ENS / Département d'Informatique **2014 / 2015**

Section : Informatique (4ème Année)

Module : Génie Logiciel

SERIE D'EXERCICE N° 1

(Introduction au Génie Logiciel – Modèles de cycle de vie)

1. **Génie Logiciel :** désigne l'ensemble des méthodes, des techniques et d'outils concourant à la production d'un logiciel de qualité avec maîtrise des coûts et délais.
   Logiciel : est l'ensemble des programmes et de documents nécessaires à leur installation, utilisation, développement et maintenance.
2. **Devoir de maison.**
3. **Les principaux facteurs de qualité d'un logiciel :**
   a. *Validité :* (correction, justesse, conformité) est la capacité que possède un produit logiciel à remplir exactement ses fonctions, définies par le cahier des charges et les spécifications. Adéquation entre :
      - Le besoin effectif de l'utilisateur,
      - Les fonctions offertes par le logiciel.

   b. *Fiabilité ou Robustesse :* la robustesse (fiabilité, sureté) est la capacité qu'offrent des systèmes logiciels à réagir de manière appropriée à la présence de conditions anormales (i.e. rien de catastrophique ne peut survenir, même en dehors des conditions d'utilisation prévues).
   c. *Facilité d'utilisation :* la facilité d'utilisation est la facilité avec laquelle des personnes présentant des formations et des compétences différentes peuvent apprendre à utiliser les produits logiciels et s'en servir pour résoudre des problèmes. Elle recouvre également la facilité d'installation, d'opération et de contrôle.
   d. *Compatibilité :* la compatibilité est la facilité avec laquelle des éléments logiciels peuvent être combinés à d'autres (un logiciel doit pouvoir interagir en synergie avec d'autres logiciels).
   e. *Efficacité :* l'efficacité est la capacité d'un système logiciel à utiliser le minimum de ressources matérielles, que ce soit le temps machine, l'espace occupé en mémoire externe et interne, ou la bande passante des moyens de communication (les logiciels doivent satisfaire aux contraintes de temps d'exécution).
   f. *Portabilité :* la portabilité est la facilité avec laquelle des produits logiciels peuvent être transférés d'un environnement logiciel ou matériel à l'autre (un même logiciel doit pouvoir fonctionner sur plusieurs machines).
   g. *Réutilisabilité :* la réutilisabilité est la capacité des éléments logiciels à servir à la construction de plusieurs applications différentes (80 % du code est du " tout venant " qu'on retrouve à peu prêt partout, 20 % du code est spécifique).
   h. *Maintenabilité :* la maintenabilité est le degré de facilité de la maintenance d'un produit logiciel.
   i. *Extensibilité :* l'extensibilité est la facilité d'adaptation des produits logiciels aux changements de spécifications.

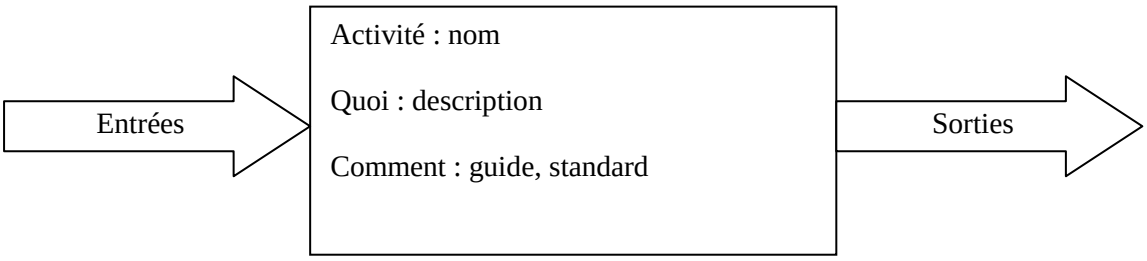
j. *Intégrité :* aptitude d'un logiciel à protéger son code et ses données contre des accès non autorisé.
k. *Ponctualité :* la ponctualité est la capacité d'un système logiciel à être livré au moment désiré par ses utilisateurs, ou avant.

Donc, un bon logiciel bien fait est un logiciel :

- correct (valide),
- fiable (robuste),
- avec un code réutilisable,
- compatible avec d'autres logiciels,
- efficace,
- portable,
- facile à utiliser,
- maintenable,
- ponctuel et
- extensible.

4. Processus de développement est l'ordre et la manière d'enchaîner les étapes d'un développement, on l'utilise pour appréhender la complexité de développement du logiciel.

5. Les activités d'un processus de développement sont décrites en termes de :
   a. Entrées de l'activité (matière première) ;
   b. Sortie de l'activité (résultat) ;
   c. Intervenant et rôles (qui ?) ;
   d. Description de l'activité (quoi – quel est le problème à traiter ?) ;
   e. Standards, guides, « best practices » à appliquer (comment ?).



6. Les étapes de développement d'un logiciel :
   a. *Faisabilité :* (POURQUOI ?)
      - Pourquoi développer le logiciel ?
      - Y a-t-il de meilleures alternatives ?
      - Comment procéder pour faire ce développement ?
      - Y a-t-il un marché pour le logiciel ?
      - Quels moyens faut-il mettre en œuvre ? A-t-on le budget, le personnel, la matériel nécessaires ?

Activité	Etude préalable
Description	Etudier la faisabilité du projet, ses contraintes techniques (coût, temps, qualité) et les alternatives possibles.
Entrées	Problème à résoudre, objectifs à atteindre.
Sorties	OUI (la décision est prise de réaliser le logiciel) ou NON (le projet est abandonné).

b. *Spécification :* (QUOI ?)

Activité	Spécifier
Description	Décrire ce que doit faire le logiciel (comportement en boite-noire). Décrire comment vérifier en boite-noire que le logiciel fait bien ce qui est exigé.
Entrées	Client qui à une idée de ce qu'il veut. Exigences, désir, besoins… concernant le système permettant de résoudre le problème.
Sorties	Cahier des charges du logiciel (ou spécification du logiciel). Des procédures de validation. Version provisoire des manuels d'utilisation et d'exploitation du logiciel.

c. *Conception :* (COMMENT ?)

Activité	Concevoir
Description	Organiser le logiciel afin qu'il puisse satisfaire les exigences de la spécification. Faire les principaux choix techniques pour satisfaire les exigences de la spécification.
Entrées	Une spécification.
Sorties	Une description des décisions de conception. Des procédures de tests qui permettent de vérifier que les décisions de conception sont correctement implémentées en code source et qu'elles contribuent à satisfaire les exigences de la spécification.

d. *Implantation :* (COMMENT ?)

Activité	Coder et tester
Description	Ecrire le code source du logiciel. Tester le comportement du code source afin de vérifier qu'il réalise les responsabilités qui lui sont allouées.
Entrées	Spécification, conception.
Sorties	Code source. Tests unitaire. Documentation.

e. *Intégration :*

Activité	Intégrer
Description	Assembler le code source du logiciel (éventuellement partiellement) et dérouler les tests d'intégration.
Entrées	Conception, code source, tests d'intégration (tests).
Sorties	Un rapport de tests d'intégration.

f. *Validation :*

Activité	Valider
Description	Construire le logiciel complet exécutable. Dérouler les tests de validation sur le logiciel complet exécutable.
Entrées	Logiciel complet exécutable à valider. Tests de validation.
Sorties	Rapport de tests de validation.

g. *Maintenance :*

Activités :

- Maintenance corrective : correction des bugs.
- Maintenance adaptative : ajuster le logiciel.
- Maintenance perfective, d'extension : augmenter / améliorer les possibilités du logiciel.

Productions :

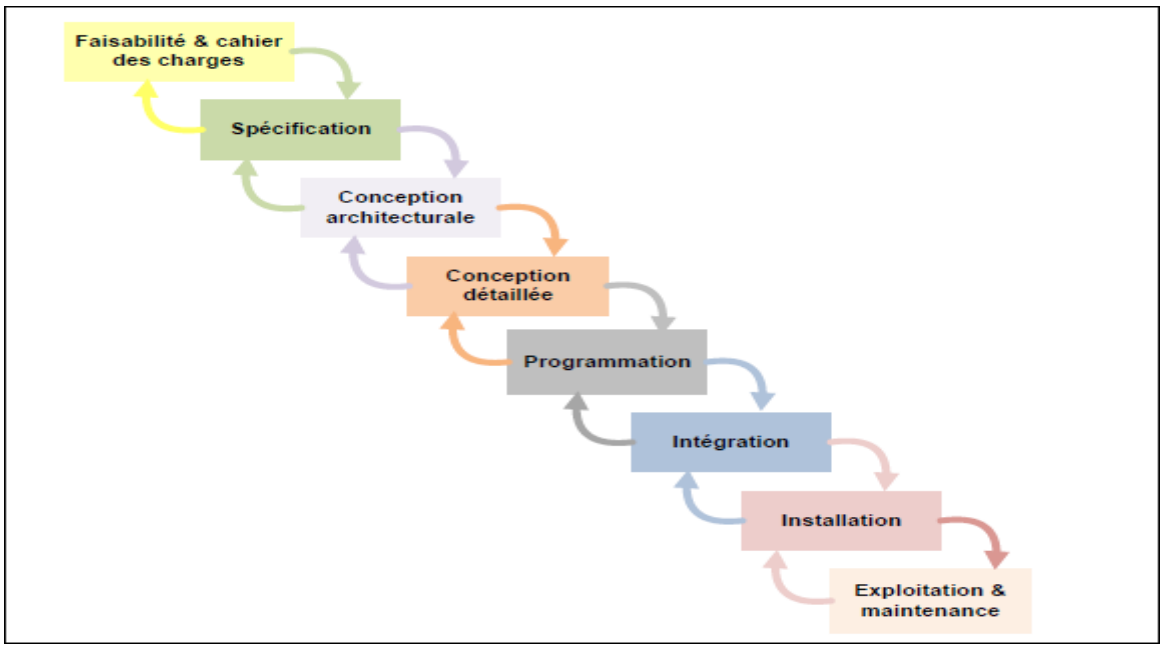
- Logiciel corrigé ;
- Mises à jour ;
- Documents corrigés.

7. Les modèles de cycle de vie d'un logiciel :

a. *Cycle de vie en CASCADE :*

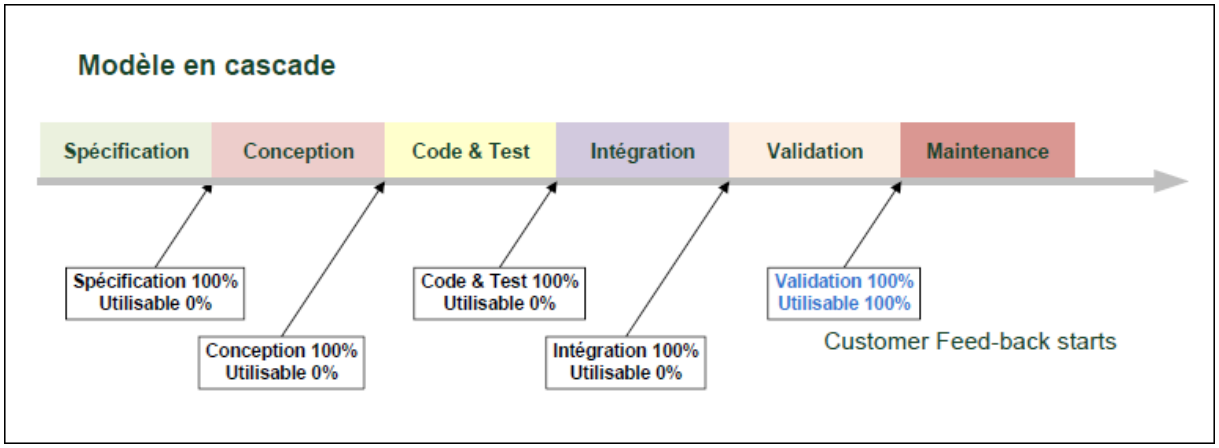
Le modèle en cascade (Waterfall model) est le plus classique des cycles de vie.

- Cycle de vie linéaire sans aucune évaluation entre le début du projet et la validation ;
- Le projet est découpé en phases successives dans le temps ;
- A chaque phase correspond une activité principale bien précise produisant un certain nombre de livrables ;
- Chaque phase ne peut remettre en cause que la phase précédente.



Inconvénients :

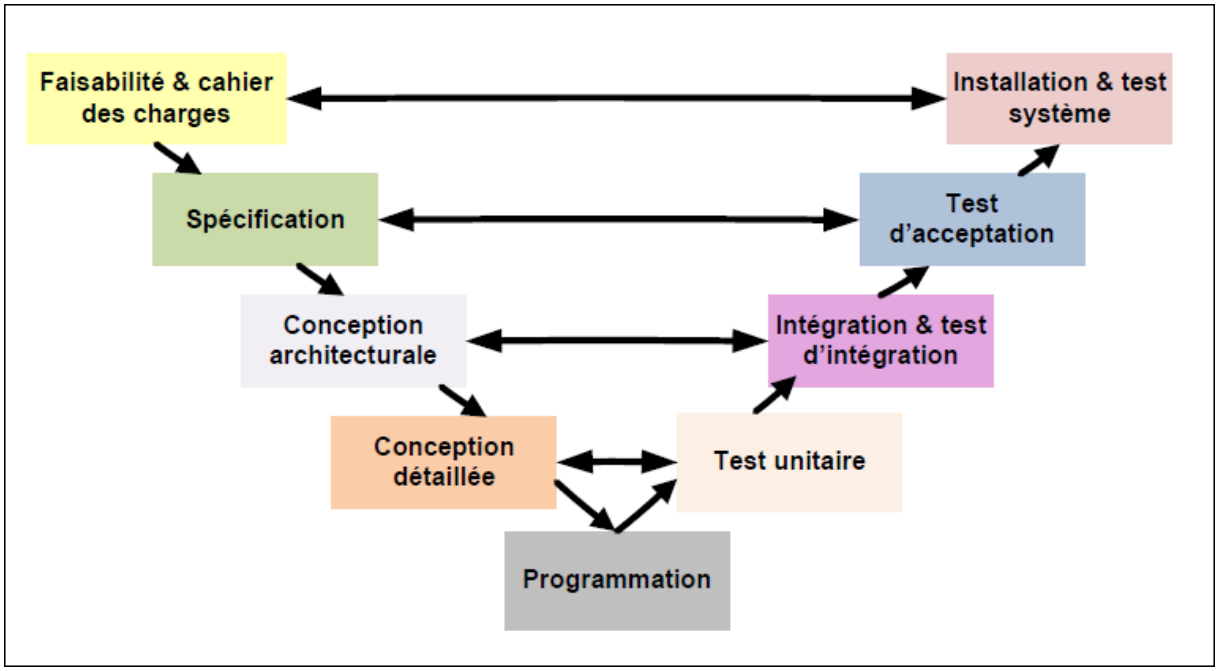
- Les vrais projets suivent rarement un développement séquentiel.
- Etablir tous les besoins au début d'un projet est difficile.
- Aucune validation intermédiaire (aucune préparation des phases de vérification) ;
- Sensibilité à l'arrivée de nouvelles exigences : refaire toutes les étapes ;



b. *Cycle de vie en V :*

A ce jour, le cycle en V reste le cycle de vie le plus utilisé. C'est un cycle de vie orienté test :

- A chaque activité créative (spécification, conception et codage) correspond une activité de vérification (validation, intégration, tests unitaire) ;
- Chaque phase en amont prépare la phase correspondante de vérification (la vérification est prise en compte au moment même de la création).



Avantage :

- La préparation des dernières phases (validation-vérification) par les premières (construction du logiciel), permet de d'éviter d'énoncer une propriété qu'il est impossible de vérifier objectivement après la réalisation.

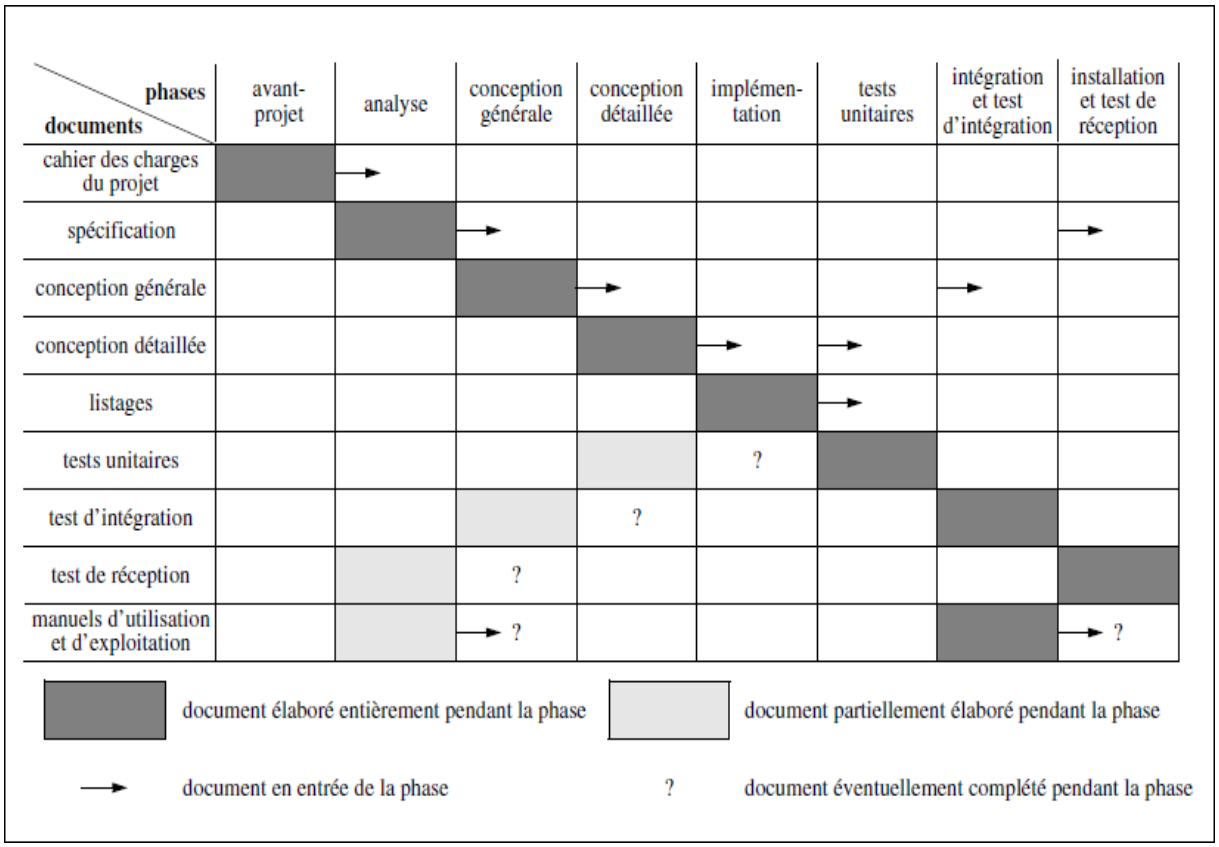
Inconvénients :

- Construit-on le bon logiciel ? le logiciel est utilisé très (trop) tard :
  - Il faut attendre longtemps pour savoir si on a construit le bon logiciel ;
  - Difficile d'impliquer les utilisateurs lorsque qu'un logiciel utilisable n'est disponible qu'à la dernière phase.
- Idéal quand les besoins sont bien connus, quand l'analyse et la conception sont claires.

8. Comparaissant :

Prototypage jetable (rapide)	Prototypage non jetable
Cette approche consiste donc à produire rapidement une maquette (prototype jetable) qui constitue une ébauche du futur système. Avec cette approche, on est capable de définir plus explicitement et de manière plus cohérente les besoins des usagers, de détecter les fonctions manquantes et identifier et améliorer les fonctions complexes, comme elle permet de démontrer la faisabilité et l'utilité de l'application. Le prototype est ensuite abandonné et le système réel est construit.	Un prototype répond à des objectifs différents : on cherche à construire un système éventuellement très incomplet, mais dans son dimensionnement réel de façon à pouvoir faire des essais en vraie grandeur. On détermine d'abord un ensemble minimum de fonctions elles-mêmes incomplètes de façon à réaliser un premier incrément du logiciel dont on se sert pour analyser le comportement du logiciel. L'utilisateur fournit en retour des informations au concepteur qui modifie le prototype. Ce processus d'évolution de prototypes continue jusqu'à ce que l'utilisateur soit satisfait du système livré.

9. Les phases intervenant dans le modèle de cycle de vie en V :



10. Les qualités requises dans un cahier de charges :

a. Bon niveau de généralité ;
b. Formulation adéquate des besoins ? Problème bien décrit ;
c. Etre précis, non ambigu malgré l'usage d'un langage naturel (≠ mathématique) ;
d. Etre complet (pas d'omission involontaire) ;
e. Etre cohérent (pas d'inférence de fonctionnalités) ;
f. Etre vérifiable : critères de validation définis. Evaluer la faisabilité des besoins ? faire éventuellement une maquette, une simulation ;
g. Etre modifiable : facilité à exprimer un changement ou ajout de besoins.

11. Devoir de maison.