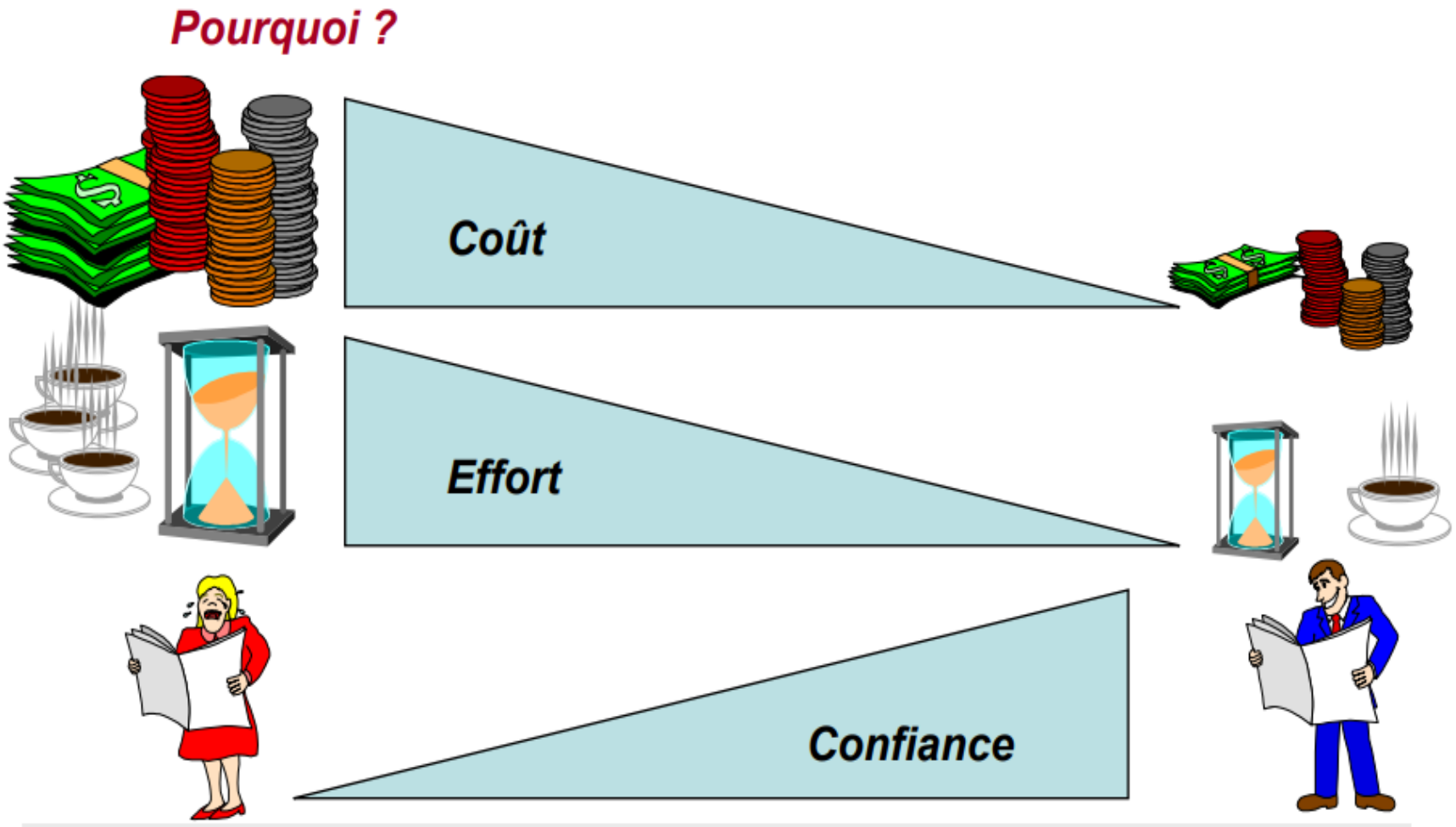
Chapitre I.
Introduction au test logiciel

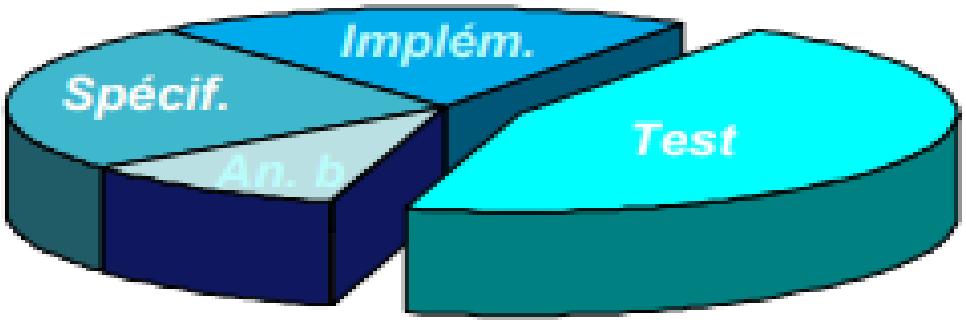
a.hettab@centre-univ-mila.dz

Introduction



Introduction

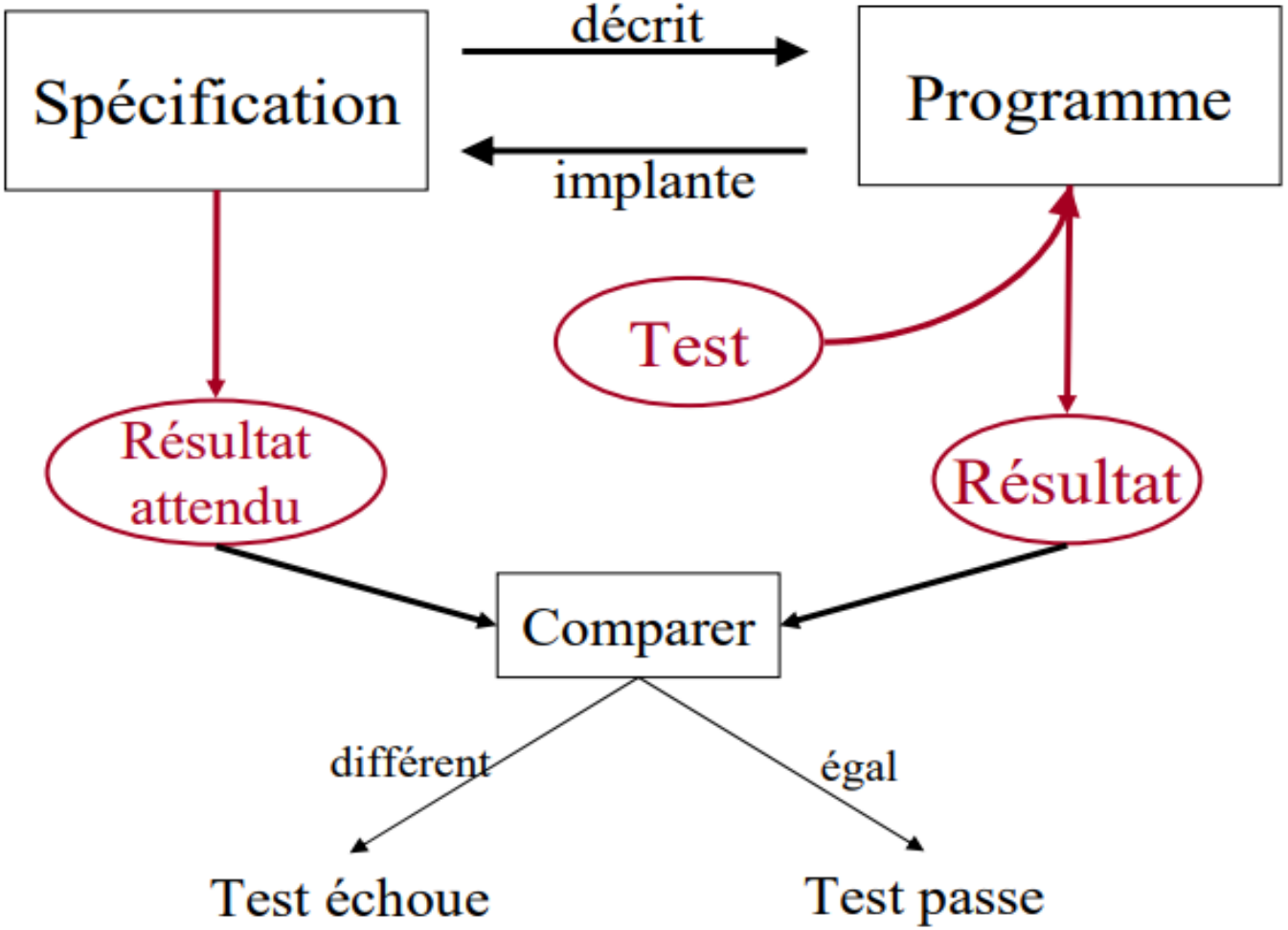
- Le test logiciel est une activité très importante dans le cycle de développement logiciel,
- Des tests logiciels efficaces sont la clé du développement de logiciels de qualité avec une plus grande satisfaction des utilisateurs, des coûts de maintenance extrêmement faibles, une validité (précision) et une fiabilité supérieures,
- Selon le type de projet logiciel, les activités de test consomment entre 40 % et 60 % du budget du pro



Introduction

- Le test logiciel est un processus ou une série de procédures conçues pour garantir que le code informatique exécute ce pour quoi il a été conçu et ne se comporte pas de manière inattendue.
- Le processus de test logiciel peut être effectué à l'aide de tests logiciels automatisés ou manuels. Pour répondre à ce besoin, plusieurs techniques de test ont été proposées, et un grand nombre d'outils d'automatisation des tests sont apparus. Le site de test spécialisé « http://www.stickyminds.com » référence des centaines d'outils d'automatisation des tests logiciels

Introduction



Termes et définitions

- Dans ce qui suit, nous présentons quelques notions de base, nécessaires pour aborder le domaine des tests logiciels.
- **Test:** Le test est un processus qui exécute un système ou un composant, en utilisant des moyens automatiques ou manuels, pour vérifier la satisfaction des spécifications ou pour s'assurer que les résultats effectifs (réels) correspondent à ceux attendus. IEEE (Standard Glossary of Software Engineering Terminology)

Termes et définitions

- **Vérification et validation:**

La vérification est la repense à la question suivante : **Faisons-nous le travail correctement ?** Elle utilise des preuves objectives pour la confirmation de la satisfaction des exigences spécifiques.

La validation est la repense à la question suivante : **Faisons-nous le travail attendu ?** Elle est une opération destinée à démontrer que les exigences déterminées sont respectées pour une utilisation spécifique escomptée.

Termes et définitions

- **Test statique et test dynamique**
  - **Test statique**

Les méthodes de tests statiques consistent en l'analyse textuelle du code du logiciel afin d'y détecter des erreurs, **sans exécution du programme**.

Les méthodes utilisées sont : **revue de code**, **analyse des types**, **analyse du domaine des variables, ...**

Les revues de code permettent **l'examen détaillé** d'une **spécification**, d'une **conception** ou d'une **implémentation** par une personne ou un groupe de personnes (lecture croisée), afin de **détecter** des **fautes**, des **violations de normes** de développement ou d'autres problèmes.

Termes et définitions

- **Test statique et test dynamique**
  - **Test statique**

Avantages :

Méthodes **efficaces et peu coûteuses** (60 à 95% des erreurs sont détectées lors de contrôles statiques). Donc, les méthodes de tests statiques sont nécessaires.

Inconvénients :

Méthodes **ne permettant pas de valider** le comportement d'un programme **au cours de son exécution**. Donc, les méthodes de tests statiques ne sont pas suffisantes.

Termes et définitions

- **Test statique et test dynamique**
  - **Test dynamique**

Les méthodes de tests dynamiques consistent en **l'exécution du programme** et le **valider** à l'aide d'un **jeu de tests**.

Elles visent à **détecter des erreurs** en confrontant les **résultats obtenus** à ceux **attendus** par **la spécification**.

Termes et définitions

- **Plan de test, scénario de test, cas de test, suite de test, script de test, et données de test:**

Un plan de test est un document qui décrit la stratégie globale de test d'un projet. Il définit l'objectif des tests, les fonctionnalités à tester, les outils utilisés, les ressources, le calendrier et les critères d'acceptation**.**

Un scénario de test permet de contrôler **la séquence des étapes prévues** de façon **formelle ou informelle** dans les **spécifications fonctionnelles.**

Un cas de test permet de **concrétiser** un **scénario de test,** sur des **données concrètes,** pour **atteindre un objectif de test,** de manière à mettre en œuvre un **chemin de programme particulier.**

Termes et définitions

- **Plan de test, scénario de test, cas de test, suite de test, script de test, et données de test (suite):**

Une suite de tests est un ensemble de **plusieurs cas de tests** destinés à être **utilisés** pour **tester un logiciel** afin de démontrer qu'il **possède un ensemble de comportements spécifiés.**

Un script de test est un **ensemble d'instructions** qui seront **exécutées** sur le système sous test pour **vérifier** que le système **fonctionne comme prévu.**

Données de test Cela concerne les **variables** et leurs **valeurs** dans le **cas de test.** Dans l'exemple d'une connexion par e-mail, il s'agirait du nom d'utilisateur et du mot de passe du compte.

Termes et définitions

- **Plan de test, scénario de test, cas de test, suite de test, script de test, et données de test (suite):**
- **Exemple:** Voici un exemple de deux cas de test

Test Case id	Test Scenario	Test Cases	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
NC1	Verify Name Field	Customer Name cannot be empty	1) Logs in 2) Click on the "New Custmer" button 3) Do not enter a value in Customer Name Field 4) Press TAB and move to next Field		An error message "Custmer name must not be blank" must shown	An error message "Custmer name must not be blank" shown	Pass
NC2	Verify Name Field	Customer name cannot have Special characters	1) Logs in 2) Click on the "New Custmer" button 3) Enter a value contains special characters in Customer Name Field	Adel@G &$"'	An error message "Special characters are not allowed" must shown	No action	Fail

Termes et définitions

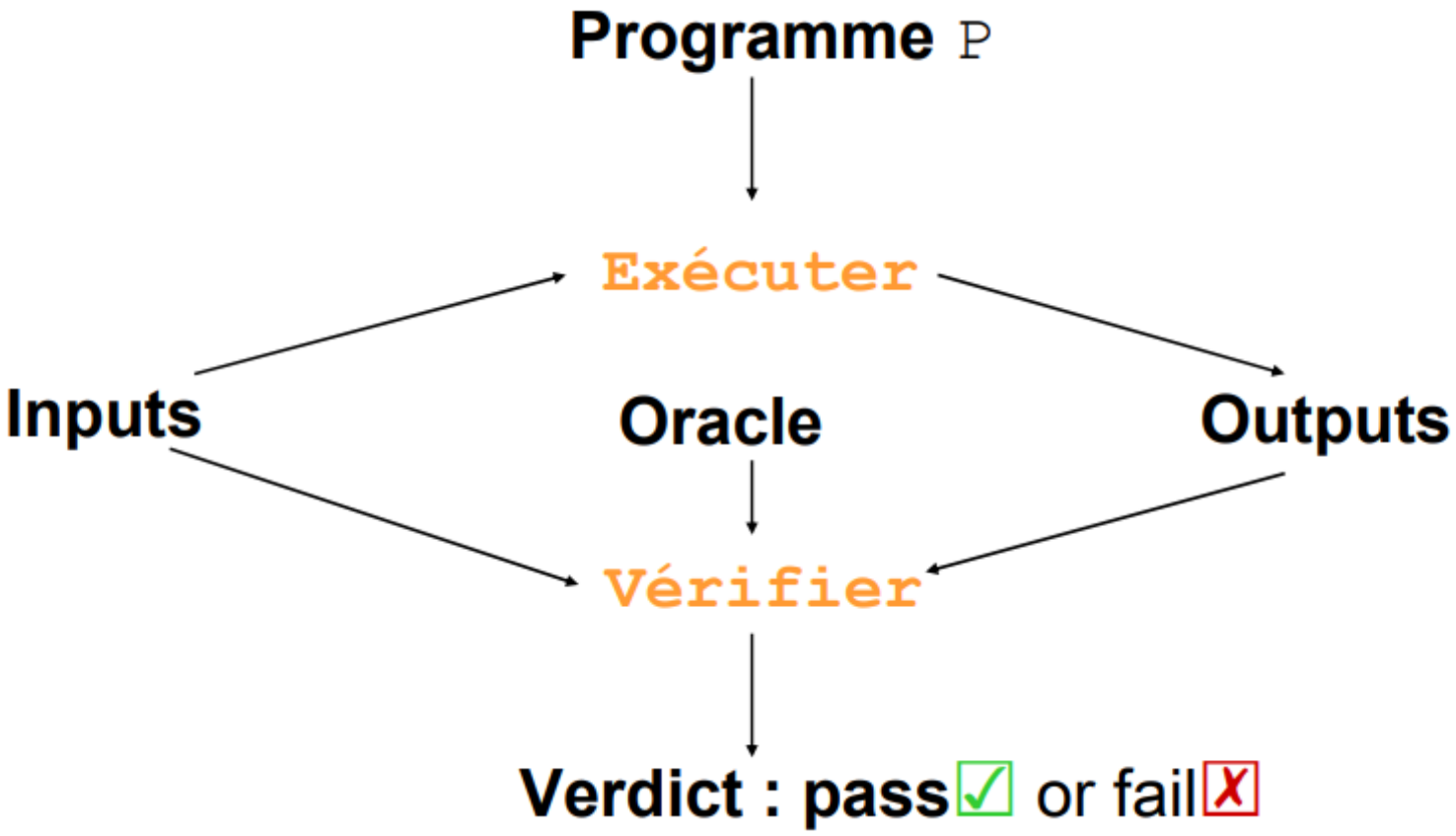
- **Résultat attendu et l'oracle de test:**

Un résultat attendu est un **comportement** fourni par les **spécifications** d'un système ou d'un composant selon des **conditions spécifiées.**

Un oracle est un mécanisme permettant de **déterminer si** le programme **a réussi** ou **a échoué** un **test.** Ceci est fait en **comparant** les **résultats obtenus** avec les **résultats attendus.**

Termes et définitions

- **Résultat attendu et l'oracle de test (suite)**



Termes et définitions

- **Erreur, Défaut et défaillance**

L'erreur Action humaine produisant des défauts.

Les défauts sont des imperfections dans le logiciel. Ces défauts peuvent engendrer des défaillances mais ce n'est pas nécessaire.

Les défaillances sont des événements au cours desquels le logiciel ne fait pas ce qu'il devrait. Pour simplifier, les défaillances sont ce que nous appelons couramment **Bug** ou **Anomalie**.

Termes et définitions

- **Priorité:**

La priorité se réfère à l'ordre dans lequel un défaut doit être corrigé. Plus la priorité est élevée, plus le défaut nécessite une résolution rapide.

Types de priorités: Les priorités des bugs ou défauts peuvent être réparties en trois catégories **:**

1. **Faible :** Le défaut est gênant, mais sa correction peut être reportée après la résolution des problèmes plus critiques.
2. **Moyenne :** Le défaut doit être traité dans le cadre normal du développement, mais peut attendre la prochaine version du logiciel.
3. **Élevée :** Le défaut doit être corrigé en urgence, car il perturbe gravement le système, le rendant inutilisable tant qu'il n'est pas résolu.

Termes et définitions

- **Gravité (Sévérité):**

La gravité des bogues, ou des défauts, en phase de tests représente l'ampleur de l'impact d'un bug sur l'application testée. Plus le bug affecte la fonctionnalité du système, plus son niveau de gravité est élevé.

Types de gravité :

Lors des tests logiciels, la gravité d'un bug ou d'un défaut peut être classée selon les catégories suivantes :

1. **Critique :** Le défaut provoque un blocage complet du processus, empêchant toute activité de continuer.
2. **Majeure :** Ce défaut est très grave, provoquant un effondrement du système, bien que certaines parties restent encore fonctionnelles.
3. **Moyenne :** Le défaut cause un comportement indésirable, mais le système demeure globalement opérationnel.
4. **Mineure :** Ce défaut n'entraîne pas de panne significative du système et a un impact limité.

Termes et définitions

- **Différence entre gravité et priorité dans les tests:**

priorité	gravité
La priorité du défaut détermine l'ordre dans lequel le développeur doit corriger un défaut.	La gravité d'un défaut désigne le niveau d'impact qu'un défaut exerce sur le fonctionnement du produit.
La priorité est liée à la planification, déterminant l'ordre dans lequel les tâches ou défauts doivent être traités.	La gravité est liée à la fonctionnalité ou aux normes, indiquant à quel point un défaut affecte le bon fonctionnement ou la conformité du produit.
La priorité des défauts est déterminée en concertation avec le gestionnaire ou le client, en tenant compte de l'impact sur le projet et des besoins opérationnels.	L'ingénieur QA évalue et détermine le niveau de gravité du défaut.
La priorité indique le délai dans lequel le bug doit être corrigé, en fonction de son urgence et de son impact sur le projet.	La gravité indique l'importance du défaut en termes d'impact sur la fonctionnalité du produit.

Termes et définitions

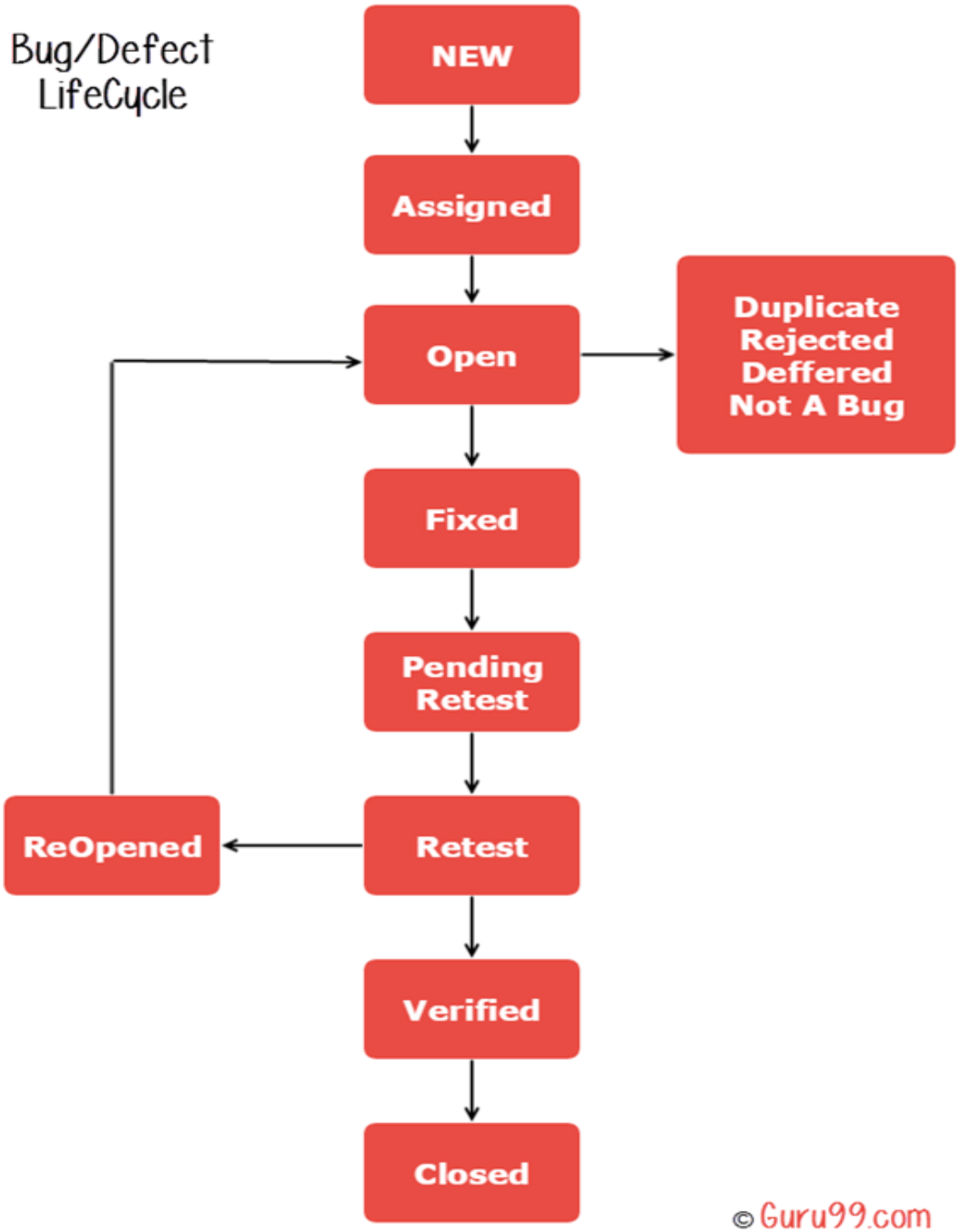
- **Différence entre gravité et priorité dans les tests (suite):**

priorité	gravité
Sa valeur est subjective et peut changer au fil du temps en fonction de l'évolution de la situation du projet.	Sa valeur est objective et moins susceptible de changer.
La priorité est influencée par la valeur commerciale, reflétant l'importance du défaut par rapport aux objectifs d'affaires.	La gravité est déterminée par l'impact sur la fonctionnalité.
Lors de l'UAT(User Acceptance Testing), l'équipe de développement corrige les défauts en se basant sur leur priorité.	Lors du SIT(System Integration Testing), l'équipe de développement corrigera les défauts en tenant compte d'abord de leur gravité, puis de leur priorité.
Un statut de priorité élevée et de gravité faible signifie que le défaut doit être corrigé rapidement, même s'il n'affecte pas l'application de manière significative.	Une gravité élevée avec un statut de priorité faible signifie que le défaut doit être corrigé, mais qu'une intervention immédiate n'est pas nécessaire.
Le statut de priorité est déterminé en fonction des exigences du client.	L'état de gravité est fondé sur les aspects techniques du produit.

Termes et définitions

- **Cycle de vie des défauts/bogues dans un test logiciel:**
- Le cycle de vie des défauts, ou cycle de vie des bogues, dans les tests logiciels désigne l'ensemble des états par lesquels un défaut ou un bogue passe au cours de son existence. L'objectif de ce cycle est de faciliter la coordination et la communication de l'état actuel du défaut aux différentes parties prenantes, tout en rendant le processus de correction systématique et efficace.
- **Le statut de défaut**, ou statut du bogue dans le cycle de vie du défaut, représente l'état actuel d'un défaut ou d'un bogue à un moment donné. L'objectif de ce statut est de transmettre avec précision l'état ou la progression d'un défaut, permettant ainsi de mieux suivre et comprendre l'évolution réelle du cycle de vie du défaut.

Terme


Bug/Defect
LifeCycle
NEW
Assigned
Open
Duplicate
Rejected
Deffered
Not A Bug
Fixed
Pending
Retest
ReOpened
Retest
Verified
Closed
© Guru99.com

Termes et définitions

Flux de travail des états de défauts

Le nombre d'états traversés par un défaut peut varier d'un projet à l'autre. Le diagramme de cycle de vie précédemment couvre les états possibles :

1.Nouveau (New) : Lorsqu'un nouveau défaut est enregistré pour la première fois, il se voit attribuer le statut "Nouveau".

2.Attribué (Assigned) : Après que le testeur a publié le bogue, le responsable du test le valide et l'assigne à l'équipe de développement.

3.Ouvert (Open) : À ce stade, le développeur commence l'analyse et travaille sur la correction du défaut.

4.Corrigé (Fixed) : Lorsque le développeur apporte les modifications nécessaires et les vérifie, le statut du bogue est alors changé en "Corrigé".

5.En attente de nouveau test (Panding retest): Une fois le défaut corrigé, le développeur fournit un code spécifique pour le retesting. Le statut est alors "En attente de nouveau test", en attendant l'action des testeurs.

6.Retester (Retest) : Le testeur effectue un nouveau test pour vérifier si le défaut a bien été corrigé, et change le statut en "Retested".

Termes et définitions

Flux de travail des états de défauts (suite)

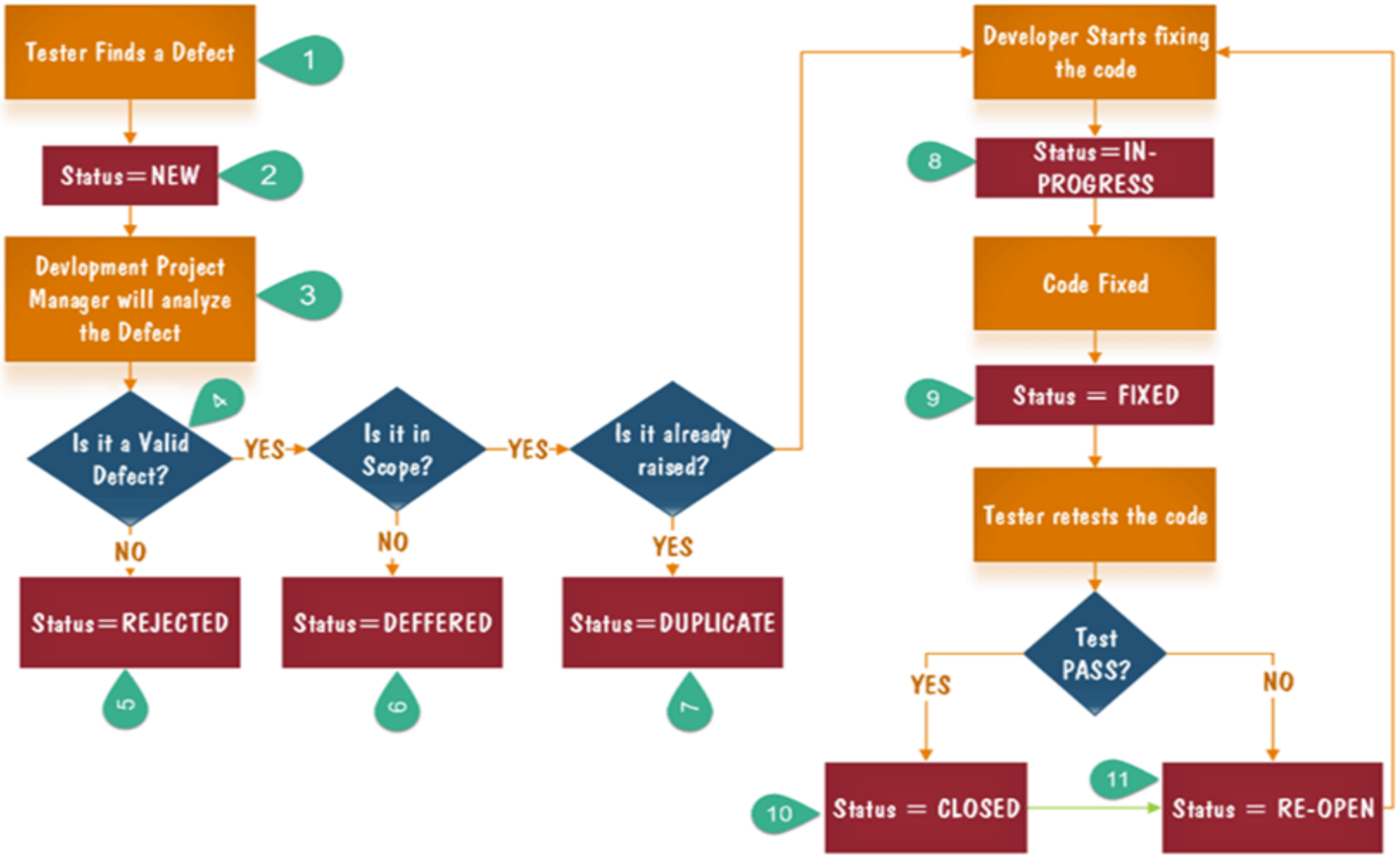
7. **Vérifié (Verified)** : Le testeur reteste le bogue après sa correction par le développeur. Si aucun problème n'est détecté, le bug est considéré comme corrigé et son statut est changé en "Vérifié".
8. **Réouvert (Reopened)**: Si le bogue persiste malgré la correction, le testeur change le statut en "Réouvert", et le bogue recommence alors son cycle de vie.
9. **Fermé (close):** Si le bogue n'existe plus, le testeur attribue le statut "Fermé".
10. **Dupliquer (Dublicate) :** Si le défaut est identique ou similaire à un autre déjà signalé, le statut est modifié en "Duplicate".
11. **Rejeté (Rejected) :** Si le développeur juge que le défaut n'est pas un véritable problème, il attribue le statut "Rejeté".
12. **Reportés (Deffered)** : Si le bogue n'est pas prioritaire et qu'il est prévu d'être corrigé dans une future version, il reçoit le statut "Différé".
13. **Pas un défaut (Not a bug):** Si le problème signalé n'affecte pas la fonctionnalité de l'application, le statut attribué est "Pas un bug".

Termes et définitions

Le diagramme en dessous illustre les étapes du cycle de vie d'un bogue:

1. **Identification du défaut** : Le testeur découvre un défaut et lui attribue le statut "Nouveau".
2. **Transmission au chef de projet** : Le défaut est soumis au chef de projet pour l'analyse.
3. **Validation du défaut** : Le chef de projet détermine si le défaut est valide. Si le défaut n'est pas valide, son statut devient "Rejeté".
4. **Vérification de l'applicabilité** : Si le défaut est valide, le chef de projet vérifie s'il fait partie du champ d'application. Par exemple, si un problème survient avec une fonctionnalité d'email qui ne fait pas partie de la version actuelle, le défaut est alors classé comme "Reporté" ou "Différé".
5. **Recherche de duplicata** : Le gestionnaire examine si un défaut similaire a été signalé auparavant. Si c'est le cas, le statut est changé en "Dupliqué".
6. **Attribution au développeur** : Si le défaut est unique, il est attribué à un développeur, qui commence à le corriger. À ce stade, le statut est "En cours".
7. **Correction** : Une fois le code corrigé, le défaut reçoit le statut "Corrigé".
8. **Retester** : Le testeur effectue un nouveau test du code. Si les cas de test passent, le statut devient "Fermé". Si les cas de test échouent à nouveau, le défaut est rouvert et réattribué au développeur

Termes et définitions



Termes et définitions

- **Rapport d'un défaut:**
- Un **Rapport d'un défaut (bug report**) est un document détaillé qui décrit l'état d'un défaut dans une application. Il sert à signaler le comportement inattendu d'un défaut afin que les développeurs puissent l'analyser et le corriger. Il contient les informations suivantes: un id, le testeur responsable, version, test case id, les étapes à suivre, résultat attendu, résultat obtenu, priorité et sévérité.

Termes et définitions

- **Rapport d'un défaut (suite):**
- **Exemple**: voici un exemple d'un rapport d'un défaut pour le cas de test de l 'exemple précédent.

Defect #	Status	Raised By	Version	TestCase Id	Steps to Reproduce	Expected Results	Actual Results	Priority	Severity
1	Open	Mohamed	v1	NC2	1) Logs in 2) Click on the "New Custmer" button 3) Enter a value contains special characters in Customer Name Field	An error message "Special characters are not allowed" must shown	No action	Medium	Medium

Types de tests

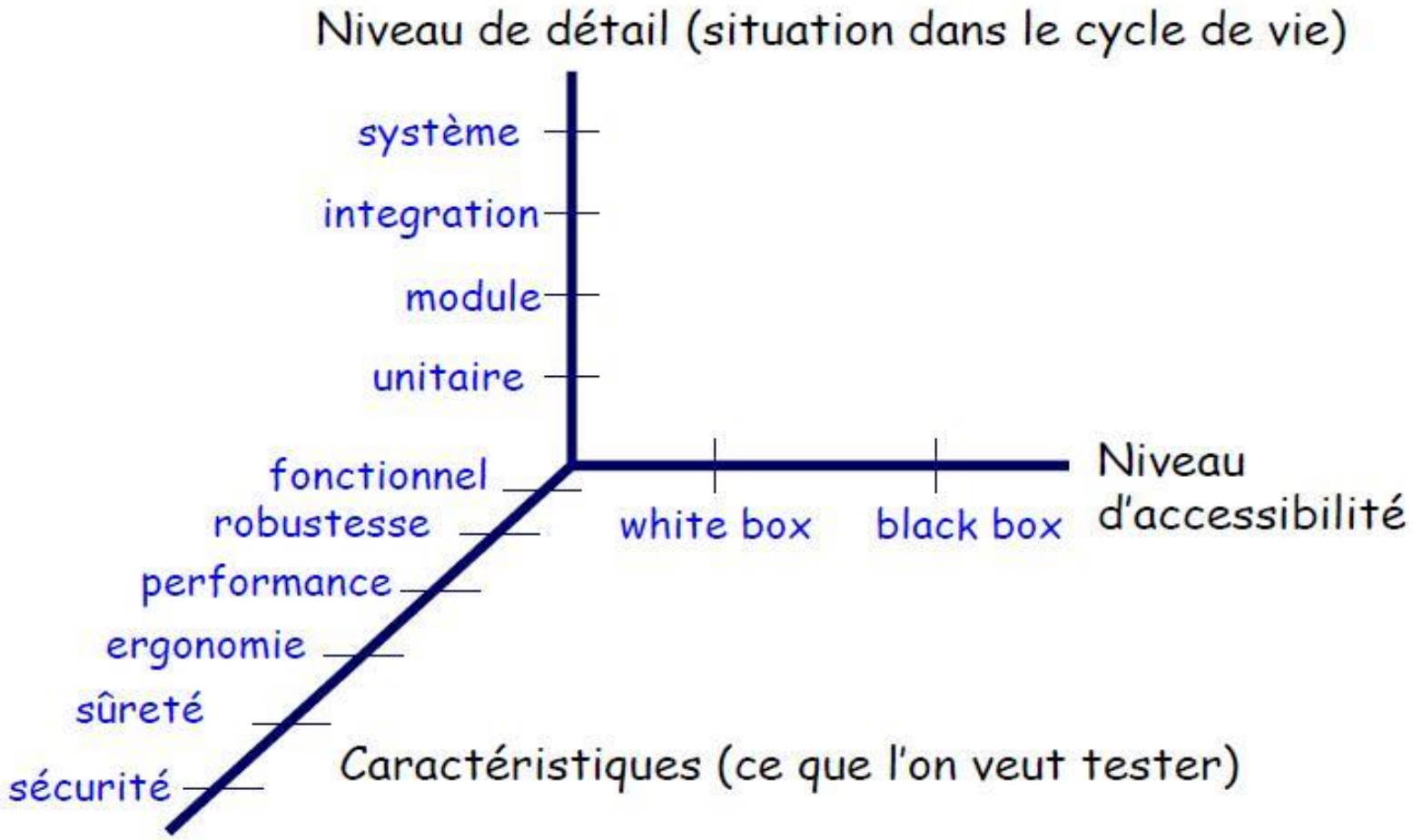
- Dans la littérature, les tests logiciels sont classés selon:

  - le niveau d'accessibilité,

  - le niveau de détail (situation dans le cycle de vie)

  - les caractéristiques (ce que l'on veut tester)

Types de tests



Types de tests

Classification des tests selon le niveau d'accessibilité

- Les techniques de test sont classées dans deux grandes familles:

  - **Les tests boîte noire (ou les tests fonctionnels)**

  - **les tests boîte blanche (ou tests structuraux)**.

Remarque: Il y a une troisième classe qui combine les deux types de test en boîte noire et en boîte blanche qu'il s'appelle **boite grise**.

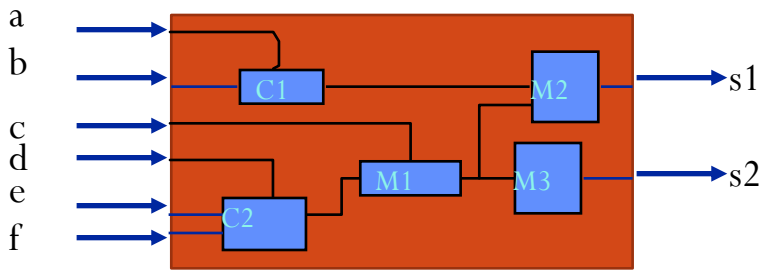
Elle mélange à la fois les fonctionnalités et le fonctionnement d'un système

Types de tests

Classification des tests selon le niveau d'accessibilité

Test fonctionnel

test structurel



Types de tests

Classification des tests selon le niveau d'accessibilité

- **Test boîte noire (test fonctionnel)**

- **Les tests boîte noire (tests fonctionnels)** sont utilisés pour vérifier **la conformité fonctionnelle** entre l'application développée et **les spécifications initiales** du système. La **structure** et le **fonctionnement interne** des applications ne sont pas étudiés par les tests fonctionnels, qui ne servent qu'à en examiner les fonctionnalités.

Types de tests

Classification des tests selon le niveau d'accessibilité

- **Test boîte noire (test fonctionnel)(suite)**

- Cette méthode sert à valider **le cahier des charges** avec l'application développée. Pour cela, les testeurs recherchent **des erreurs d'interface**, de **fonctions manquantes ou incorrectes**, **d'erreurs d'accès aux bases de données externes** ou dans **les structures de données**, etc.

Types de tests

Classification des tests selon le niveau d'accessibilité

- **Test boîte noire (test fonctionnel)(suite)**

- Les tests fonctionnels consistent à suivre **des scénarios** qui sont tracés sur les différents **chemins utilisateur** possibles sur le système sous test.

Toutes les **fonctionnalités** du système doivent être considérées dans ces **scénarios**. Pour cela, chaque **scénario** est utilisé pour vérifier certaines **fonctionnalités** en comparant **les sorties obtenues** avec les **sorties attendues**. Ainsi, les entrées valides sont acceptées, et les entrées non valides sont refusées.

Types de tests

Classification des tests selon le niveau d'accessibilité

- **Test boîte noire (test fonctionnel)(suite)**

- Avantages: Les méthodes de tests en boîte noire ont de nombreux avantages, entre autres;

- la simplicité,

- la rapidité,

- Faciliter le contrôle visuel.

- **Limites**: elles sont également leurs limites telles que:

- la superficialité

- la redondance

Types de tests

Classification des tests selon le niveau d'accessibilité

- **Test boîte blanche (tests structurel)**

Les tests boîte blanche (tests structuraux) consistent à examiner **le fonctionnement** d'un système plutôt que **ses fonctionnalités**.

Ils sont utilisés pour testés l'ensemble **des composants internes** du logiciel, dont les données de test sont produites à partir d'une analyse du **code source**. L'analyse du code et la dérivation de jeu de test se fait souvent à l'aide de **graphes de flot de contrôle (control flow graph)**, qui est utilisé comme une représentation graphique de tous les chemins pouvant être suivis par un programme lors de son exécution.

Types de tests

Classification des tests selon le niveau d'accessibilité

- **Test boîte blanche (tests structurel)(suite)**

Les tests en boîte blanche sont souvent utilisés durant **la phase d'implémentation** d'une application. Ils peuvent être utilisés aussi dans les autres phases de développement d'un projet. Cette méthode est utilisée principalement pour **les tests unitaires**, et elle est appliquée également sur les tests **d'intégration** et rarement sur les tests **système**.

Types de tests

Classification des tests selon le niveau d'accessibilité

- **Test boîte blanche (tests structurel)(suite)**

Dans cette méthode, **des scénarios de test**, qui sont dérivés à partir du **graphe de flot de contrôle** selon **des critères de couverture**, sont appliqués pour vérifier l'ensemble du **code source**. Sur la base de ces **critères de couverture**, des métriques peuvent être calculées pour déterminer dans quelle mesure le test a stimulé le système. Il existe plusieurs types de **critères de couverture**, dont les principaux sont : **tous les nœuds**, **tous les arcs**, et **tous les chemins**.

Types de tests

Classification des tests selon le niveau d'accessibilité

- **Test boîte blanche (tests structurel)(suite)**

- Avantages: Les méthodes de tests en boîte blanches ont de nombreux avantages, entre autres;

- Test complet.

- Prise en charge des tests automatisés.

- Optimisation du code par la suppression du code inutile.

- **Limites**: elles sont également leurs limites telles que:

- Le test en boîte blanche est souvent long, complexe et onéreux.

- Les testeurs doivent avoir une connaissance interne des logiciels.

Types de tests

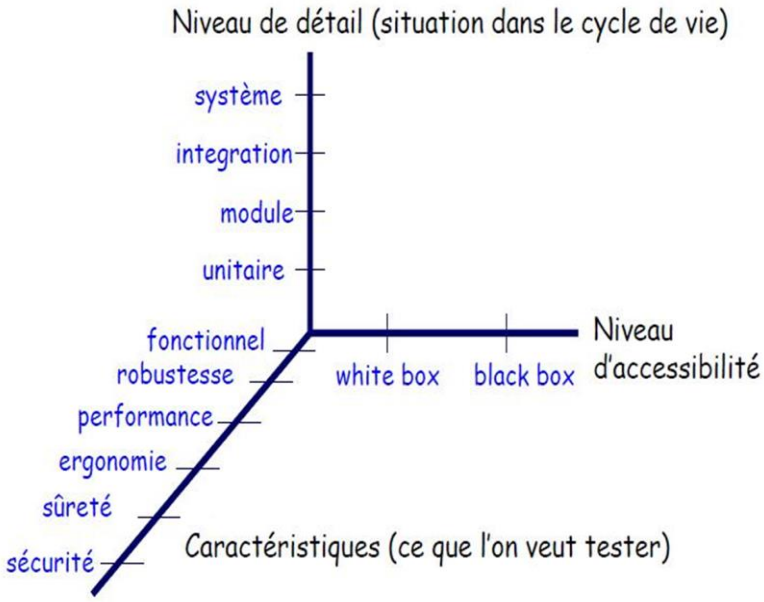
Classification des tests selon le niveau de détail

(situation dans le cycle de vie)

Les tests doivent être effectués **aux différents niveaux** du processus de développement d'un logiciel.

les tests du logiciel sont décomposés en :

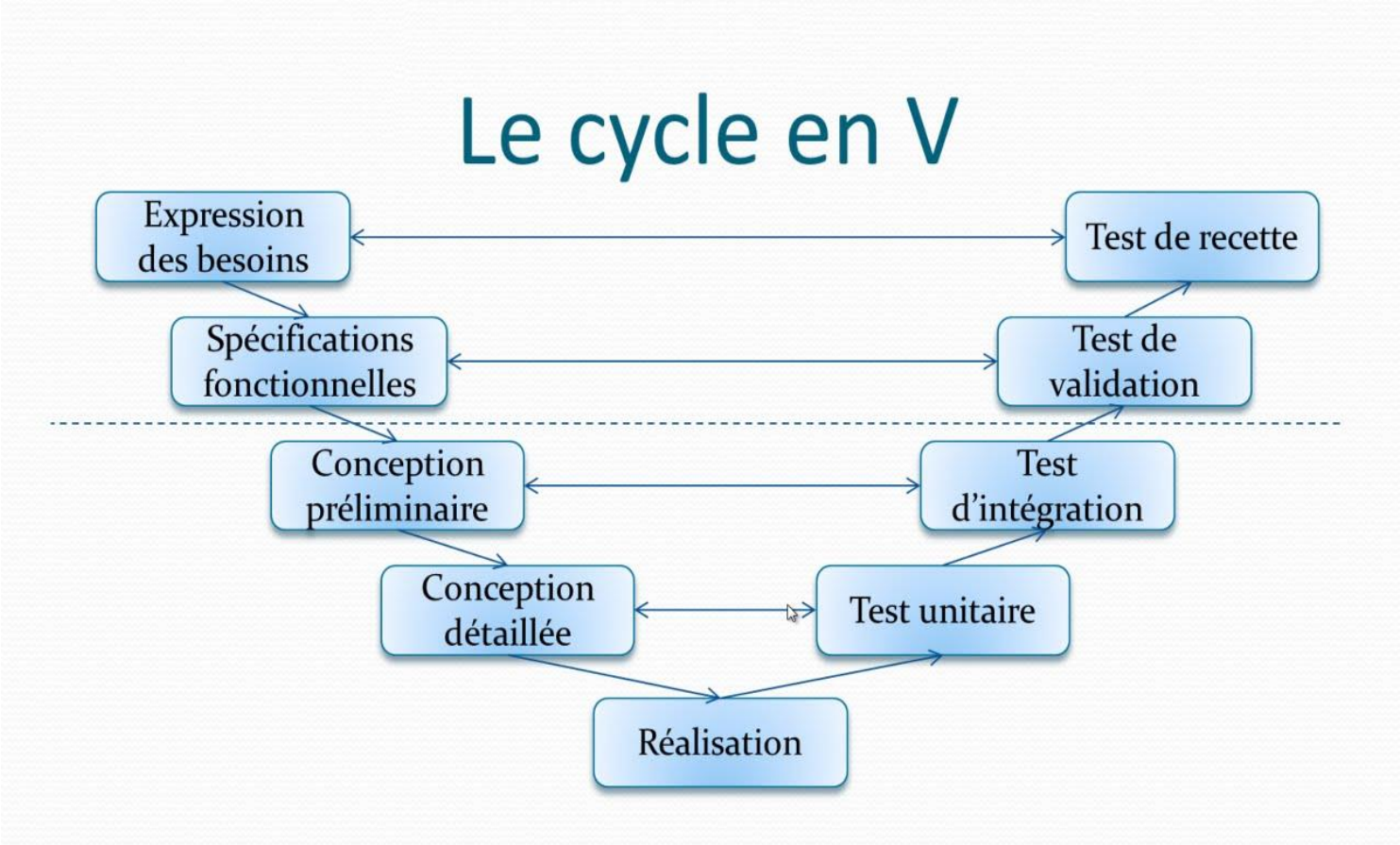
- **tests unitaires,**
- **tests d'intégration,**
- **test système**
- **test d'acceptation(recette)**



Types de tests

Classification des tests selon le niveau de détail

(situation dans le cycle de vie)



Types de tests

Classification des tests selon le niveau de détail

(situation dans le cycle de vie)

- **tests unitaires**

Le test unitaire consiste est utilisé pour tester **les unités** et **les composants** d'un logiciel **individuellement**. Il vise à valider **la qualité du code** et les performances d'un module en vérifiant que chaque unité du logiciel fonctionne comme prévu. **L'unité** qui est considérée comme **la plus petite partie testable du logiciel**, comporte généralement une ou plusieurs entrées et une seule sortie.

Types de tests

Classification des tests selon le niveau de détail

(situation dans le cycle de vie)

- **tests unitaires (suite)**

Dans ce type de test, chaque **module** est testé **indépendamment** du reste du programme, pour assurer qu'il fonctionne correctement en toutes circonstances. La méthode de **la couverture de code (couverture structurelle)** est largement utilisée pour assurer que l'ensemble des tests conduit à exécuter l'ensemble des instructions présentes dans le code.

Types de tests

Classification des tests selon le niveau de détail

(situation dans le cycle de vie)

- **tests unitaires (suite)**

L'exécution d'un test unitaire passe généralement par les étapes suivantes:

1) l'initialisation qui sert à définir un environnement de test totalement reproductible.

2) L'exécution qui consiste à exécuter le module à tester.

3) L'évaluation qui consiste à définir le succès ou l'échec de test en comparant les résultats obtenus avec les résultats définis.

4) La désactivation qui sert à désinstaller l'environnement de test installé dans la première étape pour retrouver l'état initial du système.

Types de tests

Classification des tests selon le niveau de détail

(situation dans le cycle de vie)

- **Test d'intégration**

Le **test d'intégration** est la phase du test logiciel dont des modules logiciels individuels **sont combinés et testés** en tant que **groupe**.

Les tests d'intégration se produisent **après** les **tests unitaires** et **avant** les **tests system**.

Ils **prennent** les modules d'entrée qui ont été testés par les **tests unitaire**, puis les **regroupent**, et ensuite ils **appliquent les tests** définis dans un **plan de test d'intégration** à ces groupes des unités, et finalement ils **fournissent** en sortie les résultats de test.

Types de tests

Classification des tests selon le niveau de détail

(situation dans le cycle de vie)

- **Test d'intégration (suite)**

Il existe **plusieurs types** de tests d'intégration dont les plus célèbres sont:

- **le test descendant (top-down)**
- **le test ascendant (bottom-up)**
- **le test mixtes (en sandwich)**
- **le big-bang**

Il existe également **d'autres types** de test d'intégration comme: **l'intégration de la couche, l'intégration de la collaboration**, **l'intégration de haute fréquence**, **l'intégration client-serveur**, et **l'intégration des services distribués**.

Types de tests

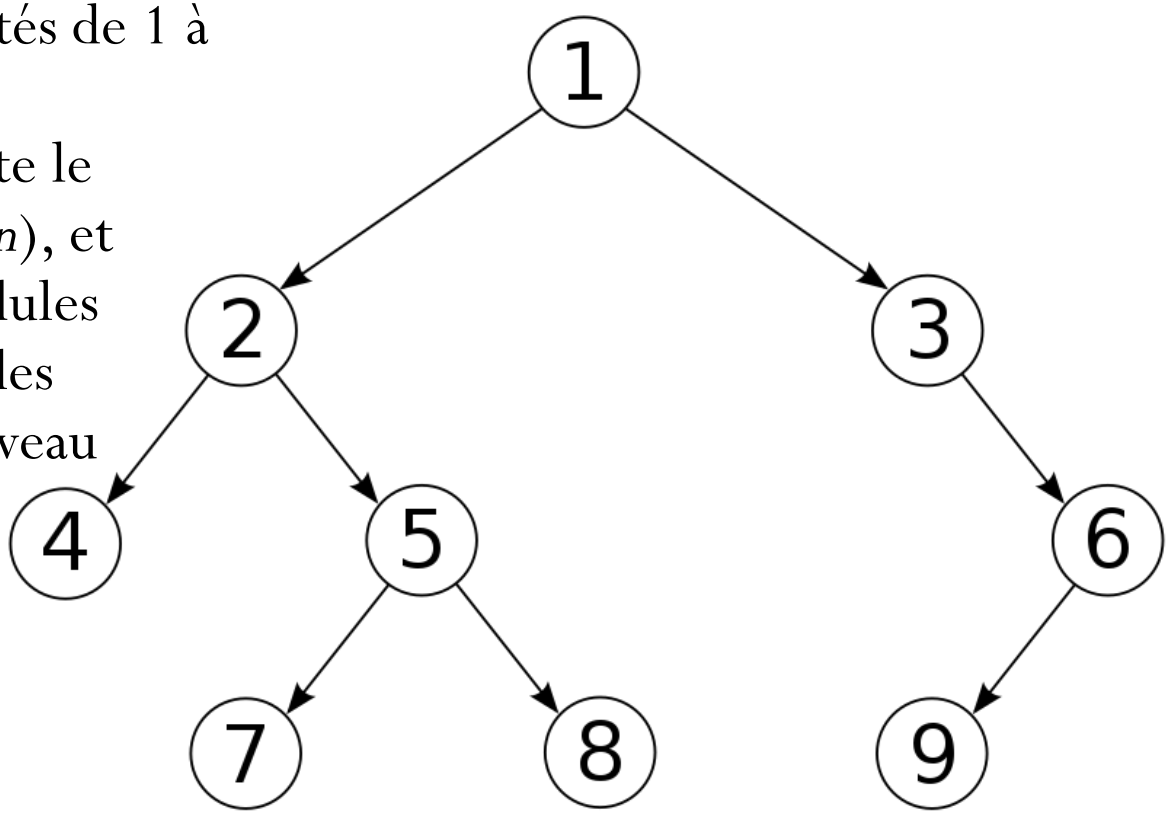
Classification des tests selon le niveau de détail

(situation dans le cycle de vie)

- **Test d'intégration (suite)**

Voici un exemple composé de neuf modules numérotés de 1 à 9.

Le module 1 représente le module principal (*main*), et chacun des autres modules appelle les méthodes des modules situés à un niveau inférieur.



Types de tests

Classification des tests selon le niveau de détail

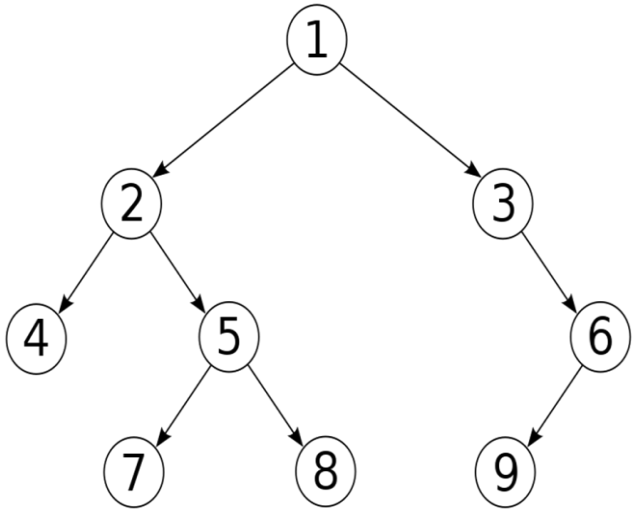
(situation dans le cycle de vie)

- **Test d'intégration (suite)**

Le test descendant (Top-down) est une approche de test intégrée dans laquelle les modules intégrés supérieurs sont testés en premier lieu, puis on test les modules en dessous étape par étape jusqu'à la fin du module correspondant.

On obtient les tests suivants:

- **première étape**
  - 1
- **seconde étape**
  - 1, 2 et 3
- **troisième étape**
  - 1, 2, 3, 4, 5 et 6
- **quatrième étape**
  - 1, 2, 3, 4, 5, 6, 7, 8 et 9



Types de tests

Classification des tests selon le niveau de détail

(situation dans le cycle de vie)

- **Test d'intégration (suite)**

Le test ascendant (bottom-up), est une approche de test intégrée dont les composants de niveau le plus bas sont d'abord testés et utilisés pour faciliter le test des composants de niveau supérieur. Ce processus est répété jusqu'à ce que le composant situé en haut de la hiérarchie soit testé. Tous les modules, procédures ou fonctions du bas niveau sont intégrés puis testés. Après les tests d'intégration des modules intégrés de niveau inférieur, le niveau suivant de modules sera formé et pourra être utilisé pour les tests d'intégration. Cette approche n'est utile que lorsque tous ou la plupart des modules du même niveau de développement sont prêts. Cette méthode permet également de déterminer les niveaux de logiciels développés et facilite le compte-rendu des progrès des tests sous forme de pourcentage.

Types de tests

Classification des tests selon le niveau de détail

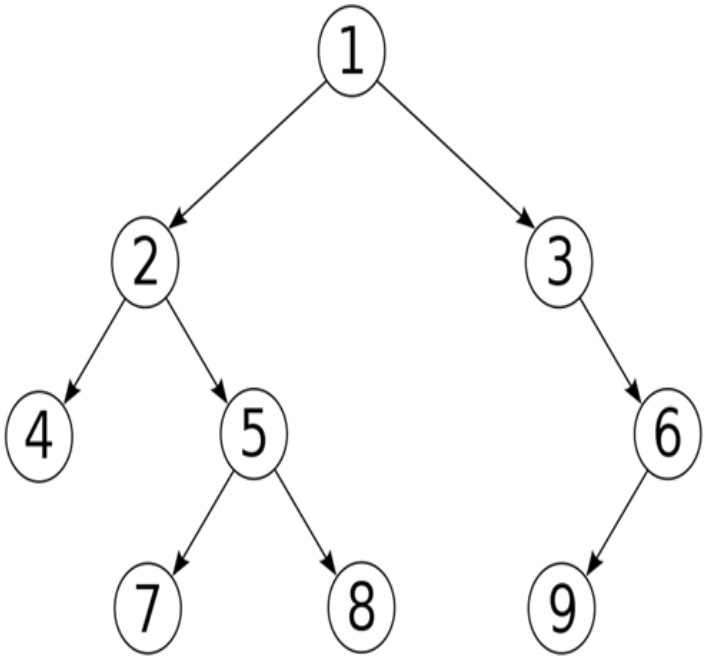
(situation dans le cycle de vie)

- **Test d'intégration (suite)**

Le test ascendant (bottom-up) (suite)

On obtient les tests suivants:

- **première étape**
- 7
- 8
- 9
- **seconde étape**
- 4
- 5, 7 et 8
- 6 et 9
- **troisième étape**
- 2, 4, 5, 7 et 8
- 3, 6 et 9
- **quatrième étape**
- 1, 2, 3, 4, 5, 6, 7, 8 et 9



Types de tests

Classification des tests selon le niveau de détail

(situation dans le cycle de vie)

- **Test d'intégration (suite)**

Le test sandwich est une approche permettant de combiner un test descendant avec un test ascendant. Le système est vu comme ayant 3 niveaux :

- **Niveau logique (haut) ;**
- **Niveau cible (milieu) ;**
- **Niveau opérationnel (bas).**

Les tests sont effectués dans les deux directions (haut et bas) et se rassemblent au milieu (couche cible).

Types de tests

Classification des tests selon le niveau de détail

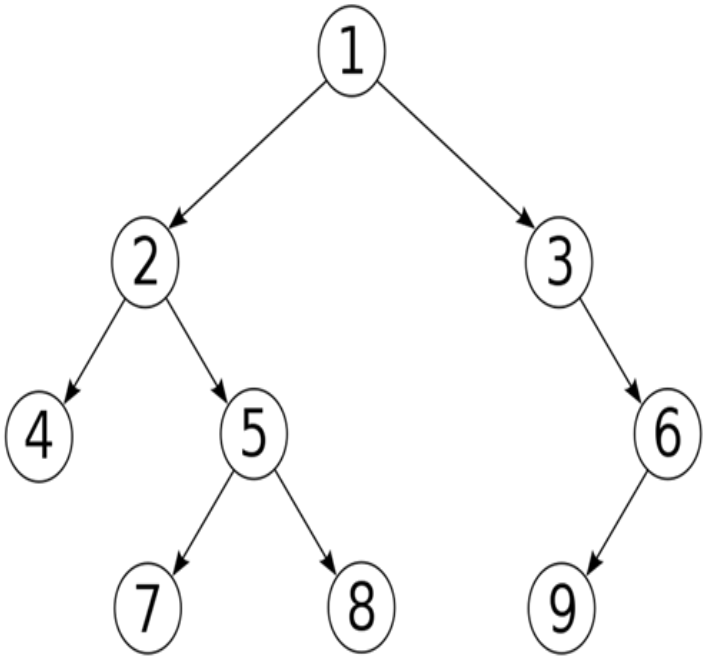
(situation dans le cycle de vie)

- **Test d'intégration (suite)**

Le test sandwich (suite)

On obtient les tests suivants:

- **première étape**
  - 1
  - 7 et 8
  - 9
- **seconde étape**
  - 1, 2 et 3
  - 5, 7 et 8
  - 6 et 9
  - 4
- **troisième étape**
  - 1, 2, 3, 4, 5 et 6
  - 2, 4, 5, 7 et 8
  - 3, 6 et 9
- **quatrième étape**
  - 1, 2, 3, 4, 5, 6, 7, 8 et 9



Types de tests

Classification des tests selon le niveau de détail

(situation dans le cycle de vie)

- **Test d'intégration (suite)**

- **L'approche de test big-bang**, est une approche de test intégrée dans laquelle la plupart des modules développés sont couplés pour former un système logiciel complet ou une partie importante du système, puis ce système sera testé en utilisant des tests d'intégration. Cette méthode est très efficace pour gagner du temps dans le processus de test d'intégration. Toutefois, si les scénarios de test et leurs résultats ne sont pas enregistrés correctement, le processus d'intégration dans son ensemble sera plus compliqué et empêchera l'équipe de test d'atteindre l'objectif de test d'intégration.

Types de tests

Classification des tests selon le niveau de détail

(situation dans le cycle de vie)

- **Test système**

Après **l'intégration** de tous les composants logiciels, ayant réussi les tests d'intégration, **les tests système** les prennent comme entrée.

Les tests du système ont pour but de **garantir** que le logiciel installé dans le matériel est conforme aux **spécifications fonctionnelles**, notamment en vérifiant **les interfaces** entre le matériel et le logiciel, les **fonctions générales**, **l'utilisation et l'allocation des ressources**, **les performances**, **le fonctionnement en temps réel**…etc.

Types de tests

Classification des tests selon le niveau de détail

(situation dans le cycle de vie)

- **Test système (suite)**

Les méthodes de tests de **boîte noire** sont largement utilisées pour effectuer **les tests système**, qui ne nécessitent aucune connaissance du code ou de la logique interne.

Remarque: Dans cette décomposition du test, en fonction du cycle de vie, il existe également **des tests de validation du logiciel**, qui permettent de valider le logiciel indépendamment du matériel sur lequel il est implanté.

Types de tests

Classification des tests selon le niveau de détail

(situation dans le cycle de vie)

- **Test d'acceptation (recette)**

Les tests d'acceptation ou de recette permettent de **contrôler** les aspects liés à **l'installation** et à **la documentation** de l'application.

Ils sont appliqués **après** les **tests de validation**, et **les tests du système** qui testent la conformité entre l'application installée et le matériel.

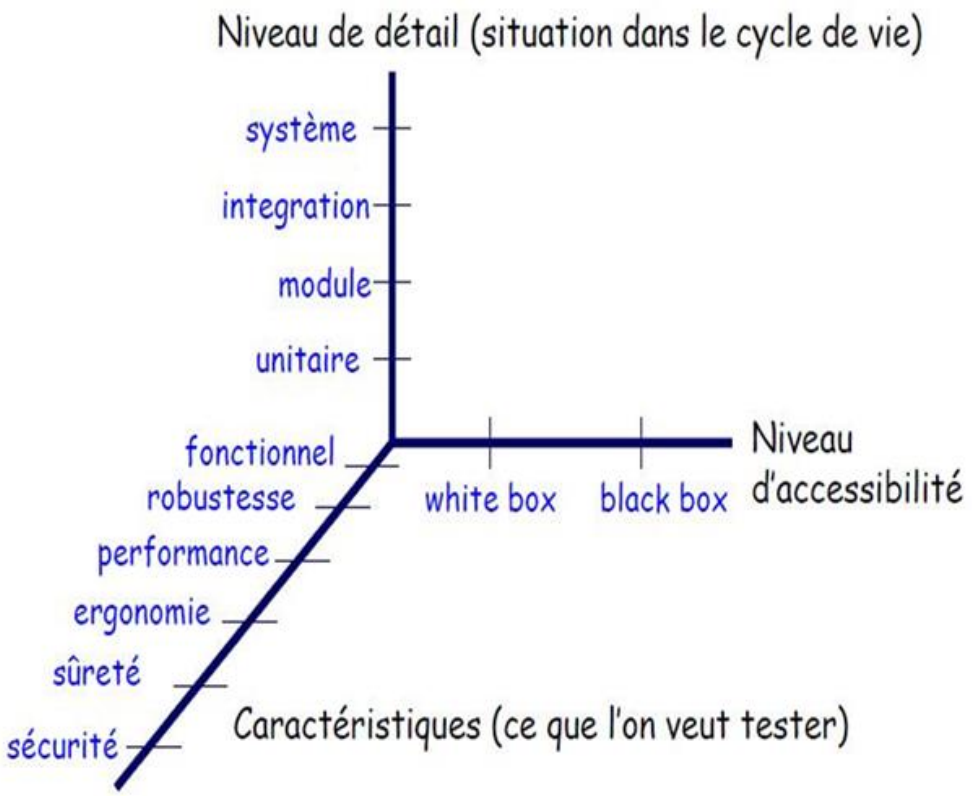
Les tests de recette consistent à vérifier que **les supports d'installation** ne contiennent pas des virus. Ils sont également utilisés afin de garantir la fiabilité de l'application lors de la phase de démarrage, en testant soigneusement toutes les procédures d'installation.

Types de tests

Classification des tests selon les caractéristiques

(ce que l'on veut tester)

- Dans la suite on présentera les tests relatifs à certaines **caractéristiques** d'un logiciel tel que :
- **la robustesse,**
- **la performance,**
- **la non-régression,**
- **l'ergonomie,**
- **la sécurité et**
- **la sûreté.**



Types de tests

Classification des tests selon les caractéristiques

(ce que l'on veut tester)

- **Test de robustesse**

La robustesse est définie comme le degré auquel un système ou un composant peut **fonctionner correctement** en présence **de conditions environnementales stressantes** ou **d'entrées non valides**.

Le test de robustesse est une méthodologie **d'assurance qualité** centrée sur **la robustesse** d'un logiciel.

Les tests de robustesse sont également utilisés pour décrire le processus de vérification de la robustesse (c'est-à-dire de la correction) des tests élémentaires dans un processus de test.

La problématique du **test de robustesse** dépend du système considéré, de son architecture et de son objectif.

Types de tests

Classification des tests selon les caractéristiques

(ce que l'on veut tester)

- **Test de performance**

Les tests de performance ont certains objectifs à savoir :

- Validation de la capacité des réseaux et des serveurs à prendre en charge des charges d'accès importantes.

- Définition d'un environnement matériel minimal pour le bon fonctionnement de l'application, en validant son comportement vis à vis des conditions extrêmes.

Les métriques de performance typiques incluent : **le taux de transfert de données**, la **vitesse de traitement**, la **mémoire**, **l'efficacité de la charge de travail et la fiabilité**, la **bande passante** et le **débit du réseau**.

Types de tests

Classification des tests selon les caractéristiques

(ce que l'on veut tester)

- **Test de performance (suite)**

Il existe **plusieurs types** de tests de performance:

1) Les tests de résistance: ils placent un système sous **une charge de trafic supérieure** aux **attentes**, afin que les développeurs puissent voir dans quelle mesure il **fonctionne** au-dessus des **limites de capacité** attendues.

2) Les tests de charge: ils aident les développeurs à comprendre le comportement d'un système sous une valeur de charge spécifique, en déterminant le nombre d'utilisateurs qu'une application ou un système peut gérer avant la mise en production de cette application ou de ce système.

Types de tests

Classification des tests selon les caractéristiques

(ce que l'on veut tester)

- **Test de performance (suite)**

Il existe **plusieurs types** de tests de performance :

3) Les tests de contrainte: ils permettent à l'équipe de développement du logiciel de comprendre l'évolutivité d'une charge de travail. Les tests de contrainte sollicitent les ressources matérielles, telles que les processeurs, la mémoire, les disques durs et les disques SSD, pour déterminer le point de rupture potentiel d'une application sur ces ressources finies.

4) Les tests d'imprégnation, également appelés tests d'endurance: ils simulent une augmentation constante du nombre d'utilisateurs finaux pour tester la durabilité à long terme des systèmes.

Types de tests

Classification des tests selon les caractéristiques

(ce que l'on veut tester)

- **Test de performance (suite)**

Il existe **plusieurs types** de tests de performance :

5) Les tests de crête, un autre sous-ensemble de tests de contrainte: ils évaluent les performances d'un système face à une augmentation soudaine et significative d'utilisateurs finaux simulés.

6) Les tests de pointe: ils aident à déterminer si un système peut gérer une augmentation de charge de travail brusque et drastique sur une courte période, de manière répétée. Comme pour les tests de résistance, une équipe informatique effectue généralement des tests de crête avant un événement important dans lequel un système subira probablement des volumes de trafic supérieurs à la normale.

Types de tests

Classification des tests selon les caractéristiques

(ce que l'on veut tester)

- **Test d'ergonomie**
- **L'ergonomie du logiciel** comprend **la détermination des besoins des utilisateurs**, **la conception de l'interface**, **l'assistance aux utilisateurs** et les **tests d'utilisabilité**.
- Les principales **méthodes** de **test d'ergonomie** sont **:**

- L'audit ergonomique consiste à vérifier si le logiciel respecte un ensemble de critères d'utilisabilité**.**

- Le test de perception permet d'évaluer la compréhension de l'interface**.**

- Le test d'utilisabilité sert à vérifier l'utilisabilité en plaçant l'utilisateur en situation.

Types de tests

Classification des tests selon les caractéristiques

(ce que l'on veut tester)

- **Test de sûreté**

La sûreté de fonctionnement est la combinaison de différents facteurs notamment :

- **la fiabilité** : ce point peut être couvert notamment par :
  - **l'exactitude fonctionnelle** (attestée par les tests fonctionnels) ;
  - **la tolérance aux pannes** ;
  - **l'utilisabilité** ;
  - **la durabilité** ;
- **la maintenabilité ;**
- **la disponibilité et les temps de réponse ;**
- **la confidentialité ;**
- **l'intégrité :** impossibilité d'altération des informations ;

Types de tests

Classification des tests selon les caractéristiques

(ce que l'on veut tester)

- **Test de sécurité**
- **Les tests de sécurité** sont utilisés pour garantir que les applications et les logiciels sont **exemptés** de toute **menace** ou **risque** qui peuvent **entraîner** une **perte importante**.
- **Les tests de sécurité** recherchent les éventuelles **lacunes** et **faiblesses** du système pouvant **entraîner une perte** d'informations, de la **réputation des employés**, **des revenus** ou d'une **organisation tierce**.

Types de tests

Classification des tests selon les caractéristiques

(ce que l'on veut tester)

- **Test de sécurité (suite)**
- **Le but des tests de sécurité** est **d'identifier les menaces** dans le système et **de mesurer ses vulnérabilités** potentielles afin que le système ne cesse de fonctionner ou ne soit pas exploité.
- **Les tests de sécurité** permettent également à détecter tous les risques de sécurité possibles dans le système et aident les développeurs à résoudre ces problèmes par le biais de codes.

Types de tests

Classification des tests selon les caractéristiques

(ce que l'on veut tester)

- **Test de sécurité (suite)**
- **Il existe sept types principaux de tests de sécurité:**

1) Analyse de la vulnérabilité: Cette opération est effectuée à l'aide d'un logiciel automatisé permettant d'analyser un système à l'aide de signatures de vulnérabilité connues.

2) Analyse de la sécurité: elle implique l'identification des faiblesses du réseau et du système, puis elle fournit des solutions pour réduire ces risques. Cette numérisation peut être effectuée manuellement ou automatiquement.

Types de tests

Classification des tests selon les caractéristiques

(ce que l'on veut tester)

- **Test de sécurité (suite)**
- **Il existe sept types principaux de tests de sécurité:**

3) Tests de pénétration: ce type de tests simule une attaque provenant d'un pirate informatique malveillant. Ces tests impliquent l'analyse d'un système particulier afin de rechercher d'éventuelles vulnérabilités lors d'une tentative de piratage externe.

4) Évaluation des risques: ce type de tests implique une analyse des risques de sécurité observés dans l'organisation. Les risques sont classés comme faible, moyen et élevé. Ce type de tests recommande des contrôles et des mesures pour réduire les risques.

Types de tests

Classification des tests selon les caractéristiques

(ce que l'on veut tester)

- **Test de sécurité (suite)**
- **Il existe sept types principaux de tests de sécurité:**

5) Audit de sécurité: Il s'agit d'une inspection interne d'applications et de systèmes d'exploitation pour détecter les failles de sécurité. Un audit peut également être effectué via une inspection du code ligne par ligne.

6) Piratage éthique: Il s'agit de pirater des systèmes de logiciel d'une organisation. Contrairement aux pirates malveillants, qui volent pour leurs propres gains, l'objectif est de révéler les failles de sécurité du système.

7) Evaluation de la posture: elle combine l'analyse de sécurité, le piratage éthique et les évaluations de risques pour montrer une posture de sécurité globale d'une organisation.