Chapitre II : La Modélisation des Processus en génie Logiciel.

I. Introduction sur le cycle de Développement du Logiciel

A l'instar de tout produit, un logiciel a un cycle de vie ; il naît, vit et meurt ; autrement dit, il est créé, distribué sur un marché ou bien mis en exploitation et disparaît du fait d'obsolescence. La phase de maintenance évolutive a dans ce cas pour objectif de retarder la phase de disparition.

Le cycle de développement des logiciels s'insère dans le cycle de vie global du produit, et on parle généralement de **cycle de vie des logiciels**.

II. Généralités sur la Notion de Processus de Développement logiciel

A) Définitions de Quelques Concepts

1. Processus

Un processus est un ensemble structuré d'acteurs, d'activités pour chaque acteur (code, planification…), d'artefacts pour chaque activité (exécutable, planning), de Workflows (un workflow = un ensemble d'activités).

Un processus peut lui-même englober des sous-processus.

N.B. : Il n'existe pas de processus idéal, les entreprises adaptent les processus existants à leurs besoins.

2. Processus de développement logiciel

Un processus de développement logiciel est un ensemble (structuré) d'activités qui conduisent à la production d'un logiciel.

3. Modèle de Développement

Un Processus de Logiciel étant un ensemble structuré d'activités nécessaires pour développer un logiciel, Un Modèle de développement de logiciel est une représentation abstraite d'un processus.
Les principaux modèles adaptables à tous sont :

- Spécification : on définit ce que le système devra faire,
- Conception et implémentation : on définit l'organisation du système et on l'implémente,
- Validation : on vérifie que le système fait bien ce que veut le client,
- Evolution : on modifie le système en réponse aux changements des besoins du client.

Remarque

Un processus de développement décrit une ***méthode*** *qui permet de construire, déployer et éventuellement maintenir un logiciel. Un processus de développement définit une* ***séquence d'étapes****, partiellement ordonnées, qui permettent d'obtenir un système logiciel ou faire évoluer un système existant.*

4. Projet de Développement de Logiciel

Ensemble d'activités organisées permettant de créer un logiciel ou un service lié avec une qualité définie dans le cadre d'un budget fixé et sur un délai fixé.

La maîtrise du processus de développement logiciel passe par un **découpage** en activités

- ✓ ***Séquentielles*** (verticales) : Spécifications, Conception,
- ✓ ***Permanentes*** (horizontales) : Gestion de projet, Gestion des configurations.

Chaque activité sera validée avant de passer à la suivante.

Chaque projet est découpé en éléments maîtrisables, en modules aussi indépendants que possible, ce qui facilitera le codage, l'intégration et la maintenance.

B) Les Activités du développement logiciel

Les activités des processus de développement logiciels se regroupent en 5 grandes catégories :

a) La spécification du logiciel définit ses fonctionnalités et leurs contraintes.
b) La conception, qui concerne la mise en œuvre des spécifications techniques du logiciel,
c) L'implémentation qui concerne la réalisation du logiciel, en conformité avec sa spécification.
d) La validation s'assure effectivement du respect de la spécification par le logiciel produit.
e) L'évolution adapte le logiciel aux besoins futurs des utilisateurs.

Il est difficile d'appliquer un processus comme une unique séquence des 5 activités précédentes, du fait du principe d'incrémentalité.

En général, un logiciel complet est le fruit de plusieurs itérations, chaque itération contenant les 5 activités de : spécification, conception, implémentation, validation et évolution.

Il existe différents modèles de processus qui organisent de façons différentes ces activités, entre eux : le modèle en *cascade*, le Modèle en V, le modèle de *développement incrémental* et le modèle de *développement par composants (par Prototypes)*.

C) Les étapes d'un processus de développement logiciel

Un processus regroupe un ensemble d'étapes (dites activités, phases) :

- ✓ ***Séquentielles*** :
  - → Définition des besoins,
  - → Analyse / Conception,
  - → Implémentation,
  - → Validation et tests,
  - → Déploiement et maintenance,
- ✓ ***Permanentes (horizontales)*** :
  - → Produits intermédiaires,
  - → Plan d'assurance qualité et documentation,
  - → Gestion de projet, des ressources.

On peut de ce fait dire que le ***processus de développement logiciel*** décrit les étapes nécessaires pour développer des fonctions logicielles en garantissant ***maintenabilité*** et ***traçabilité***.

Afin de conserver un niveau de documentation minimum et utilisable, ce plan documentaire définit les documents qui doivent être remplis par les intervenants d'un projet.

Ce processus est inspiré des **normes ISO/IEC 15288 et ISO/IEC 12207** dont le champ d'application est réduit au « cycle de développement des logiciels » et permet d'assurer le cycle de vie du logiciel depuis sa création, jusqu'à son retrait, en passant par sa distribution sur un marché.

D) La Description du processus de développement de logiciel

La description des processus, concerne surtout celle de ses activités au sein telles que :

- ✓ spécifier un modèle de données,
- ✓ concevoir une interface et,
- ✓ l'ordonnancement de ces activités.

La description du processus peut aussi inclure :

- ✓ Les produits, qui sont les résultats des sorties d'une activité d'un processus,
- ✓ Les rôles, qui reflètent les responsabilités des personnes impliquées dans le processus,
- ✓ Les pré- et post-conditions, qui sont des conditions imposées avant et après l'activité d'un processus.

III. Généralités sur le Cycle de vie d'un logiciel

A) Définitions

Le « **cycle de vie d'un logiciel** » (en anglais *software lifecycle*), désigne toutes les étapes du développement d'un logiciel, de sa conception à sa disparition.

L'objectif d'un tel découpage est de permettre de définir des jalons intermédiaires permettant la **validation** du développement logiciel, c'est-à-dire la conformité du logiciel avec les besoins exprimés, et la **vérification** du processus de développement, c'est-à-dire l'adéquation des méthodes mises en œuvre.

B) Les Principales Activités du Cycle d'un Logiciel

Le cycle de vie du logiciel comprend généralement au minimum les activités suivantes :

- ***a)*** **Définition des objectifs**, consistant à définir la finalité du projet et son inscription dans une stratégie globale (Schéma Directeur).
- ***b)*** **Analyse des besoins et faisabilité,** c'est-à-dire l'expression, le recueil et la formalisation des besoins du demandeur (le client) et de l'ensemble des contraintes.
- ***c)*** **Conception générale**. Il s'agit de l'élaboration des spécifications de l'architecture générale du logiciel.
- ***d)*** **Conception détaillée**, consistant à définir précisément chaque sous-ensemble du logiciel.
- ***e)*** **Codage** (Implémentation ou programmation), soit la traduction dans un langage de programmation des fonctionnalités définies lors de phases de conception.
- ***f)*** **Tests unitaires**, permettant de vérifier que chaque sous-ensemble du logiciel est implémenté conformément aux spécifications.
- ***g)*** **Intégration**, dont l'objectif est de s'assurer de l'interfaçage des différents éléments (modules) du logiciel. Elle fait l'objet de *tests* d'intégrations consignées dans un document.
- ***h)*** **Qualification** (ou *recette*), c'est-à-dire la vérification de la conformité du logiciel aux spécifications initiales.
- ***i)*** **Documentation**, visant à produire les informations nécessaires pour l'utilisation du logiciel et pour des développements ultérieurs.
- ***j)*** **Mise en production**, C'est le déploiement sur site du logiciel.
- ***k)*** **Maintenance**, comprenant toutes les actions correctives (maintenance corrective) et évolutives (maintenance évolutive) sur le logiciel.

C) Les Documents applicables dans le Processus de Développement Logiciel

Phase de développement	Document de sortie	Responsable
Evaluation	Cahier des Charges	Client
Spécification	Spécifications fonctionnelles	Responsable logiciel
Conception	Spécifications de conception	Développeur
Ecriture du code	Fichiers sources, Code généré	Développeur
Tests unitaires	Fiche de tests unitaires	Développeur
Tests d'intégration	Fiche de tests d'intégration Fiches d'anomalie Fiche de livraison	Testeur Responsable Logiciel
Tests fonctionnels	Fiche de tests fonctionnels Fiches d'anomalie	Testeur
Validation	Rapport de validation Questionnaire interactif client	Client

D) Les Modèles de Cycle de Vie Logiciel

Comme pour toutes les fabrications, il est important de détenir un procédé de fabrication du logiciel bien défini et explicitement décrit et documenté. En GL, il s'agit d'un type de fabrication un peu particulier : en un seul exemplaire, car la production en série est triviale (recopie).

Les ***modèles de cycle de vie du logiciel*** décrivent de manière abstraite les différentes manières d'organiser la production d'un logiciel.

Les ***étapes***, leur ***ordonnancement***, et parfois les ***critères de passage*** d'une étape à une autre, sont explicités (*critères de terminaison d'une étape - revue de documents -, critères de choix de l'étape suivante, critères de démarrage d'une étape).*

- Il faut souligner la différence entre ***étapes*** (découpage temporel), et ***activités*** (découpage selon la nature du travail).
- Il y a des activités qui se déroulent ***dans plusieurs étapes*** (ex : **la spécification, la validation et la vérification**), voire ***dans toutes les étapes*** (ex : la **documentation**).

Rappelons aussi la différence entre *vérification* et *validation* (**B. Boehm, 1976**) :

- ***Verification***: *« are we building the product right ? »* (« Construisons-nous le produit correctement ? » - correction interne du logiciel, concerne les développeurs),
- ***Validation*** : *« are we building the right product ?»* (« construisons-nous le bon produit ? » - adaptation du produit vis-à-vis des besoins des utilisateurs).

E) Descriptions des Modèles linéaires, séquentiels

1. Le Modèle en Cascade

Le ***modèle en cascade*** découle des méthodes classiques d'ingénierie, Il s'adapte de ce fait dans un contexte où le logiciel fait partie d'un système complexe englobant. La production de documents entre chaque phase améliore le suivi du projet.

Ses caractéristiques principales sont :

- Chaque phase doit se terminer pour commencer la suivante.
- Des documents sont produits pour concrétiser la réalisation de chaque phase.

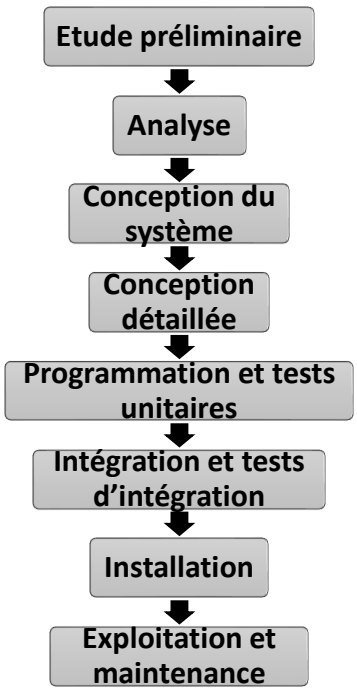
- Lorsqu'une erreur a été commise dans une phase et qu'elle est détectée dans une phase suivante, il faut faire remonter cette information dans la phase incriminée et recommencer le processus à partir de celle-ci. On doit alors reproduire de nouveaux documents.
- Ce modèle de processus impose une importante réflexion sur les choix faits en amont car le coût de la correction d'une erreur est important.

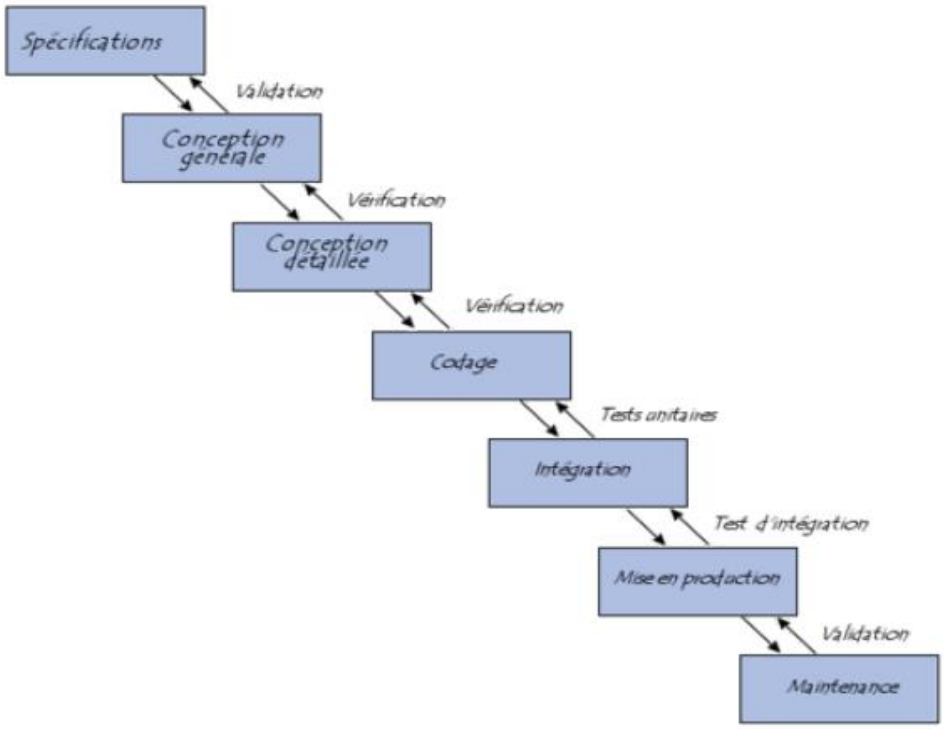
Avantages :

→ aisé à comprendre et à mettre en œuvre,
→ forte structuration : définition puis réalisation,
→ la documentation guide les étapes,
→ Découpage des taches simple et intuitive,
→ Responsabilité humaine facile à associer,
→ Production de la documentation obligatoire,
→ Maintenance facile.

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,
→ Le modèle en cascade rend coûteux le développement itératif puisque la rédaction des documents de validation de chaque phase demande beaucoup de travail.
→ Ce modèle est inadapté au développement de systèmes dont la spécification est difficile à formuler *a priori*.
→ Difficulté d'avoir tous les besoins du client.
→ Risques d'erreurs élevés.





Description des Etapes

Les Etapes	descriptions
Etude préliminaire ou étude de faisabilité ou planification	• définition globale du problème, • différentes stratégies possibles avec avantages/inconvénients, • ressources, coûts, délais. • On produit un rapport d'analyse préliminaire qui fournit le schéma directeur.
Analyse des exigences ou analyse préalable	• qualités fonctionnelles attendues en termes des services offerts, • qualités non fonctionnelles attendues : efficacité, sûreté, sécurité, utilisation, portabilité, etc. • qualités attendues du procédé de développement (ex : procédures de contrôle qualité). • On élabore un cahier des charges plus un plan qualité. Le cahier des charges contient les besoins du client.
Analyse du système	• modélisation du domaine, • modélisation de l'existant (éventuellement), • définition d'un modèle conceptuel (ou spécification conceptuelle), • plan de validation. • On élabore un dossier d'analyse plus un plan de validation.
Conception	• proposition de solution au problème spécifié dans l'analyse • organisation de l'application en modules et interface des modules (architecture du logiciel), • description détaillée des modules avec les algorithmes essentiels (modèle logique) • structuration des données. On élabore un dossier de conception plus un plan de test global et par module.
Programmation et tests unitaires	• traduction dans un langage de programmation, • tests avec les jeux d'essais par module selon le plan de test. On élabore un dossier de programmation et codes sources des modules.
Intégration et tests de qualification	• composition progressive des modules, • tests des regroupements de modules, • test détaillé du système complet selon le plan de test global. On élabore des traces de tests et de conclusions de tests.
Installation	Mise en fonctionnement opérationnel chez les utilisateurs. Parfois restreint dans un premier temps à des utilisateurs sélectionnés.
Maintenance	• maintenance corrective (ou curative), • maintenance adaptative, • maintenance perfective.

2. Le Modèle en V

Le modèle en V est actuellement le cycle de vie le plus connu et le plus utilisé. Il s'agit d'un modèle en cascade dans lequel le développement des tests et des logiciels sont effectués de manière synchrone.
La représentation en V tient compte de la réalité, dans ce cas, le processus de développement n'est pas réduit à un enchaînement de tâches séquentielles. Elle montre que:

- C'est en phase de spécification que l'on se préoccupe des procédures de qualification.
- C'est en phase de conception globale que l'on se préoccupe des procédures d'intégration.
- C'est en phase de conception détaillée que l'on prépare les tests unitaires.

Le modèle de cycle de vie en V permet de commencer plus tôt:

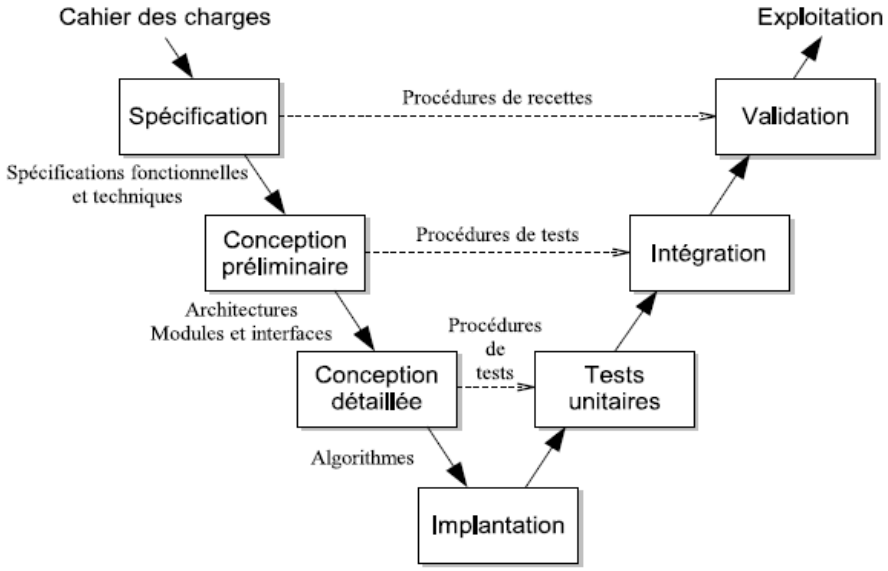
- Le Plan de tests de qualification.
- Le Plan d'évaluation des performances.

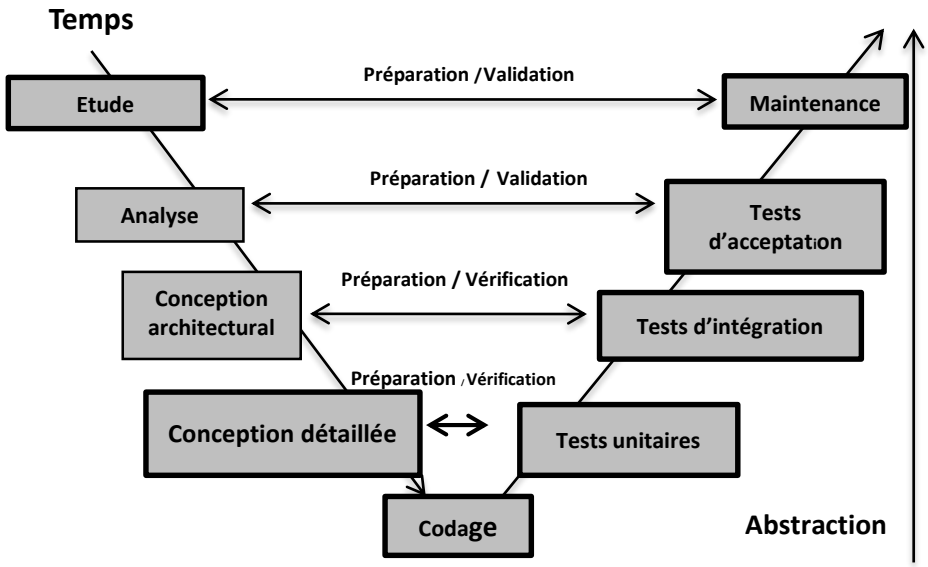
Avantages

- Plus réactif que le modèle en cascade,
- Forte structuration des étapes de test,
- Force l'identification de blocs fonctionnels.

Inconvénients

- Hypothèse stricte de séparation entre implantation et spécification,
- Logiciel délivré seulement à la fin du projet,
- Ce modèle souffre toujours du problème de la vérification trop tardive du bon fonctionnement du système.





3. Le Modèle en Spirale

(***spiral model***) est un modèle de cycle de développement logiciel qui reprend les différentes étapes du cycle en V. Par l'implémentation de versions successives, le cycle recommence en proposant un produit de plus en plus complet. Le cycle en spirale met plus l'accent sur la gestion des risques que le cycle en V. On distingue quatre phases dans le déroulement du cycle en spirale :

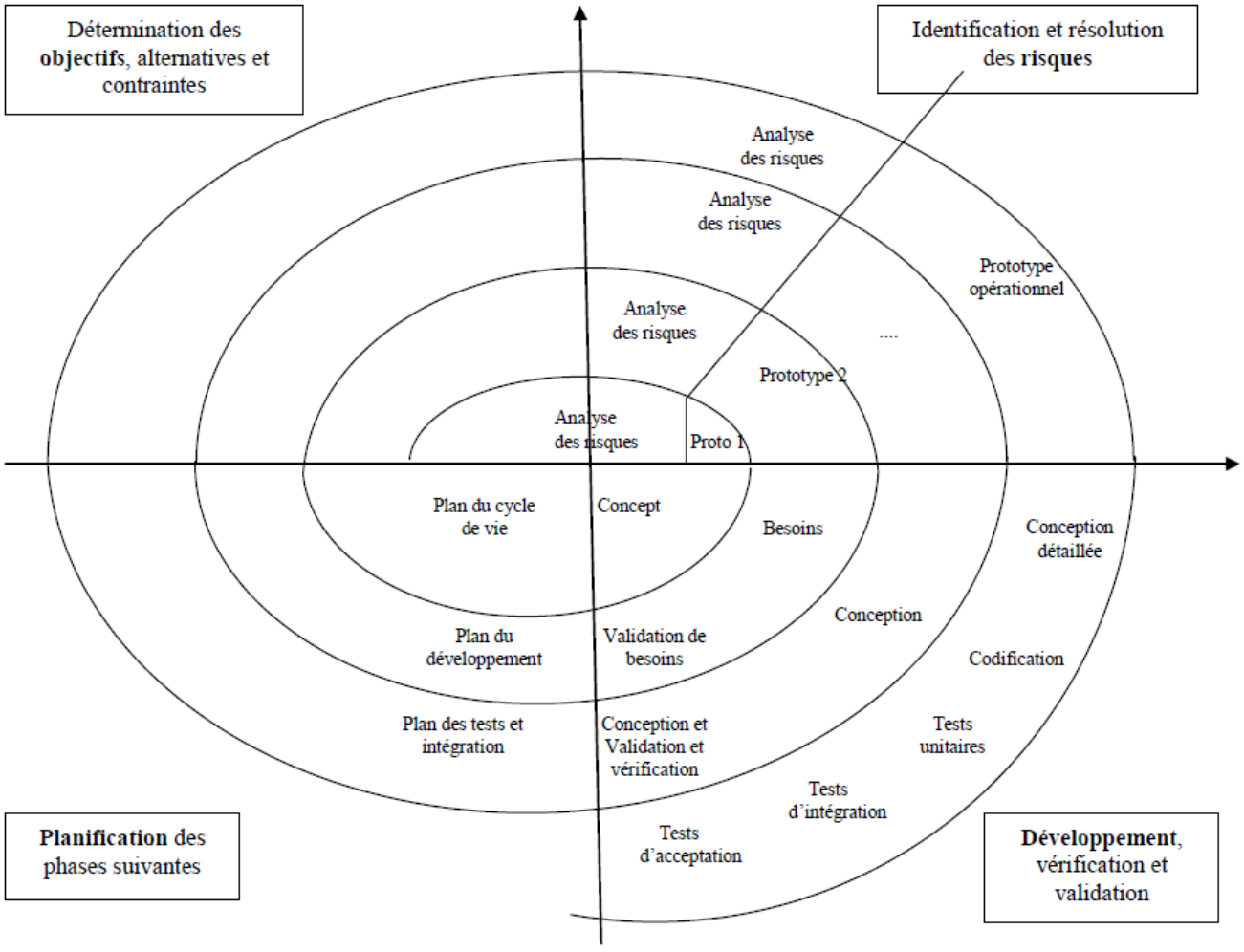
- La détermination des objectifs, des alternatives, et des contraintes.
- L'analyse des risques, l'évaluation des alternatives.
- Le développement et la vérification de la solution retenue.
- La revue des résultats et la vérification du cycle suivant.

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 une capacité à bien analyser les risques,
- nécessite gestionnaires compétents.



Le *modèle en spirale,* de **Boehm (1988),** met l'accent sur l'évaluation des risques.

Les risques à prendre en compte sont très divers : indisponibilité de matériel ou de certains logiciels, changement dans les exigences, dépassement de délais, dépassement de coûts, taille de projet sous-estimée, changement de technologie, apparition de concurrents, personnel qui quitte l'équipe.
La gestion du risque fait appel à différentes compétences dans le domaine du logiciel, de la gestion de produit, de Marketing.

Chaque boucle de la spirale représente une phase du processus du logiciel. La boucle la plus interne concerne la faisabilité du système, la boucle suivante concerne la conception, etc. Chaque boucle de la spirale est décomposée en quatre secteurs : détermination des objectifs, Identification des risques et leur réduction, Développement et Validation/Vérification, Planification de la boucle suivante.

Les principaux *risques et leurs remèdes*, tels que définis par **Boëhm**, sont résumés dans le tableau suivant :

Risque	Remède
Défaillance de personnel	Embauches de haut niveau, formation mutuelle, leaders, adéquation profil/fonction
Calendrier irréaliste	Estimation détaillée, développement incrémental, réutilisation, élagage des exigences
Risque financier	Analyse des coûts/bénéfices, conception tenant compte des coûts
Développement de fonctions inappropriées	Revues d'utilisateurs, manuel d'utilisation précoce...
Développement d'interfaces inappropriées	Maquettage, analyse des tâches
Volatilité des exigences	développement incrémental de la partie la plus stable d'abord, masquage d'information
Problèmes de performances	simulations, modélisations, essais et mesures, maquettage
Exigences démesurées par rapport à la technologie	analyses techniques de faisabilité, maquettage
Tâches ou composants externes défaillants	audit des sous-traitants, contrats, revues, analyse de compatibilité, essais et mesures...

4. Le Modèle Incrémental

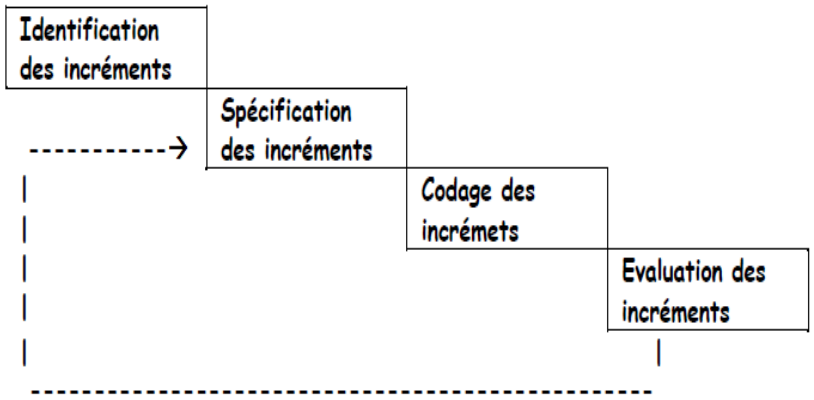
Pour ce Modèle, on évolue par incréments, en partant d'un noyau de base qui est par la suite complété progressivement. Généralement, une bonne planification et une spécification rigoureuse sont nécessaires, ainsi qu'une forte indépendance au niveau fonctionnel, et au niveau du calendrier.

Avantages

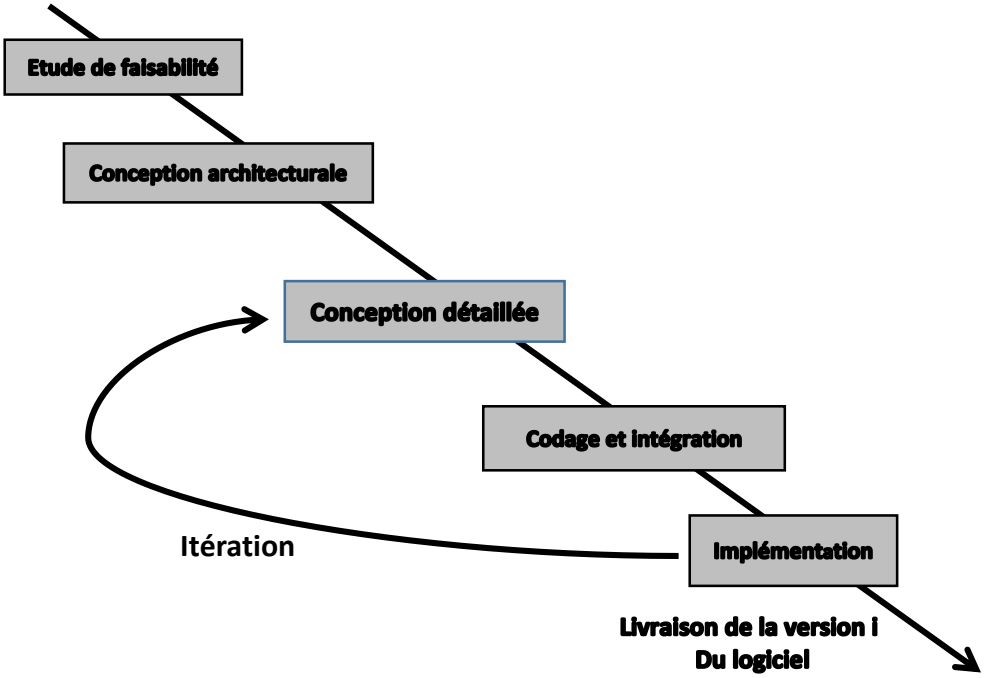
- Appliqué pour un logiciel de grande taille,
- Réduire les délais de livraison.

Inconvénients

- Maintenance exhaustive,
- Risques d'interruption de développement.



Ce cycle de vie permet de prendre en compte l'analyse de risques et de faire accepter progressivement un logiciel par les utilisateurs plutôt que de faire un changement brutal des habitudes.



5. Le Modèle de Prototypage

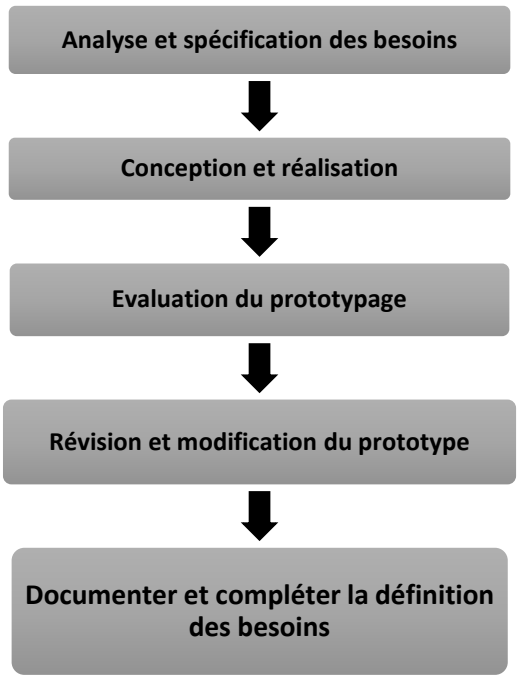
Un **prototype** est une version initiale/intermédiaire d'un système, utilisée pour démontrer des concepts et faire des essais de choix de conception.

Un prototype peut être utilisé pour :

- Le processus de spécification pour aider à classification des exigences et leur validation,
- L'étape de conception, pour explorer des choix et proposer diverses versions d'interfaces,
- Comparer des versions lors de la phase de tests.

On distingue plusieurs types de prototypes :

- Prototype exploratoire (maquette),
- Pour expliciter plus clairement l'expression des besoins (exigences),
- Horizontal : permet de tester toutes les fonctionnalités à un niveau abstrait,
- Vertical : quelques fonctions sont testées complètement,
- Prototype expérimental,
- Étude de choix de conception,
- Prototype évolutif,
- Réalisé par raffinements successifs.



Avantages du prototypage.

- Améliore la facilité d'utilisation du système.
- Meilleur adéquation avec les besoins réels.
- Améliore la qualité de la conception.
- Améliore la maintenabilité.
- Réduit les efforts de développement.
- Permet de se concentrer sur les points critiques du système.
- Un modèle évolutif.
- Simplifie l'élaboration des besoins.
- Aide à l'élaboration de l'interface Homme/Machine.

Inconvénients

- Problème de gestion de projet.
- Difficulté à mettre en œuvre des procédures de validation et de vérification.

Conclusion

Il n'y a pas de modèle idéal pour les processus de fabrication du logiciel, car tout dépend des circonstances.

- ✓ **Le modèle en cascade ou en V** est risqué pour les développements innovants car les spécifications et la conception risquent d'être inadéquates et souvent remises en cause.
- ✓ Le **modèle incrémental** est risqué car il ne donne pas beaucoup de visibilité sur le processus complet.
- ✓ Le **modèle en spirale** est un canevas plus général qui inclut l'évaluation des risques.

Souvent, un même projet peut mêler différentes approches, comme le prototypage pour les sous-systèmes à haut risque et la cascade pour les sous-systèmes bien connus et à faible risque.