 de ce chapitre
- Fibonacci: 0, 1, 2, 3, 5, 8 Story Points

❑ Priorisation

- Cf. section 4 de ce chapitre
- Valeur client / Risque technique

➔ Placer les US dans le product backlog

Team Capacity

- ❑ Estimer le nombre de jours de présence de chaque TM sur la Release
 - Peut varier selon temps partiels, multi-projets...

➔ Exemple (Release de 60 jours ouvrés)

Membre	Jours de présence (sur la Release)
Mr X	40
Mme Y	60
Mr Z	50
Mlle T	30
TOTAL	180

Inclure les US dans la Release

- ❑ « Remplir » le Release Backlog avec les US à hauteur du Team Capacity



Nécessite de traduire le Team Capacity en nombre de SP réalisables par l'équipe sur la Release

➔ Calculer sa **Vélocité** (Ingé 3 seulement)



Calculer la Vitesse

- ❑ Techniques possibles:
 - **Historique:** se baser sur la vitesse de l'équipe observée sur des projets antérieurs.
 - **Dynamique:** effectuer quelques itérations, en mesurer la vitesse.
 - **Estimée:** Cf. pages suivantes.



Estimer la Vitesse (1/6): Choix des US

- ❑ Sélectionner quelques US représentatives de chaque taille.
- ❑ Exemple:

US	Story Points
US 1	1
US 2	3
US 3	3
US 4	5
US 5	5
US 6	8



Estimer la Vitesse (2/6): Décomposer en Tasks

- ❑ Identifier toutes les Tasks nécessaires à la réalisation de l'US

[Lien vers la notion de Task](#)

- ❑ Exemple:

US 6 : Tâches
Décrire les règles métier
Ecrire les acceptance tests
Concevoir l'IHM
Démontrer une maquette d'IHM au product owner
Coder l'IHM
...
Coder le middle tier
Mettre à jour la DB
Automatiser les tests unitaires

Estimer la Vitesse (3/6): Estimer les Tasks en heures idéales



❏ Exemple:

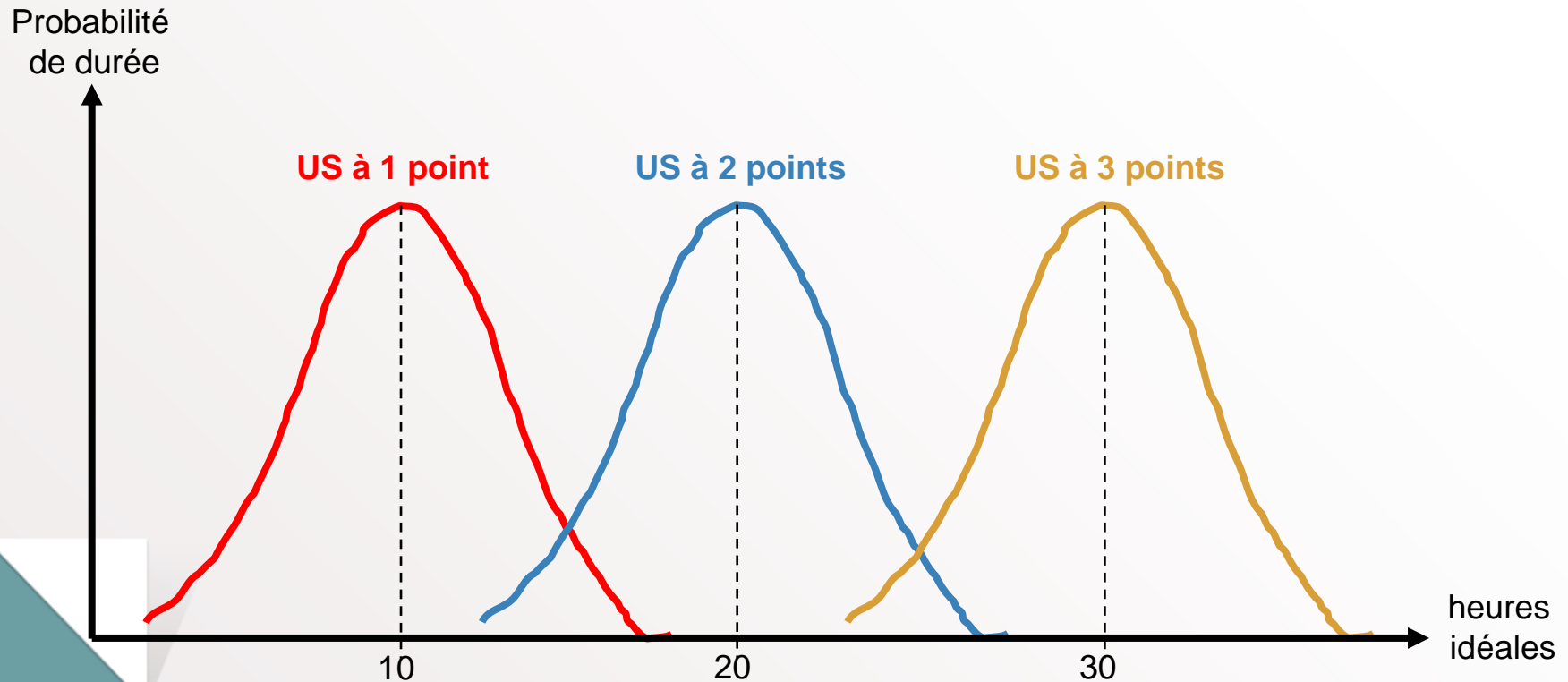
US 6 : Tâches	Heures idéales
Décrire les règles métier	12
Ecrire les acceptance tests	8
Concevoir l'IHM	16
Démontrer une maquette d'IHM au product owner	2
...	...
Coder l'IHM	8
Mettre à jour la DB	6
Automatiser les tests unitaires	6
TOTAL	79



Estimer la Vitesse (4/6):

Conversion story points \Leftrightarrow heures idéales

- ❑ Toutes les US estimées à X points ne dureront pas le même nombre d'heures idéales.
- ❑ On peut établir une distribution type autour d'une moyenne:





- ❑ Calculer la « durée moyenne du point » sur la base des quelques US sélectionnées
- ❑ Exemple:

US	Story Points	Heures idéales
US 1	1	13
US 2	3	30
US 3	3	22
US 4	5	57
US 5	5	48
US 6	8	79
TOTAL	25	249

➔ On fera l'hypothèse que 1 SP est équivalent à 10 heures idéales.

Hypothèse simplificatrice et imprécise (certaines US sont très éloignées de la moyenne) mais à l'échelle d'une release entière les écarts s'équilibrent.



Estimer la Vitesse (5/6): Heures disponibles par itération ou release

- ❑ Estimer le nombre d'heures idéales disponibles par jour pour TM.
 - ➔ ce nombre peut varier d'un membre à l'autre (CP très sollicité,...).
 - ➔ généralement proche de 6 heures idéales
- ❑ Déterminer le nombre de jours de présence de chaque membre par itération (temps partiels, multi-projets...).

Membre	Heures idéales par jour	Jours de présence (par itération de 2 semaines)	Heures idéales (par itération de 2 semaines)
Mr X	4	10	40
Mme Y	6	10	60
Mr Z	6	8	48
Mlle T	6	6	36
TOTAL heures idéales par itération (team capacity)			184



Estimer la Vitesse (6/6): Vitesse

- ❑ Utiliser le coefficient de conversion story points \Leftrightarrow heures idéales
- ❑ Exemple:

Total heures idéales par itération	184 heures
Valeur moyenne Story Point	10 heures
Vitesse (par itération)	18 points



Estimer le Team Capacity en SP

- ❑ Si on est en Schedule driven
 - calculer le nombre total d'itérations dans la release
 - en déduire le nombre total de points réalisables dans la release
- ❑ Exemple:

Durée de la Release	3 mois (12 semaines)
Nombre d'itérations dans la Release	6
Vélocité (par itération)	18 points
Total de points de la Release (Team Capacity)	6 x 18 = 108 points

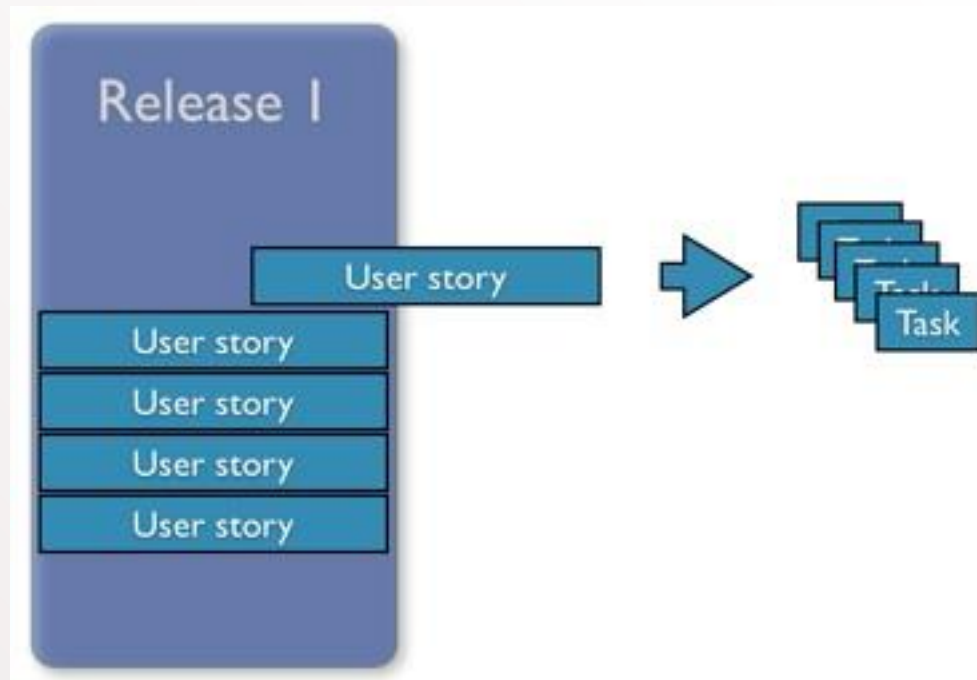
Conclure la session de planification

- ❑ Ajuster l'objectif (réduire scope ou reculer livraison) si nécessaire
 - Ajouter les US dans le Release backlog par ordre de priorité jusqu'à atteindre le Team Capacity



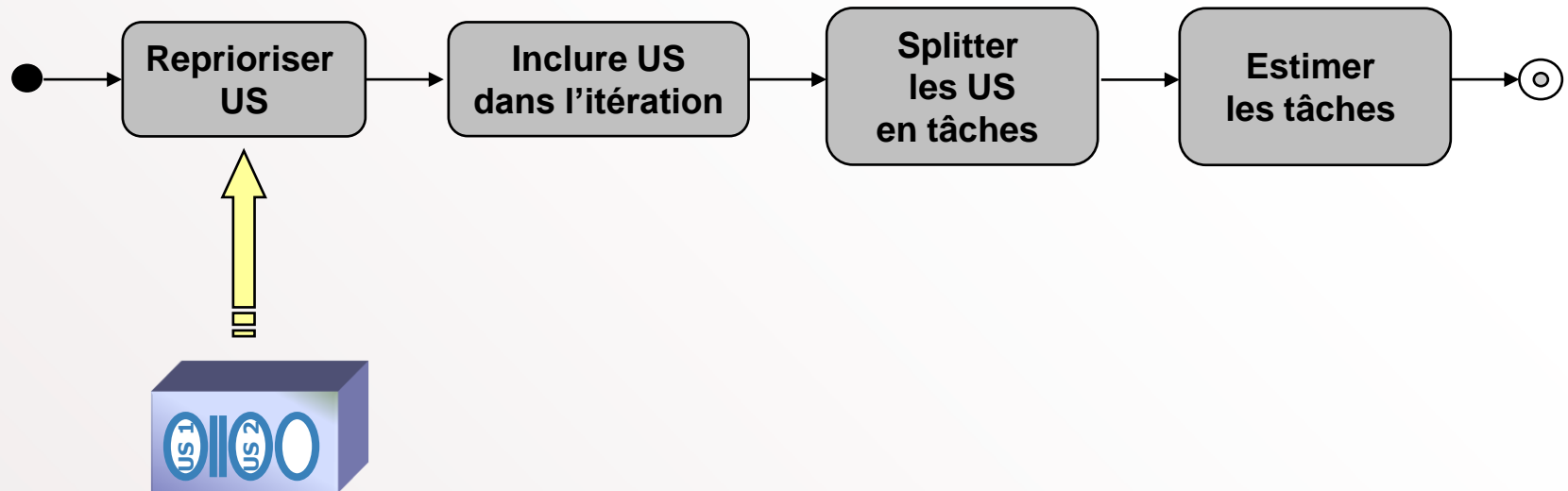
- ❑ A l'issue de la session de planification de Release:
 - Les US ne sont pas attribuées à un TM
 - Les US ne sont pas affectées à une itération

8 – Planification au Niveau Itération



Session de planification d'itération

- Session d'une **demi-journée**, en tout début d'itération, impliquant toute l'équipe.



RELEASE BACKLOG Mis à Jour (i-1)

- US i-1 non acceptées
- US nouvelles/supprimées
- bugs apparus

Reprioriser les US

☐ Changements dans le Release Backlog:

- Clotûre d'itération i-1
 - ✓ les US non acceptées ont rejoint le release backlog
 - ✓ des bugs sont apparus
- Planification de l'itération i
 - ✓ des US sont ajoutées ou supprimées



- ☐ Insérer ces nouveaux éléments dans la pile
- ☐ Reconsidérer les priorisations antérieures

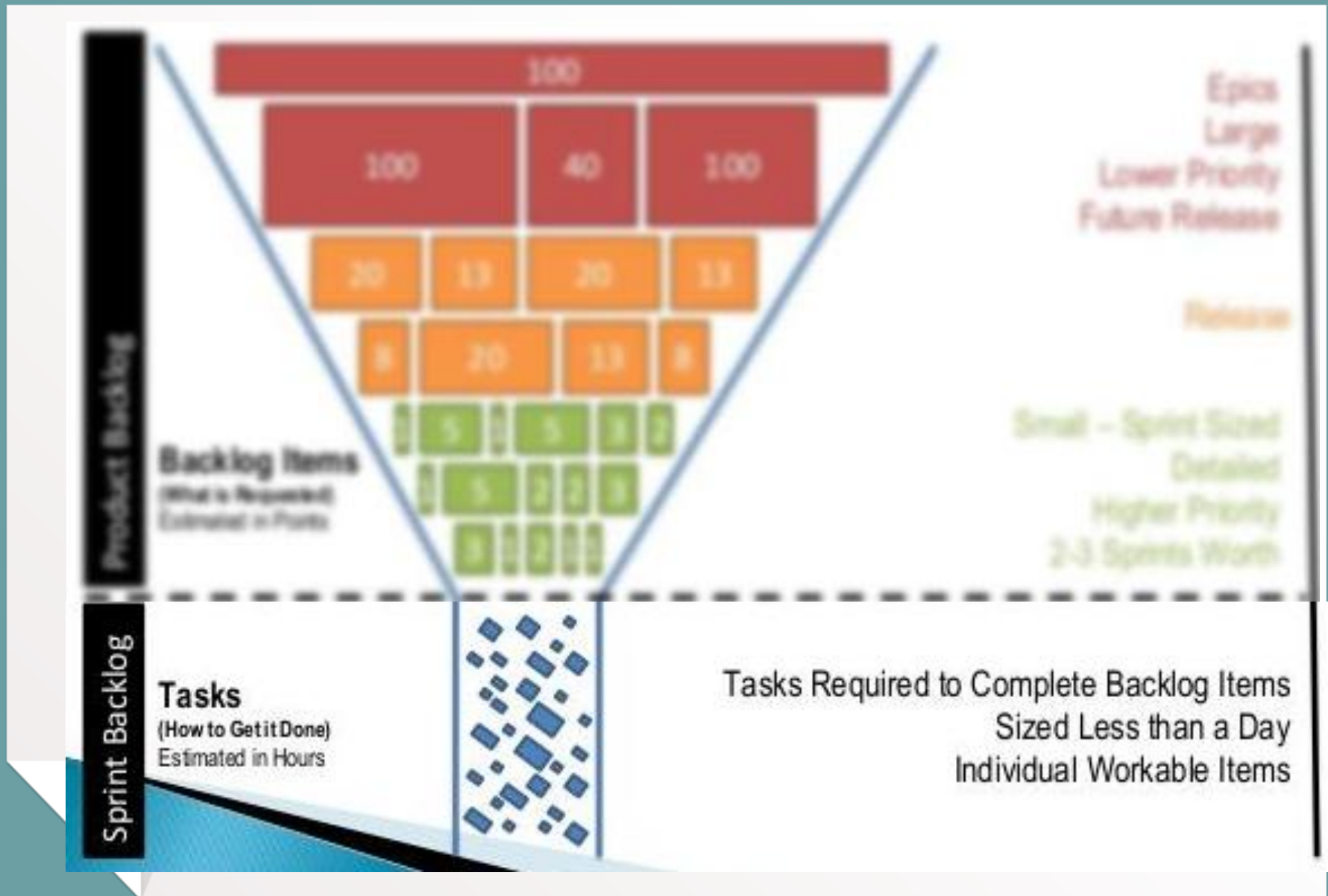
Inclure les US dans l'itération

- ❑ Remplir le Team Capacity [pour une itération]
 - Eg: Capacity = 18 SP → 6 US de 3 SP, ou 9 de 2 SP, etc...
 - Si le Capacity n'est pas connu en SP (version Ing 2):

Team Capacity pour la Release	180 jours
Nombre d'itérations dans la Release	6
Team Capacity pour l'itération	30 jours*
Poids de l'itération dans la Release	$(30/180) = 1/6$
Charge totale de la Release (que l'on espère réaliser)	120 SP
Charge possible pour l'itération	$330 \times 1/6 = 20 \text{ SP}$

* pas forcément 1/3 du total. A comptabiliser réellement pour chaque itération

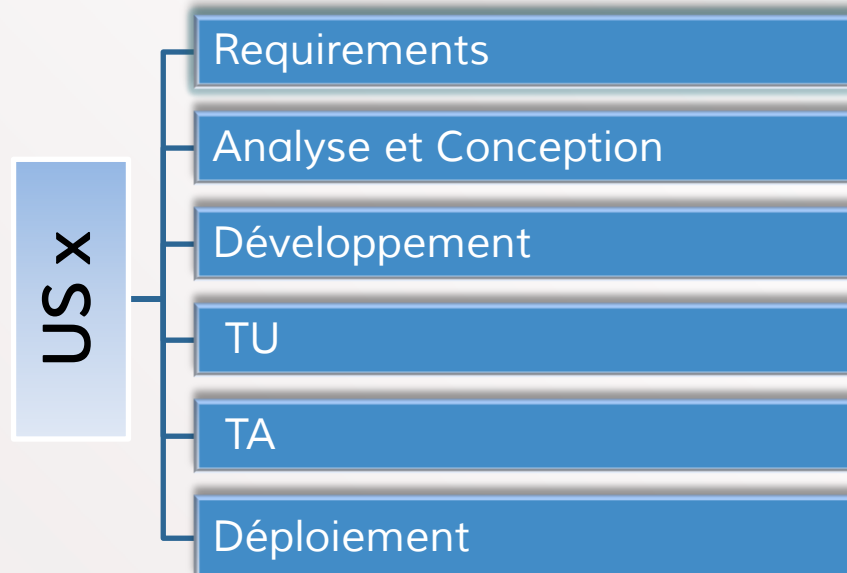
Split: User Stories -> Tasks



Tasks

- ❑ Travaux nécessaires pour réaliser un WI
- ❑ Réalisés par le TM en charge de l'US [sans intérêt pour le PO]
- ❑ Représentent quelques heures de travail (1 ou 2 jours max)

exemple simplifié





❑ Exemple plus complet (pour une US « de développement »)

- Décomposer les tasks de l'exemple simplifié
- En analyse et Conception chaque diagramme = 1 task
- En développement chaque composant ou classe = 1 task

Méta task	Tasks
Requiements	Description textuelle de l'UC Details et Glossaire
Analysis & Design	Diagramme de sequence système
	Diagramme de classes métier
	Diagramme de classes participantes
	Diagramme de sequence conception préliminaire
	Diagramme de classes de conception préliminaire
	Diagramme de sequence conception détaillée
	Diagramme de classes de conception détaillée
	Diagramme de composants
Implementation	Implémentation de l'IHM
	Implementation du Controller
	Implementation du Service
	Implementation DAO
	Implémentation requête base de données
	Revue de code
Tests	Rédaction Cahier de recette
	Implémentation Tests Unitaire
	Exécuter recette fonctionnelle
	Exécuter tests de recette avec Product Owner
Deploiement	Déploiement

❑ Estimation des tasks:

- l'estimation doit être partagée par toute l'équipe car on ne sait pas encore à qui la tâche sera affectée
- Unité = **heures idéales**: durée normale pour effectuer la tâche, sans interruption
 - ✓ réunions et questions collègues hors projet, temps personnel, lecture mail, maladie, ...

➔ **Heures idéales < temps écoulé**

Conclure la session de planification



- ❑ A l'issue de la session de planification d'itération:
 - Les US ne sont (toujours) pas attribuées à un TM
 - L'attribution se fera au fil de l'itération (FIFO)