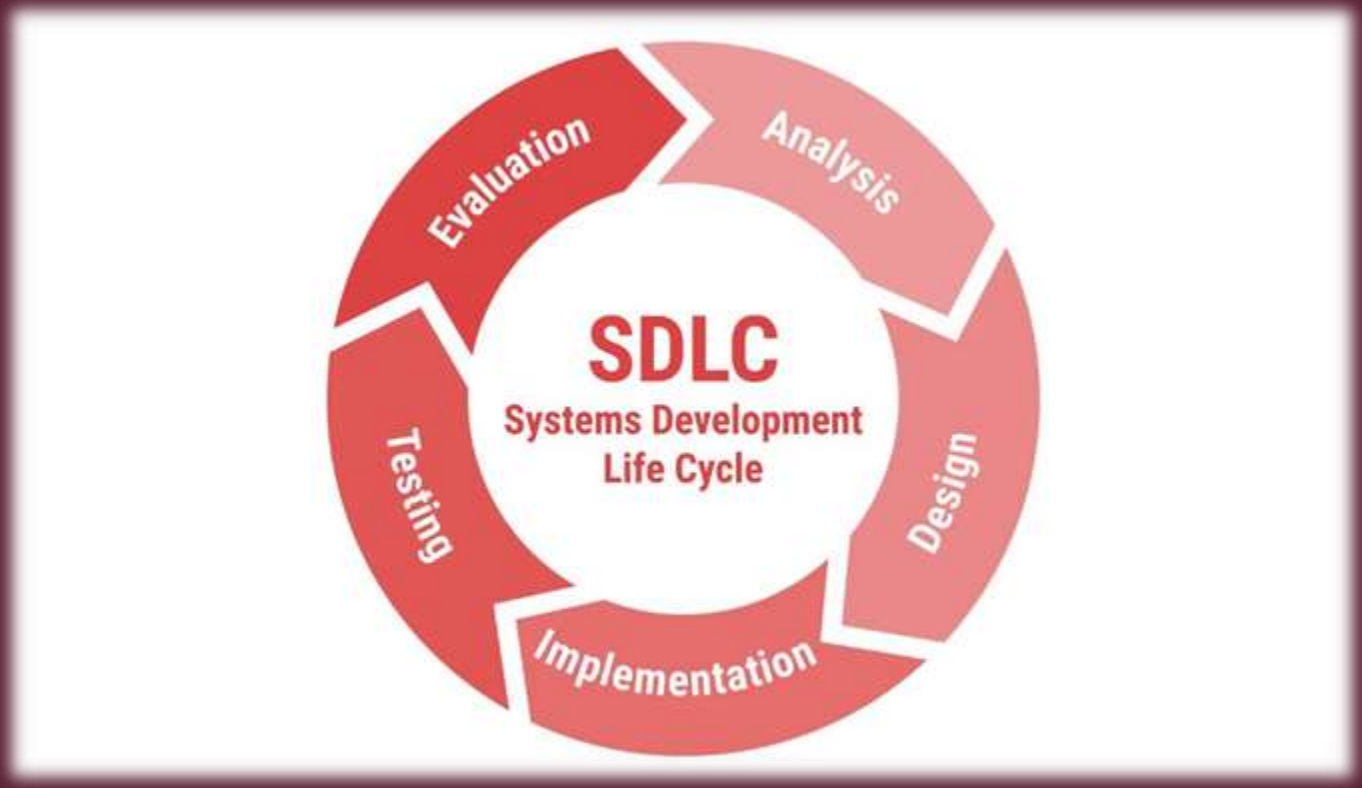
LE CYCLE DE DÉVELOPPEMENT LOGICIEL



Réalise par :

AABIDA Sara
ELABSY Aya
KARIM Hajar
MOUMEN Aya

Plan :

- Introduction
- LE MODELE EN CASCADE (WATERFALL MODEL)
- LE MODELE EN V
- LE CYCLE DE VIE EN SPIRALE
- Conclusion

Introduction :

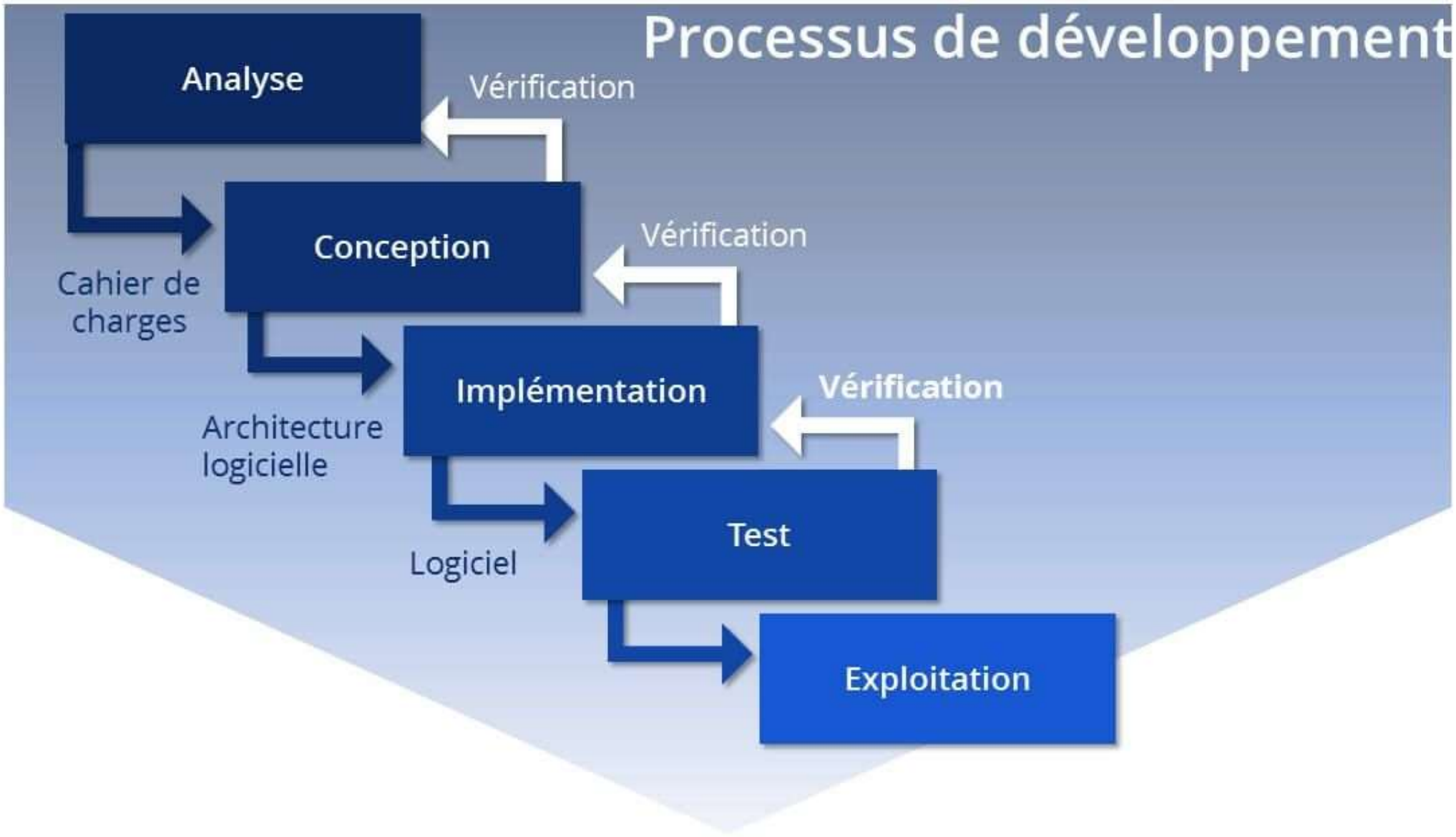
Le cycle de vie du développement logiciel (SDLC - Software Development Life Cycle) représente les étapes suivies pour concevoir, développer, tester et déployer un logiciel. Il permet d'assurer une bonne gestion du projet, de minimiser les erreurs et d'optimiser la qualité du produit final. Plusieurs modèles existent pour organiser ces étapes, parmi lesquels le modèle en cascade, le modèle en V et le cycle de vie en spirale.

LE MODELE EN CASCADE (WATERFALL MODEL):

Le modèle en cascade est une méthode de développement logiciel où chaque phase doit être complétée avant de passer à la suivante. Il suit une séquence linéaire et rigide, souvent représentée sous forme de cascade.

Les étapes du modèle en cascade :

1. **Analyse des besoins → Comprendre ce que le client veut.**
2. **Conception → Planifier l'architecture et la structure du logiciel.**
3. **Implémentation → Écrire le code et développer l'application.**
4. **Tests → Vérifier si le logiciel fonctionne correctement.**
5. **Déploiement → Mettre le logiciel en production.**
6. **Maintenance → Corriger les erreurs et améliorer le logiciel après son lancement.**


Processus de développement
Analyse
Vérification
Conception
Cahier de charges
Vérification
Implémentation
Architecture logicielle
Vérification
Test
Logiciel
Exploitation

Avantages :

- **Clarté et simplicité : La structure linéaire facilite la compréhension du processus.**
- **Planification rigoureuse : Chaque phase est bien définie, ce qui permet une planification précise.**
- **Documentation complète : Une forte documentation accompagne généralement chaque étape du projet.**

Inconvénients :

- **Rigidité : Le modèle ne s'adapte pas facilement aux changements en cours de développement.**
- **Retard dans la détection des erreurs : Les problèmes peuvent être découverts tardivement, ce qui augmente le coût de leur correction.**
- **Moins adapté aux projets innovants : Pour des projets aux exigences susceptibles d'évoluer, des modèles plus agiles sont souvent préférables.**

Exemple du modèle en cascade Imaginons qu'une entreprise souhaite développer un site e-commerce pour vendre des vêtements en ligne.

Voici comment le modèle en cascade serait appliqué :

1.Analyse des besoins :

- **Définir les fonctionnalités du site : catalogue de produits, panier d'achat, paiement en ligne, gestion des utilisateurs, etc.**
- **Identifier les technologies à utiliser (PHP, MySQL, HTML/CSS, etc.).**

2.Conception :

- **Élaborer l'architecture du site (base de données, structure des pages, design)**
- **Définir les interfaces utilisateur et créer des maquettes.**

3.Implémentation :

- **Création des pages web, intégration du back-end et de la base de données.**
- **Développement du site selon les spécifications définies.**

4.Tests :

- **Vérifier le bon fonctionnement des fonctionnalités (ajout au panier, paiement sécurisé, connexion utilisateur).**
- **Identifier et corriger les bugs éventuels.**

5.Déploiement :

- **Mise en ligne du site pour que les clients puissent l'utiliser.**

6.Maintenance :

- **Corrections des bugs après le lancement.**
- **Ajout de nouvelles fonctionnalités si nécessaire.Dans ce modèle, chaque phase doit être terminée avant de passer à la suivante. Si un problème est détecté tardivement (par exemple, une erreur dans l'architecture du site), il peut être coûteux et long à corriger.**

Le modèle en V :

Le modèle en V est une amélioration du modèle en cascade qui met un fort accent sur la vérification et la validation à chaque étape du développement.

Il suit une structure en forme de “V”, où chaque phase de développement a une phase de test correspondante.

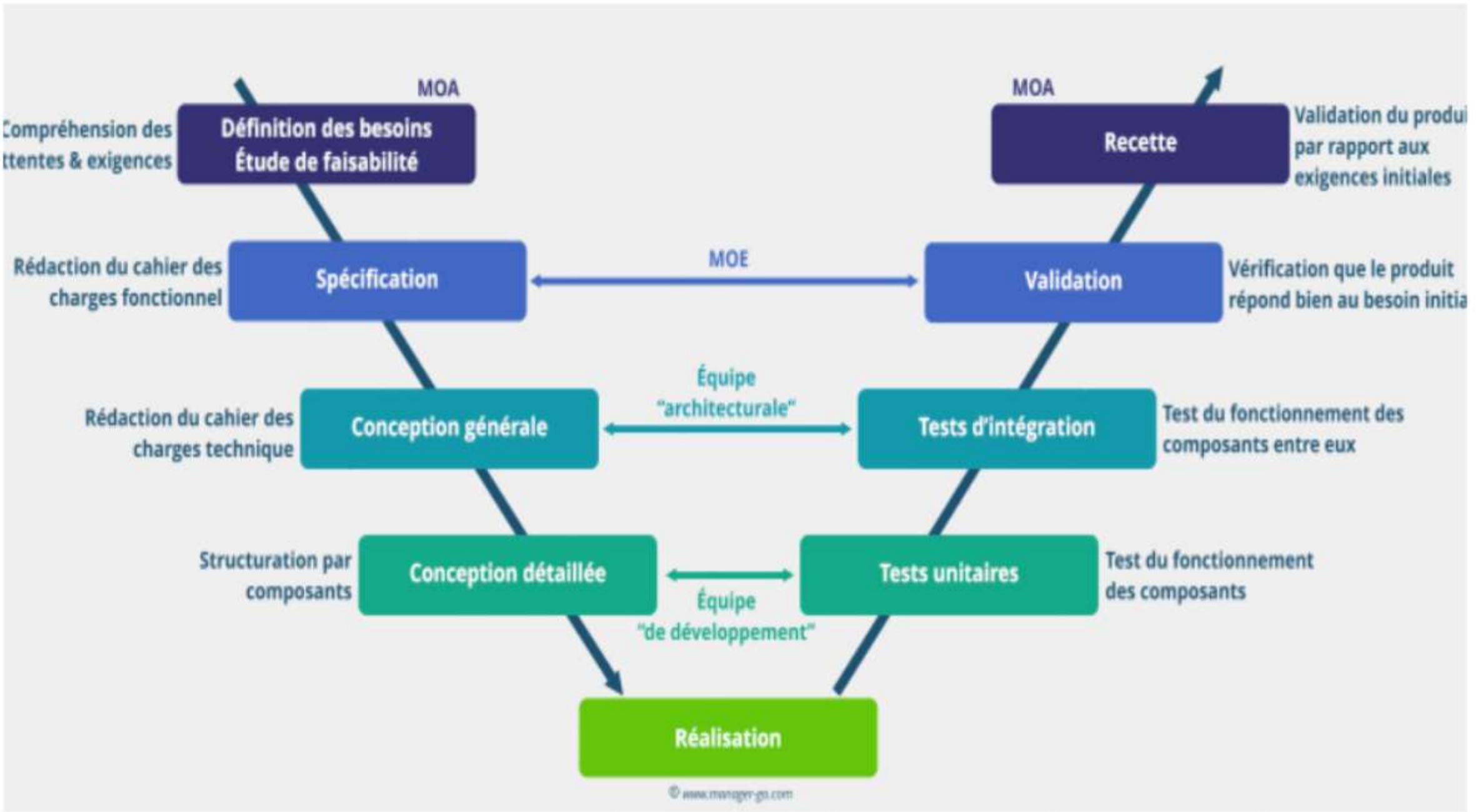
Les étapes du modèle en V :

Phase de développement :

1.Analyse des besoins → Comprendre les attentes du client.
2.Spécifications fonctionnelles → Définir ce que le système doit faire.
3.Conception architecturale → Planifier la structure globale du système.
4.Conception détaillée → Détailler chaque composant du système.
5.Implémentation → Développement du code.

Phase de validation et de test :

1. Tests unitaires → Vérifier chaque composant séparément.
2. Tests d'intégration → Vérifier l'interaction entre les composants.
3. Tests de validation → Vérifier si le système répond aux besoins fonctionnels.
4. Tests d'acceptation → Vérifier avec le client que le produit final correspond à ses attentes.


MOA
Définition des besoins
Étude de faisabilité
Compréhension des
ttentes & exigences
MOA
Recette
Validation du produi
par rapport aux
exigences initiales
Rédaction du cahier des
charges fonctionnel
Spécification
MOE
Validation
Vérification que le produit
répond bien au besoin initia
Équipe
"architecturale"
Rédaction du cahier des
charges technique
Conception générale
Tests d'intégration
Test du fonctionnement des
composants entre eux
Structuration par
composants
Conception détaillée
Équipe
"de développement"
Tests unitaires
Test du fonctionnement
des composants
Réalisation
© www.manager-go.com

Avantages du modèle en V :

✓ **Détection des erreurs plus rapide grâce aux tests à chaque étape.**

✓ **Meilleure qualité du logiciel à la fin du projet.**

✓ **Idéal pour les projets où les exigences sont bien définies.**

Inconvénients du modèle en V :

✗ **Rigidité : difficile de modifier le projet en cours de développement.**

✗ **Long temps de développement, car les tests sont nombreux.**

✗ **Moins adapté aux projets évolutifs ou incertains (préférer des méthodes agiles).**

Exemple : Développement d'une application bancaire en ligne

1. Phase de définition des exigences

- **Exigences fonctionnelles :**

•L'utilisateur doit pouvoir se connecter à son compte bancaire en ligne.

•L'utilisateur peut effectuer des virements entre comptes.

•L'utilisateur peut consulter son solde bancaire.

•Exigences non fonctionnelles :

•L'application doit être sécurisée (authentification forte, chiffrement des données).

•L'application doit être performante et capable de gérer un grand nombre d'utilisateurs simultanés.

2. Phase de conception

•Conception de l'architecture :

•Définition des composants du système (front-end, back-end, base de données).

•Conception des interfaces utilisateur (UI/UX).

•Conception des modules :

•Conception détaillée des fonctionnalités telles que la gestion des comptes, la consultation du solde, les virements, etc.

3. Phase de codage

•Développement des différents modules de l'application en suivant les spécifications de conception.

•Implémentation de la logique métier (par exemple, gestion des transactions bancaires).

4. Phase de tests (Validation et vérification)

• Vérification (à gauche du "V") :

•Tests unitaires :

Chaque fonction (par exemple, la fonction de connexion) est testée indépendamment pour s'assurer qu'elle fonctionne comme prévu.

•Tests d'intégration :

S'assurer que les différents modules (par exemple, la gestion des comptes et des transactions) fonctionnent ensemble de manière cohérente.

•Validation (à droite du "V") :

•Tests système :

Tester l'application dans son ensemble pour s'assurer qu'elle répond bien aux exigences définies (par exemple, vérification de l'authentification, des virements, de la consultation du solde).

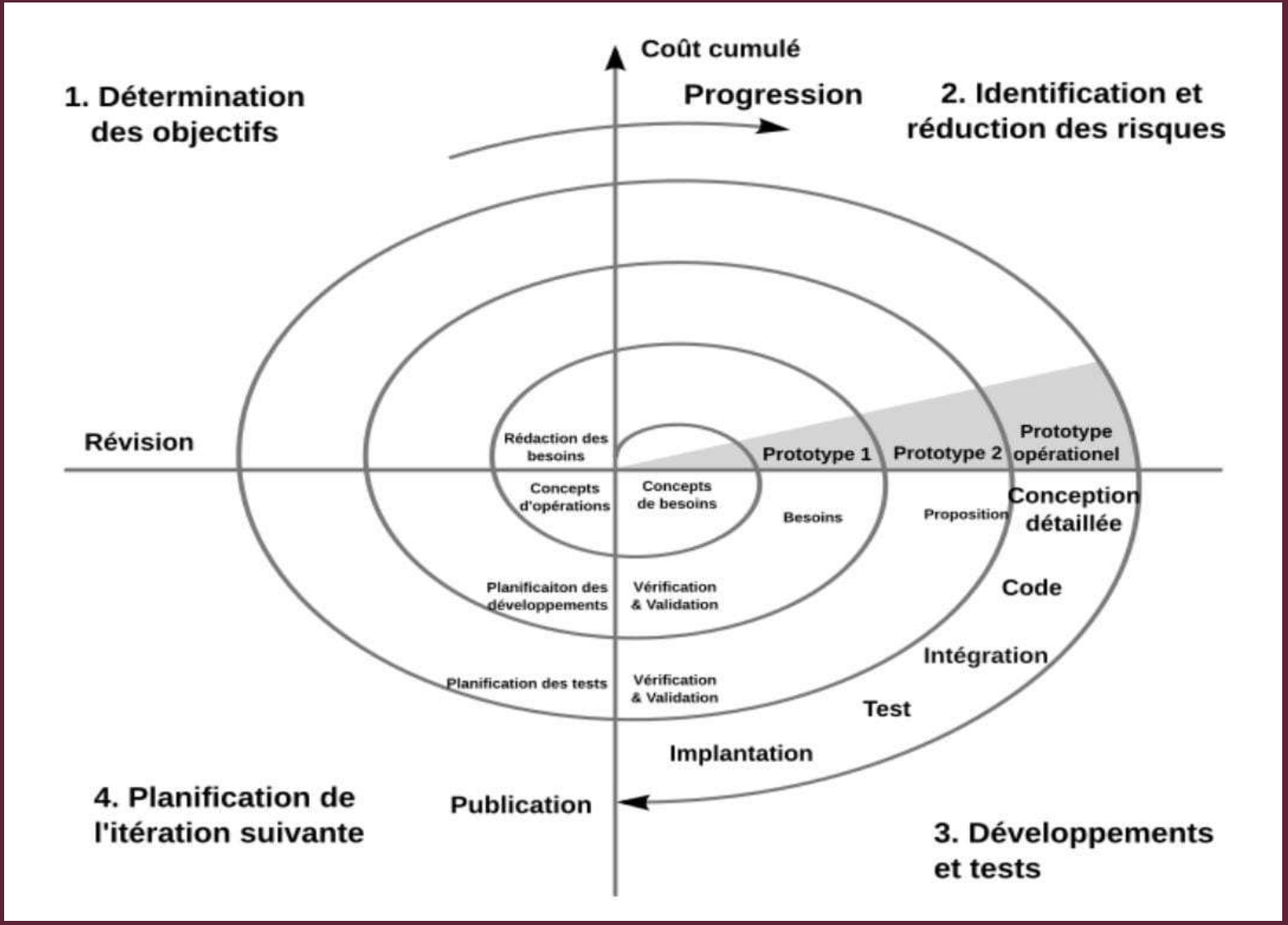
•Tests d'acceptation utilisateur :

S'assurer que l'application répond aux attentes de l'utilisateur final, en vérifiant que les fonctionnalités sont intuitives et faciles à utiliser.

Modèle en spirale :

le modèle en spirale est un **mode opératoire de développement logiciel** inventé par Barry W. Boehm en 1986. Il part du principe que le développement d'applications représente un **cycle itératif**, qui doit être répété jusqu’à ce que le but fixé soit atteint. Par une analyse régulière des risques et des contrôles réguliers du produit intermédiaire, le modèle en spirale diminue considérablement le risque d’échec lors des projets logiciels de grande taille.

Représentation graphique du modèle en spirale selon Boehm :



Phases du modèle en spirale :

Définir les objectifs : Identifier les objectifs, contraintes, et alternatives pour chaque phase.

Identifier et résoudre les risques : Analyser les risques, et trouver des solutions pour minimiser leur impact.

Développement et tests : Concevoir, développer et tester le prototype ou le produit.

Planification de la prochaine itération : Réviser les résultats et planifier l'itération suivante pour continuer à améliorer le produit.

les avantages et les inconvénients du modèle en spirale :

Avantages:

Modèle générique flexible.

Implication précoce du client et des utilisateurs possible.

Contrôle périodique dû aux risques.

Coordination parfaite entre exigences techniques et conception.

Maîtrise maximale des coûts, ressources et qualité du projet logiciel.

Adapté aux environnements techniques novateurs.

Inconvénients:

Effort de gestion important.
Les décisions régulières peuvent retarder le processus de développement.
À cause de la subdivision du processus de développement, des erreurs et incohérences de conception peuvent facilement se retrouver dans le produit final.
Connaissance en analyse et gestion des risques indispensable, mais souvent manquante.
Inadapté aux petits projets aux risques raisonnables.

Exemple : Développement d'une application mobile de gestion de tâches

1. Première itération (Phase de planification et de spécification):

Objectif : Comprendre les besoins de base des utilisateurs et définir les fonctionnalités principales.
Activités :Identification des besoins des utilisateurs : fonctionnalités essentielles telles que la création, la suppression et la gestion des tâches.
Estimation des coûts et des ressources.
Livrables : Un cahier des charges initial.

2. Deuxième itération (Phase de conception préliminaire et prototypage)

Objectif : Créer un prototype fonctionnel de base pour visualiser l'interface utilisateur.
Activités :Conception de l'interface utilisateur.
Création d'un prototype fonctionnel simple pour recueillir les retours d'utilisateurs.
Livrables : Prototype de l'interface utilisateur.

3. Troisième itération (Phase de développement et d'évaluation)

Objectif : Implémenter les fonctionnalités principales identifiées dans les itérations précédentes.
Activités :Développement de la fonctionnalité de gestion des tâches.Tests internes pour s'assurer du bon fonctionnement.
Livrables : Version alpha de l'application avec gestion des tâches.

4. Quatrième itération (Phase d'évaluation des risques et d'amélioration)

Objectif : Évaluer les risques techniques et les performances du logiciel.

Activités :Identification des risques potentiels liés à la sécurité et aux performances.

Amélioration des fonctionnalités basées sur les retours des utilisateurs.

Livrables : Version bêta avec des correctifs et des optimisations.

5. Cinquième itération (Phase de livraison)

Objectif : Finaliser l'application pour la livraison aux utilisateurs finaux.

Activités :Finalisation du développement avec des tests approfondis.

Livraison de la version finale aux utilisateurs.

Livrables : Application prête à être déployée.

Conclusion:

Le choix du modèle de développement logiciel dépend de la nature du projet :

- **Le modèle en cascade est adapté aux projets simples et bien définis.**
- **Le modèle en V est utile lorsque la qualité et les tests sont essentiels.**
- **Le modèle en spirale convient aux projets complexes nécessitant une évaluation continue des risques.**

"Merci pour votre attention ! "