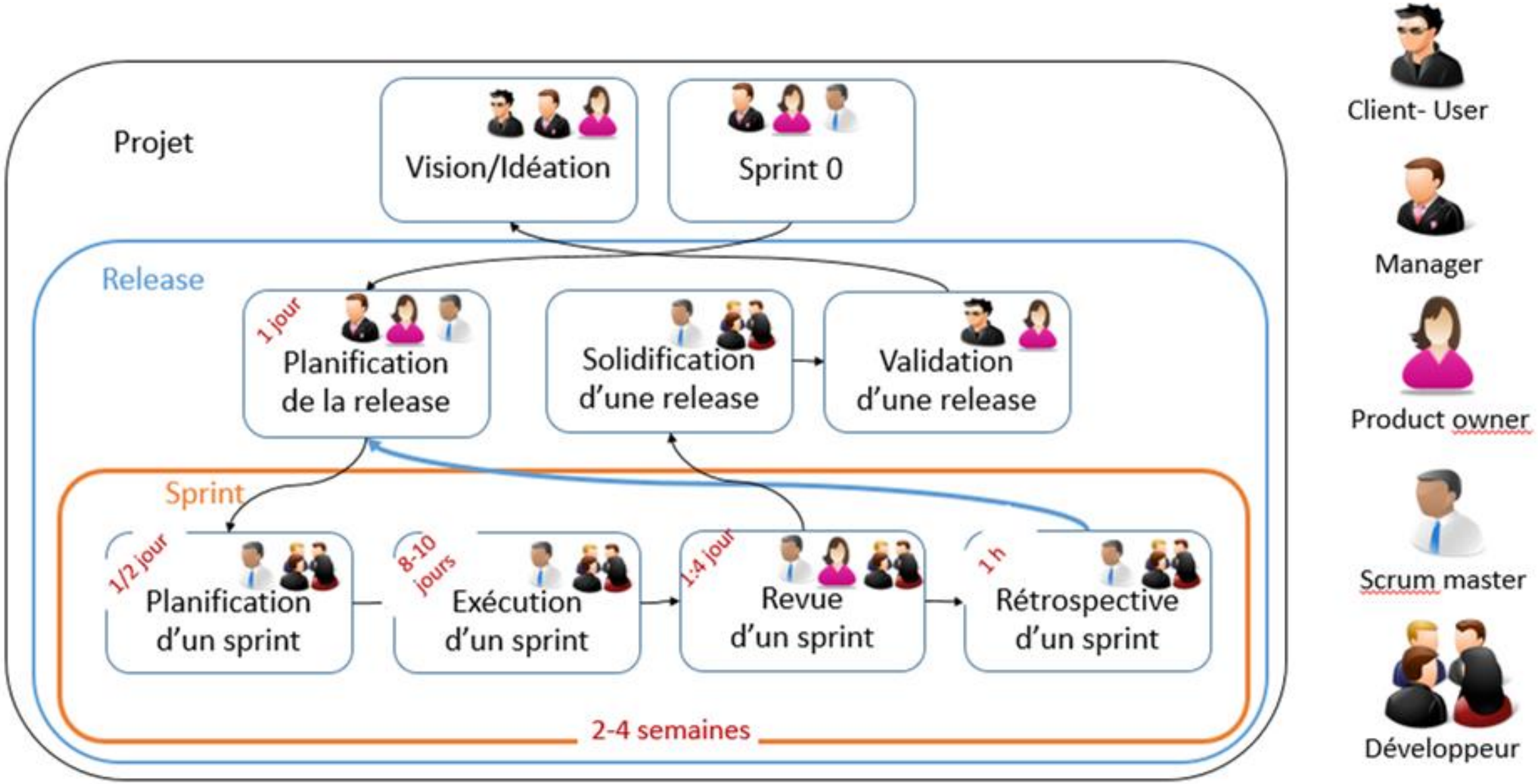
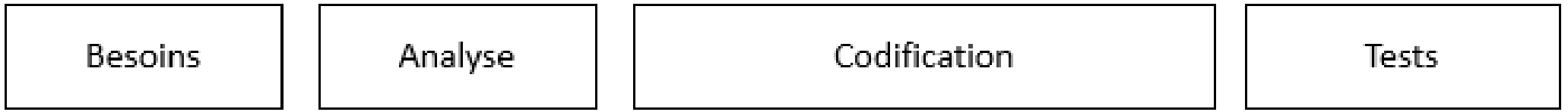
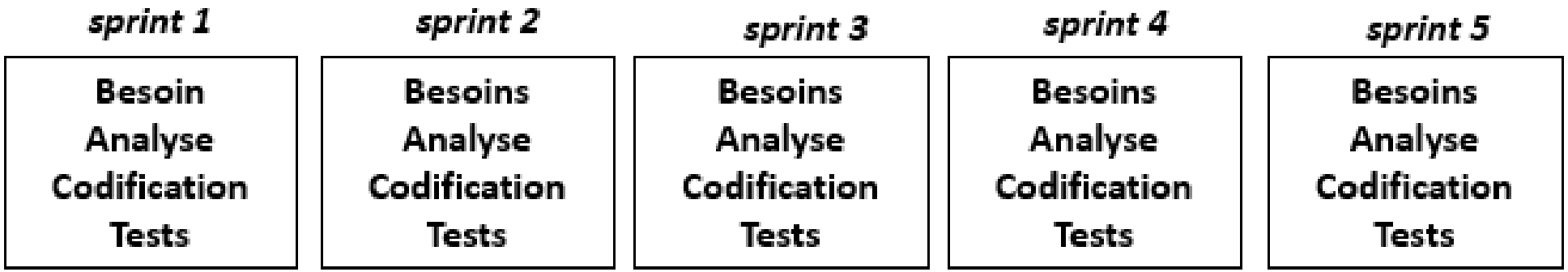
Processus logiciel Scrum



Processus logiciel SCRUM (*sprints* et *releases*)



Cycle séquentiel



Cycle SCRUM

Processus logiciel SCRUM

- Une phase d'idéation
- Une succession de sprints de taille fixe
- Chaque *sprint* produit un logiciel fonctionnel
- Chaque *sprint* réalise toutes les activités de développement logiciel
- Les *sprint*s sont groupés en *Release*s (de taille fixe)

Release SCRUM vs. *Release* traditionnelle

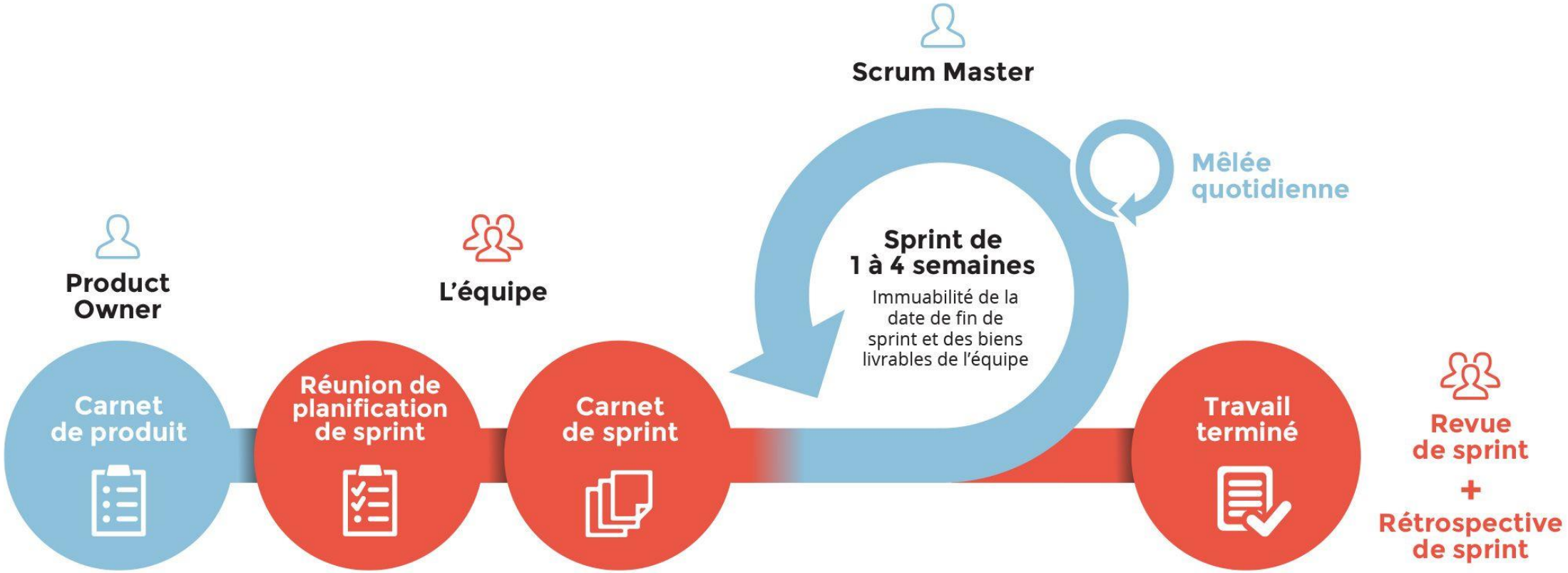
- Traditionnellement, une *release* (livrable) correspond à une nouvelle version d'un logiciel
  - Incrément fonctionnel (une *release* correspond à la réalisation d'un objectif fonctionnel)
  - Evolution technique
- SCRUM
  - Une *release* :
    - correspond à la livraison d'une *feature* (fonctionnalité)
    - travail réalisé pendant une période de temps fixe qui comprend plusieurs *sprints*
    - pour faire coïncider ces deux objectifs, on décompose/regroupe les *features*
  - Mais une nouvelle version du produit (livrable) peut-être produite à la fin de n'importe quel sprint

Périodes d'une *Release*

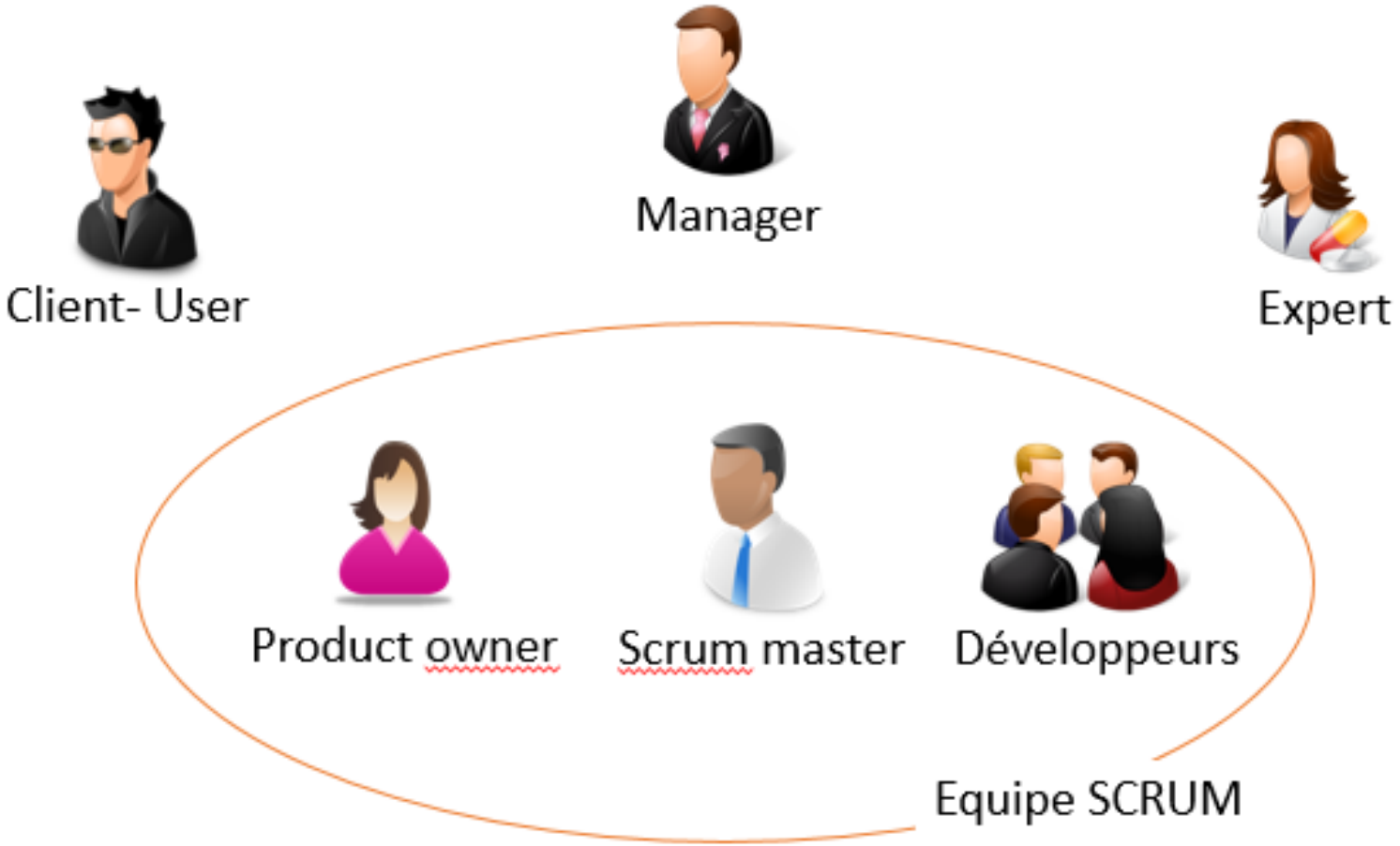
- La période avant le premier *sprint* de développement, appelée « sprint zéro » : planification de la *release* en plusieurs *sprints*
- La période des *sprints* (de développement)
- La période après le dernier *sprint* et avant la fin de la release

Sprint

- Chaque release est décomposée en sprints au cours de l'activité de planification de la release (la première fois « sprint zéro »)
- Un sprint est une période de développement de taille fixe (en moyenne 2 à 4 semaines)
  - Durée fixe, équipe stable
- Un sprint se décompose en un ensemble de tâches
- Mêlée journalière : réunion d'équipe pour faire le point sur le travail réalisé depuis le début du sprint et le travail à réaliser avant la fin du sprint
- Chaque *sprint* termine par :
  - une revue du produit
  - une rétrospective sur le processus


Scrum Master
Mêlée quotidienne
Product Owner
L'équipe
Sprint de 1 à 4 semaines
Immuabilité de la date de fin de sprint et des biens livrables de l'équipe
Carnet de produit
Réunion de planification de sprint
Carnet de sprint
Travail terminé
Revue de sprint
+
Rétrospective de sprint

Les acteurs


Client- User
Manager
Expert
Product owner
Scrum master
Développeurs
Equipe SCRUM

L’équipe Scrum

- Composition:
  - 1 Product Owner
  - 1 Scrum Master
  - 2 à 7 développeurs
    - Le product owner et le Scrum master peuvent aussi prendre le rôle de développeur
- Principes
  - Auto-organisation
  - Pluridisciplinarité
  - Stabilité
  - Valeurs communes

Product owner

Responsabilités

- Fait partager la vision globale du produit
- Gère le backlog du produit (liste ordonnée des « choses » à faire)
- Définit les priorités
- Accepte ou rejette les *Releases* (livrables)

Scrum master

Responsabilités

- n’est pas « le chef », mais un facilitateur
- Motive l’équipe
- Fait appliquer les bonnes pratiques de Scrum
- Gère les obstacles

Les objets de SCRUM

FEATURE, STORY, TÂCHE, BACKLOG

Feature

Une *feature (fonctionalité)* est un service ou une fonctionnalité du produit à développer

Elle se décompose en *stories* (histoires) de tailles différentes. On distingue :

- les *stories* complexes (épiques) qui seront affinées en *stories* plus simples
- Les stories atomiques qui ne se décomposent pas et sont réalisées dans les *sprints*

Feature (exemple)

un site pour propriétaires d'animaux domestiques
Des *features* (fonctionalités, chapites …)

https://pablopernot.fr/2017/01/cartographie-plan-action/

Workflow d'une *feature*

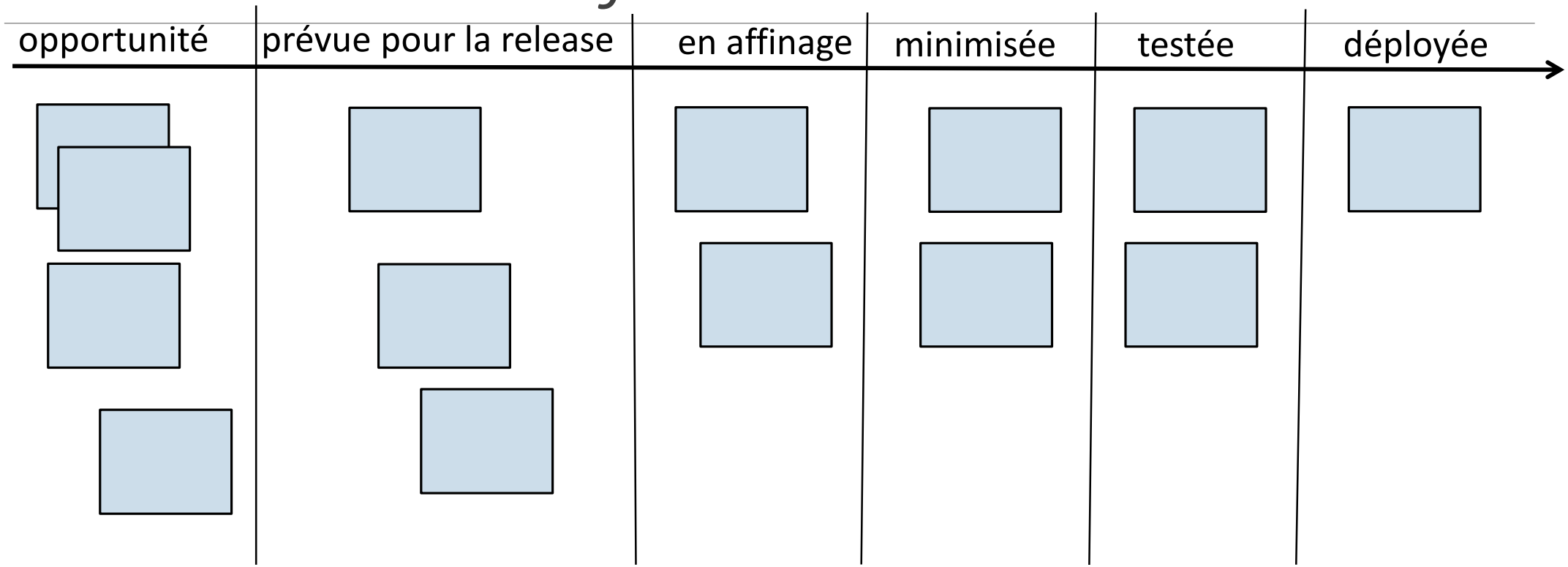
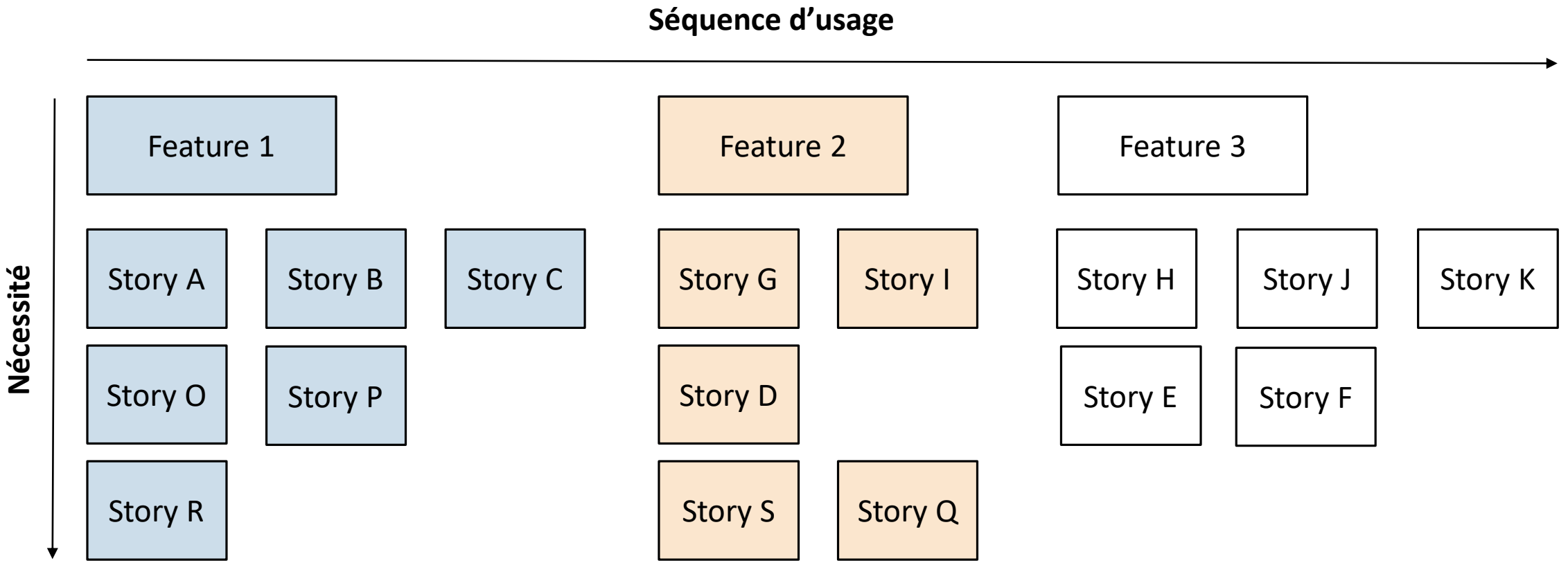


Tableau des *features*

Workflow d'une *feature*

- Opportunité : idée de feature qui présente une opportunité pour le produit
- Prévue pour la release : l'étude d'opportunité a abouti, et la feature est prévue pour la release en cours
- En affinage : initialement sous forme d'épique, est décomposée en stories
- Minimisée : *Minimal Maketable Feature*
- Testée : … avec des stories finies
- Déployée : utilisable par des utilisateurs qui peuvent fournir du feedback

Story map : décomposition d'une *feature* en *stories*



Exemple

http://winnipegagilist.blogspot.com/2012/03/how-to-create-user-story-map.html

Story

- Une *story* est une exigence du système à développer, formulée en une ou deux phrases dans le langage de l'utilisateur.
- Les *Stories* émergent au cours d'ateliers de travail menés avec le Métier, le Client et/ou les Utilisateurs.
- On distingue :
  - *Story* fonctionnelle
  - *Story* technique
  - Correction de bug
  - Remboursement de la « dette technique »

Story

- Les 3C
  - Carte : l'histoire est courte et sa description tient sur une carte (demi-page)
  - Conversation : l'histoire est définie avec les gens du métier
  - Confirmation : l'histoire est confirmée par des tests d'acceptation rédigés au même moment que celle-ci
- Workflow de la *story*
  - Idée d'une *story* rédigée sur une **C**arte (1/2 feuille)
  - **C**onversation dirigée par le *product owner* qui inclut les gens du métier
  - L'équipe apporte sa **C**onfirmation que la *story* est prête
  - L'équipe réalise la story
  - Le *product owner* apporte sa **C**onfirmation que la story est finie

Estimation des points d’une story

- Planning Poker
  - Les participants s'installent autour d'une table, placés de façon que tout le monde puisse se voir.
  - Le responsable de produit explique à l'équipe un scénario utilisateur (*user story*).
  - Les participants posent des questions au responsable de produit, discutent du périmètre du scénario, évoquent les conditions de satisfaction qui permettront de le considérer comme "terminé".
  - Chacun des participants évalue la complexité de ce scénario, choisit la carte qui correspond à son estimation et la dépose, face vers le bas, sur la table devant lui.
  - Au signal du facilitateur, les cartes sont retournées en même temps.
  - S'il n'y a pas unanimité, la discussion reprend.
  - On répète le processus d'estimation jusqu'à l'obtention de l'unanimité.
  - Une procédure optimisée consiste, après la première "donne", de demander aux deux acteurs ayant produit les évaluations extrêmes d'expliquer leurs points de vue respectifs. Ces explications achevées et comprises de tous, une nouvelle estimation est produite et c'est alors la moyenne arithmétique de ces estimations qui est prise en compte.

Indicateurs

Sprint

- *Burndown* de *sprint* (orienté reste à faire)
- *Burnup* de *sprint* (orienté ce qui a été déjà fait)

Release

- *Burndown* de *release*
- *Burnup* de *release*

Equipe

- Vélocité (capacité de l'équipe)
- Suivi des obstacles
  - Perturbations exogènes ou endogènes qui perturbent le bon déroulement du sprint
  - Peut de générer de nouvelles tâches (dette) et/ou leur réorganisation

Vélocité

- Estime la capacité de l'équipe en nombre de points de stories par sprint
- Utilisé pour la planification de la release
- Affinée à la fin de chaque sprint
- Tendance à la stabilité

Les activités propres à SCRUM

Idéation

Définition d'une vision commune

- Identification des *features*
  - Impact mapping
- Identification des parties prenantes
  - Acteurs

Création d'un *backlog* de haut niveau du produit

- Tableau ordonné des features
- (éventuellment Story map haut niveau)

Sprint « Zéro »

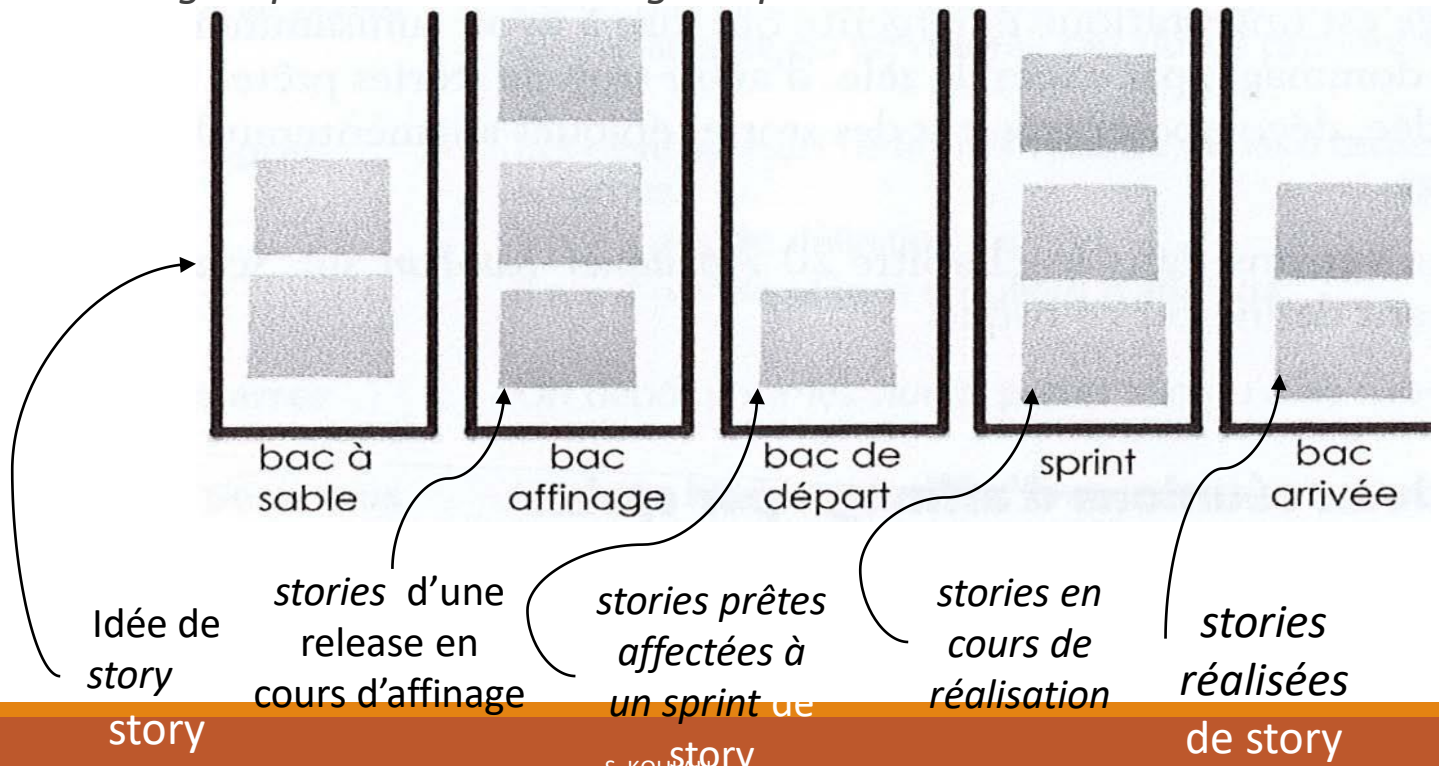
Affinage du backlog

- *Story mapping* (*feature* → stories)(partiel)
- Description des stories (description, conditions d'acceptation)
- Ordonnancement des stories
- Planification de la première *release*
  - Approvisionnement du bac d'affinage
  - Approvisionnement du bac de départ du sprint 1

Peut durer plusieurs journées

Le *backlog* (carnet de route)

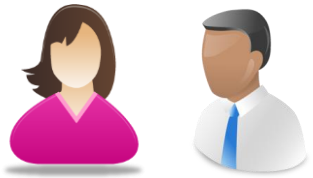
- Liste ordonnée des choses (stories) à faire
- En pratique, plusieurs (sous-)*backlogs*
  - *On peut distinguer le backlog de produit et le backlog de sprint*



Story mapping

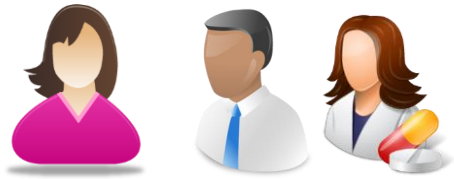
- Ordonnancement des features
- Décomposition en stories
- (Organisation des releases)

Planification des *releases*



- Affiner les risques, les incertitudes en fonction des retours de la revue et de la rétrospective du dernier sprint
- Ajuster la vélocité de l'équipe (capacité de travail de l'équipe en nombre de points de story)
- Affiner, (re-)planifier le(s) prochain(s) sprint(s)
  - ✓Définition de prêt et fini
  - ✓Nombre de points
- Affiner, (re-)planifier la future release

Affinage du backlog



1. Approvisionner le bac de départ en stories prêtes
2. Identifier les *epics* à décomposer en stories simples
3. Identifier les stories du bac à sable qu peuvent entrer dans le bac d'affinage
4. Evaluer les éléments du bac d'affinage
5. Réordonner le bas d'affinage
6. Purger les bacs