

Génie Logiciel : TD 1

Exercice 1 :

Une entreprise développe des applications sur mesure pour ses clients. Son processus de développement s'organise comme suit :

1. Une première phase consiste à recueillir les besoins du client sous forme de spécifications détaillées. Ces spécifications doivent être validées avant de passer à la phase suivante.
2. Une fois les spécifications validées, une équipe produit un prototype rapide permettant au client de visualiser une première version simplifiée de l'application. Ce prototype est soumis à plusieurs itérations de feedback avec le client pour ajuster les fonctionnalités et le design.
3. Une fois le prototype validé, le développement complet de l'application commence, incluant la programmation, les tests unitaires et l'intégration.
4. L'application est ensuite soumise à des tests d'acceptation par le client avant la livraison finale.

Question : Quel est le modèle de cycle de vie utilisé par l'entreprise ? Justifiez votre réponse.

- **Modèle évolutif**

- **Pourquoi ce modèle ?**

Ce modèle est utilisé car il commence par une validation des besoins pour garantir que tout est clair dès le départ. Un prototype est ensuite créé pour tester et ajuster les fonctionnalités selon les retours. Une fois validé, le développement complet peut se faire de manière structurée. Ce processus progressif assure un produit final adapté aux besoins du client.

Exercice 2 :

Une entreprise est spécialisée dans le développement d'applications pour la gestion de l'énergie. Ses projets sont souvent complexes et impliquent des technologies innovantes. Le processus de développement suivi par l'entreprise est le suivant :

1. Phase initiale : Une analyse approfondie est menée pour identifier les besoins, les contraintes et les risques majeurs. À ce stade, des ressources importantes sont investies dans la planification et l'évaluation des risques potentiels.
2. Prototypage : Un prototype est développé pour tester les concepts clés, valider les hypothèses et réduire les incertitudes liées aux technologies utilisées.
3. Cycle itératif : Chaque itération du développement inclut une planification détaillée, une analyse des risques mise à jour, un développement partiel, des tests et une évaluation des résultats obtenus.
4. Décision stratégique : À la fin de chaque cycle, une décision est prise quant à la poursuite, la modification ou l'abandon du projet, en fonction des résultats obtenus et des risques restants.
5. Livraison finale : Une fois que le projet est jugé prêt, une version complète du logiciel est livrée au client.

Question : Quel est le modèle de cycle de vie adopté par l'entreprise ? Justifiez votre réponse.

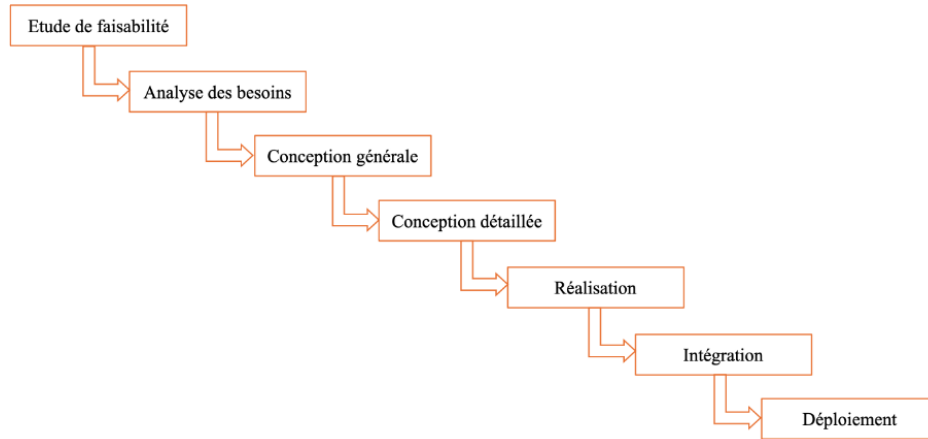
- **Modèle en spirale**

- **Pourquoi ce modèle ?**

Ce modèle est utilisé car il débute par une analyse des besoins et des risques pour poser des bases solides. Un prototype permet ensuite de tester les idées clés et de valider les choix. La gestion des risques se fait en continu, avec une décision stratégique à chaque cycle : poursuivre, ajuster, ou arrêter. Ce modèle convient particulièrement aux projets complexes et innovants.

Exercice 3 : Étapes du modèle en cascade

Dans la représentation graphique ci-dessous du modèle en cascade, **veuillez préciser les entrées et sorties principales pour chaque phase.**



1. **Étude de faisabilité**
 - **Entrées** : L'idée du projet, ce que le client veut, les contraintes (budget, délais).
 - **Sorties** : Un rapport qui dit si le projet est réalisable ou non.
2. **Analyse des besoins**
 - **Entrées** : Les objectifs validés et les premiers besoins du client.
 - **Sorties** : Un cahier des charges détaillé.
3. **Conception générale**
 - **Entrées** : Le cahier des charges.
 - **Sorties** : Un plan global (avec schémas et technologies choisies).
4. **Conception détaillée**
 - **Entrées** : Le plan global.
 - **Sorties** : Des détails précis sur les modules, les interfaces, et les algorithmes.
5. **Réalisation (Programmation)**
 - **Entrées** : Les plans détaillés et l'environnement de travail.
 - **Sorties** : Le code source et les premiers tests.
6. **Intégration**
 - **Entrées** : Les morceaux de code testés séparément.
 - **Sorties** : Le logiciel complet, testé pour s'assurer qu'il fonctionne bien.
7. **Déploiement**
 - **Entrées** : Le logiciel finalisé.
 - **Sorties** : Le logiciel est mis en production, avec un guide d'utilisation et un support technique.

Exercice 4 : Méthode agile pour l'application de pointage

Une entreprise souhaite réaliser une application de pointage permettant de gérer les heures d'entrée, de sortie, les absences, les retards, ainsi que le travail durant les jours fériés.

Dans le cadre de ce projet, quelle méthode agile serait la plus appropriée : XP (Programmation Extrême) ou FDD (Développement Dirigé par les Fonctionnalités) ? Justifiez votre choix.

La méthode : Extreme Programming (XP).

Justification :

1. Le projet doit avancer vite et nécessite des ajustements en continu selon les retours des utilisateurs.
 2. XP est parfait pour les projets où il faut être flexible et livrer des fonctionnalités rapidement.
-

Exercice 5 :

Une grande entreprise décide de lancer un portail e-commerce pour vendre ses produits à travers le monde. Le projet est vaste, avec des fonctionnalités bien définies dès le départ. Le portail doit inclure :

1. La gestion des produits (ajout, suppression, mise à jour des stocks).
2. Une recherche avancée par catégories et filtres.
3. Un système de gestion des utilisateurs (inscription, connexion, profils).
4. Une fonctionnalité de panier d'achat et de paiement sécurisé.
5. Des rapports sur les ventes et les statistiques de performance.

Dans le cadre de ce projet, quelle méthode agile serait la plus appropriée : XP (Programmation Extrême) ou FDD (Développement Dirigé par les Fonctionnalités) ? Justifiez votre choix

La méthode : La méthode appropriée est **FDD (Feature-Driven Development)**.

Justification :

- Le projet est vaste, avec des fonctionnalités bien définies dès le départ.
 - FDD se concentre sur le développement par fonctionnalités, adapté aux grands projets structurés.
-

Exercice 6 :

Une société est spécialisée dans les formations en ligne. Elle souhaite développer une plateforme e-learning pour ses utilisateurs. Cette plateforme doit inclure :

1. Une gestion des utilisateurs (étudiants et formateurs).
2. Un espace de cours interactif (vidéos, quiz, documents).
3. Un système de messagerie entre étudiants et formateurs.
4. Des statistiques détaillées sur la progression des étudiants.
5. Une compatibilité avec les appareils mobiles.

L'entreprise a une idée globale des fonctionnalités souhaitées, mais les besoins détaillés sont susceptibles d'évoluer pendant le développement, en fonction des retours des parties prenantes et des tests sur les premières versions.

Dans le cadre de ce projet, quelle méthode agile est utilisée : XP (Programmation Extrême) ou FDD (Développement Dirigé par les Fonctionnalités) ? Justifiez votre choix.

La méthode : Extreme Programming (XP).

Justification :

1. Les besoins évoluent au fur et à mesure du projet, avec des ajustements basés sur les retours des parties prenantes.
2. XP offre la flexibilité nécessaire pour répondre aux changements fréquents. XP est parfait pour les projets où il faut être flexible et livrer des fonctionnalités rapidement.

Exercice 7 :

Une entreprise spécialisée dans les transports publics souhaite développer un logiciel pour améliorer l'expérience des voyageurs. Ce logiciel devra inclure les fonctionnalités suivantes :

1. Une interface de planification des trajets (calcul des itinéraires, horaires, correspondances).
2. Un système de suivi en temps réel des véhicules (bus, trains, etc.).
3. Une gestion des paiements électroniques pour l'achat de billets.
4. Une section d'assistance utilisateur pour signaler des problèmes ou poser des questions.

À partir de cette demande, identifiez et décrivez les principales étapes du développement de ce logiciel.

- **Étude de faisabilité :**
On analyse si le projet est faisable en tenant compte des contraintes techniques, financières, et organisationnelles. On évalue les ressources nécessaires pour valider sa viabilité.
 - **Analyse des besoins :**
On identifie et décrit clairement ce que le projet doit faire, comme la planification des trajets, le suivi en temps réel, les paiements électroniques, et l'assistance aux utilisateurs.
 - **Conception générale :**
On définit la structure globale du projet (par exemple, une architecture en trois parties) et on choisit les technologies qui conviennent. On planifie aussi les interactions principales entre les éléments.
 - **Conception détaillée :**
On détaille les modules, les algorithmes, et les interfaces. On précise comment les différentes parties vont interagir entre elles.
 - **Réalisation :**
On développe les différentes parties du projet et on teste chacune pour vérifier qu'elles fonctionnent bien individuellement.
 - **Intégration :**
On assemble tous les modules pour former le logiciel complet et on vérifie que tout fonctionne correctement ensemble.
 - **Déploiement :**
Le logiciel est mis en production et les utilisateurs reçoivent des guides et de l'aide pour l'utiliser facilement.
-

Exercice 8 :

Une entreprise envisage de développer un logiciel complexe pour répondre à des besoins spécifiques. Avant de choisir le modèle de développement approprié, elle souhaite comprendre les avantages et les inconvénients des différents modèles.

Questions :

• Complétez le tableau suivant en présentant deux avantages et deux inconvénients pour chaque modèle de développement :

Modèle	Avantages	Inconvénients
XP (Extreme Programming)	<ul style="list-style-type: none"> - Permet une livraison rapide. - Forte collaboration avec le client. 	<ul style="list-style-type: none"> - Moins adapté aux grands projets complexes. - Demande une forte implication continue des parties prenantes.
FDD (Feature-Driven Development)	<ul style="list-style-type: none"> - Approprié pour les grands projets avec des fonctionnalités bien définies. - Bonne organisation. 	<ul style="list-style-type: none"> - Moins flexible. - Plus long en cas de modifications fréquentes.
Modèle en cascade	<ul style="list-style-type: none"> - Simple à comprendre. - Adapté aux projets bien définis dès le départ. 	<ul style="list-style-type: none"> - Rigidité en cas de changements tardifs. - Corrections difficiles en fin de cycle.
Modèle en V	<ul style="list-style-type: none"> - Validation continue à chaque étape. - Réduit les risques d'erreur. 	<ul style="list-style-type: none"> - Processus plus coûteux. - Adapté uniquement aux projets critiques.
Modèle en spirale	<ul style="list-style-type: none"> - Gère les risques dès le départ. - Approche flexible et itérative. 	<ul style="list-style-type: none"> - Approche complexe et coûteuse. - Nécessite des compétences avancées en gestion de projets.
Modèle évolutif	<ul style="list-style-type: none"> - Ajustements fréquents grâce aux retours utilisateurs. - Permet une amélioration continue. 	<ul style="list-style-type: none"> - Difficulté à planifier précisément coût et durée. - Risque de dérive des objectifs.

• Sur la base de vos réponses, expliquez dans quel type de projet chaque modèle serait le plus adapté.

- **XP** convient aux petits projets nécessitant des ajustements fréquents et une livraison rapide.
- **FDD** est idéal pour des projets de grande envergure avec des exigences bien définies.
- **Modèle en cascade** s'adapte aux projets simples, avec des spécifications stables dès le départ.
- **Modèle en V** est parfait pour les projets critiques nécessitant une validation rigoureuse, comme les systèmes embarqués.
- **Modèle en spirale** est adapté aux projets complexes et innovants où l'évaluation des risques est essentielle.
- **Modèle évolutif** est recommandé pour les projets où les besoins évoluent au fil du développement, grâce aux retours continus des utilisateurs.