Introduction aux méthodes de développement

ESIR 2 SNR - GLA

Génie Logiciel

Le génie logiciel est une science qui s'intéresse en particulier aux procédures systématiques qui permettent d'arriver à ce que des logiciels de grande taille correspondent aux attentes du client, soient fiables, aient un coût d'entretien réduit et de bonnes performances tout en respectant les délais et les coûts de construction

Les Processus de développement logiciel

- Ensemble d'activités successives organisées en vue de la production d'un logiciel
- Mais quelles sont les différentes activités ?

Activités du développement logiciel

Quelles sont les différentes activités de développement logiciel ?

Activités du développement logiciel

- Etude de faisabilité
- Analyse des besoins
- Spécification
- Conception
- Programmation
- Vérification & Validation
- Livraison
- Maintenance

Faisabilité (pourquoi développer le logiciel)

- Etude préalable ou étude de faisabilité : Elle concerne la définition globale du problème.

- Pourquoi développer le logiciel ?
- Comment faire ce développement ?
- Quels moyens faut-il mettre en oeuvre ? Y aura t-il un budget suffisant, assez de personnel ?
- Y a t-il le matériel nécessaire.

Analyse des besoins (Que faire ?)

- Comprendre les besoins du client :
  - Objectifs généraux, environnement du futur système, ressources disponibles, contraintes de performance
  - Les besoin sont fournies par le client (qui est considéré comme expert du domaine, ou le futur client)
- Spécification :
  - Mettre en place une description claire de ce que doit faire le logiciel (fonctionnalités détaillées, exigences de qualité, interface...)
  - Clarifier le cahier de charges (ambiguïtés, contradictions) en listant les exigences fonctionnelles (fonctionnalités) et non fonctionnelles (des attributs de qualité comme la maintenabilité, la rapidité, …)

Exigences fonctionnelles et non fonctionnelles

- **Exigences fonctionnelles :**
  - Décrivent les fonctionnalités du système, ce que le système doit faire

- **Exigences non fonctionnelles :**
  - Décrivent les propriétés et contraintes du système

- Listez quelques EF et non F sur la plate-forme Netflix ?

Exigences fonctionnelles et non fonctionnelles

- **Exigences fonctionnelles :**
  - Décrivent les fonctionnalités du système, ce que le système doit faire
  - Exemple (Netflix) : regarder un film, création de comptes, souscription, paiement

- **Exigences non fonctionnelles :**
  - Décrivent les propriétés et contraintes du système
  - Exemple : fiabilité, temps de réponse, taille de stockage, sécurité, disponibilité, maintenabilité

Conception (Comment faire ?)

- **Conception :**
  - Imaginer, modéliser, et élaborer une solution concrète qui permet de réaliser la spécification

- **Conception Globale :**
  - décrire l'architecture en composants (avec interface et fonctionnalités)

- **Conception Détaillée :**
  - Détailler et représenter chaque composant
  - Prévoir des tests unitaires pour s'assurer que les composants réalisés sont conformes à la description

Programmation

- **Programmation :**
  - Implémenter la solution conçue précédemment
  - Choisir l'environnement de développement, du/des langages de programmation, des normes,

Vérification & Validation

- **Vérification**
  - S'assurer que le logiciel développé est conforme à la spécification
- **Validation :**
  - S'assurer que les besoins du client sont satisfaits

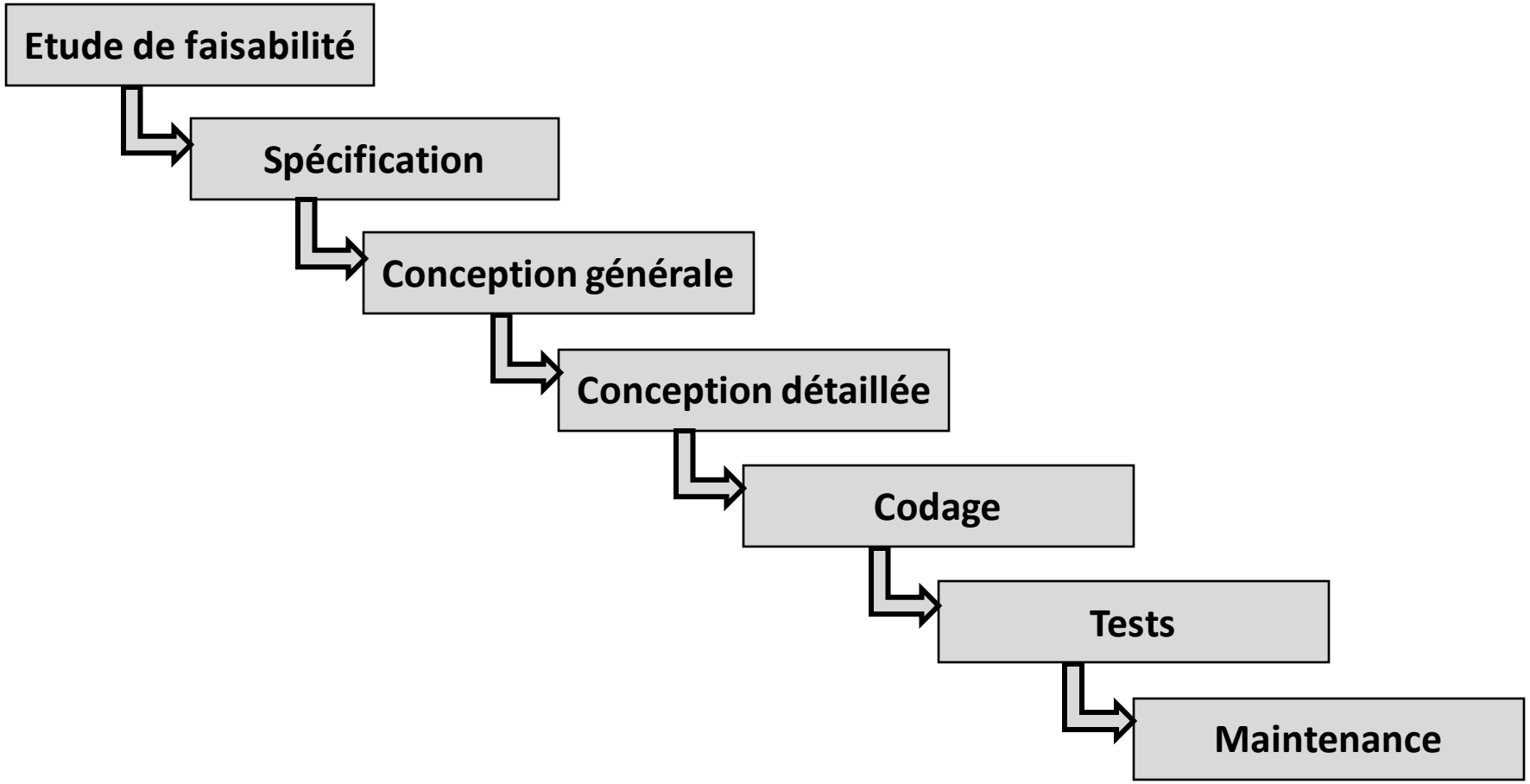
Maintenance

- **Corrective :**
  - Identifier des erreurs dans le logiciel et les corriger.
- **Adaptive :**
  - Adapter le logiciel au changements dans l'environnement.
- **Perfective :**
  - Améliorer la performance, ajouter des fonctionnalités, améliorer la maintenabilité du logiciel.

Activités du développement logiciel

- Etude de faisabilité
- Analyse des besoins
- Spécification
- Conception
- Programmation
- Vérification & Validation
- Livraison
- Maintenance

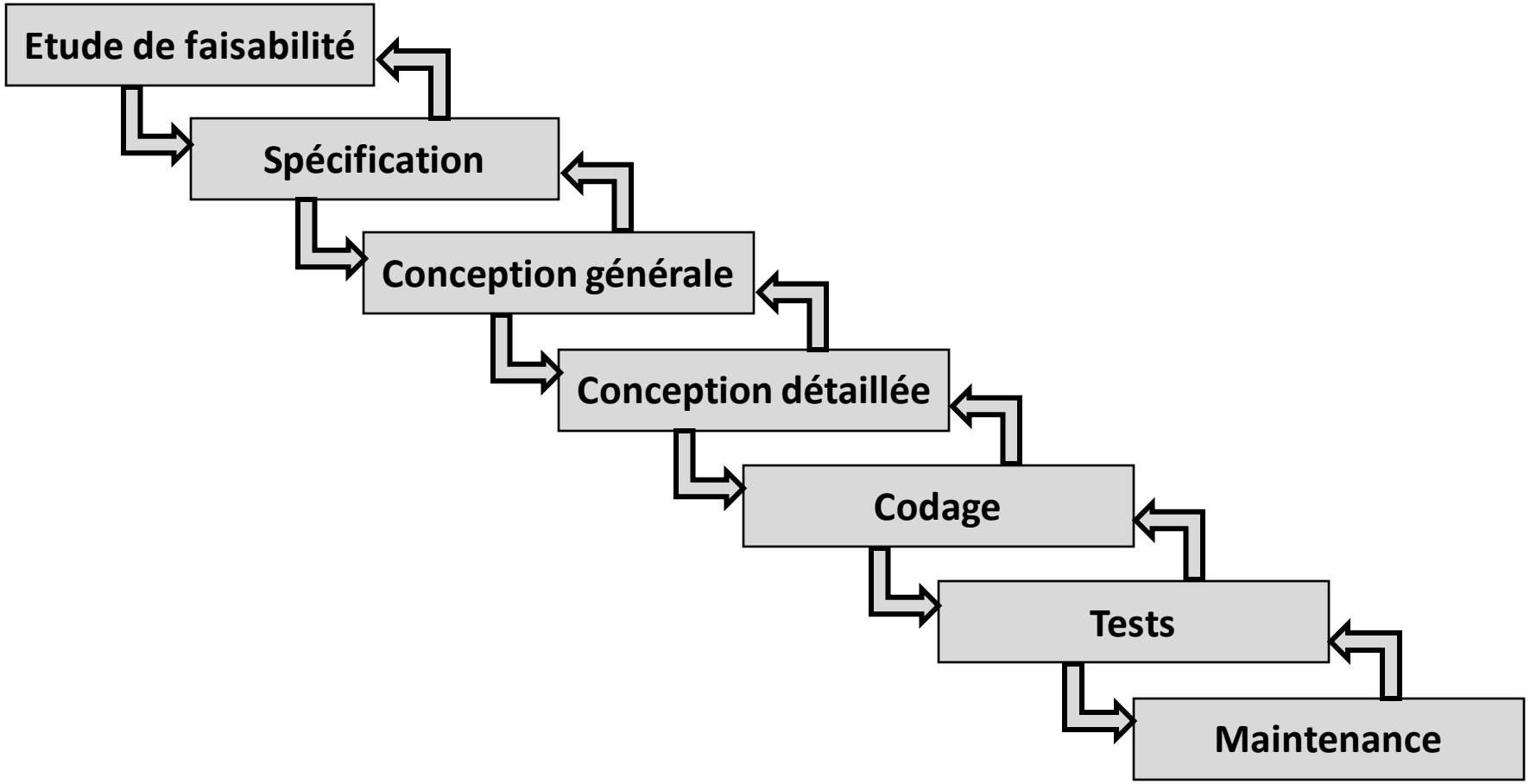
Cycles de développement : Cycle en cascade (waterfall)



Cycles de développement : Cycle en cascade (waterfall)

- C'est le modèle le plus classique des cycles de vie
- Cycle de vie linéaire sans aucune évaluation entre le début du projet et la validation
- Le projet et découpé en phases successives dans le temps
- A chaque phase correspond une activité principale bien précise produisant un certain nombre de livrables
- Chaque phrase ne peut remettre en cause la phrase précédénte

Cycles de développement : Cycle en cascade avec retour en arrière



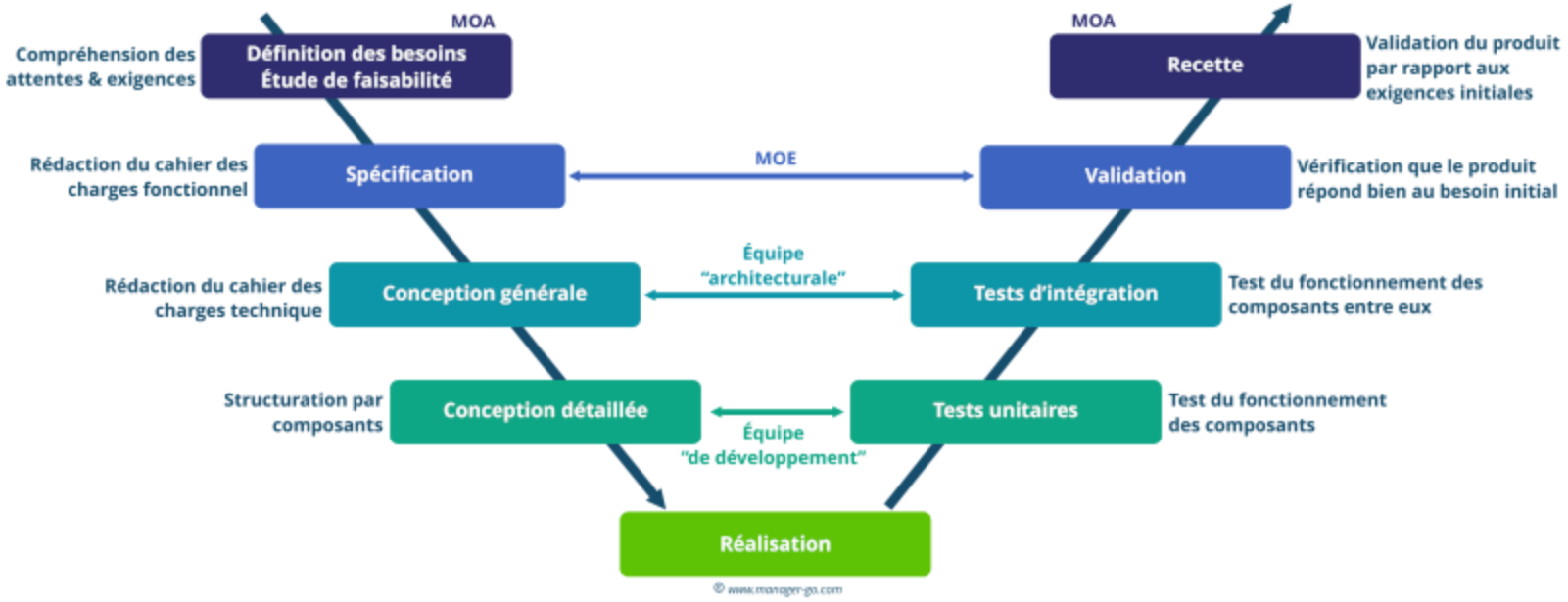
Cycle de vie en cascade

- Ce cycle est séquenciel, il est divisé en phases, dans chaque phase est produit un document

- **Limites :**
  - Les vrais projets suivent rarement un développement séquentiel
  - Etablir tous les besoins au début d'un projet est difficile
  - Sensibilité à l'arrivée de nouvelles exigences : refaire toutes les étapes
  - Adapté quand les besoins sont **clairement** identifiées et stables

Cycles de développement : cycle en V

- 80% des développements logiciels sont organisés selon un cycle V



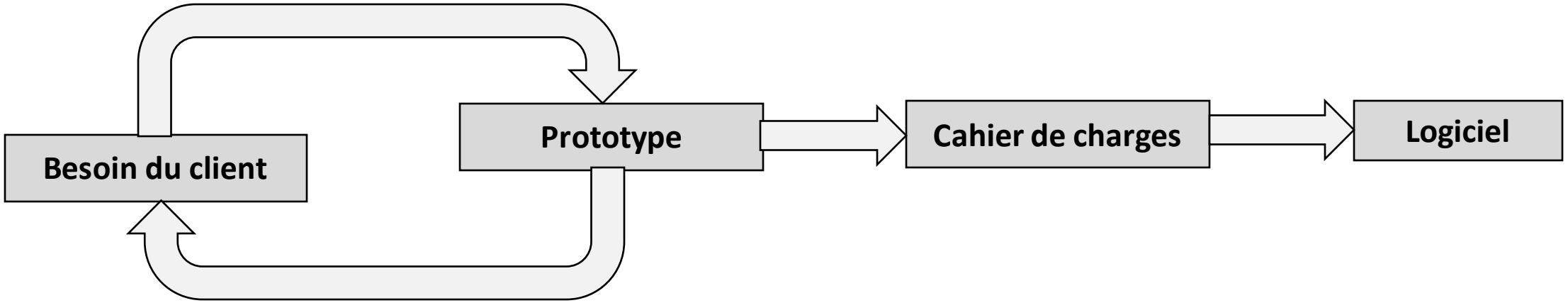
Cycles de développement : cycle en V

- **Avantages :**
  - Découpage du développement rationnel, logique
  - Identification claire des documents à produire, des rôles des équipes
  - Convient aux grosses équipes

- Inconvénients :
  - Manque de souplesse : le client ne peut pas changer ses objectifs en cours de route
  - Découverte tardive des erreurs de spécification et conception par le client
  - Intégration finale risquée, elle s'effectue à la fin du cycle

Prototypage

- Développement rapide d'un prototype avec le client pour valider ses besoins
- Ecriture de la spécification à partir d'un prototype, puis processus de développement linéaire.
- Avantage : Validation concrète des besoins, moins de risque d'erreur de spécification



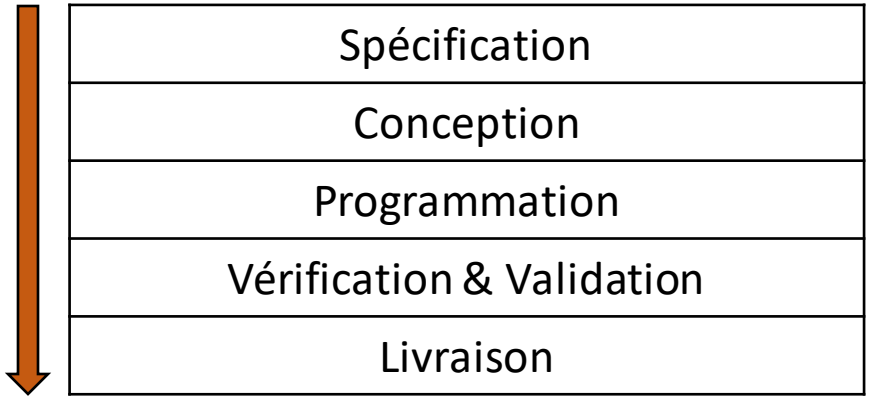
Prototypage

- **Prototype Jetable :**
  - Le prototype n'est créé que dans le but d'exprimer puis de valider les besoins de l'utilisation
- **Prototype Evolutif :**
  - Le prototype est conservé tout au long du cycle de développement. Il est complété pour obtenir le logiciel final

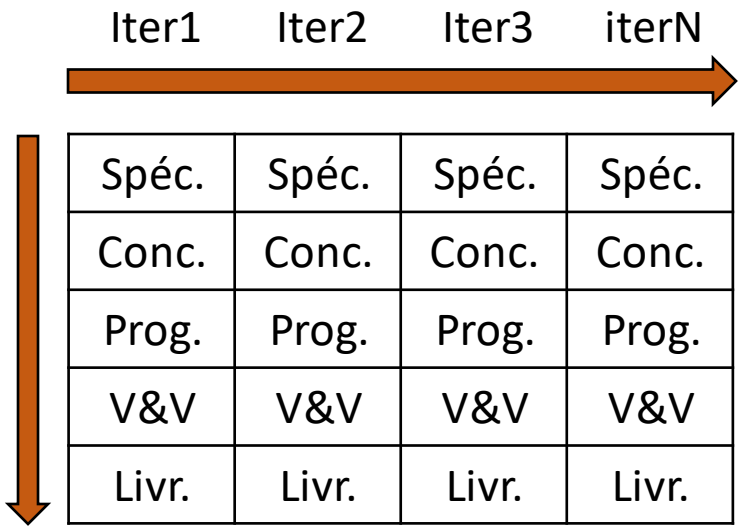
- **Avantages :**
  - les efforts consacrés au développement d'un prototype sont le plus souvent compensés par ceux gagnés à ne pas développer de fonctions inutiles

Cycle itératif/Incrémental

- Sous-ensemble minimal et fonctionnel du système
- Ajouts d'incréments jusqu'à la fin du processus



Développement en cascade



Développement incrémental

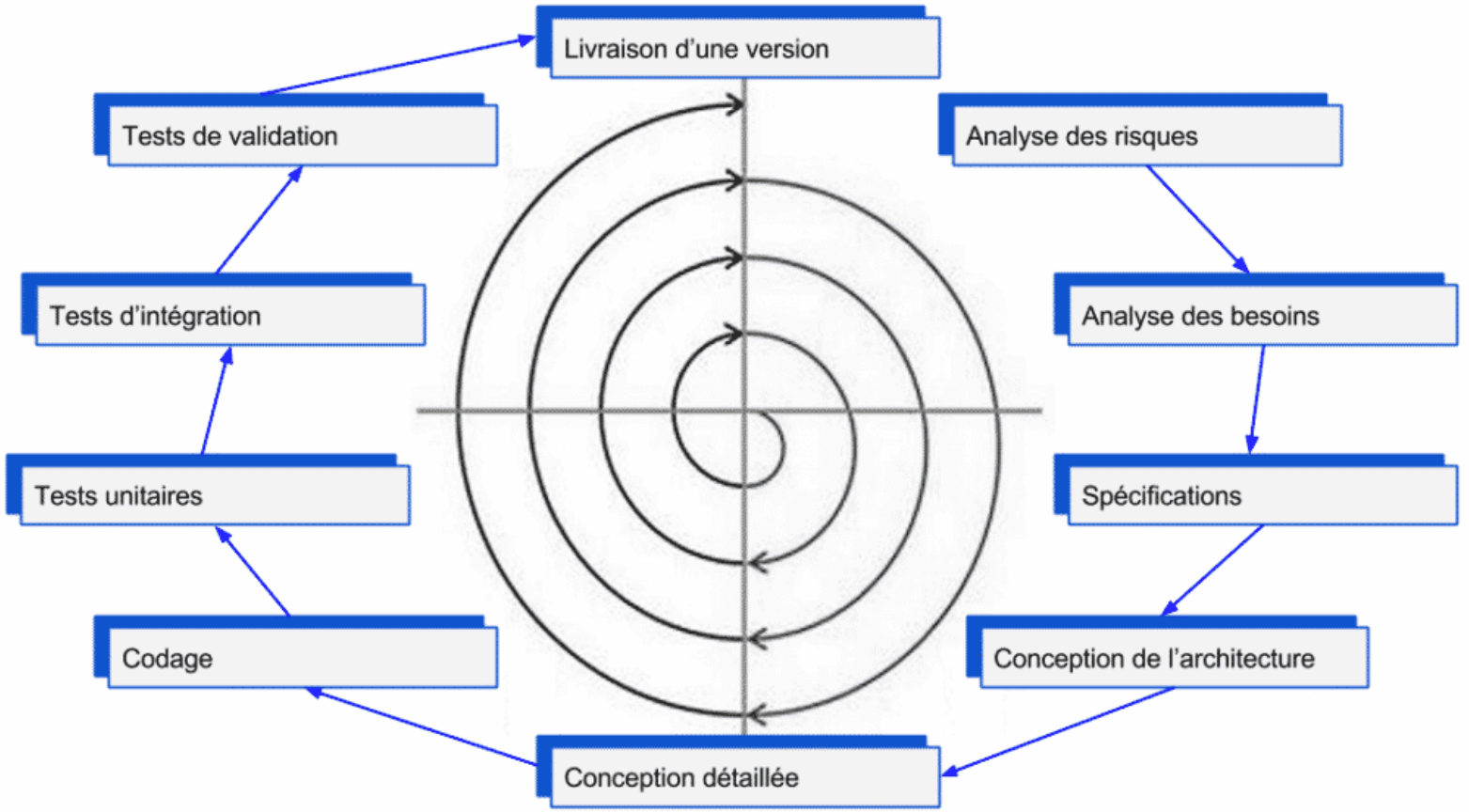
Cycle incrémental/itératif

- **Avantages :**
  - Le développement est moins complexe (diviser pour mieux régner)
  - Les intégrations sont progressives
  - Possibilité de livraisons et de mises en service après chaque incrément
  - Intégration continue : dispose en permanence d'un logiciel fonctionnel
  - Validation par le client rapide à la fin de chaque itération

- **Inconvénients :**
  - La remise en cause du noyeau de départ
  - La remise en cause des incréments précédents
  - L'impossibilité d'intégrer un nouvel incrément (incompatibilité entre incréments, à cause de l'absence de tests d'integration entre les deux incréments)
  - Mise en place complexe pour les grosses équipes (plus de 12 personnes) et pour des équipes sur plusieurs localisations.

Cycle en spirale

- Chaque version implémente toutes les fonctionnalités possibles contrairement au prototypage où le prototype n'est pas fonctionnel



Pour résumer

- **En pratique :**
- Par de processus idéal
- Choix du processus en fonction des contraintes (taille des équipes, temps, qualité, …)
- Adaptation du processus types aux besoins réels

Documentation

- **Objectif :** garder une traçabilité du projet et de la phase passée
- **Pour l'équipe :**
  - Regrouper et structurer les décisions prises
  - Faire référence pour les décisions futures
  - Garantir la cohérence entre les modèles et les produits
  - Pas de perte d'information lors d'un turn-over
  - Intégration rapide pour les nouveaux arrivants
- **Pour le client :**
  - Donner une vision claire de l'état d'avancement du projet

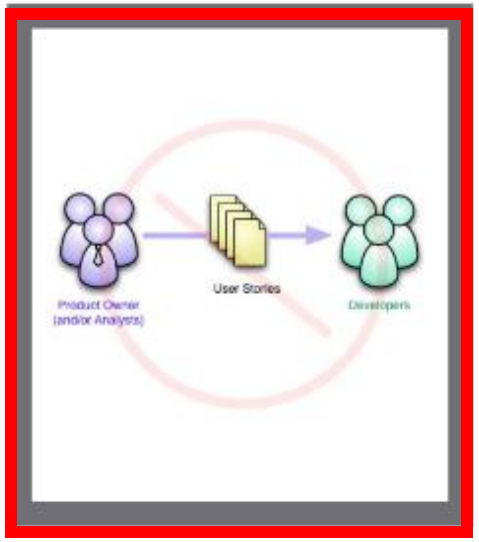
Agile

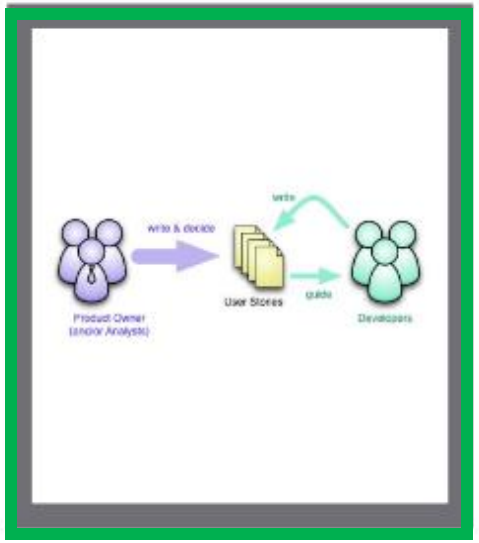
Quelle est votre expérience avec les méthodes agiles en entreprise ?

Agile

- Décomposition du projet en plusieurs étapes
- Améliorer continuellement le produit
- Mettre le client au coeur du processus, avec une collaboration constante entre les parties prenantes
- Organisation faite selon les besoins de l'équipe
- Méthode axée sur l'adaptation, la flexibilité et sur l'humain et la communication

Agile

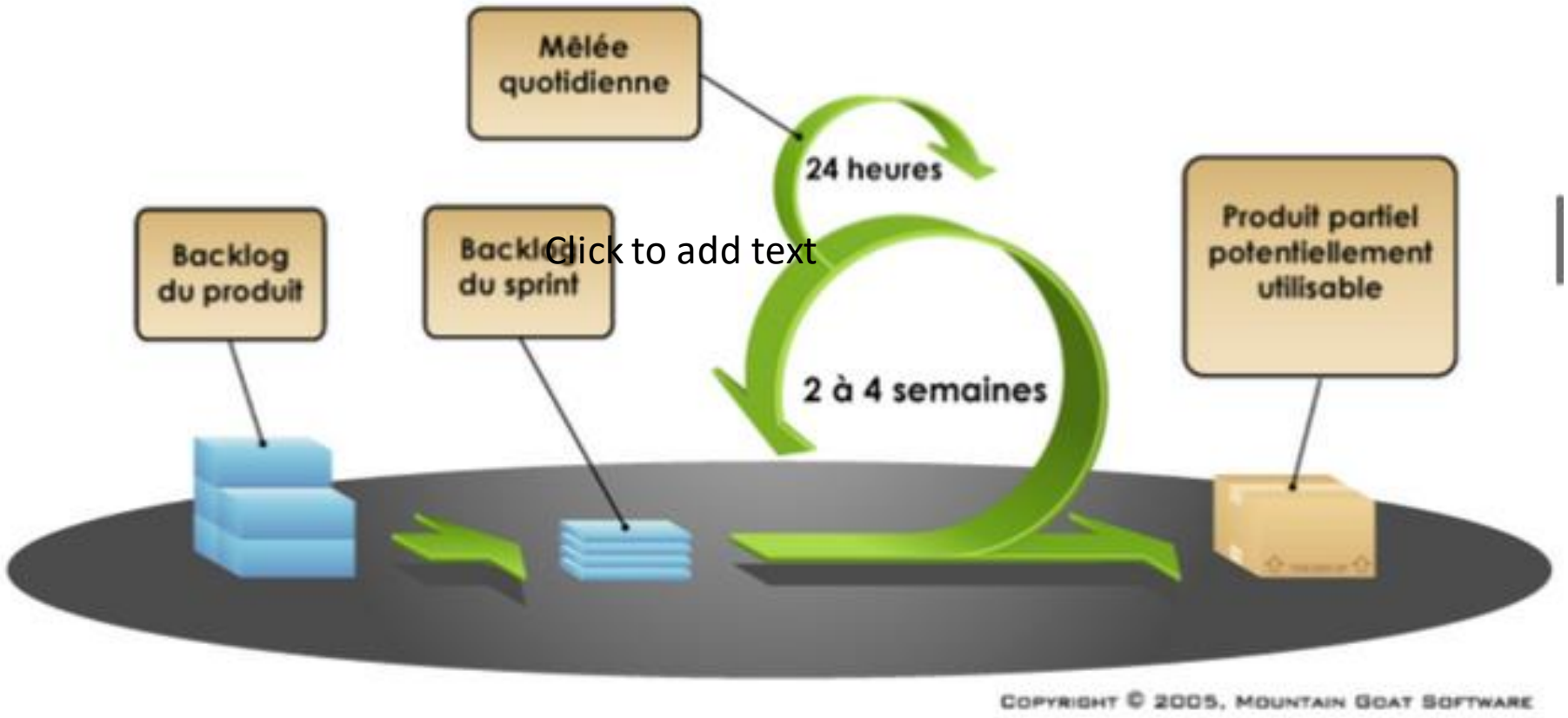

Product Owner
(and/or Analyst)
User Stories
Developers


write & decide
write
guide
Product Owner
(and/or Analyst)
User Stories
Developers

Agile – Principes scrum

- Scrum est un processus Agile qui vise à produire la plus grande valeur métier dans la durée la plus courte.
- Du logiciel qui fonctionne est produit à chaque itération appelée sprint (toutes les 2 à 4 semaines).
- Le client définit les priorités. L'équipe s'organise pour déterminer la meilleure façon de produire les exigences les plus prioritaires.
- A chaque fin de sprint, tout le monde peut voir fonctionner le produit courant et décider ce qui doit être fait dans le prochain sprint ou livrer le produit s'il est jugé satisfaisant par le client.

Agile – Principes scrum



Click to add text

Le spring

> Un sprint à part, au démarrage du projet

> Permet de définir les bases du produit

> Permet de valider les points techniques durs

> Se concrétise par une première fonctionnalité, même partielle, qui démontre que l'ensemble de la construction tient.

> Permet de faire une première estimation macroscopique du backlog produit

Le spring

- Planification
- Réunion quotidienne
- Revue du sprint
- Rétrospective

Git

- Système de contrôle de versions
- GitHub, Gitlab : outils de gestion de travail collaboratif

- Pourquoi ?
- Performances, sécurité, flexibilité, contrôle de versions

Quelques commandes Git

https://education.github.com/git-cheat-sheet-education.pdf

TP GIT

https://github.com/Cherfalyes/GLA-2022/blob/main/GIT/TP-Git.md