TD2

Partie 1 :

Ex 1 : Une entreprise LOG de production logiciel adopte un processus de développement logiciel qui consiste à enchaîner les différentes phases de développement : étude de faisabilité, spécification, conception, implémentation, tests et livraison. Les retours en arrière entre ces différentes phases ne sont pas planifiés mais si des erreurs sont détectées pendant les tests, il est possible que l'équipe de développement réadapte la conception et/ou l'implémentation du logiciel. Le succès des projets de développement logiciel de cette entreprise est garanti seulement s'il s'agit de reproduire un projet déjà réalisé

Déterminez le modèle de cycle de vie utilisé par cette entreprise.

Le modèle de cycle de vie utilisé par cette entreprise est le modèle de cycle de vie en cascade. Le modèle de cycle de vie en cascade est un processus de développement logiciel linéaire et séquentiel dans lequel les phases de développement sont exécutées de manière séquentielle, chaque phase devant être terminée avant de passer à la suivante. Dans ce modèle, les retours en arrière entre les phases ne sont pas planifiés, mais il est possible de réadapter la conception et/ou l'implémentation du logiciel si des erreurs sont détectées pendant les tests.

Dans le modèle de cycle de vie en cascade, l'étude de faisabilité est la première phase, suivie de la spécification, de la conception, de l'implémentation, des tests et de la livraison. Le succès des projets de développement logiciel de cette entreprise est garanti seulement s'il s'agit de reproduire un projet déjà réalisé, ce qui est en accord avec le modèle en cascade, qui est efficace pour les projets ayant des spécifications claires et bien définies.

Ex2 : Les jalons (milestones) sont des évènements qui servent à indiquer le degré d'avancement d'un projet de logiciel comme l'achèvement du manuel d'utilisateur.

1) En quoi un modèle de cycle de vie divisé en phases aide-t-il à la gestion du développement d'un logiciel ?

Un modèle de cycle de vie divisé en phases et associé à des jalons aide à la gestion du développement de logiciels en fournissant une structure claire pour le processus de développement, en permettant une meilleure estimation des ressources et des délais, et en permettant une surveillance et un contrôle plus efficaces de l'avancement du projet.

2)Quelles sont les deux caractéristiques obligatoires d'un jalon (milestone) ?

1. Il doit être mesurable : cela signifie que le jalon doit être spécifique, objectif et quantifiable, de sorte qu'il soit clair pour toute l'équipe de développement quand il a été atteint. Il doit y avoir un critère ou une définition claire pour déterminer si le jalon a été atteint ou non.
2. Il doit être significatif : cela signifie que le jalon doit marquer une étape importante dans le développement du projet. Il doit y avoir un changement notable dans l'état ou l'avancement du projet lorsqu'un jalon est atteint, de sorte que cela puisse être reconnu et célébré comme une réalisation importante pour l'équipe de développement.

Ex3 : En considérant le cycle de vie d'un logiciel

1) Indiquer la ou les phases où est produit chacun des documents suivants : Manuel d'utilisation, conception architecturale, plan d'assurance qualité, spécification des modules, code source, cahier de charges, plan de test, manuel utilisateur préliminaire, conception détaillée, estimation des couts, calendrier du projet, rapport des tests, documentation.

- Cahier des charges : Phase d'analyse et de spécification.
- Conception architecturale : Phase de conception.
- Spécification des modules : Phase de conception.
- Conception détaillée : Phase de conception.
- Code source : Phase de développement.
- Plan d'assurance qualité : Tout au long du cycle de vie du logiciel.
- Plan de test : Phase de développement et de tests.
- Rapport des tests : Phase de tests.
- Documentation : Tout au long du cycle de vie du logiciel.
- Manuel d'utilisation : Phase de tests et de livraison.
- Manuel utilisateur préliminaire : Phase de conception.
- Estimation des coûts : Phase d'analyse et de spécification.
- Calendrier du projet : Tout au long du cycle de vie du logiciel.

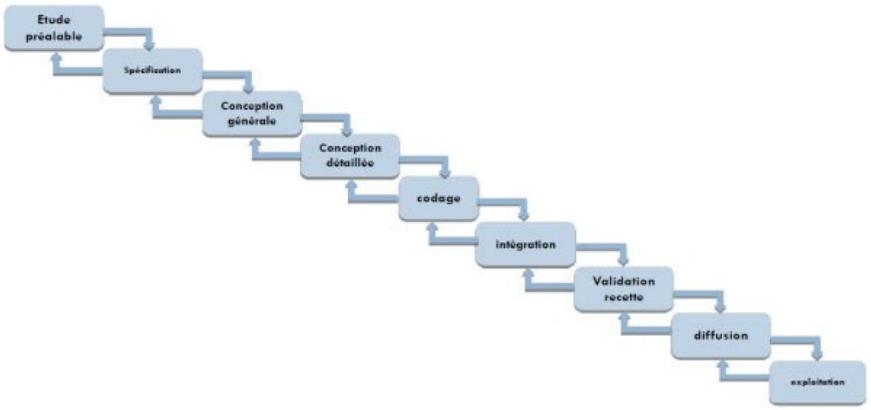
2) Quelles différences y a-t-il avec un modèle de processus ?

Le cycle de vie d'un logiciel décrit les différentes étapes du développement d'un logiciel, tandis que le modèle de processus décrit comment les différentes activités doivent être réalisées pour atteindre les objectifs du cycle de vie. Le modèle de processus est un outil pratique pour appliquer et mettre en œuvre les principes du cycle de vie du logiciel.

Ex4 : Comment peut-on combiner le modèle en cascade ou en V avec le modèle en spirale ?

Il est possible de combiner les modèles de cycle de vie en cascade ou en V avec le modèle en spirale en utilisant une approche itérative et incrémentale. Cette approche consiste à décomposer le projet en plusieurs étapes et itérations, avec des jalons pour mesurer l'avancement du projet. Chaque itération suit un modèle en spirale avec des phases d'analyse des risques, de conception, de développement et de test. Une fois qu'une itération est terminée, le modèle en cascade ou en V est utilisé pour valider la qualité du produit. Cette combinaison permet d'intégrer les avantages de chaque modèle et d'adapter le processus de développement en fonction de l'avancement du projet et des risques identifiés.

Ex5 : Dans la représentation graphique suivante du modèle en cascade :



1)Préciser les entrées et sorties principales (pas forcément des documents) pour chaque phase.

1. **Analyse des besoins / spécifications**

- Entrées : demande du client, analyse des concurrents, étude de marché, etc.
- Sorties : spécifications du produit, exigences fonctionnelles et non fonctionnelles, plan de validation, etc.

2. **Conception**

- Entrées : spécifications du produit, exigences, architecture logicielle, etc.
- Sorties : conception détaillée du produit, schémas UML, plans de test, etc.

3. **Implémentation**

- Entrées : conception détaillée, plan de test, environnement de développement, etc.
- Sorties : code source, exécutables, modules, etc.

4. **Tests**

- Entrées : code source, spécifications, plans de test, environnement de test, etc.
- Sorties : rapports de test, correction des erreurs, validation, etc.

5. **Déploiement / Maintenance**

- Entrées : version finale du produit, documentation utilisateur, etc.
- Sorties : produit déployé, maintenance corrective et évolutive, mises à jour, etc.

2) Quelles sont les phases concernées par la vérification et/ou la validation ?

La vérification est associée aux phases de spécification, de conception et de codage et vise à s'assurer que le logiciel est conforme aux exigences et aux spécifications établies pour lui, ainsi qu'aux normes et aux standards de développement.

La validation est associée à la phase de tests et vise à s'assurer que le logiciel est apte à remplir les fonctions pour lesquelles il a été conçu et que les besoins et les attentes des utilisateurs sont satisfaits.

Partie 2 :

Ex1 : Pour la peinture des murs d'une pièce, on considère :

1. les tâches suivantes : choisir la couleur, acheter la peinture, nettoyer les murs, préparer la peinture et peindre les murs ;
2. les artefacts suivants : choix de la couleur, pots de peinture achetés, murs propres, peinture mélangée, murs peints.

Dessiner un modèle de processus pour la peinture des murs.

1. Choix de la couleur
    - Entrée : préférences de couleur du client
    - Sortie : choix de la couleur de peinture
2. Achat de la peinture
    - Entrée : choix de la couleur de peinture
    - Sortie : pots de peinture achetés
3. Nettoyage des murs
    - Entrée : murs à peindre
    - Sortie : murs propres
4. Préparation de la peinture
    - Entrée : pots de peinture achetés, mélangeur de peinture
    - Sortie : peinture mélangée
5. Peinture des murs
    - Entrée : murs propres, peinture mélangée
    - Sortie : murs peints

Ex2 : Pour assurer un enseignement à distance aux étudiants, l'instructeur divise les élèves en équipes et affiche un problème sur une page Web. Les équipes travaillent sur le problème en utilisant le tchat, ils posent des questions à l'instructeur en utilisant un forum, et ils soumettent les solutions par email. L'instructeur évalue ensuite les solutions en fonction d'un barème préétabli.

Dessiner un modèle de processus pour préparer les sessions interactives.

1. Planification de la session : l'instructeur planifie la session en choisissant le sujet, la date, l'heure et la durée de la session.
2. Préparation du matériel : l'instructeur prépare le matériel nécessaire pour la session, comme les problèmes à résoudre et les instructions pour les élèves.
3. Publication du matériel : l'instructeur publie le matériel sur la page Web et informe les élèves par e-mail de la mise à disposition des ressources.
4. Séance interactive : les élèves travaillent en équipes pour résoudre les problèmes en utilisant le tchat pour communiquer et le forum pour poser des questions à l'instructeur si nécessaire.
5. Soumission des solutions : les élèves soumettent leurs solutions par e-mail.
6. Évaluation des solutions : l'instructeur évalue les solutions en fonction du barème préétabli et envoie les notes aux élèves

Ex3 : Soit les trois types de tests: tests unitaires, d’intégration et d’acceptation.

Dessiner un modèle de processus pour chaque type de test.

Modèle de processus pour les tests unitaires :

1. Identifier les fonctions ou les modules à tester.
2. Écrire les scénarios de test.
3. Écrire le code de test.
4. Exécuter les tests.
5. Analyser les résultats des tests.
6. Corriger les erreurs et répéter les étapes 2 à 5 jusqu'à ce que les tests réussissent.
7. Documenter les résultats des tests.

Modèle de processus pour les tests d'intégration :

1. Identifier les différents modules à intégrer.
2. Écrire les scénarios de test.
3. Définir l'ordre d'intégration des modules.
4. Intégrer les modules et exécuter les tests.
5. Analyser les résultats des tests.
6. Corriger les erreurs et répéter les étapes 2 à 5 jusqu'à ce que les tests réussissent.
7. Documenter les résultats des tests.

Modèle de processus pour les tests d'acceptation :

1. Identifier les scénarios de test d'acceptation.
2. Écrire les scénarios de test.
3. Exécuter les tests.
4. Analyser les résultats des tests.
5. Corriger les erreurs et répéter les étapes 2 à 4 jusqu'à ce que les tests réussissent.
6. Documenter les résultats des tests.
7. Fournir les résultats des tests aux parties prenantes pour approbation.