Résumé

Chapitre 1 :Fondamentaux de Test

Cours:

Le chapitre 1 explore les principes de base des tests logiciels. Il commence par souligner l'importance de tester les logiciels, car un logiciel défectueux peut entraîner des problèmes financiers, de réputation et de temps. Les tests logiciels ne se limitent pas à l'exécution de tests, mais englobent un processus complet, y compris la planification, l'analyse, la conception et la mise en œuvre des tests, ainsi que le suivi des résultats.

- **Qu'est-ce que les tests ?** Les tests logiciels consistent à évaluer la qualité du logiciel en vérifiant s'il répond aux exigences, s'il fonctionne correctement et en identifiant les défauts.
- **Pourquoi tester ?** Les tests sont essentiels pour éviter des problèmes tels que la perte d'argent, de temps ou la détérioration de la réputation de l'entreprise. Ils servent à évaluer, valider, prévenir des défauts, trouver des défaillances, et réduire les risques liés à une qualité logicielle insuffisante.
- **Les 7 principes de test** : Ces principes incluent le fait que les tests révèlent la présence de défauts, que les tests exhaustifs sont impossibles, qu'il faut tester tôt, etc.

1. Les tests montrent la présence de défauts : Les tests sont conçus pour détecter des défauts dans le logiciel. En d'autres termes, les tests révèlent des anomalies ou des problèmes potentiels.
2. Les tests peuvent prouver la présence de défauts, mais ne peuvent en prouver l'absence : Les tests peuvent démontrer que des défauts existent, mais il est impossible de garantir que le logiciel est exempt de défauts à 100 %, car il est impossible de tester toutes les combinaisons possibles.
3. Les tests exhaustifs sont impossibles : Il est généralement impossible de tester toutes les combinaisons possibles d'entrées, de scénarios et de conditions d'un logiciel en raison de contraintes de temps et de ressources. Il faut donc prioriser les tests.
4. Tout tester veut dire gérer la focalisation des efforts de tests en fonction des risques et priorités : Les tests doivent être concentrés sur les aspects du logiciel les plus importants et les plus susceptibles de causer des problèmes, en fonction des risques et des priorités.
5. Tester tôt : Les tests devraient commencer dès que possible dans le processus de développement. Plus tôt les défauts sont détectés, plus il est facile et moins coûteux de les corriger.

6. Regroupement des défauts : Il est typique que les défauts se produisent dans des groupes, ce qui signifie qu'ils se concentrent dans certaines parties du logiciel plutôt que d'être uniformément répartis.
7. Paradoxe du pesticide : Si les mêmes tests sont répétés de nombreuses fois, ils deviendront moins efficaces car ils ne détecteront que les défauts déjà connus. Il est essentiel de renouveler et d'ajuster régulièrement les tests

- **Le processus de test :** Il consiste sur la planification des tests, l'analyse et la conception des tests, l'implémentation et l'exécution des tests, l'évaluation des critères de sortie et l'information, ainsi que les activités de clôture des tests.
- **Psychologie de test :** Les tests logiciels impliquent des êtres humains, et il est essentiel de comprendre la psychologie humaine pour communiquer efficacement sur les défauts et les résultats des tests.

QCM:

Question : Lequel est visible pour l'utilisateur final comme écart entre la spécification technique et le comportement attendu ? Réponse : C. Un défaut

Question : Que veut dire un test objectif ? Réponse : A. Les tests unitaires

Question : Lesquels de ces objectifs de test sont valides ? Réponse : A. 1, 2 et 3

Question : Soit la liste suivante d'activités du processus de test :

Analyse et conception
Activités de clôture de test
Évaluer les critères de sortie et informer
Planification et contrôle
Implémentation et exécution Dans quel ordre logique ces activités se déroulent ? Réponse : B. 4, 1, 5, 3 et 2

Question : Lequel des énoncés suivants décrit un principe clé des tests logiciels? Réponse : B. Pour un système de logiciel, il est normalement impossible de tester toutes les combinaisons d'entrée et de sortie.

Question : Quels sont les défauts les moins coûteux à corriger? Réponse : C. Les défauts détectés tôt dans le cycle de vie.

Question : Lequel des problèmes ci-dessous représente une conséquence directe des défaillances logicielles? Réponse : A. Mauvaise réputation.

Chapitre 2 :Test pendant le cycle de vie

Cours:

1. **Introduction :** Cette section introduit le chapitre en énonçant l'objectif principal comprendre comment les tests logiciels sont intégrés tout au long du cycle de vie du développement logiciel. Elle établit le contexte pour les discussions à venir sur les modèles de développement logiciel, les niveaux de tests, les types de tests, et conclut avec des exercices pour renforcer la compréhension.
2. **Modèles de développement logiciel :** Dans cette partie, le chapitre aborde les différents modèles de développement logiciel, tels que le modèle en cascade, le modèle en V et le modèle itératif. Il explique comment le choix du modèle impacte la manière dont les tests sont planifiés, conçus et exécutés tout au long du cycle de vie du logiciel.

Modèle en V : Ce modèle met l'accent sur la vérification et la validation à chaque étape du développement. Les tests sont intégrés dès le début, et les phases de développement et de test se reflètent en forme de "V."

Modèle itératif : Ce modèle divise le projet en itérations. Chaque itération passe par des phases de développement et de test. Les tests sont effectués fréquemment tout au long du projet.

3. **Niveaux de tests :** Cette section se concentre sur les différents niveaux de tests, notamment les tests unitaires, les tests de composants, les tests d'intégration, les tests systèmes et les tests d'acceptation. Elle explique les caractéristiques et les objectifs spécifiques de chaque niveau de test, soulignant leur pertinence à différentes étapes du développement.

Tests unitaires : Ils se concentrent sur la vérification des composants individuels (unités) du logiciel, tels que les fonctions, les classes ou les méthodes.

Tests de composants : Ils évaluent le comportement des groupes de composants qui interagissent, ce qui permet de détecter des erreurs résultant de ces interactions.

Tests d'intégration : Ces tests vérifient la manière dont les composants s'intègrent pour former des modules plus importants. Ils peuvent mettre en évidence des problèmes liés à la communication et à l'intégration.

Tests systèmes : Ces tests portent sur le système dans son ensemble et évaluent s'il remplit les exigences spécifiées.

Tests d'acceptation : Ils s'assurent que le système est conforme aux attentes des utilisateurs et des clients, ce qui inclut des tests d'acceptation utilisateurs, des tests opérationnels, des tests contractuels, etc.

4. **Types de tests :** Les types de tests sont explorés ici. Il s'agit de distinguer les tests fonctionnels des tests non fonctionnels, et les tests boîte blanche des tests boîte noire. Cette partie clarifie les différents aspects du logiciel qui sont évalués à travers ces types de tests.

Tests fonctionnels : Ils se concentrent sur ce que le logiciel doit faire et s'alignent sur les spécifications. Ils vérifient si les fonctionnalités du logiciel fonctionnent correctement.

Tests non fonctionnels : Ces tests évaluent comment le logiciel fonctionne. Ils incluent des tests de performance, de sécurité, de fiabilité, d'utilisabilité, etc.

Tests boîte blanche : Ils examinent la structure interne du logiciel, notamment le code source. Ces tests sont particulièrement utiles pour détecter des erreurs de programmation.

Tests boîte noire : Ils s'appuient sur des spécifications et évaluent le comportement du logiciel sans se préoccuper de la structure interne. Ils sont essentiels pour vérifier si le logiciel répond aux besoins spécifiés

Tests liés au changement : Ces tests sont effectués en réponse à des modifications apportées au logiciel ou à son environnement. Ils comprennent :

1/Tests de confirmation : Ils vérifient que les actions de correction, effectuées en réponse à des échecs de test précédents, ont été couronnées de succès. Ces tests sont essentiels pour s'assurer que les problèmes ont été résolus correctement.

2/Tests de régression : Ces tests sont réalisés sur un programme qui a déjà été testé, après avoir apporté des modifications. L'objectif est de garantir que les modifications n'ont pas introduit de nouveaux défauts et que les parties non modifiées du logiciel restent intactes. Les tests de régression sont cruciaux pour maintenir la stabilité du logiciel au fil du temps.

Tests de maintenance : Ils se concentrent sur les systèmes logiciels opérationnels existants. Ces tests sont déclenchés par des modifications, des migrations ou la suppression de logiciels ou de systèmes. Les activités de maintenance comprennent les tests des modifications et les tests de régression, qui visent à garantir que les systèmes opérationnels restent fonctionnels et conformes aux exigences malgré les changements.

5. **Conclusion :** La section de conclusion récapitule les points clés du chapitre et réaffirme l'importance d'intégrer les tests tout au long du cycle de vie du développement logiciel. Elle rappelle également l'importance de choisir le modèle de développement logiciel adapté à un projet donné.

QCM:

Question : Les tests système doivent examiner :

Réponse : Les exigences non fonctionnelles et fonctionnelles

Question : Laquelle des ces affirmations définissent le test de régression?

Réponse :1-2

Question : Quelle affirmation sur le test fonctionnel est exacte?

Réponse : Le test fonctionnel est utile tout au long du cycle de vie et peut être appliqué par des analystes métiers, des testeurs, des développeurs et des utilisateurs.

Question : Laquelle de ces informations est la plus souvent correcte ?

Réponse : Le test de composant cherche les défauts dans les programmes qui sont testables séparément.

Question : Les tests de confirmation sont :

Réponse : Les tests qui exécutent les cas de test qui ont été en échec la dernière fois qu'ils furent exécutés.

Question : Les tests réalisés pour une intégration de haut en bas (Top-Down)

Réponse : Nécessite des pilotes

Chapitre 3 :Techniques statiques

Cours:

- **Les Tests Statiques**

Les tests statiques sont une composante essentielle de l'assurance qualité logicielle, se distinguant des tests dynamiques par l'absence d'exécution du code. Ils se concentrent sur l'examen manuel des produits d'activités ou l'évaluation automatisée du code et d'autres livrables.

- **Processus de Revue**

Les revues sont un aspect clé des tests statiques. Elles consistent en une évaluation formelle des produits d'activités, offrant une opportunité de détection précoce des défauts et d'amélioration de la qualité. Les revues peuvent varier en formalité, de l'informelle à la formelle.

- **Activités du Processus de Revue**

Le processus de revue comprend plusieurs étapes, notamment la planification, le lancement de la revue, la revue individuelle, la communication et l'analyse des problèmes, ainsi que la correction et la production de rapports. Chacune de ces étapes joue un rôle crucial dans le succès de la revue.

- **Rôles et Responsabilités**

Les participants aux revues assument divers rôles, dont le manager, le facilitateur, l'auteur, le responsable de la revue, les réviseurs, et le scribe. Chacun de ces rôles a des responsabilités spécifiques dans le processus de revue.

- **Types de Revue**

Revue Informelle : Elle ne suit pas un processus formel, peut ne pas comporter de réunion de revue, les résultats peuvent être documentés selon le choix, et l'utilisation de checklists est facultative. Elle est couramment utilisée dans le développement Agile, avec pour principal objectif de détecter d'éventuels défauts.

Relecture technique : Généralement dirigée par l'auteur, optionnellement précédée d'une réunion de préparation. Elle vise à trouver des défauts, à améliorer le produit logiciel, à envisager des implémentations alternatives et à évaluer la conformité aux normes et aux spécifications.

Revue technique : Un processus de détection de défauts défini impliquant des pairs et des experts techniques. Elle vise à obtenir un consensus, à détecter des défauts potentiels, à évaluer la qualité et à renforcer la confiance dans le produit examiné.

Inspection : Dirigée par un facilitateur formé, c'est un processus formel basé sur des règles et des check-lists pour l'examen visuel de documents en vue de détecter des défauts. Elle vise à trouver des défauts, à améliorer la qualité du document, à tirer des leçons et à améliorer le processus.

- **Application des Techniques de Revue**

Plusieurs techniques de revue peuvent être utilisées pour découvrir des défauts, notamment l'approche ad hoc, les checklists, les scénarios et les essais à blanc, ainsi que les techniques basées sur les rôles et les perspectives. Chaque technique a ses avantages et ses inconvénients.

- **Facteurs de Réussite des Revues**

Le succès des revues dépend de divers facteurs, notamment la clarté des objectifs, la sélection appropriée des participants, l'acceptation objective des défauts identifiés, la gestion des aspects personnels et psychologiques, l'adaptation des techniques de revue, la formation des participants, et le soutien de la direction.

- **Conclusion**

Les tests statiques, bien qu'ils ne puissent pas valider le comportement du logiciel en cours d'exécution, sont essentiels pour identifier des défauts qui ne sont pas facilement détectables par les tests dynamiques. Ils contribuent à améliorer la qualité des produits d'activités, réduisent les coûts et les délais de développement, et constituent une composante essentielle de l'assurance qualité logicielle. Cependant, ils sont complémentaires aux tests dynamiques pour une évaluation complète de la qualité du logiciel.

QCM:
Question : Parmi les affirmations suivantes concernant les revues de spécification, laquelle est vraie ?

Réponse : Les revues sont une façon rentable de faire tôt du test statique sur le système.

Question : Les défauts typiques qui sont faciles à dégager par les revues plutôt que par les tests dynamiques

Réponse : Toutes les trois réponses ci-dessus.

Question : Les revues, les tests statiques et dynamiques ont le même objectif :

Réponse : Identifier les défauts.

Question : Pourquoi les tests statiques et les tests dynamiques sont complémentaires?

Réponse : Parce qu'ils ont des objectifs différents, mais ils trouvent les mêmes types de défauts.

Question : Quel est théoriquement l'ordre des étapes à respecter lors d'une revue formelle ?

Réponse : Planification, préparation, séance de revue, suivi.

Question : Quels rôles et actions faut-il pour les vérifications/revues ?

Réponse : Pour une inspection, il faut prévoir une phase de préparation pour les réviseurs.

Chapitre 4 :Technique de tests

Cours:

Objectifs

Différencier la spécification de conception de test de la spécification des cas de test et des spécifications de procédures de test.
Comprendre et différencier les termes : condition de test, cas de test et procédure de test.
Savoir différencier les techniques basées sur les spécifications (boîte blanche) et la technique boîte noire.

Introduction

Le chapitre commence par une introduction générale aux tests, mettant en évidence le besoin de tests dans le processus de développement logiciel.

Processus de développement de Test(1)

- Phase d'analyse de conception des tests : Cette phase implique l'analyse de la documentation de la base des tests pour déterminer ce qui doit être testé, identifier les conditions de test et établir la traçabilité des exigences.
- Classification des exigences : Les exigences sont classées en fonction du type de test, du niveau de l'exigence, de l'importance de l'exigence et de la fragilité potentielle du produit.

Processus de développement de Test(2)

Le chapitre explique les différentes catégories de techniques de test, à savoir les techniques basées sur les spécifications (boîte blanche), les techniques basées sur la structure (boîte blanche) et les techniques basées sur l'expérience.

Techniques basées sur les spécifications fonctionnelles(boîte noire)

- Partitions d'équivalence : Cette technique consiste à regrouper des entrées en classes d'équivalence qui montrent un comportement similaire.
- Analyse des valeurs limites : Elle se concentre sur les valeurs aux limites, car les erreurs sont souvent commises à proximité des limites des domaines de définition.
- Table de décision : Cette technique consiste à construire une table montrant les combinaisons des entrées et/ou stimuli et de leurs sorties et/ou actions associées.
- Tests de transition d'États : Cette technique est utilisée lorsque différents états peuvent être définis dans l'application.

Techniques de Tests Basées sur la Structure Boîte Blanche :

Ces techniques examinent la structure interne du logiciel et visent à obtenir une couverture complète de la structure. Elles comprennent la couverture des instructions, la couverture des branches et la couverture des conditions.

Techniques Basées sur l'Expérience :

Ces techniques s'appuient sur l'expérience des testeurs pour anticiper les erreurs possibles. Elles incluent :

- Tests Exploratoires : Des tests informels conçus, exécutés, enregistrés et évalués dynamiquement pendant l'exécution des tests.
- Tests Basés sur des Checklists : Des tests basés sur des listes de contrôle qui contiennent les conditions de test.

QCM:

1. Le bout de code donné contient une erreur potentielle de division par zéro. La méthode la plus efficace pour détecter cette erreur serait le "Test aux limites."
2. Le test effectué avec les valeurs "1er janvier 2009", "31 avril 2010", "30 juin 2009" et "1er octobre 2008" est un "Test basé sur l'expérience."
3. Pour couvrir à 100 % les décisions dans le fragment de code donné, il faut 3 cas de test.
4. Le test "Table de décision" n'appartient pas aux techniques boîtes noires.
5. La réponse correcte est : "La partition d'équivalence, les transitions d'états, les tests des cas d'utilisation et la table de décision sont des techniques de test de type White Box."
6. Les valeurs aux limites pour la température sont "15, 19 et 25 °C."
7. La réponse correcte est : "Couverture d'instructions : 2, couverture de branches : 4."
8. La couverture d'instructions ne peut pas contrôler "Le code mort."
9. La réponse correcte est : "Couverture des instructions; couverture des branches; analyse du flux de données."
10. La mesure de couverture "Est une mesure partielle du test."
11. Les valeurs de tests aux limites sont "1900 et 2004."
12. La partition d'équivalence est "Une technique de test de catégorie black box appropriée à tous les niveaux de test."

Chapitre 5 :Gestion des tests

Cours:

Introduction :

Dans ce chapitre, on se penche sur l'organisation des tests, l'estimation et la planification des tests, le suivi et le contrôle du déroulement des tests, la gestion de configuration, les tests et les risques, la gestion des incidents, et enfin, une conclusion.

Rôles des Acteurs :

Plusieurs acteurs interviennent dans les tests, notamment les testeurs, les développeurs, le chef de projet, et le responsable du test.

Gestion des Tests :

Il existe différentes techniques pour gérer les tests, notamment en établissant une stratégie de test appropriée en fonction des besoins du projet.

Indépendance des Testeurs :

L'indépendance des testeurs est cruciale pour garantir l'objectivité et l'impartialité. Différentes options d'indépendance sont possibles, comme les développeurs testant leur propre code ou des testeurs externes à l'organisation.

Tâches du Responsable de Test :

Les responsables de test ont plusieurs tâches, notamment la planification des tests, la surveillance et le contrôle de leur exécution, l'adaptation du planning en fonction des résultats, la gestion de configuration, la mesure de l'avancement des tests, la sélection des outils, et la création de rapports de synthèse de test.

Tâches des Testeurs :

Les testeurs passent en revue les plans de test, analysent et évaluent les exigences, créent des spécifications de test, préparent l'environnement de test, obtiennent les données de test, exécutent les cas de test, mesurent les performances des composants, et passent en revue les tests développés par d'autres.

Estimation et Planification des Tests :

L'estimation et la planification des tests sont essentielles pour garantir que les tests sont effectués de manière structurée et rentable. Les critères d'entrée et de sortie des tests sont définis pour déterminer quand commencer et quand arrêter les tests.

Suivi et Contrôle des Tests :

Le suivi et le contrôle des tests permettent de fournir un retour sur les activités de test, de collecter des données pour les projets futurs, de mesurer l'avancement par rapport au calendrier et au budget planifiés, et de prendre des décisions basées sur les informations recueillies.

Gestion de Configuration :

La gestion de configuration assure la traçabilité des éléments du test et des changements apportés, tout en garantissant que les documents et les éléments du logiciel sont référencés de manière non ambiguë dans la documentation de test.

Test et Risques :

Les tests doivent prendre en compte les risques liés au projet et au produit, car les défauts logiciels peuvent avoir des conséquences graves.

Gestion des Incidents :

Les rapports d'incidents permettent de fournir un retour sur les problèmes rencontrés, d'assurer l'amélioration du processus de test, et de maintenir la traçabilité des anomalies.

Conclusion :

La gestion des tests est essentielle pour garantir la qualité du logiciel et la conformité aux normes et aux normes de l'industrie.

QCM:

Pourquoi est-il nécessaire de définir une stratégie de test?

Il est nécessaire de définir une stratégie de test car il existe de nombreuses façons de tester un logiciel, et il est essentiel de réfléchir pour décider de la méthode la plus efficace pour tester le projet en cours.
Démarrer les tests sans une planification préalable peut conduire à des tests chaotiques et inefficaces.

Les défauts logiciels peuvent entraîner des pertes financières, des retards, des atteintes à la réputation de l'entreprise, voire, dans des cas extrêmes, des blessures ou des décès. Il est donc essentiel d'avoir une stratégie de test appropriée en place. La stratégie de test est nécessaire pour informer le chef de projet de la manière dont l'équipe de test va planifier les cycles de test.

Quel critère ne peut pas être classé comme critère de sortie des tests?

Le critère qui ne peut pas être classé comme critère de sortie des tests est "Le coût."

Quelle information ne figure pas dans un rapport de problème?

L'information qui ne figure généralement pas dans un rapport de problème est "Le nombre d'erreurs."

Les tests du plus haut niveau d'indépendance sont faits par :

Les tests du plus haut niveau d'indépendance sont faits par "Une entité en dehors du sphère du projet."

Les Tests doivent s'arrêter quand :

Les Tests doivent s'arrêter quand "Tous les tests planifiés sont exécutés."

Parmi les propositions suivantes, laquelle ne fait pas partie des fonctions de la gestion des tests ?

Parmi les propositions, celle qui ne fait pas partie des fonctions de la gestion des tests est "Mise en place du budget et de l'intervalle de temps."

Quelles sont les informations généralement consignées dans un rapport d'incident?

Les informations généralement consignées dans un rapport d'incident comprennent :

Les informations permettant une évaluation de l'avancement du test et de la fin du test.

Les informations sur le nombre total d'erreurs susceptibles de survenir pendant l'ensemble du processus de test.

Les informations permettant la reproduction de l'erreur potentielle.

Quelles sont les exigences relatives à la gestion de la configuration devant être remplies du point de vue du test?

Les exigences relatives à la gestion de la configuration devant être remplies du point de vue du test incluent :

Gestion des versions, gestion de la configuration, suivi de l'état des erreurs et des modifications, audits de configuration.

Gestion de la configuration, prise en charge des outils de test, audits de configuration.

Gestion des versions, gestion de la configuration, planning d'intégration et gestion des mises à jour.