Qualité de développement

CM5 : Génie logiciel

Mickaël Martin Nevot

V2.2.0

Qualité de développement

Génie logiciel

- **L'ensemble des activités de conception et de mise en œuvre des produits et des procédures tendant à rationaliser la production du logiciel et son suivi**
- **Procédures**, **méthodes**, **langages**, **ateliers**, imposés ou préconisés par les **normes** adaptées à l'environnement d'utilisation afin de favoriser la **production et la maintenance de composants logiciels de qualité**
- Depuis 1983. Né de constatations (avant les années 1980) :
  - Logiciels non fiables
  - Logiciels très difficiles à réaliser dans les délais
  - Logiciels ne satisfaisant pas le cahier des charges

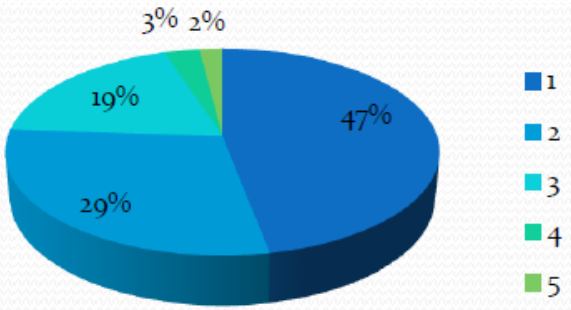
Quelques erreurs célèbres

- Perte de la 1ère sonde Mariner vers Venus suite à une erreur de programmation dans un programme Fortran
- Perte, en 1971, de 72 ballons d'expérimentation météorologique à cause d'un bogue logiciel
- Panne, en 1990, du réseau téléphonique de la côte Est des USA suite à un changement de version d'un des modules du système de gestion du réseau
- Abandon d'un projet d'informatisation de la City après 4 ans de travail : 100 M£ de perte
- Invalidation de version de Windows suite au changement de version du Windows Genuine Advantage
- Échec du vol 501 d'Ariane 5 (explosion) suite à un bogue logiciel

La crise du logiciel

- Étude du gouvernement américain réalisée en 1979 :

1.	Payés mais jamais livrés	3.2 MUSD	47%
2.	Livrés mais jamais utilisés	2.0 MUSD	29%
3.	Abandonnés ou refaits	1.3 MUSD	19%
4.	Utilisés après modifications	0.2 MUSD	3%
5.	Utilisés tels quels	0.1 MUSD	2%



La crise du logiciel

- En 1994 :
  - Logiciels livrés avec succès 20 %
  - Livrés en retard et hors budget 60 %
  - Échecs 30 %

Prise de conscience

- L'espacement des rails au USA : 4 pieds et 8.5 pouces ?

Les lois générales du temps

- **Parkinson** : le travail se dilate jusqu'à remplir la durée disponible pour son accomplissement
- **Douglas** : dossiers et documents s'entassent jusqu'à remplir l'espace disponible pour leur rangement
- **Pareto** : l'essentiel prend 20 % du temps, l'accessoire 80 %
- **Murphy** : rien n'est aussi simple qu'il n'y parait
- **Carison** : faire un travail en continue prend moins de temps qu'en plusieurs fois
- **Illich** : au delà d'un seuil de travail horaire, le temps passé n'est plus efficace
- L'**Ecclésiaste** : il y a un moment pour tout et un temps pour chaque chose

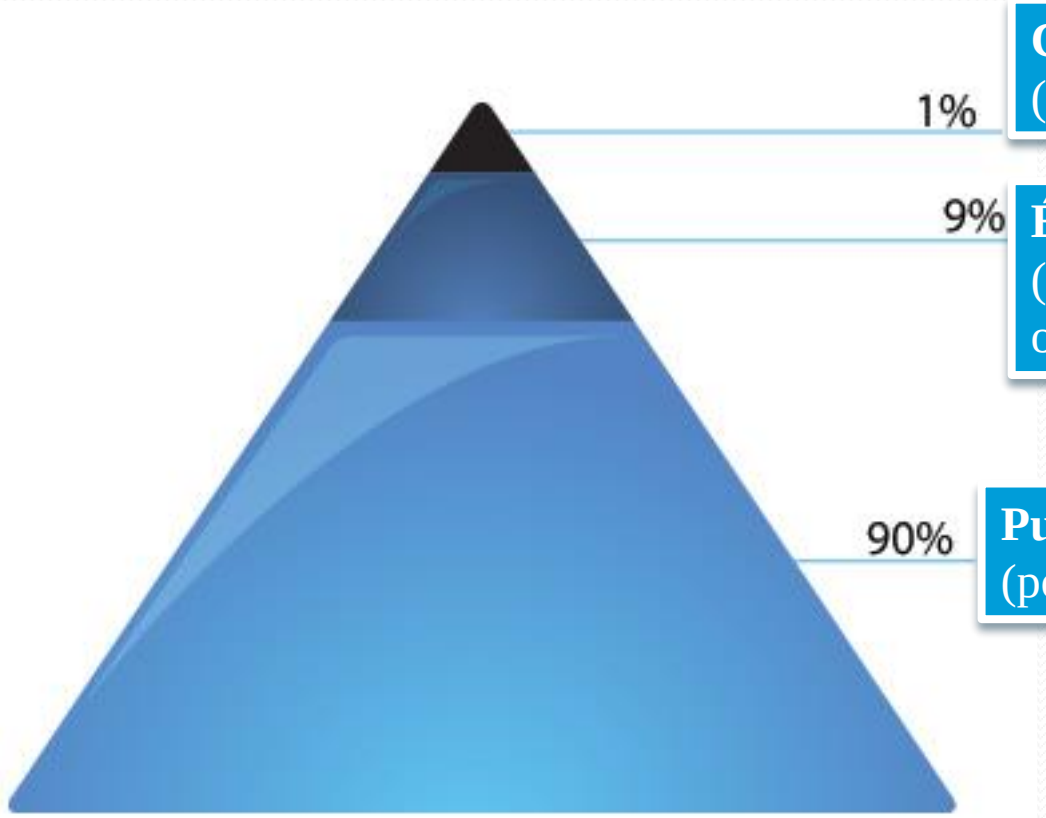
Les lois du génie logiciel

- **Amdahl** : 10 % d'instructions séquentielles peuvent multiplier jusqu'à dix fois les performances du logiciel si elles sont parallélisées
- **90 – 10** :
  - 90 % du développement prennent 10 % du temps
  - 10 % (restants) du développement prennent 90 % du temps
- **Wirth** : un logiciel perd en efficacité plus rapidement qu'un matériel en gagne
- **Brook** : ajouter de la main d'œuvre à un projet en retard le retarde encore plus

Les lois du génie logiciel

- **Peter** (principe de) : dans une hiérarchie, tout employé a tendance à s'élever à son niveau d'incompétence
- **Norvig** : toute technologie qui dépasse 50 % de taux de pénétration ne doublera jamais plus
- **Hofstadter** : cela prend toujours plus de temps que prévu même lorsque vous prenez en compte la loi de Hofstadter
- **Metcalfe** : la valeur s'accroit au carré du nombre de personnes

Règle du 1% / Règle 90–9–1



Créateurs
(population proactive)

Éditeurs
(population occasionnelle et opportuniste)

Public
(population observatrice)

Règle valable pour toute communauté virtuelle

Développement de logiciels

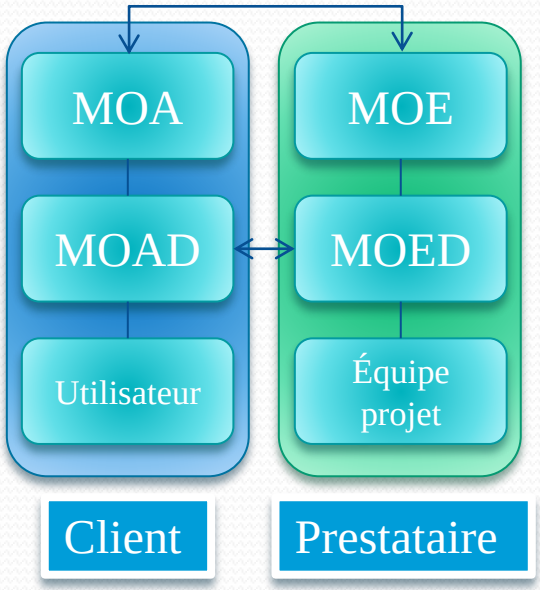
- Gestion des **exigences** (spécifications)
- **Planification** de projet
- Supervision et **suivi** de projet
- Gestion des **achats**
- Assurance **qualité**
- Gestion de la **configuration**
- Mesures et **analyses**

Le développement de logiciel comprend l'ensemble des étapes et processus qui permettent de passer de l'expression d'un besoin informatique à un logiciel fonctionnel et fiable

Équipe de production

MOA/MOE

- **MOA** : **m**aître d'**o**uvr**a**ge
  - MOAD : **m**aître d'**o**uvr**a**ge **d**élégué
  - AMOA : **a**ssistant **m**aître d'**o**uvr**a**ge
- **MOE** : **m**aître d'**œ**uvre
  - MOED : **m**aître d'**œ**uvre **d**élégué

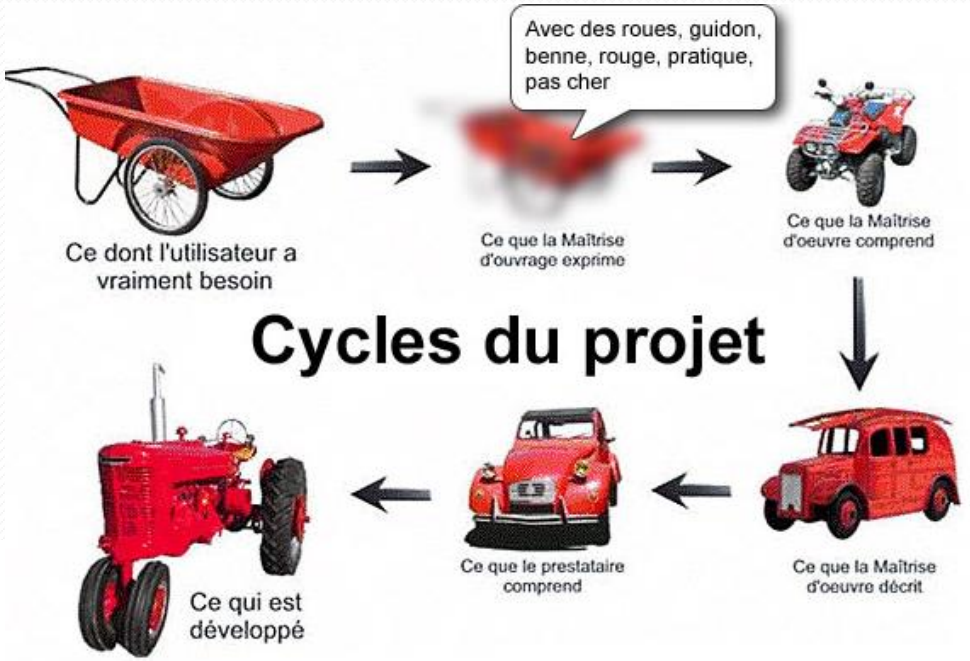


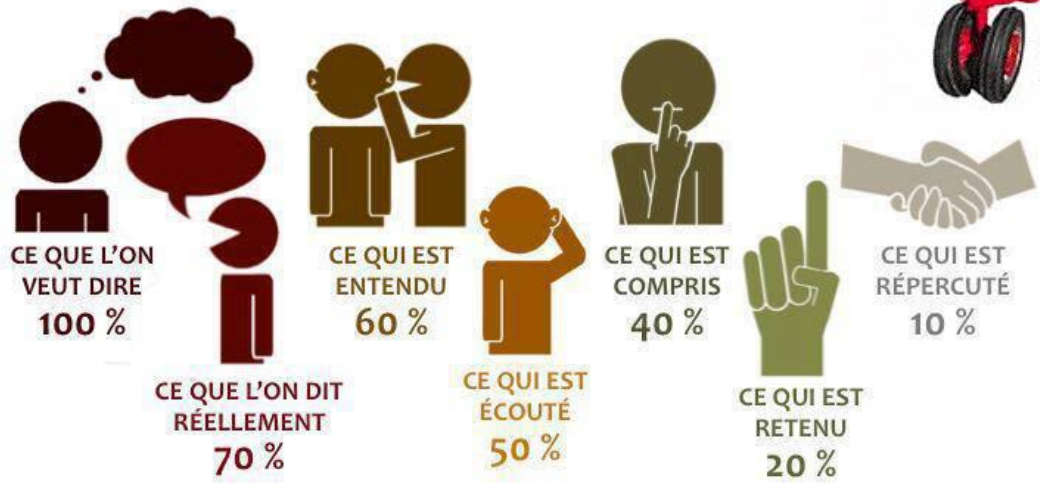
Répartition des rôles en fonction des étapes

Niveau de Détail	Rôles	Besoins et Faisabilité	Spécification	Conception Architecturale	Conception Détaillée	Codage	Test unitaire	Test d'intégration	Test de Validation	Recette
Système	MOA + AMOA	X								X
Fonctionnel	MOE + MOED		X						X	
Technique et Métier	Equipe Architecturale			X				X		
Composant	Equipe de Développement				X	X	X			

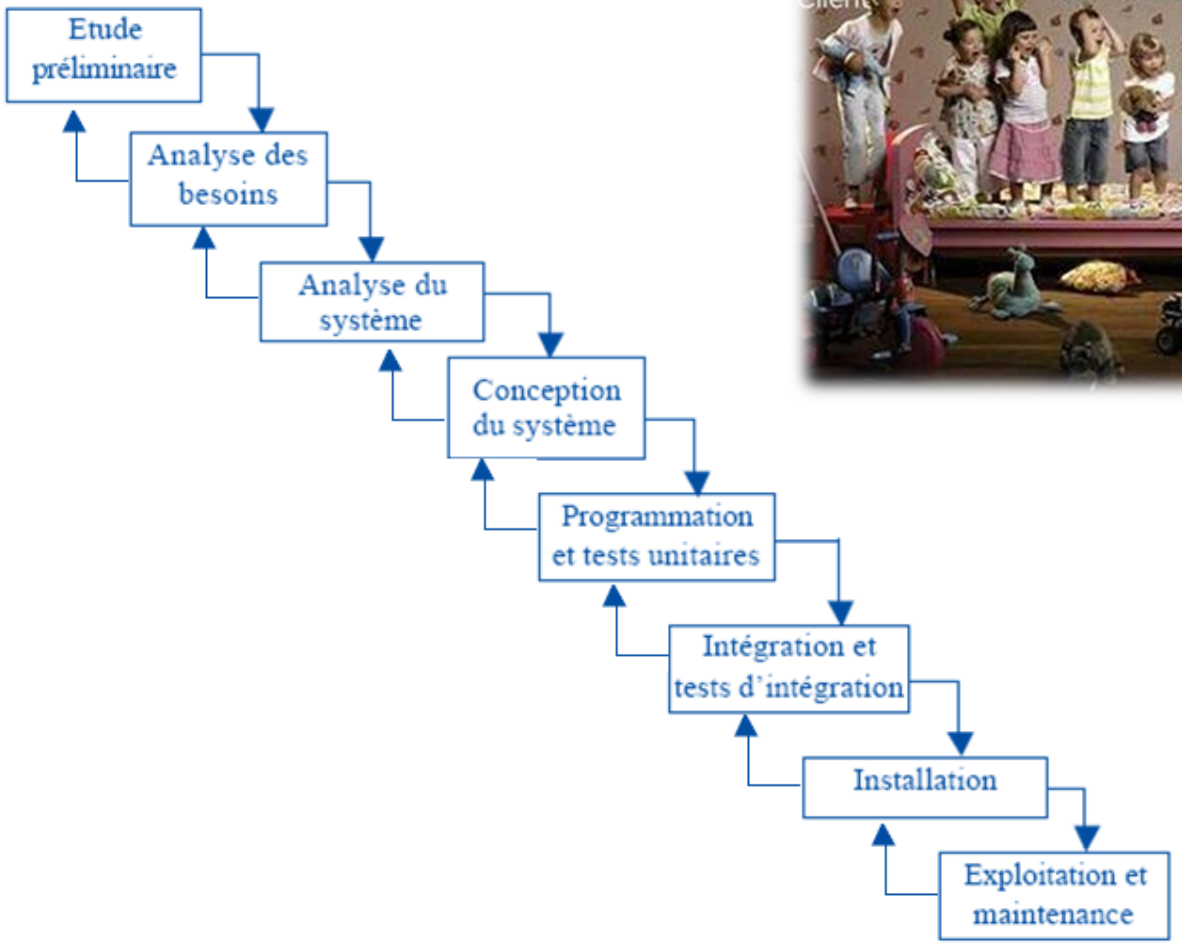
Cycles de développement

- **Modèle en cascade**
- **Cycle en V**
- Cycle en spirale
- Cycle semi-itératif
- **Cycle itératif**

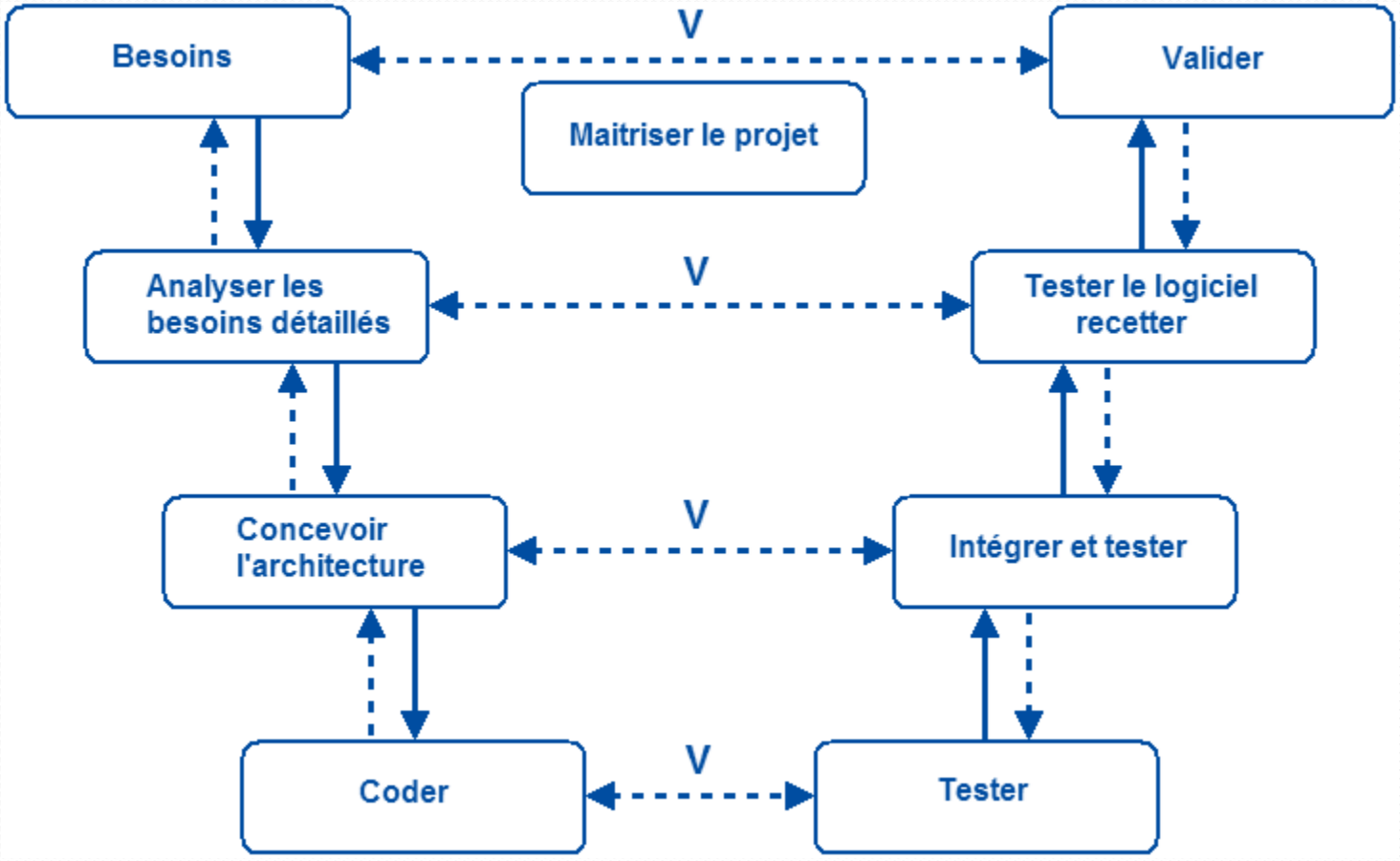




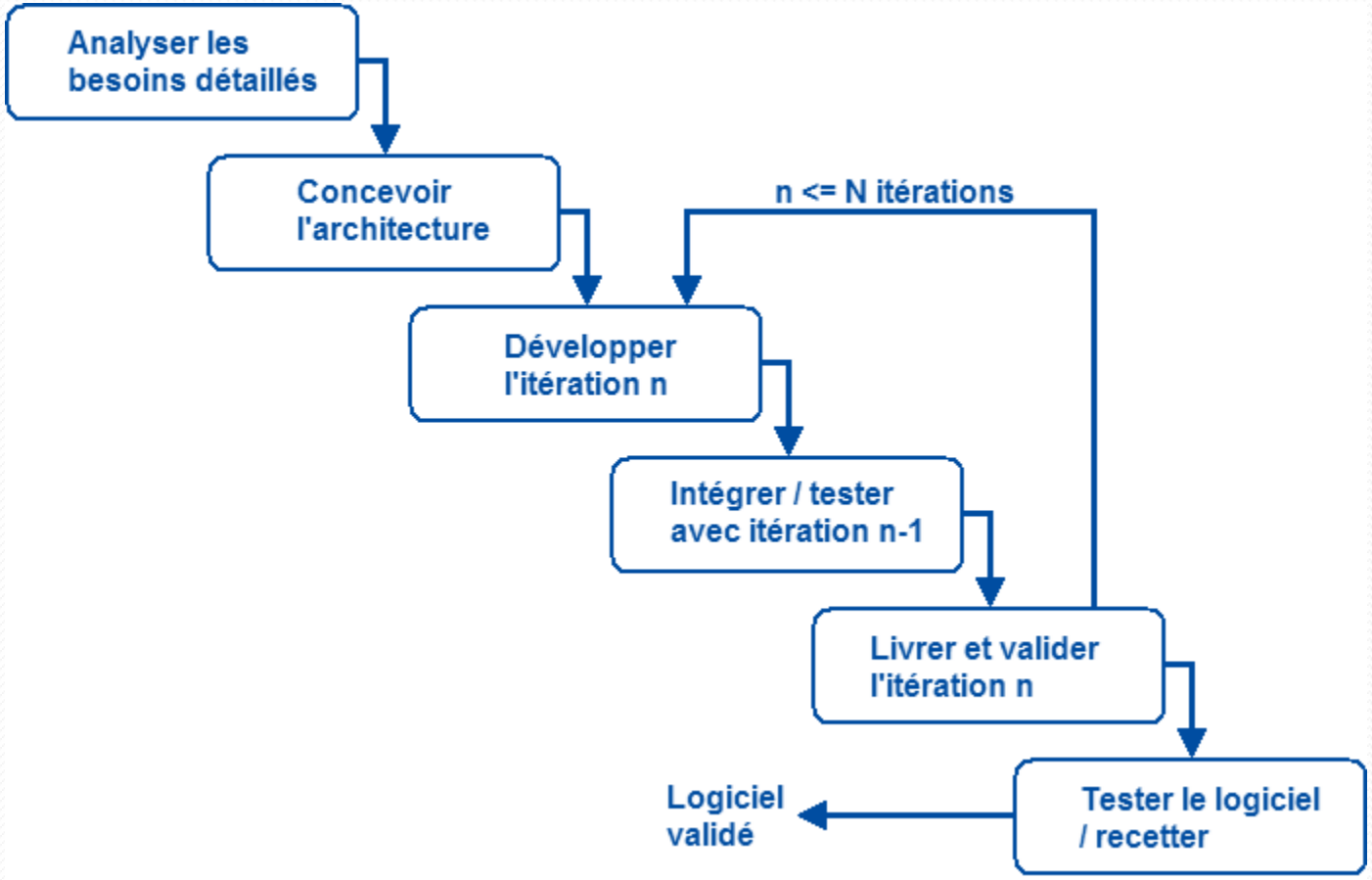
Modèle en cascade



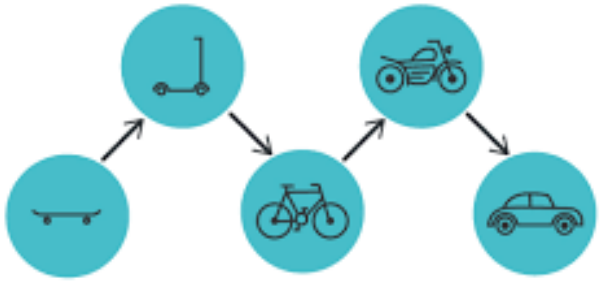
Cycle en V



Cycle itératif



Design itératif

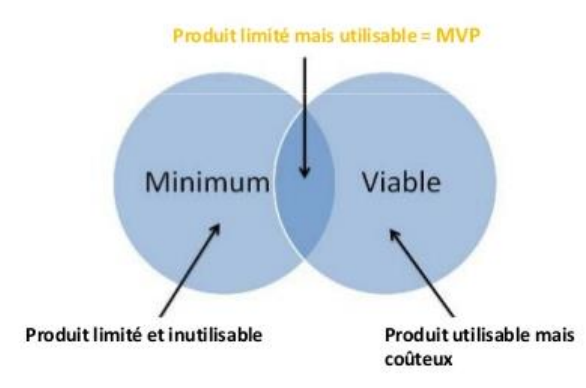


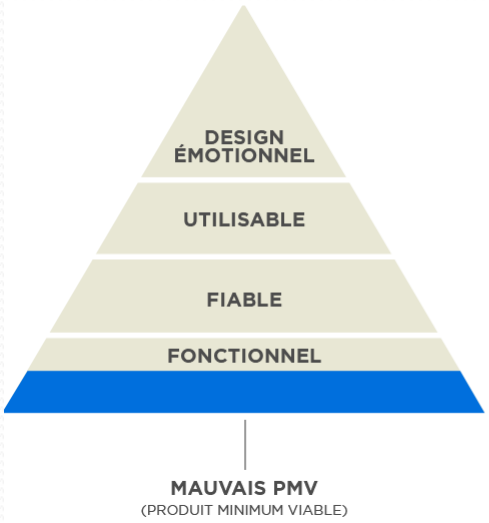
« Même les meilleurs experts en utilisabilité ne peuvent *designer* de parfaites interfaces utilisateurs en une seule tentative. »
– Jakob Nielsen

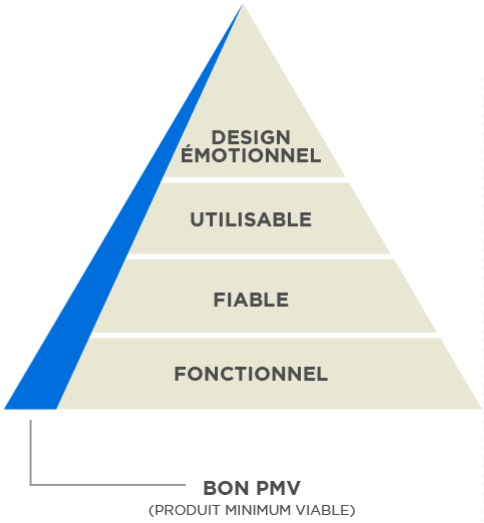
Produit minimum viable (MVP)

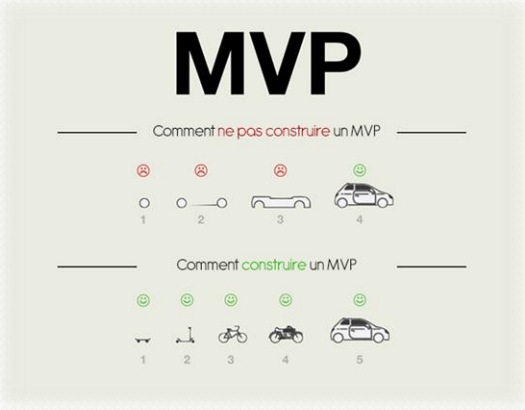
- Stratégie de développement
- Tests rapides et quantitatifs de mise sur le marché

Le produit le plus simple possible répondant à un objectif minimum pour lequel un client est prêt à payer ou à en mesurer la valeur









Étude préliminaire / analyse

- Étude préliminaire :
  - Définition **globale** du système
  - Choix de **stratégies** (ressources, coûts, délais, etc.)
  - Guidée par l'**expérience**
- Analyse :
  - Besoins :
    - **Fonctionnels** : services offerts
    - **Non fonctionnels** : efficacité, sécurité, utilisation, portabilité, etc.
  - Système :
    - **Modélisation** de l'existant et du domaine d'application

Cahier des charges

- Issu de l'**analyse** du projet
- Document de référence et **contractuel**
- Expression de besoins **précis** (sans ambiguïté) du **client** :
  - **Fonctionnels**
  - **Interfaces** (entre les logiciels)
  - **Non fonctionnels** (performances, contraintes)
- Élaboré par le **MOA** : ce qu'attend le MOA du MOE
- Permet au client d'évaluer :
  - L'étendue des travaux
  - Les coûts
  - Les délais

Cahier des charges : structure

- **Contexte** : politique et stratégie du projet
- **Objectif** : buts recherchés
- **Dictionnaire** : culture et vocabulaire communs
- **Périmètre** : équipes de production et autres ressources
- **Calendrier** : création claire dans le temps
- **Clauses juridiques** : entre les cosignataires

Conception (du système)

- Proposition de **solution** aux besoins de l'analyse
- **Architecture** logicielle (définition des **modules**)
- **Structuration** des données
- Descendante :
  - Décompositions des modules principaux en sous-modules
- Ascendante :
  - Agrégation de modules élémentaires en modules supérieurs
- Mixte

Spécification (fonctionnelle) : SF

- La spécification englobe la **conception**
- **Ensemble de documents** qui, par des textes et des diagrammes, décrit de manière **formelle** et **exhaustive** les **exigences** du produit informatique à réaliser
- **Contrat** entre client et producteur, **exprimé par le producteur** :
  - SF générale (**SFG**) : élaborée par le **MOA**
  - SF détaillée ou technique (**SFD**) : élaborée par le **MOE**
- Décrit les **caractéristiques attendues**
- Définit **l'architecture en modules**
- Définit **la technologie à utiliser**

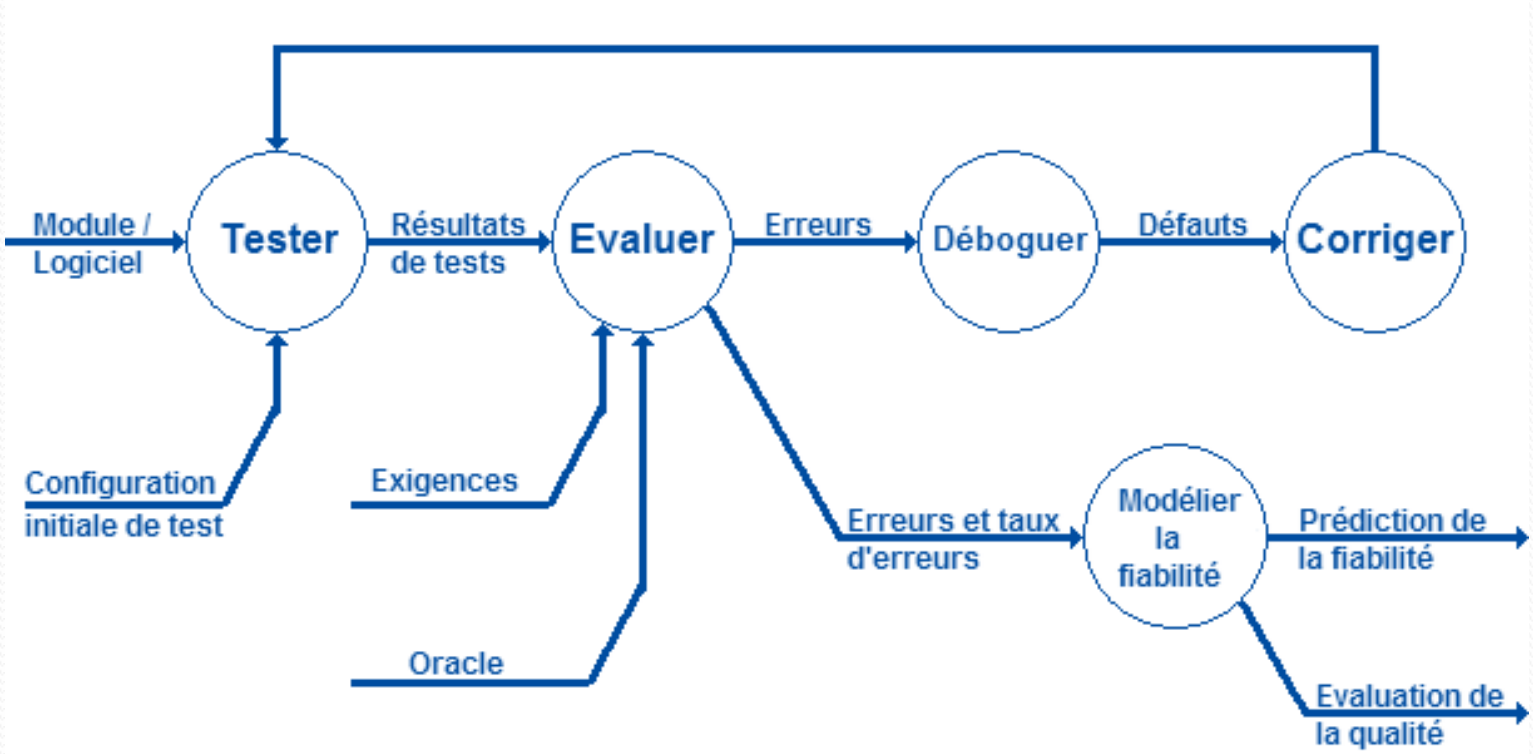
Vérification/tests

- Dynamique (expérimenter le **comportement**) :
  - Tests :
    - Prouvent la **présence d'erreur**, pas l'absence
    - Certains résultats anormaux peuvent être tolérés
  - **Jeux d'essais** :
    - Aléatoires (efficacité très variable, pas de cas limite)
    - **Fonctionnels** (boîte noire, très tôt dans le développement)
    - Structurels (boîte blanche, prise en compte de critères)
- Statique (analyser les **propriétés**, sans exécution) :
  - Techniques **formelles** (assertions)
  - Techniques **informelles** (revues, inspection, *walkthrough*)

Vocabulaire de test

- **Oracle** : résultat normal / comportement attendu
- **Défaut** : résultat anormal
- **Défaillance** : comportement différent de celui attendu
- **Erreur** : activation d'un ou plusieurs défauts
- **Tester** : vérifier la présence de défauts/défaillances
- **Évaluer** : déterminer les erreurs
- **Déboguer** : localiser l'origine des erreurs
- **Corriger** : rectifier des défauts/défaillances
- **Fiabilité** : taux d'erreurs, détermine la qualité
- **Vérification** : affirme la conformité aux exigences
- **Validation** : affirme la conformité aux besoins

Processus de test



Programmation / test unitaire

- Programmation : traduction en code source
- **Tests unitaires** :
  - Tester chaque sous-programme et structure d'un module
  - Vérifier la fidélité à la spécification fonctionnelle
  - Environnements de test : **JUnit**, etc.
- **Couverture** de code :
  - Taux de code source testé (avec les tests unitaires)

Test d'intégration (fonctionnel)

- Test d'**intégration** :
  - Permet de s'assurer que l'application se comporte correctement dans sa globalité (par rapport aux fonctionnalités exigées)
  - Permet de tester que les parties développées indépendamment fonctionnent bien ensemble de façon cohérente
- **Intégration continue** :
  - Fusion des tests unitaires et des tests d'intégration (développement de l'application entière par un seul développeur)

Installation/maintenance

- Installation :
  - Mise en **fonctionnement opérationnel**
- Maintenance :
  - **Corrective** (ou curative) : erreur par rapport au contrat
  - **Adaptative** : mise à jour
  - **Perfective** : nouvelle version

Attention : il faut savoir faire la différence !

Dette technique : quand on code au plus vite et de manière non optimale, on contracte une dette technique que l'on rembourse tout au long de la vie du projet sous forme de temps de développement de plus en plus long et de bugs de plus en plus fréquents

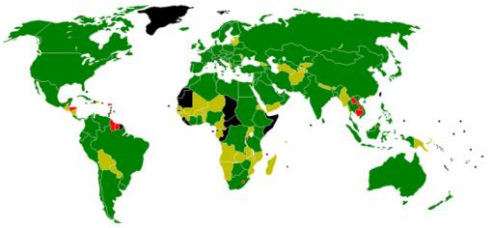
Norme logicielle

- ISO :
  - **ISO 9000** (9003) et ISO SPICE :

    entreprise qui suit un processus qualité
  - ISO 9000:2000 :

    capacité à atteindre les objectifs opérationnels visés
- IEEE :
  - *Institute of Electrical and Electronics Engineers*

ISO Organisation internationale de normalisation



IEEE

Outils, organisationnel et humain

- Outils :
  - **AGL** (atelier de génie logiciel) : plusieurs parties du cycle de développement
  - Environnements intégrés (tout le cycle de développement)
- Organisationnel et humain :
  - Organisation des équipes
  - Planification
    - Diagramme de **Gantt**
  - Estimation des coûts

« Je me fiche du nombre de femmes que vous me donnez, cela prend toujours neuf mois pour faire un bébé ! »

Les autres, et notre perception

Aller plus loin

- Différentes fonctions de maîtrise d'ouvrage
- Différentes fonctions de maîtrise d'œuvre
- Valeur acquise (méthode de la)
- Clôture du projet

Liens

- Document électronique :
  - http://tibo.lelore.free.fr
- Documents classiques :
  - Cours :
    - William Boher-Coy. *Cours d'initiation au génie logiciel.*
    - Nicolas Tassara. *Génie logiciel*.

Crédits

Auteur

Mickaël Martin Nevot
mmartin.nevot@gmail.com

Carte de visite électronique

Relecteurs

Cours en ligne sur : www.mickael-martin-nevot.com