

Génie Logiciel

Chapitre 7:

Introduction aux méthodes de développement

Niveau: 3^{ème} année License informatique

Année: 2020/2021

Processus de développement logiciel

Processus de développement logiciel

- Il ne faut pas oublier que la notation UML est un langage de modélisation objet et **non pas** une méthode objet.
- Nous sommes donc obligé de choisir une méthode de développement. Mais pourquoi?

Et pourquoi une méthode ?



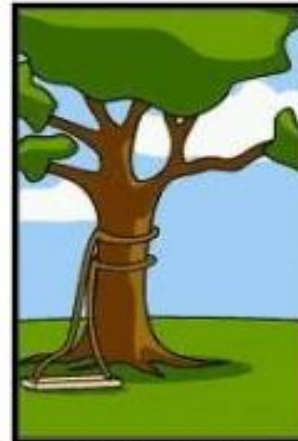
Comment le client a exprimé son besoin



Comment le chef de projet l'a compris



Comment l'ingénieur l'a conçu



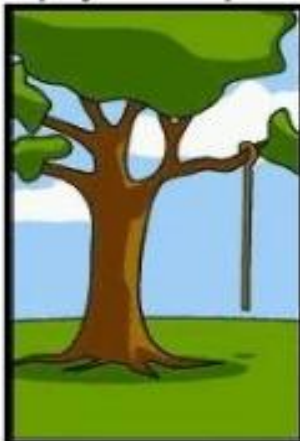
Comment le programmeur l'a écrit



Comment le responsable des ventes l'a décrit



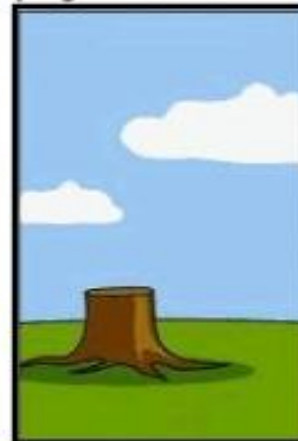
Comment le projet a été documenté



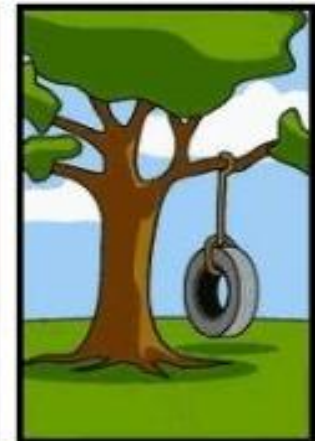
Ce qui a finalement été installé



Comment le client a été facturé



Comment la hotline répond aux demandes



Ce dont le client avait réellement besoin

Processus de développement logiciel

- Un processus définit une **séquence d'étapes**, en partie ordonnées, qui concourent à l'obtention d'un système logiciel ou à l'évolution d'un système existant.
- L'objectif d'un processus de développement est de **produire** des logiciels de qualité qui répondent aux besoins de leurs utilisateurs dans des **temps** et des **coûts** prévisibles.

Processus de développement logiciel

- **Un processus correctement défini aide à choisir:**
 - les artefacts à produire
 - les procédures à développer
 - les intervenants chargés de leur création et de leur gestion
 - la manière d'employer ces artefacts pour évaluer et diriger le projet dans son ensemble

Génération de méthodologies

■ Méthodologies classiques

- Modèles stricts
- Etapes très clairement définies
- Documentation très fournie
- Fonctionne bien avec les gros projets et les projets gouvernementaux
- Exemple: Cascade, V, Incrémental, Spiral ..etc

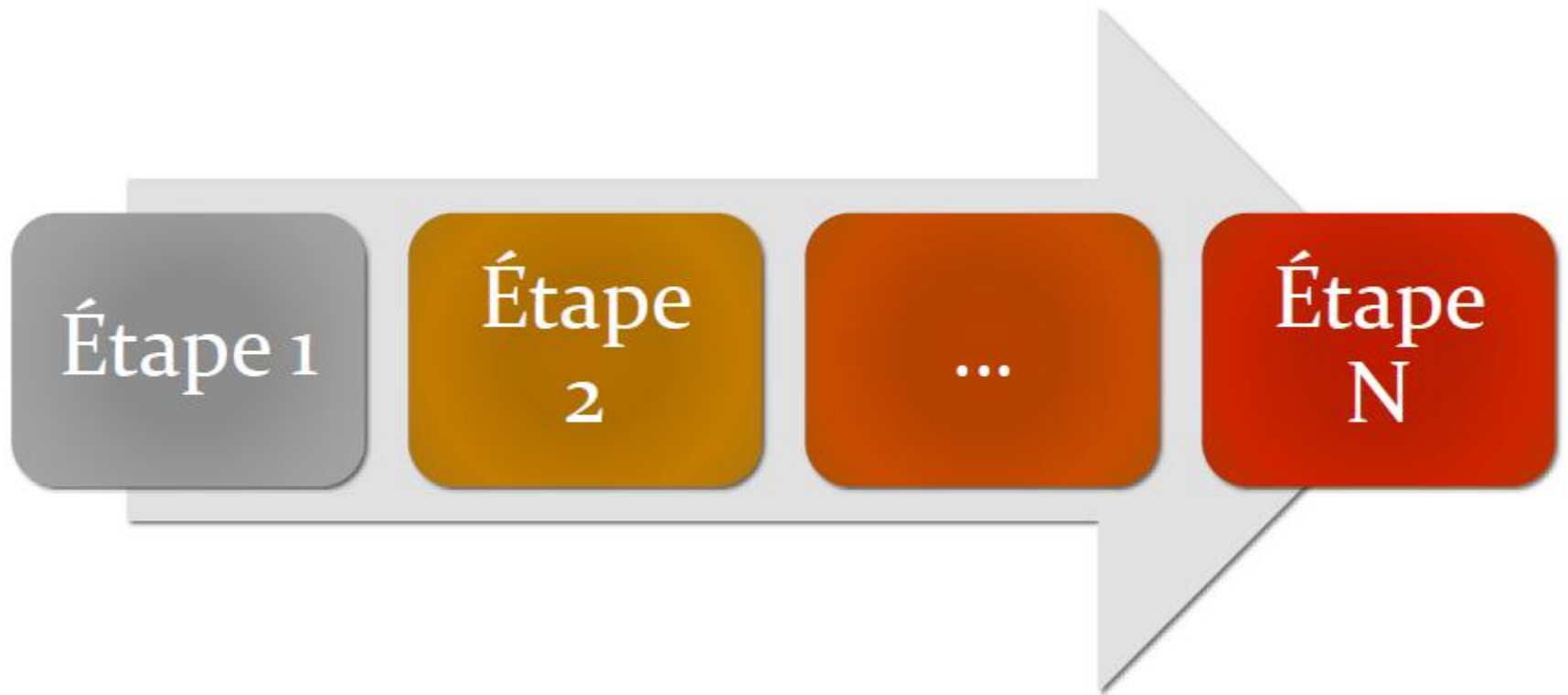
■ Méthodes agiles

- Modèles incrémentaux et itératifs
- Petites et fréquentes livraisons
- Accent sur le code et moins sur la documentation
- Convient aux projets de petite et moyenne taille
- Exemple: Scrum, Extreme Programming (XP), Rapid Application Development (RAD)..etc

Typologie

- Modèle séquentiel
- Modèle incrémental
- Modèle itératif

Modèle séquentiel



Modèle incrémental

1



2



3



Modèle itératif

1



2



3



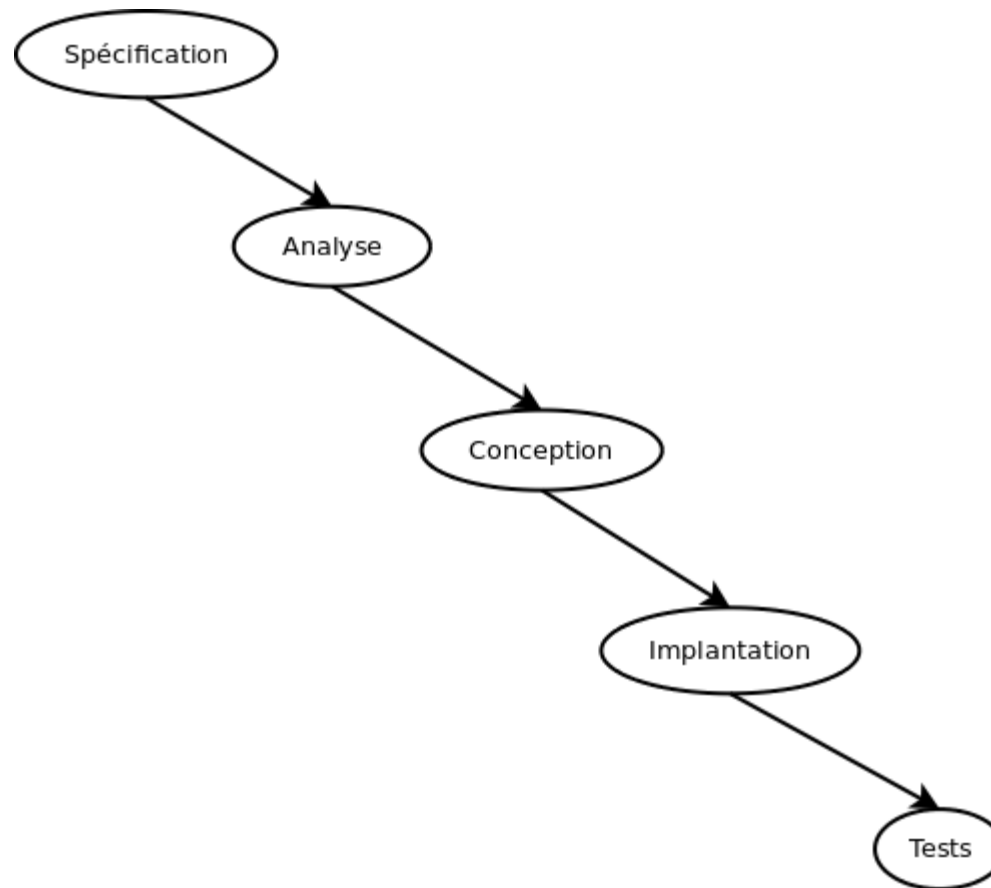
Quand Utiliser Une Méthodologie X ?

- Nature du Projet
- Taille du Projet
- Nature du client
- Exigences du contrat
- Compétences de l'équipe

Méthodologies de Développement Classiques

- Méthodologies de Développement
Classiques

Cycle en cascade



Cycle en cascade

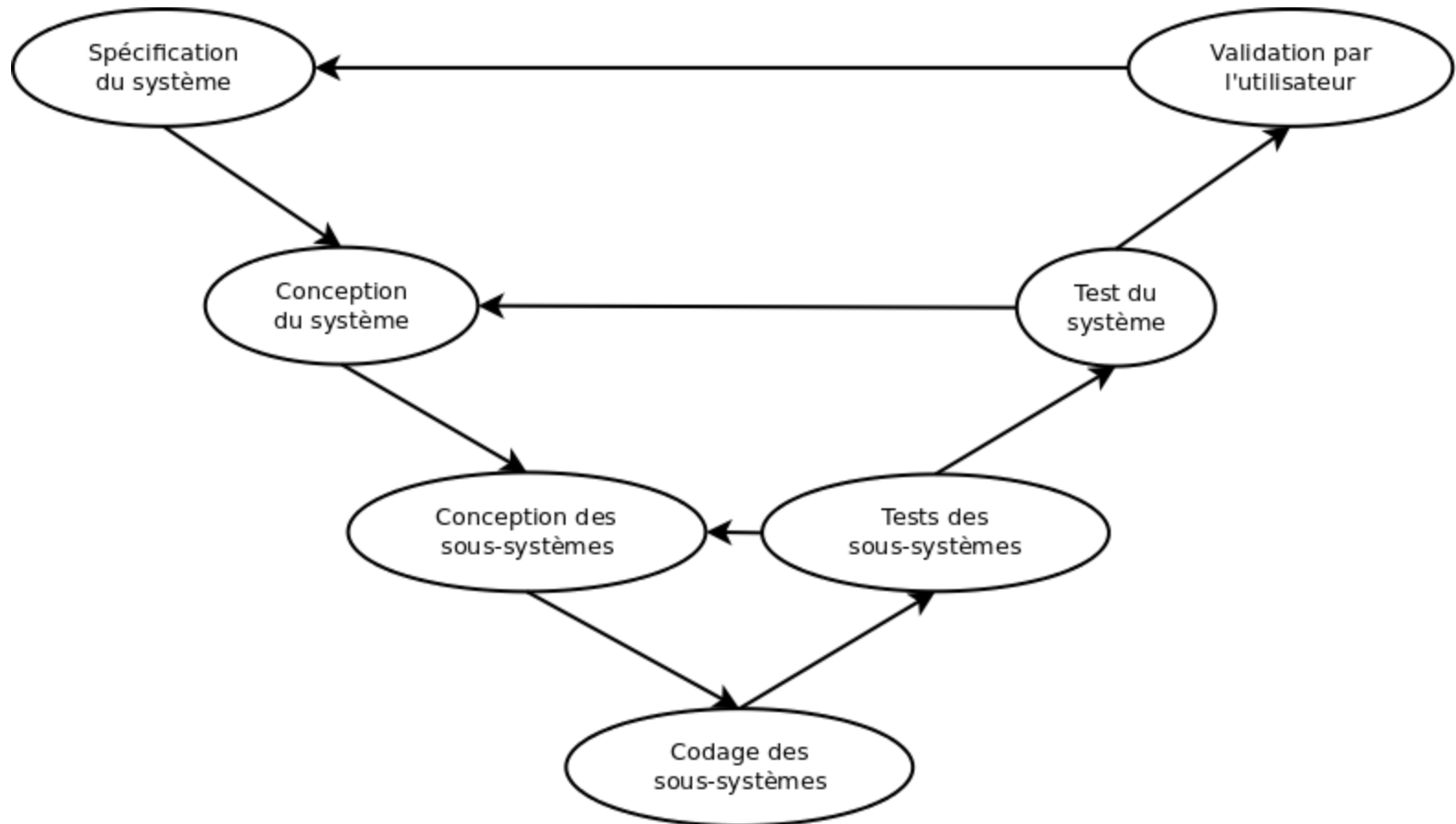
■ Avantages :

- aisé à comprendre et à mettre en œuvre
- forte structuration : définition puis réalisation
- la documentation guide les étapes

■ Inconvénients :

- modèle idéalisé, ne tient pas compte de la nature itérative d'un projet
- logiciel délivré seulement à la fin du projet
- coût de gestion en amont important

Cycle en V



Cycle en V

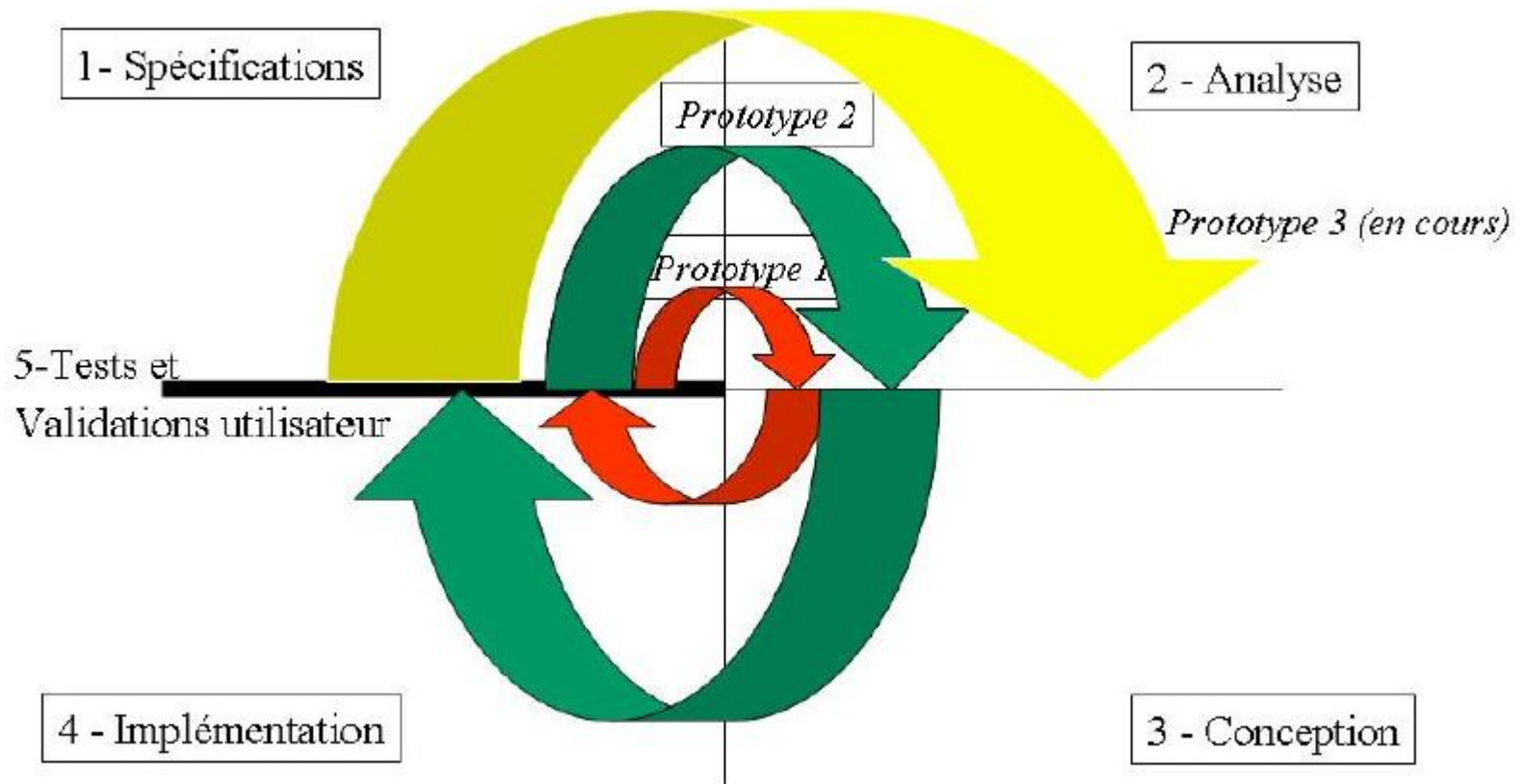
■ Avantages :

- plus réactif que le modèle en cascade
- force l'identification de blocs fonctionnels
- forte structuration des étapes de test

■ Inconvénients :

- hypothèse de stricte séparation entre implantation et spécification
- logiciel délivré seulement à la fin du projet

Cycle en spirale



Cycle en spirale

■ Avantages :

- combine les avantages des modèles en cascade/V
- tient compte de la nature itérative d'un projet
- bonne visibilité au cours du cycle de vie

■ Inconvénients :

- difficile à comprendre sans être expert technique
- nécessite capacité à bien analyser les risques
- nécessite gestionnaires compétents

Méthodes Agiles

■ Historique

- ❑ Au milieu des années 90, un groupe d'expert en gestion de projet de développement voulaient proposer de nouvelles méthodologies
- ❑ Les nouvelles méthodologies sont plus « légères » : moins de documentation et moins de contrôle sur le procédé
- ❑ Ces méthodologies s'adressent à des projets de petite ou moyenne taille avec une équipe réduite
- ❑ Ces méthodologies permettent de s'ajuster rapidement aux changements des spécifications tout en garantissant des livraisons fréquentes
- ❑ Ces méthodologies sont qualifiés de « méthodes agiles »

Méthodes Agiles

■ Les douze (12) principes Agiles

1. Toujours satisfaire le client à travers des livraisons rapides et continues
2. Bien accueillir tous les changements même les tardifs
3. Livrer fréquemment un système fonctionnel
4. Les clients et les développeurs doivent collaborer
5. Conduire le projet autour d'équipes motivées
6. La meilleure méthode de faire circuler l'information c'est le contact direct entre collaborateurs
7. La première mesure d'avancement c'est un logiciel fonctionnel
8. Le développement doit être durable et à un rythme constant
9. La bonne conception et l'excellence technique augmentent l'agilité
10. Simplifier au maximum
11. Les meilleurs architectures, besoins et conceptions proviennent d'équipes qui s'organisent d'elles-mêmes
12. L'équipe s'améliore d'une manière autonome et régulière

Principales Méthodes Agiles

- Adaptive Software Development (ASD)
- Feature Driven Development (FDD)
- Crystal Clear
- Dynamic Software Development Method (DSDM)
- Rapid Application Development (RAD)
- Scrum
- Extreme Programming (XP)

Méthodologie XP

- eXtreme Programming
- Créée en 1995
- XP est un moyen léger, efficace, à bas risques, flexible, scientifique et amusant de développer des logiciels
- Destinée à des équipes de moyenne taille avec des spécifications incomplètes et / ou vagues
- Le codage est le **noyau** de XP

XP - Fondamentaux

- Implication massive du client
- Test unitaire continu (TDD)
- Programmation par paires
- Itérations courtes et livraisons fréquentes

Méthodologie XP – Principales activités

- Codage (noyau de XP)
- Tests (pendant le codage)
- Écoute (le client ou le partenaire)
- Conception (encore basée sur le codage)

Pratiques XP




Organisation

- Programmation par binômes
- Travail énérgisé
- Espace de travail informatif
- Analyse de cause racine
- Rétrospective



Collaboration

- Confiance
- S'asseoir ensemble
- Implication du client
- Langage universel
- Réunion debout
- Standard de codage
- Démo d'itération
- Rapports



Livraison

- Fait Fait
- Pas de bugs
- Contrôle de version
- Génération de 10 minutes
- Intégration Continue
- Propriété Collective
- Documentation



Planification

- Vision
- Plan de livraison
- Jeu de planning
- Gestion des risques
- Planification de l'itération
- Relâchement
- Estimation



Développement

- Exigences Incrémentales
- Tests d'acceptation
- TDD
- Refactoring
- Conception Simple
- Conception et Architecture Incrémentales
- Solutions de Pointe
- Optimisation de Performance
- Tests Exploratoire

Méthodologie XP – Inconvénients

- Demande une certaine maturité des développeurs
- La programmation par paires n'est toujours pas applicable
- Difficulté de planifier et de budgétiser un projet
- Stress dû aux devoirs de l'intégration continue et des livraisons fréquentes
- La faible documentation peut nuire en cas de départ des développeurs

Méthodologie Scrum

- Utilisé comme méthodologie dans le livre : « Agile Software Developmentt with SCRUM” par K. Schwaber et M.. Beedlle en 2001



Méthodologie Scrum - Principes

■ Simple

- ❑ Peut être combiné avec d'autres méthodes
- ❑ Compatible avec les *best practices*

■ Empirique

- ❑ Itérations courtes (sprints)
- ❑ Feedback continu

■ Techniques simples

- ❑ *Sprints* de 2 à 4 semaines
- ❑ Besoins capturés en tant que *user stories*

■ Equipes

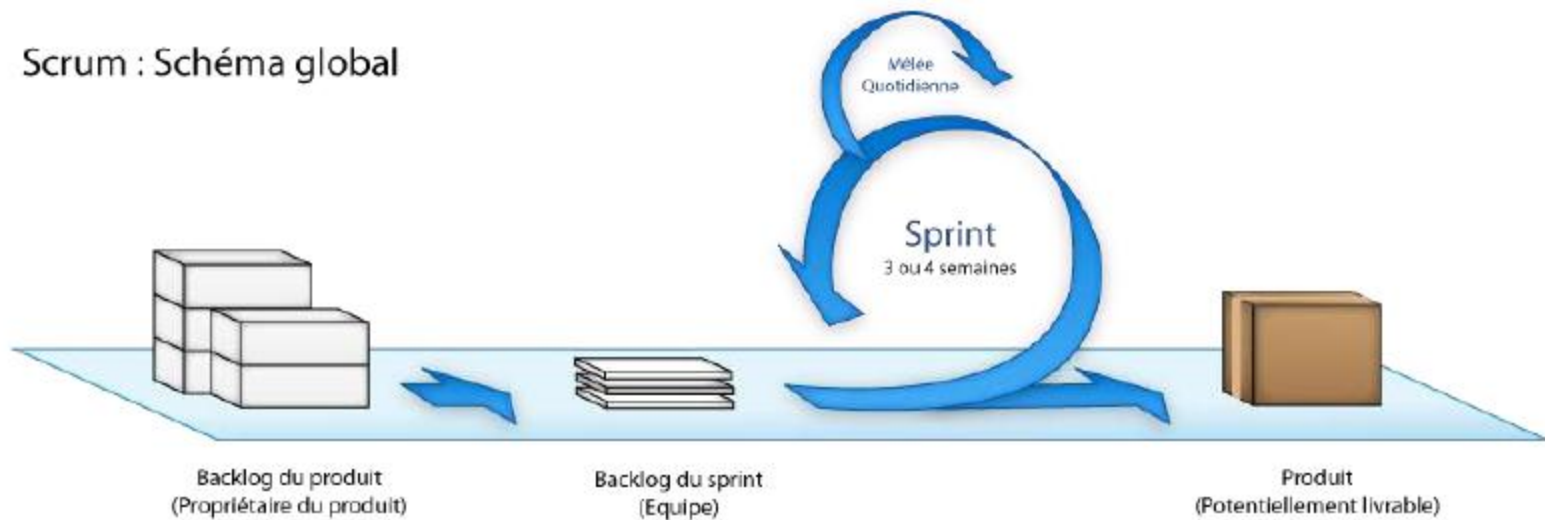
- ❑ Auto-organisation
- ❑ Multi-compétences

■ Optimisation

- ❑ Détection rapide des anomalies
- ❑ Organisation simple
- ❑ Requiert l'ouverture et la visibilité

Méthodologie Scrum - Déroulement

Scrum : Schéma global



Méthodologie Scrum – Principes et Concepts

- Processus
 - Sprint
- Backlog
 - Product Backlog
 - Sprint Backlog
- Equipe
 - Scrum Master
 - Product Owner
 - Team
 - Users and stakeholders
- Réunions
 - Scrum quotidien
 - Planning
 - Revue
 - Rétrospective
- Suivi
 - Rapports
 - Vitesse

Méthodologie Scrum - Avantages

- Méthode très simple et très efficace
- Adoptée par les géants du marché : Microsoft, Google, Nokia..
- Orientée projet contrairement à XP qui est orientée développement
- Peut inclure d'autres activités venant d'autres méthodologies (surtout XP)

Méthodologie Scrum - Inconvénients

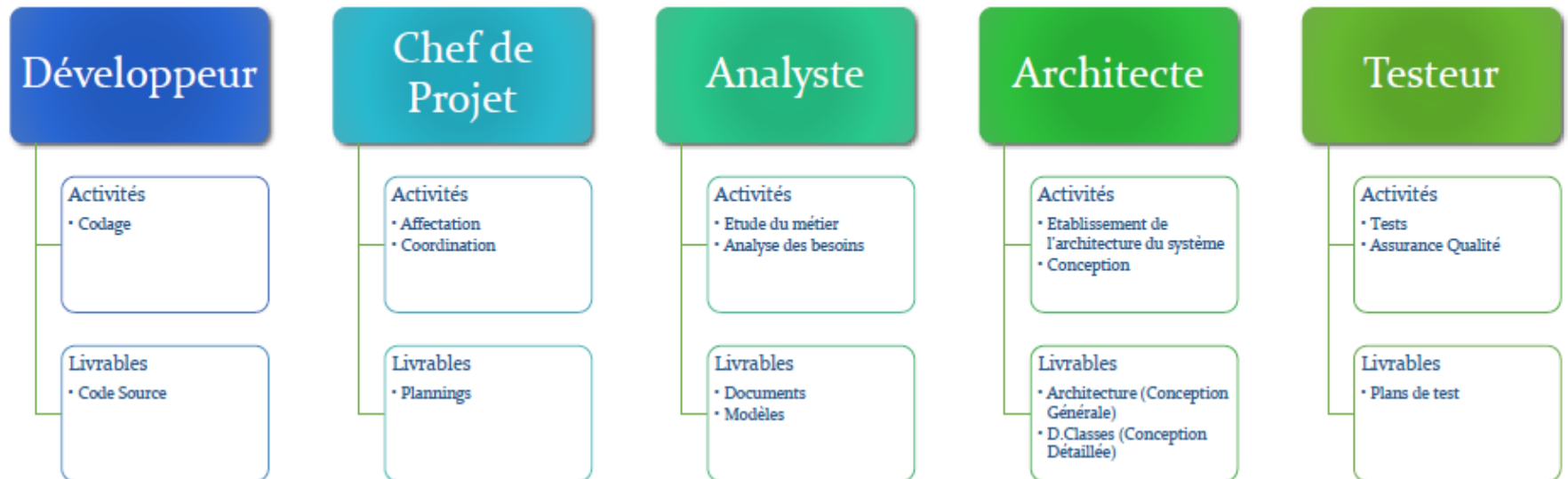
- N'est pas 100% spécifique au GL
- Difficulté de budgétiser un projet

Autres procédés

- Nombreuses autres processus:
 - **Processus Unifié (UP) :**
 - Rational Unified Process (RUP)
 - Agile Unified Process (AUP)
 - Basic Unified Process (BUP)
 - Enterprise Unified Process (EUP)
 - Essential Unified Process (EssUP)
 - Open Unified Process (OpenUP)
 - Oracle Unified Method (OUM)
 - **Ambler**
 - **Merise**
 - **SADT**
 - **HERMES. . .**
- Un autre classement: trois grandes familles :
 - **ascendante**
 - **descendante**
 - **Agile**
- **Noter bien:** Parfois: **Méthodologie = Langage + Processus**
 - **Exemple: UML & Y**

Outils et Métiers

■ Principaux Métiers de Développement



Outils et Métiers

■ Métiers et Activités

| | Expression de besoins | Analyse | Conception | Implémentation | Tests |
|----------------|-----------------------|---------|------------|----------------|-------|
| Développeur | | | | | |
| Analyste | | | | | |
| Architecte | | | | | |
| Testeur | | | | | |
| Chef de Projet | | | | | |

Outils et Métiers

- Outils CASE (Computer-aided software engineering)
 - CASE est un nom donné aux **logiciels** utilisés dans les différentes activités de GL (besoins, conception,...)
 - Exemples : compilateurs, éditeurs, débogueurs, ...etc.
 - Le but des outils CASE est d'automatiser les tâches et / ou gérer le projet de développement

Outils et Métiers

- Classification des CASE: Les outils CASE peuvent être classés :
 - D'un point de vue **fonctionnel** : selon la fonction de l'outil.
 - D'un point de vue **activité** : selon les activités dans lesquelles intervient l'outil

Outils et Métiers

■ Classification fonctionnelle:

| Outil | Exemples | Références | Métier |
|---------------------------------|---|---|------------------------------|
| Outils de planification | Outils PERT, outils d'estimation, tableurs | Microsoft Project, Excel, GanttProject, DotProject | CDP |
| Editeurs | Editeurs de texte, éditeurs d'image, éditeurs de diagrammes | vi, bloc notes, GIMP, Photoshop, Visio | Tous |
| Gestion de configuration | Gestion de versions, gestion de builds | SVN, CVS, Team Foundation Server, ClearCase | CDP, Développeur, Architecte |
| Outils de support de procédé | Générateurs de code, outils d'assistance, IDE | Team Foundation Server, Accurev, Enterprise Architect | Tous |
| Outils de traitement de langage | Compilateurs, interpréteurs, débogueurs | | Développeur, Architecte |

Outils et Métiers

■ Classification fonctionnelle – Suite:

| Outil | Exemples | Références | Métier |
|-------------------------|--|---|----------------------|
| Outils de test | Environnements de tests, outils de tests unitaires | Junit, Nunit, TestWorks, Bugzilla | Testeur, Développeur |
| Outils de documentation | Documents de projet, documents de code | Word, Open Office, Sandcastle, Doxygen, javadoc | Développeur |