Ingénierie Logicielle

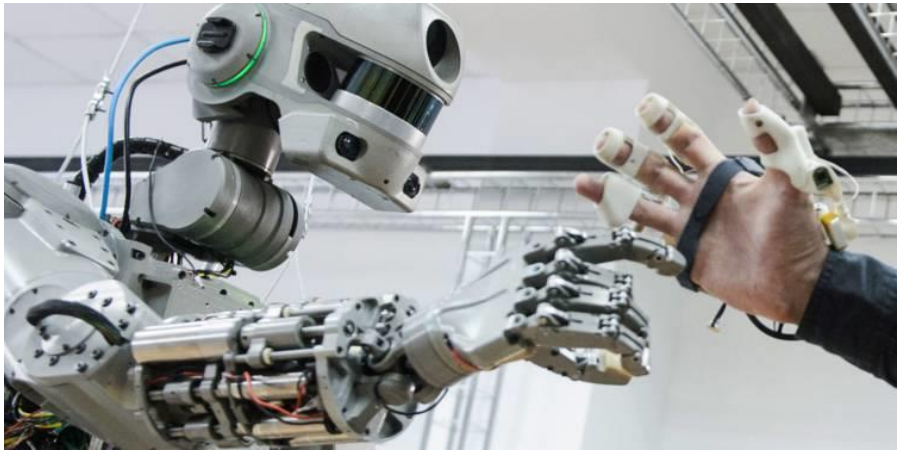
Processus de développement du logiciel

Prof. Ousmane SALL, Maître de Conférences CAMES en Informatique

Objectifs de la séquence

- A l'issue de cette séquence vous serez capable de :
  - Expliquer le cycle de vie d'un logiciel;
  - Comprendre les concepts de processus logiciels et modèle de processus logiciels
  - Décrire et comparer certains processus et modèles de développement classiques et plus récents;
  - Lister les enjeux et livrables liés à chacune des phases du développement.

Les logiciels et la société



Génie logiciel

- Mettre en place des méthodes efficaces de management
- Créer de nouveaux outils et améliorer les outils existants
- Établir des normes de gestion, d'organisation, de communication
- Une discipline qui comprend:
  - Le processus de développement de logiciels
  - La méthodologie pour l'analyse, la conception, le développement, la vérification et la maintenance de logiciels
  - Des outils qui supportent le processus et la méthodologie

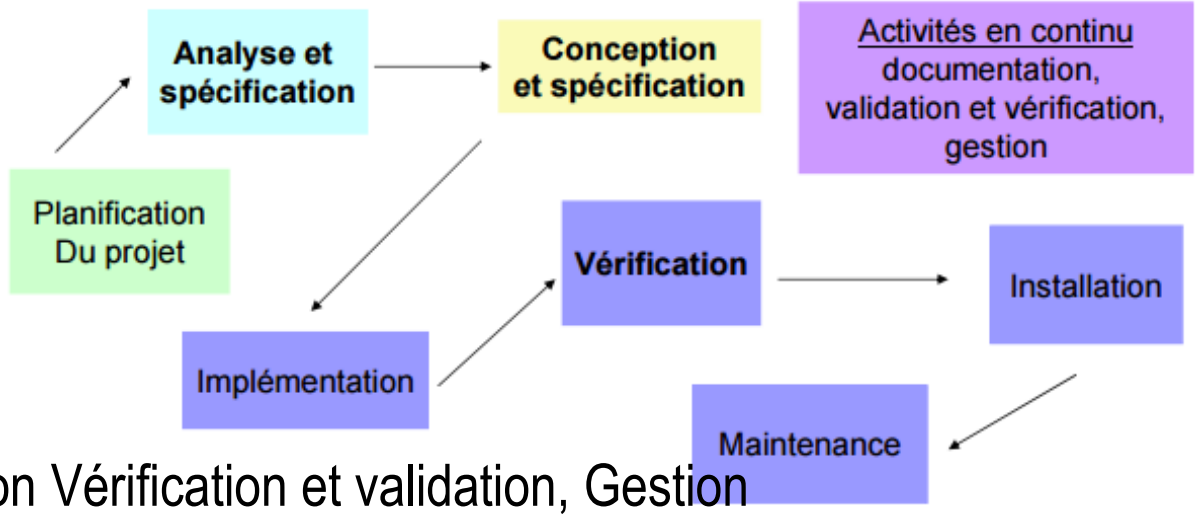
Génie logiciel / Ingénierie logicielle: Science qui s'intéresse aux méthodes de travail et aux bonnes pratiques de développement.

Processus de développement du logiciel:
Cycle de vie du logiciel

Cycle de vie du logiciel : Activités de développement (phases) ou Processus de Développement de Logiciel

- Planification du projet
- Analyse et spécification
- Conception
- Implémentation
- Vérification
- Installation
- Maintenance

Organisation de ces activités dans le temps (processus de développement).



- En continu: Documentation Vérification et validation, Gestion

Cycle de vie du logiciel: Planification(étude préliminaire)

- Question : **Est-ce possible ?**
- Définition globale du problème
- Confirmer la faisabilité
  - évaluation des stratégies possibles
  - évaluation des ressources, coûts et délais, effort
- Produire le calendrier du projet
- Trouver le personnel
- Estimation des coûts réels, devis
- Lancer le projet

Documents: Rapport de planification

Cycle de vie du logiciel: Analyse des besoins/Spécifications

- Question : **Quoi faire ?**
- Cueillette d'informations
  - exigences fonctionnelles
  - qualités non-fonctionnelles (contraintes)
- Spécification du système
  - accord entre le développeur du système et le client / l'utilisateur

- Documents:
  1. Cahier des charges
  2. Document de spécification (analyse)
  3. Prototype
  4. Pan de test.

Cycle de vie du logiciel: Analyse des besoins

- Construction de prototypes (pour élaborer la spécification)
  - Prioriser les éléments de la spécification
  - Produire et évaluer des solutions alternatives
  - Examiner les recommandations avec le chef de projet et/ou le client…

- Documents
  1. Cahier des charges/ document de spécification (analyse)
  2. Prototype et Plan de test

Cycle de vie du logiciel: Conception

- Question: **Comment faire ?**
  - Conception architecturale: décomposition et organisation de l'application en modules plus simples définis par une interface. Bases de données, environnement d'exploitation, interfaces.
  - Conception détaillée: pour chaque module, description de la manière dont les services et fonctions sont réalisés:
    - algorithmes essentiels
    - structures de données utilisées, etc
- Concevoir l'architecture d'application
- Concevoir les interfaces utilisateur
- Concevoir les interfaces du système
- Concevoir et intégrer la base de données

Documents

1. Document de conception (spécification) prototype
2. Plan de test global
3. Plan de test par module

Cycle de vie du logiciel: Implémentation

- Traduction de la conception dans un langage de programmation ou mise en œuvre en utilisant des outils de développement
- Construire les composantes logicielles

- Documents
  1. Dossiers de programmation
  2. Code source commenté
  3. Prototype

Cycle de vie du logiciel: Vérification

- Q: **Est-ce bien fait ?**
- Évaluation de la solution en fonction de la spécification
- Différents niveaux de tests
  - Tests unitaires: par module
  - Tests d'intégration: composition de modules
  - Tests de système: logiciel entier
  - Tests d'acception: définis par le client

- Documents : Rapport de vérification par test

Cycle de vie du logiciel: Installation / déploiement

- Mise en fonctionnement opérationnel chez les utilisateurs
- Conversion des données
- Parfois restreint à des utilisateurs sélectionnés
  - alpha / beta testing

Cycle de vie du logiciel: Maintenance

- **Maintenance corrective**: corriger les erreurs
- **Maintenance adaptative**: s'adapter à des changements d'environnement
- **Maintenance perfective**: améliorations
- **Maintenance préventive**: pour faciliter les opérations de maintenance à venir

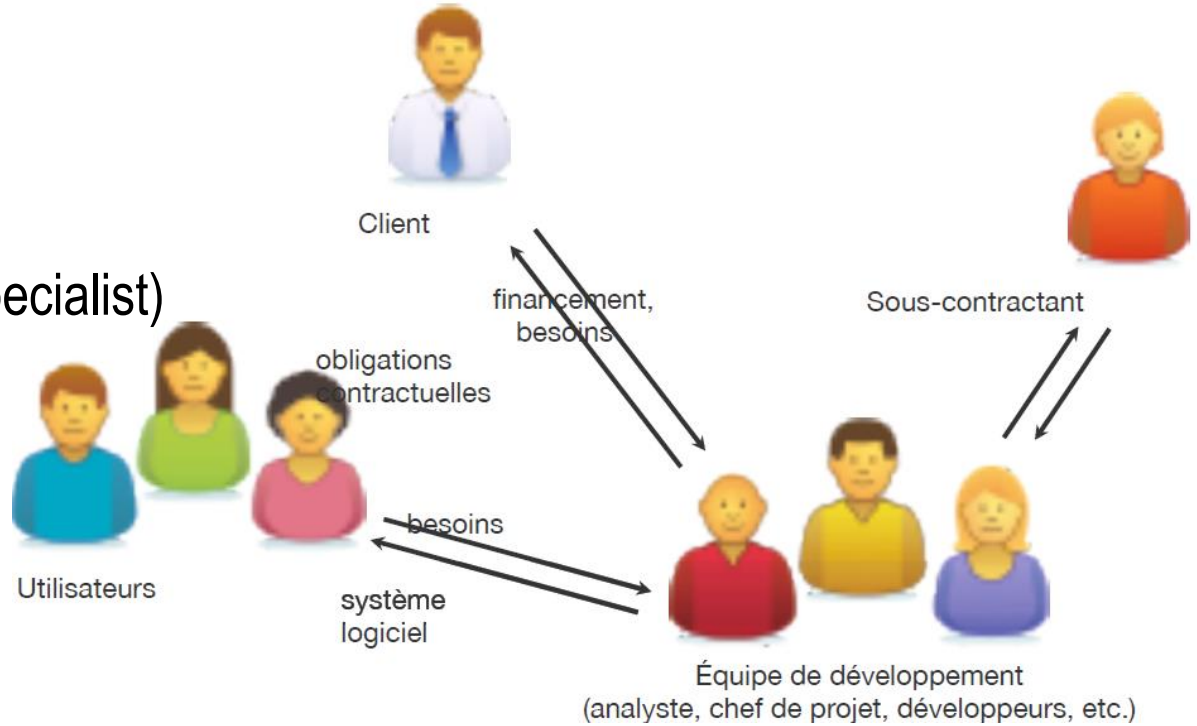
Cycle de vie du logiciel: Activités en continu

- **Gestion**
  - Du processus de développement (suivi de projet, révision, etc.)
  - De la configuration: politique de gestion des versions, des documents, politique de réutilisation
  - Des ressources humaines
  - Du risque
- Vérification: «**Construit-on le produit comme il faut ?**»
  - Le produit est-t-il correct (par rapport à la spécification)?
  - S'assurer de la qualité du produit (révisions et inspections)
  - S'assurer de satisfaire la spécification
- Validation: «**Construit-on le bon produit ?**»
  - Le produit répond-il aux besoins du client ?

- Documentation: Traçabilité

L'équipe de développement et acteurs externes

- Analyste
- Concepteur
- Programmeur
- Testeur
- Formateur (Training specialist)



UVS
UNIVERSITE VIRTUELLE DU SENEGAL

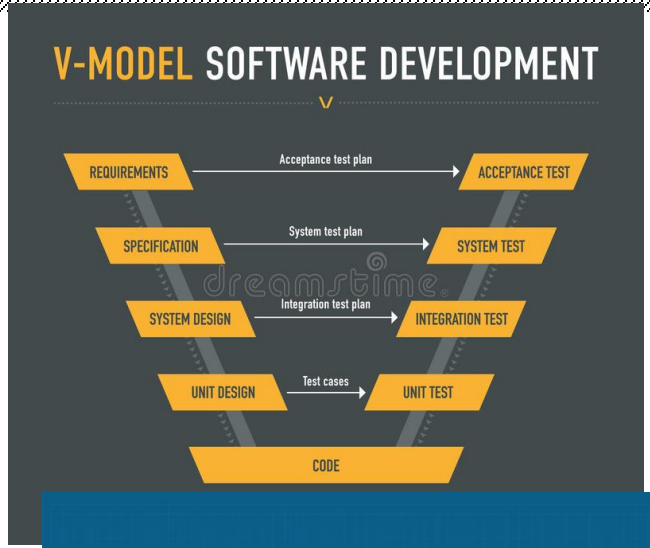
Processus de développement du logiciel:
Modèles de développement logiciel

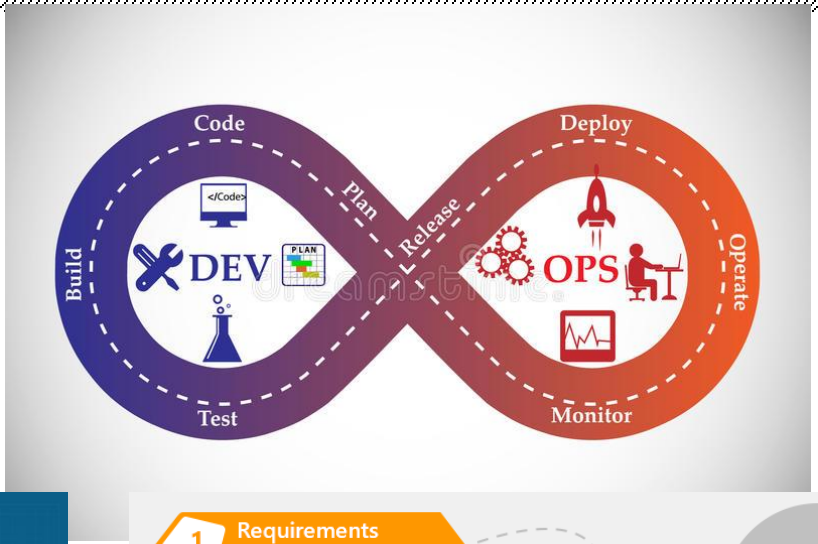
Modèles de développement logiciel

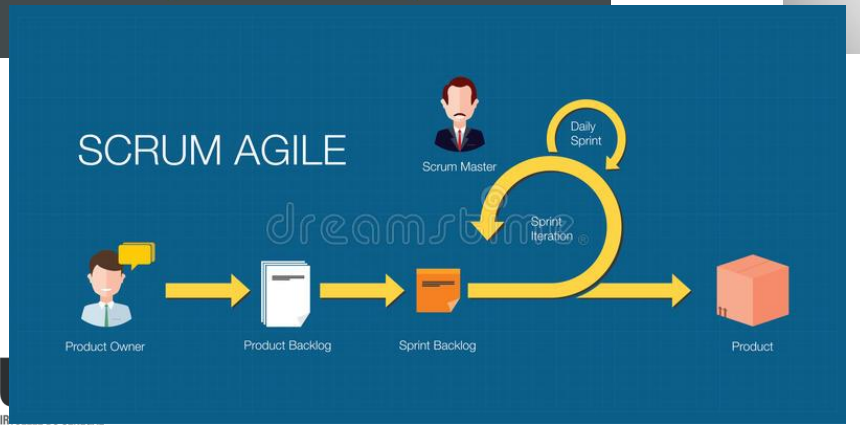
- Un processus de développement définit un ensemble d'activités et leur enchaînement
- Une activité comprend : des tâches, des contraintes, des ressources, une façon d'être réalisée
- La plupart des modèles des processus reprennent les activités fondamentales mais les organisent différemment:
  - De nombreux modèles ont été définis
  - Un modèle peut être spécifique à une organisation et à un type de logiciels (ex: embarqué)
  - Il existe malheureusement peu d'outils supportant les processus des directives pour organiser la manière dont les activités de processus logiciel doivent être effectuées et dans quel ordre.

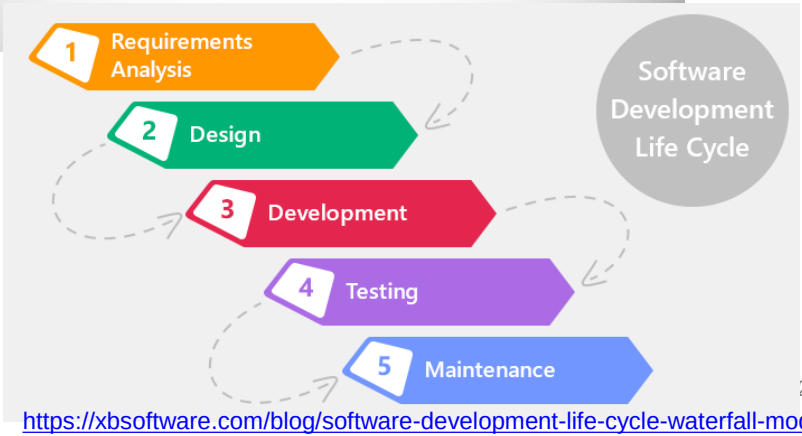
Modèles de développement logiciel

https://fr.dreamstime.com/illustration-stock-d%C3%A9veloppement-de-logiciel-de-v-mod%C3%A8le-image72624533





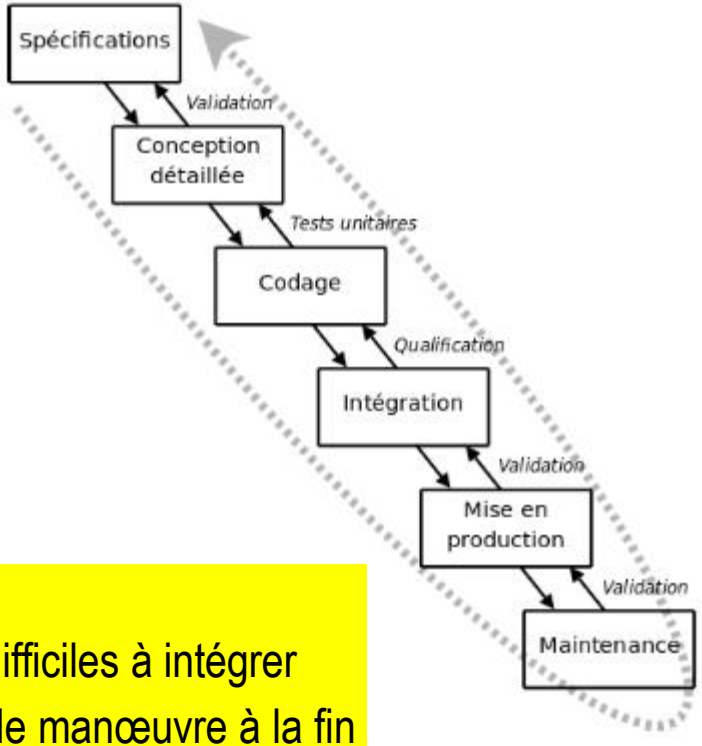




https://xbsoftware.com/blog/software-development-life-cycle-waterfall-model/

Modèle en cascade

- Modèle hérité du BTP
  - "fondation avant toiture"
- Étapes effectuées
  - séquentiellement
- Retour sur les étapes
  - précédentes à la fin



Avantages:

- Facile à mettre en œuvre
- Facile à comprendre
- Efficace pour les développements internes

Inconvénients:

- Changements difficiles à intégrer
- Peu de marge de manœuvre à la fin
- Une phase ne démarre que lorsque la précédente est finie
- Produit visible uniquement à la fin
- Implication du client faible

Modèle en V

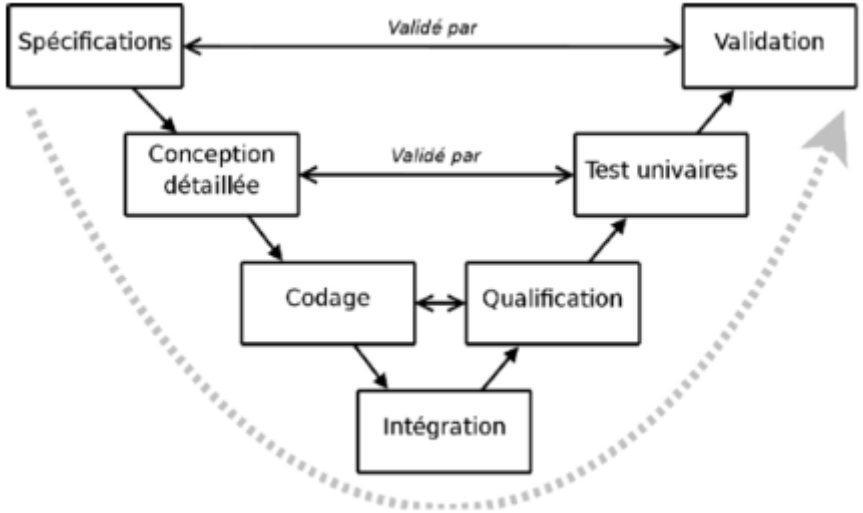
- Proposer pour pallier les problèmes de réactivité
- Chaque étape est validée avant de passer à la suivante
- Limite le retour brutal aux étapes précédentes

Avantages:

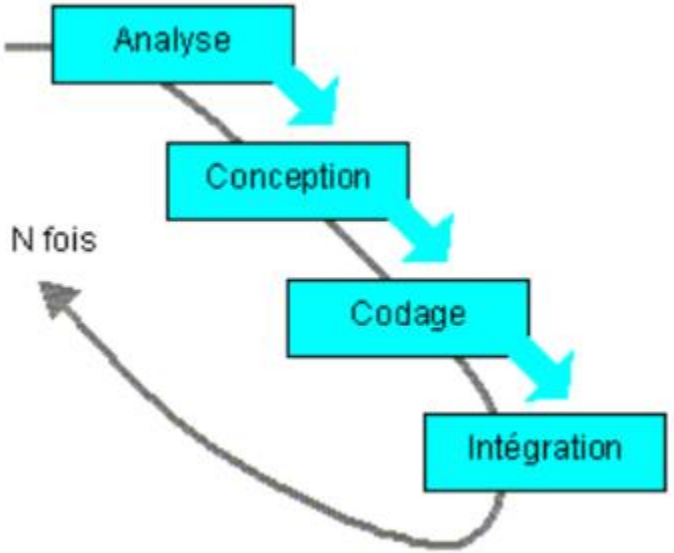
- Facile à mettre en œuvre
- Chaque étape est validée
- Permet d'anticiper les étapes suivantes

Inconvénients:

- Parallélisassions des étapes impossible
- Très sensible aux évolutions des besoins
- Produit visible à la fin



Cycle de vie: Modèle itératif



- Trier les besoins par priorités/dépendances
- Chaque itération est une construction qui répond à un besoins (ou +)
- Les phases d'analyse tiennent compte de la connaissance acquise
- Livrer régulièrement des versions de tests au client

Avantages:

Met une priorité sur les parties à risques
Chaque itération fournit un produit fonctionnel
Le client impliqué à la fin de chaque itération

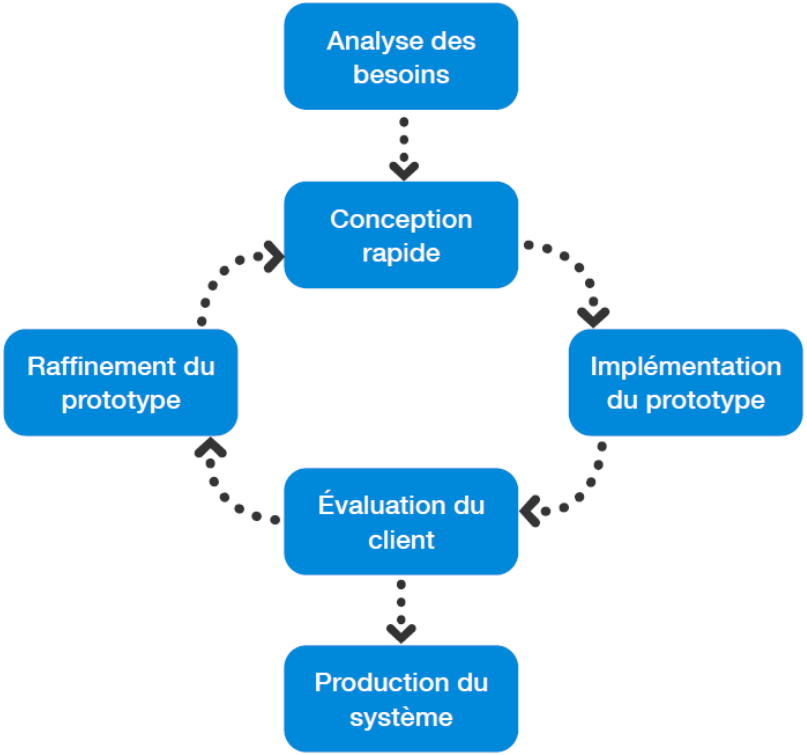
Inconvénients:

Nécessite d'avoir des le départ une vision d'ensemble sur le produit pour bien le subdiviser
Exigence une bonne planification et une bonne conception

Modèle par prototypage

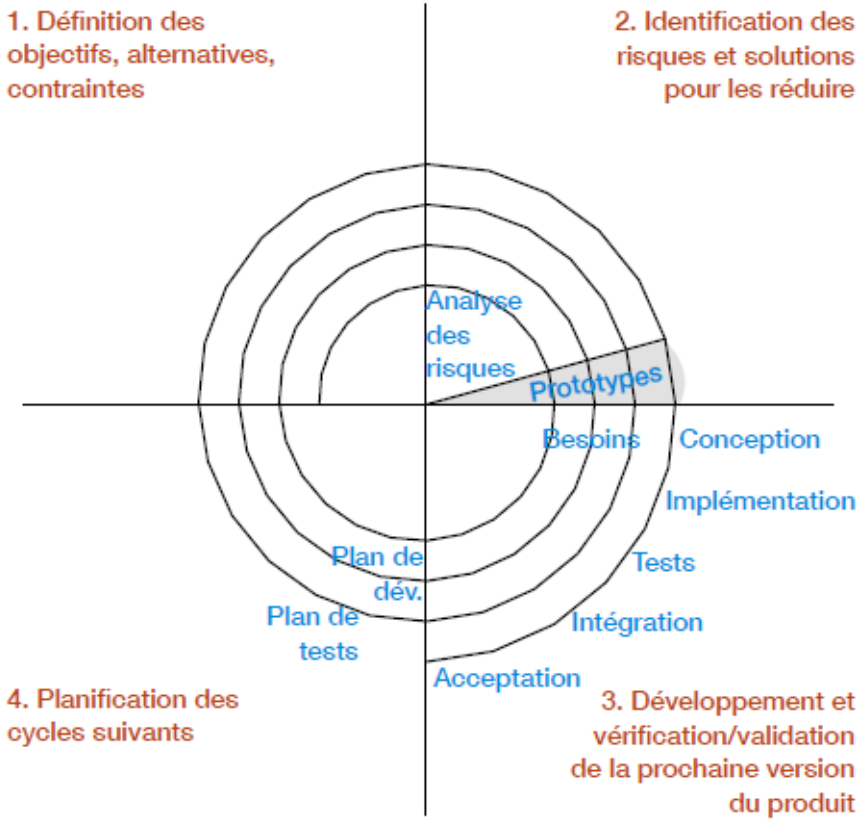
Construire un prototype jetable pour mieux comprendre les points durs (exigences, technologies)

- À chaque étape, un ou plusieurs prototypes sont soumis au client pour évaluation ou révision.
- Permet d'examiner et d'explorer certains aspects du système pour évaluer et choisir les meilleures stratégies/solutions.

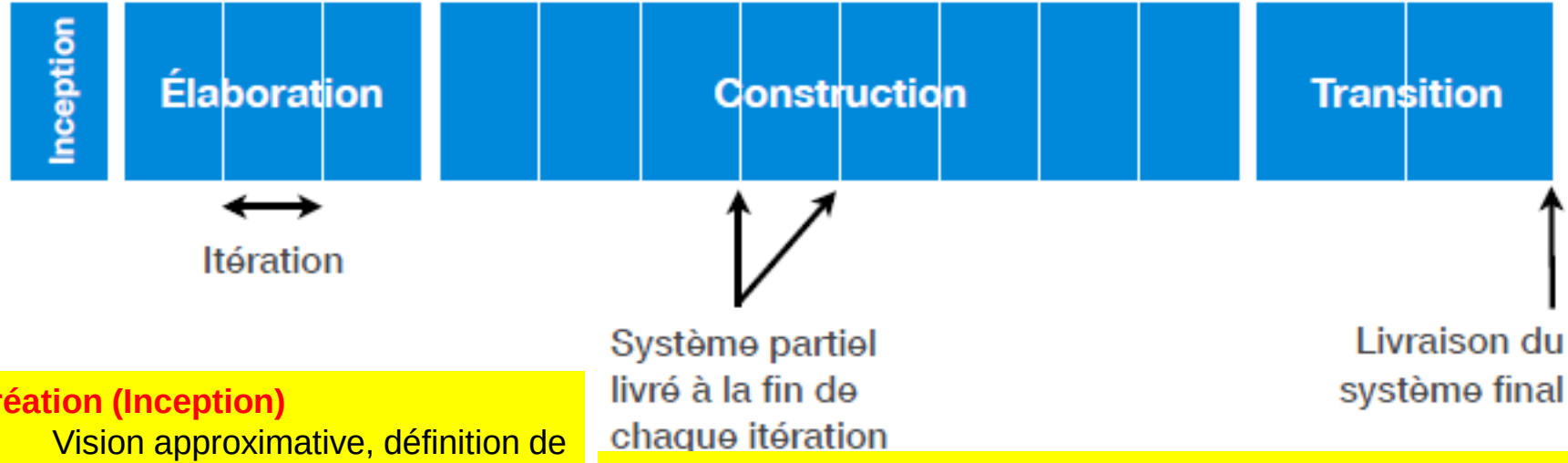


Modèle en spirale

- Axé sur les risques
- Chaque cycle comprend 4 étapes
  - Définition et identification des objectifs, alternatives, contraintes
  - Evaluation (par simulation, prototypage, etc.) des alternatives et risques
  - Développement: analyse, conception, implémentation, test
  - Evaluer les résultats des étapes traversées planifier le prochain cycle.
- Le modèle en cascade en un cas spécial d'une spirale à un cycle



Processus unifié



Création (Inception)

Vision approximative, définition de l'étendue du projet, estimés vagues
Le projet est-il réalisable?
Création ≠ analyse

Construction

Implémentation itérative des éléments plus simples / à plus faible risque
Préparation pour le déploiement

Élaboration

Vision raffinée
Développement itératif de l'architecture de base
Résolution des risques les plus élevés
Identification de la plupart des besoins et Estimés plus réalistes

Transition

Implantation du système dans un environnement de production

Méthodes Agiles

- Variables d'ajustement d'un projet:
  - **Coût**
  - **Qualité**
  - **Durée**
  - **Périmètre fonctionnel**
- Règle du jeu :
  - Le client a le droit de fixer 3 variables
  - L'équipe de développement ajuste la dernière.
- Le **périmètre fonctionnel** est la variable qui fournit la maîtrise la plus efficace.

Barry W. Boehm a introduit en 1986 un nouveau modèle de développement itératif et incrémental, précurseur des méthodes Extreme programming (XP), Scrum ou Crystal clear…
En 2001, un manifeste écrit par 17 experts introduit 4 valeurs fondamentales déclinées en 13 principes permettant de définir une nouvelle façon de développer des logiciels.
http://www.agilemanifesto.org/

Les 4 valeurs de l'Agilité

- **L'équipe** :Les individus et leurs interactions avant les processus et les outils.
- **L'application** : Des fonctionnalités opérationnelles avant la documentation.
- **La collaboration** : Collaboration avec le client plutôt que contractualisation des relations.
- **L'acceptation du changement** : Adaptation au changement plutôt que conformité aux plans

Responsabilisation de l'équipe de développement Agile

- Les méthodes Agiles responsabilise l'équipe :
  - l'équipe connaît les besoins et les priorités,
  - elle fait les estimations,
  - elle décide de son organisation,
  - elle produit un travail de qualité,
  - elle remonte les problèmes.

Les différentes méthodes Agiles

- Adaptative Software Development (ADS)
- Crystal
- Extreme Programming (XP)
- **Scrum**
  - Objectif : Améliorer la productivité des équipes
    - Eviter l'utilisation de méthodes lourdes
    - Produire la + grande valeur métier dans la durée la + courte
  - Gestion de projets de développement de logiciels
  - Créée par *Ken Schwaber et Jeff Sutherland* En 1996

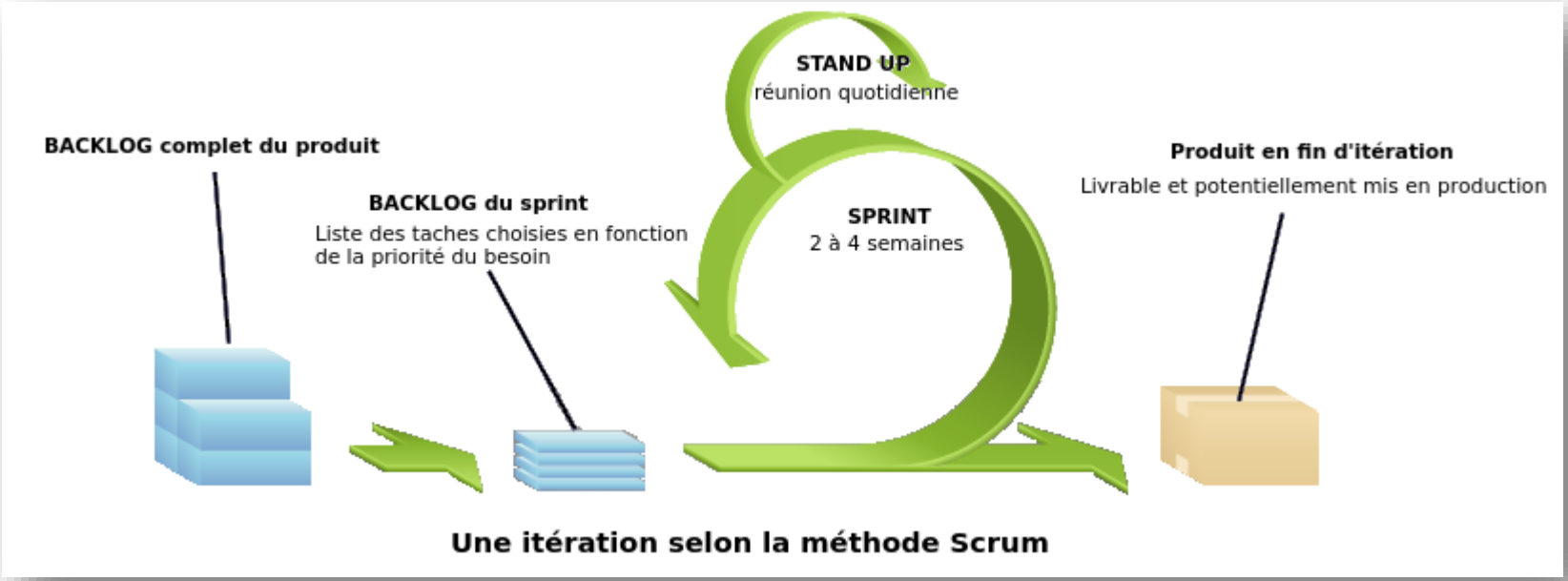
Scrum

- Origine du terme
  - En rugby « mêlée »
  - Equipe soudée qui cherche à atteindre un but précis
- Méthode itérative
  - Réalisation d'un ensemble de fonctionnalités par itération
  - Itération d'une durée fixe (2 à 4 semaines) → **Sprints**
  - Livraison d'un produit partiel fonctionnel par itération
- Participation du client
  - Définition des fonctionnalités prioritaires
  - Ajout de fonctionnalités en cours de projet (pas pendant un sprint !)

Scrum

- Scrum est une méthode Agile qui permet de produire la plus grande valeur métier dans la durée la plus courte.
- Du logiciel qui fonctionne est produit à chaque sprint, c'est à dire toutes les 2 / 4 semaines.
- Le métier définit les priorités, l'équipe s'organise elle-même 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 soit de le livrer dans l'état, soit de continuer à l'améliorer pendant un sprint supplémentaire.

Cycle de vie de Scrum



Une itération selon la méthode Scrum

Les rôles dans une équipe Scrum

- Un directeur de produit (**product owner**) qui est soit le client, soit une personne représentant le client, il:
  - définit les fonctionnalités du produit
  - choisit la date et le contenu de la release
  - responsable du retour sur investissement
  - définit les priorités dans le backlog en fonction de la valeur métier
  - ajuste les fonctionnalités et les priorités à chaque sprint si nécessaire
  - accepte et rejette les résultats

Les rôles dans une équipe Scrum

- Un Scrum Master qui:
  - représente le management de projet est responsable de faire appliquer les valeurs et les pratiques de Scrum par l'équipe
  - résout les problèmes
  - s'assure que l'équipe est complètement fonctionnelle et productive
  - facilite une coopération poussée entre tous les rôles et fonctions
  - protège l'équipe des interférences extérieures

Les rôles dans une équipe Scrum

- Les équipiers qui:
  - se composent de 5 à 10 personnes
  - regroupent tous les rôles: architecte, concepteur, analyste, développeur, testeur, …
  - sont à plein temps sur le projet
  - s'organisent eux-mêmes
  - ne changent pas de composition pendant un sprint
  - se concentrent sur un sprint à la fois (sprint courant)

Les événements Scrum

- **Planification du Sprint (2 à 4h)**
  - Définir le but du sprint
  - Définition du périmètre du sprint
  - Identification les tâches à partir des éléments sélectionnés
  - Estimation des tâches
  - Attribution des tâches
  - Obtenir l'engagement de l'équipe
- **Participants** : Partie une : Product Owner, Equipe, ScrumMaster. Partie deux : EquipeScrumMaster, Product Owner.

Les événements Scrum

- **Planification du Sprint (2 à 4h)**
  - liste de tâches est appelée le Sprint Backlog.

				Nouvelles estimations de l'effort						
				Nombre de jours restants						
Elément du Product Backlog	Tâche du Sprint	Volontaire	Effort initial estimé	1	2	3	4	5	6	7
En tant qu'acheteur, je veux placer un livre dans mon panier	Modifier le base de données		5							
	Créer la page web (UI)		8							
	Créer la page web (logique Javascript)		13							
	Ecrire les tests automatisés de recette		13							
	Mettre à jour la page d'aide pour l'acheteur		3							
	...									
Améliorer la performance du traitement des transactions	Fusionner le code DCP et terminer les tests		5							
	Terminer la commande machine pour pRank		8							
	Changer DCP et lecteur pour utiliser API HTTP pRank		13							

Un exemple de Sprint Backlog

Les événements Scrum

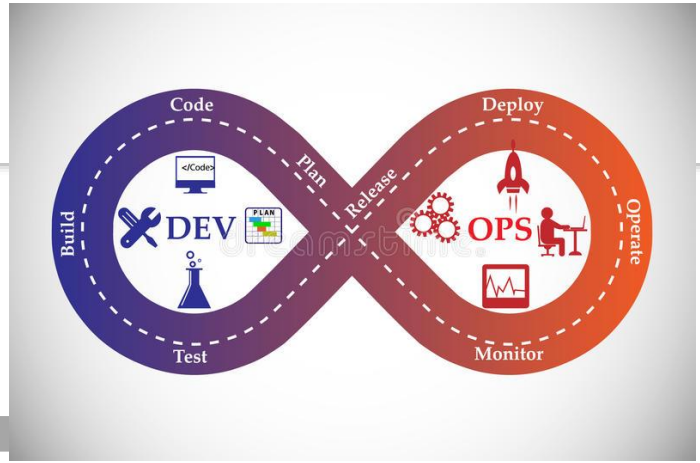
- **Scrum quotidien (15mn debout) ou Mêlée quotidienne**
  - Qu'as-tu fait depuis la dernière fois ?
  - Que prévois-tu de faire jusqu'à la prochaine réunion ?
  - Qu'est-ce qui te gêne pour réaliser ton travail aussi efficacement que possible ?
- **Participants** : l'ensemble de l'Equipe est requise; le Product Owner est optionnel ; le ScrumMaster est généralement présent mais s'assure que l'Equipe est autonome.

- **Revue de sprint (2 à 4h)**
  - Préparer la démonstration
  - Rappeler les objectifs du sprint
  - Effectuer la démonstration
  - Évaluer les résultats du sprint
  - Calculer la vélocité réelle et ajuster le plan de release
- **Participants :** l'Equipe; le Product Owner, le SrumMaster. Toutes parties prenantes en fonction du besoin, conviées par le Product Owner.

DevOps (Développement et Opérations)

Le **devops** est un mouvement visant à l'alignement de l'ensemble des équipes du système d'information sur un objectif commun, à commencer par les équipes de dev ou dev engineers chargés de faire évoluer le système d'information et les ops ou ops engineers responsables des infrastructures (exploitants, administrateurs ...

Devops — Wikipédia
https://fr.wikipedia.org/wiki/Devops



Quel modèle choisir ?

- Le choix d'un modèle dépend des conditions et des exigences du projet.
- Une combinaison de modèles est parfois utilisée pour tirer parti des avantages de plusieurs modèles.
- Critères d'évaluation des modèles:
  - Gestion des risques
  - Contrôle qualité / coût
  - Visibilité du progrès
  - Première fonctionnalité du système
  - Implication client et feedback

Gestion de projets (Voir Cours de Gestion de Projets)

«Ensemble d'outils, de techniques et de méthodes qui permettent au chef de projet et à son équipe de conduire, coordonner et harmoniser les diverses tâches exécutées dans le cadre du projet.»

Réf.: Association Française de Normalisation (AFNOR) Source image: : http://www.dianedelaraitrie.com/wp-content/uploads/2013/06/gestion-de-projet-t-shirt-copie-1024x511.jpg

Processus de développement du Logiciel:
A Retenir...

A retenir…

- Les processus logiciels sont les activités impliquées dans la production d'un système logiciel. Les modèles de processus logiciels sont des représentations abstraites de ces processus.
- Les modèles de processus généraux décrivent l'organisation des processus logiciels. Des exemples de ces modèles généraux incluent le modèle en cascade, le développement incrémental et le développement orienté vers la réutilisation.
- Les processus de conception et de mise en œuvre consistent à transformer une spécification des exigences en un système logiciel exécutable. Des méthodes de conception systématiques peuvent être utilisées dans le cadre de cette transformation.

A retenir…

- L'évolution du logiciel a lieu lorsque vous modifiez les systèmes logiciels existants pour répondre aux nouvelles exigences. Les changements sont continus et le logiciel doit évoluer pour rester utile.
- Les processus doivent inclure des activités pour faire face au changement. Cela peut impliquer une phase de prototypage qui permet d'éviter de mauvaises décisions sur les exigences et la conception. Les processus peuvent être structurés pour le développement et la livraison itératifs afin que des modifications puissent être apportées sans perturber le système dans son ensemble.

A retenir…

- Les méthodes agiles sont des méthodes de développement incrémentielles axées sur le développement rapide, les mises à jour fréquentes du logiciel, la réduction des frais généraux de processus et la production de code de haute qualité. Ils impliquent le client directement dans le processus de développement.
- La décision d'utiliser une approche de développement agile ou basée sur un plan doit dépendre du type de logiciel développé, des capacités de l'équipe de développement et de la culture de l'entreprise développant le système.


thank you
danke
謝謝
ngiyabonga
рахмат
teşekkür ederim
gracias
merci
obrigado
dziękuję
bedankt
go raibh maith agat
sukriya
kop khun krap
terima kasih
grazie
arigatô
dakujem
mochchakkeram
dank je
tapadh leat
спасибо
Баярлалаа
faafetai lava
kiitos
dankie
mauruuru
hvala
dhanyavad
nanni
nandri
köszönöm
enkosi
bayarlalaa
grâçie
vinaka
спасибі
blagodaram
mersi
barka
kia ora
welalin
tack
spas
misaotra
matondo
paldies
grazzi
mahalo
хвала
asante
manana
obrigada
murakoze
tenki
djiere dieuf
tau
дякую
mamnun
chnorakaloutioun
gralias ago
gràcies
sulpáy
akun
dankon
aciù
sobodi
dekuji
sagolun
mèsi
didi madloba
kam sah hamnida
তোমাকে ধন্যবাদ
najis tuke
rahmat
ありがとう
tanemirt
rahmet
diolch
dhanyavadagalu
shukriya
мерси
merce
takk
taiku
trugarez
ευχαριστώ
xièxie
감사합니다