Chapitre 1

Activités de développement Logiciel

1. Spécification des besoins : **Exigences**
2. Conception
3. Réalisation : Implémentation (code)
4. Test
5. Maintenance

➔ Le test concerne toute activité de développement logiciel

Exigences fonctionnelles : Ce que fait le système

Exigences non fonctionnelles : contraintes associées (temps de réponse), liées à une qualité (sécurité, utilisabilité)

Test statique : Révision d'exigences

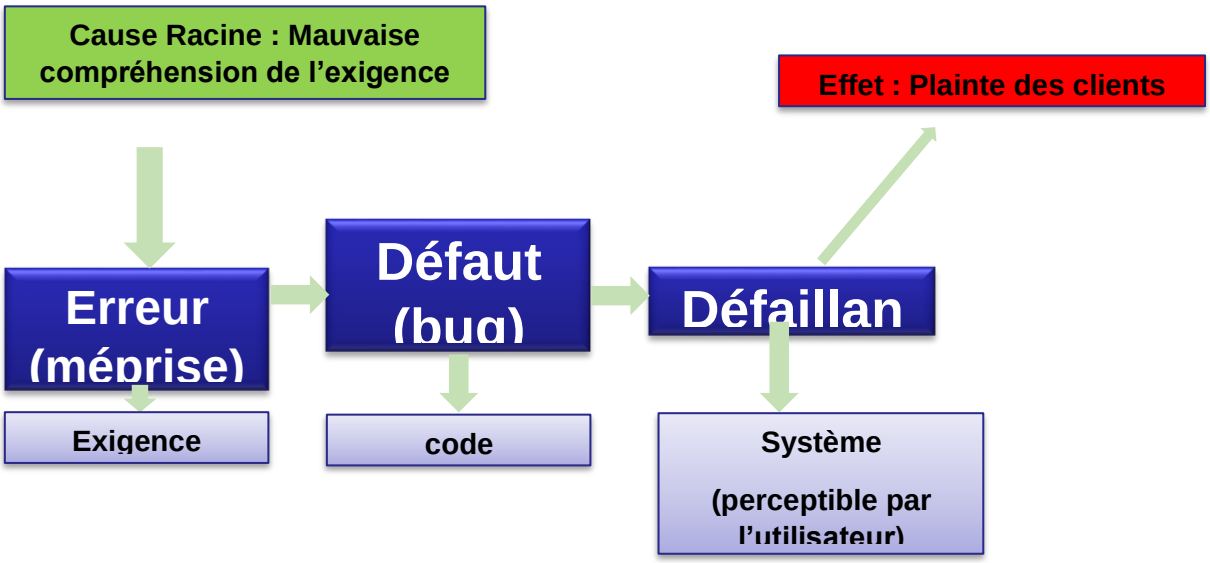
Test dynamique : Exécuter une application

Vérification : Réponse aux exigences spécifiques

Validation : Réponse aux exigences utilisateur +salification + environnement opérationnel

Tester : Trouver les défaillances -→ Testeur (ne corrige pas les défauts)

Débogage : analyser et supprimer les causes de défaillances : → Développeur



Origine des défauts : contraintes temporelles, communication, complexité, etc

Les tests améliorent la qualité du logiciel

Les 7 principes de Test

1. **Les tests cherchent la présence de défauts et non pas leur absence**
2. **Les tests exhaustifs sont impossibles : on ne peut pas tester toutes les combinaisons**
3. **Tester tôt**
4. **Regroupement de défauts**
5. **Paradoxe de pesticide : les cas de test doivent être toujours révisés**
6. **Les tests dépondent du contexte**
7. **L'absence d'erreurs est une illusion**

Suite de test : Ensemble de cas de test

Activités de Test

1. **Planification**
2. **Pilotage et contrôle : Voir l'avancement , progression – Rapport d'avancement**
3. **Analyser : Déterminer qu'est ce qu'on va tester**
4. **Conception : Comment va-t-on tester : préparation des cas de test, environnement , données de test**
5. **Implémentation : Planification, calendrier – Suite de test**
6. **Exécution : comparer le résultat attendu avec le résultat trouvé—Rapport de défauts**
7. **Clôture et synthèse : Leçons apprises – Rapport de synthèse**

Chapitre 2 : Tester pendant le cycle de vie du développement logiciel

1. CVDL : Cycle de vie de développement Logiciel

1.1 Le modèle séquentiel : Chaque phase ne peut commencer que lorsque la précédente est terminée

1.1.1 Modèle en cascade : Tester tard

1.1.2 Modèle en V : Tester tôt

1.2 Le modèle incrémental / Itératif

Incrémental : par morceaux

Itératif par version

➔ **Le choix du CVDL dépond de la nature/type du projet**

2. Niveaux de test

2.1 Test unitaire (de composant) effectué par le développeur , TDD (développement dirigé par les tests)

2.2 Test d'intégration (communication, interface), effectué par le testeur

Approaches: Big bang, stub-driver, bottom-up, top-down

2.3 Test système : effectué par le testeur

2.4 Test d'acceptation : phase d'établissement de confiance, effectué par testeur, client, utilisateur

Acceptation alpha : en usine

Acceptation beta : hors usine

3. Types de test

3.1 Tests fonctionnels : ce que fait le système- exigences

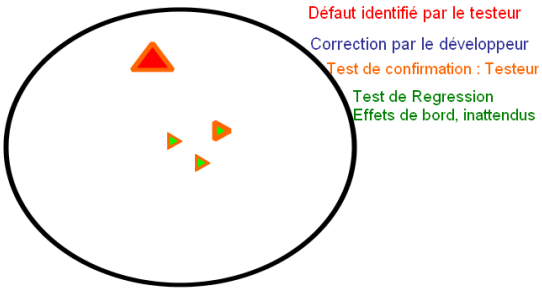
3.2 Tests non fonctionnels : contrainte, lié à une qualité-- exigences

3.3 Tests boites blanches : code, architecture/ structure interne

3.4 Test liés aux changements

3.4.1 Tests de confirmation : s'assurer qu'un défaut précédemment repéré a été fixé

3.4.2 Tests de régression : effets de bord, inattendus—sont souvent automatisés



4- Facteurs déclencheurs des tests de maintenance

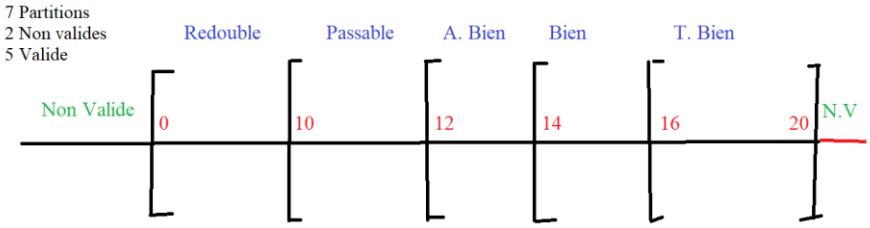
1- **Migration vers une nouvelle technologie**
2- **Ajout/Suppression**
3- **Déclassement d'une application (score a chuté..)**

Chapitre 4 Techniques de test

1. **Boites noires : Exigences**
2. **Boites blanches : Code, Architecture interne/Composant**
3. **Tests basés sur l'expérience : expérience testeur/développeur**

1. Boites noires

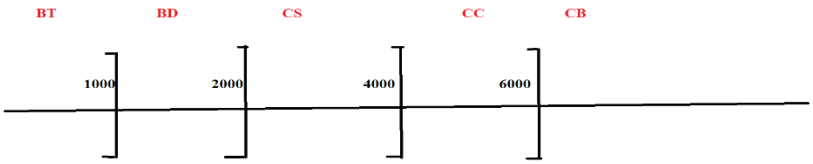
1.1 Partitions d'équivalences



Chercher la meilleur couverture

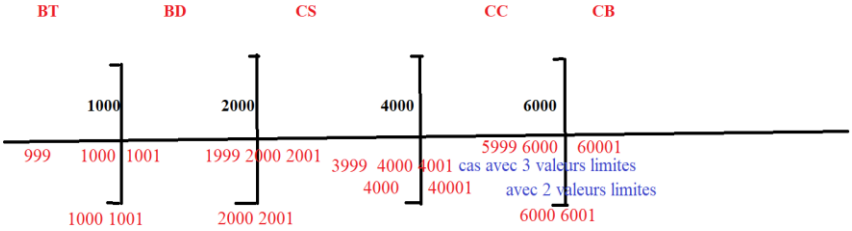
a)	**8 -**	**10 -**		**14- 15.5 -**	**17**	**c=4/7**
b)	**8-**	**11-**	**13-**		**16.5 -20**	**c=4/7**
c)	**2-**	**10-11-**	**13-**		**16**	**c=4/7**
d)		**10-**	**12-**	**14-**	**16-**	**21 c= 5/7**

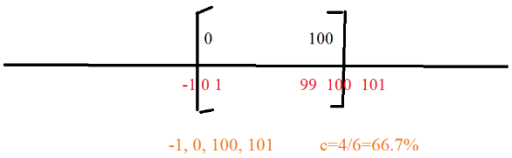
Couverture =Nombre de cas test effectués/ Nombre de cas de test nécessaires



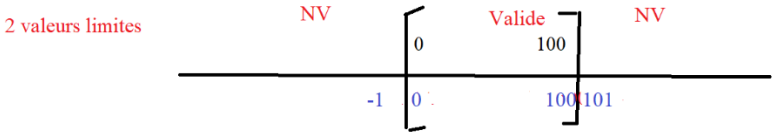
a) 0, 1000, 2000, 3000, 4000 c=3/5
b) 1000, 2001, 4000, 4001, 6000 c= 3/5
c) 123, 2345, 3456, 4567, 5678 c=3/5
d) 666, 999, 2222, 5555, 6666 c=4/5

1.2 Analyse de valeurs Limites (Extension des partitions d'équivalence)





-1, 0, 100, 101 c=4/6=66.7%



0 -1 -99 -101 c=2/4=50%

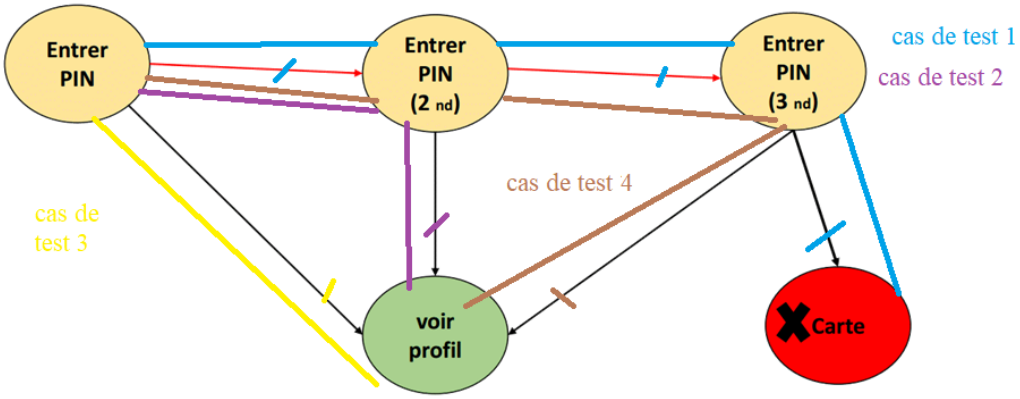
1.3 Table de décisions

Nombre de cas de test = nombre de règles

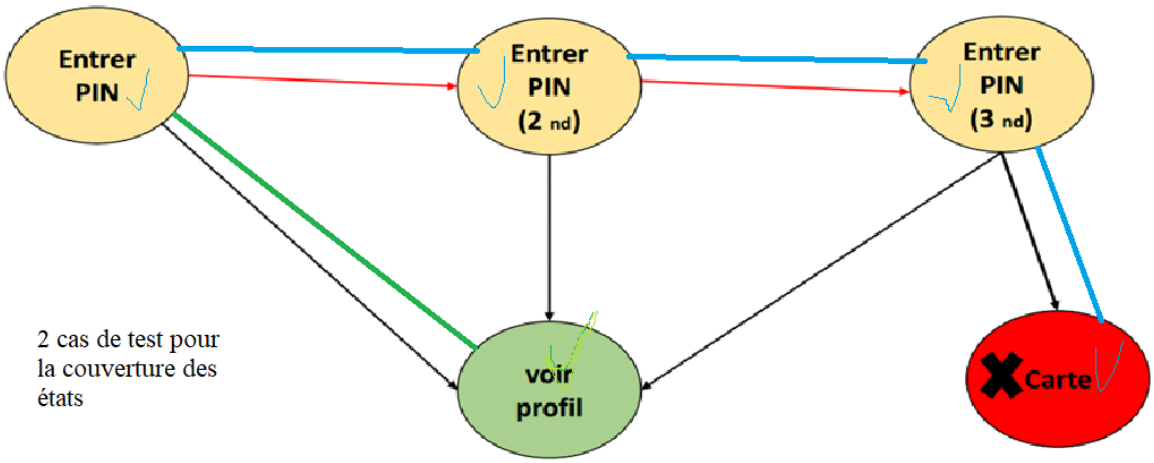
Condition	1	2	3	4	5	6	7	8
Graduate	Yes	Yes	Don't Care	Don't Care	No	No	No	No
Employed	Yes	Yes	No	No	Yes	Yes	No	No
CS	Yes	No	Yes	No	Don't Care	No	Yes	No
Actions								
Basics 0 %								
Basics 20 %								
Adv. 0 %								
Adv. 20 %								

Dans cet exemple on a 5 règles ➔ On a besoin de 5 cas de test

1.4 Transitions d'états

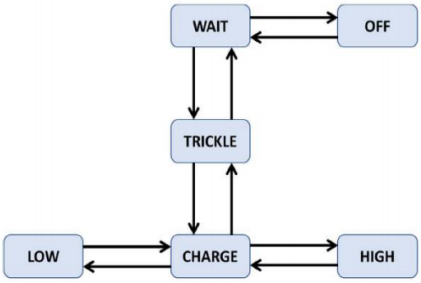


4 Cas de test pour couvrir toutes les transitions



2 cas de test pour la couverture des états

Soit le diagramme d'état suivant pour un chargeur de batterie de portable :



Quelle séquence apporte la MEILLEURE couverture des transitions pour ce modèle ?

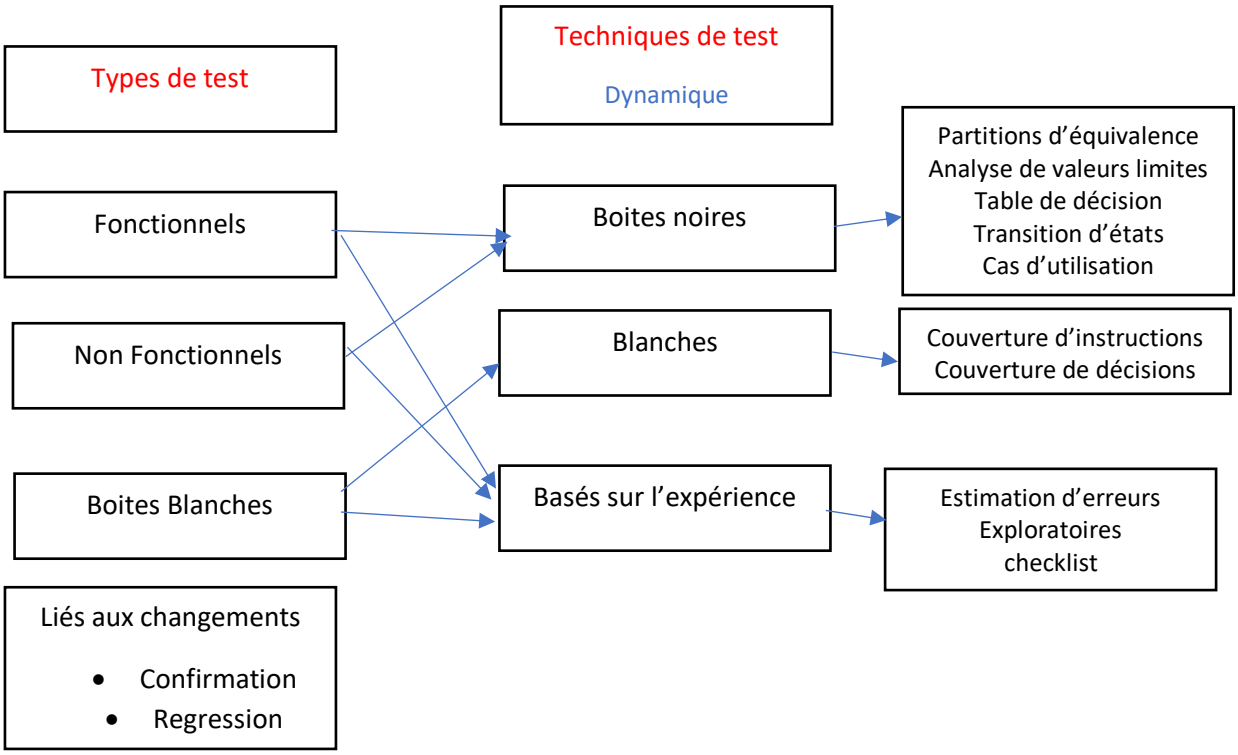
a) OFF WAIT OFF WAIT TRICKLE CHARGE HIGH CHARGE LOW c=7/10
b) WAIT TRICKLE WAIT OFF WAIT TRICKLE CHARGE LOW CHARGE c=7/10
c) HIGH CHARGE LOW CHARGE TRICKLE WAIT TRICKLE WAIT TRICKLE CHARGE 7/10
d) WAIT TRICKLE CHARGE HIGH CHARGE TRICKLE WAIT OFF WAIT c=8/10

1.5 Cas d'utilisation

Comportement de base (scénario nominal)

Comportement exceptionnel (scénario alternatif)

2. **Tests boites blanches**
   Couverture d'instructions
   Couverture de décision
   100% de couverture de décision implique 100% de couverture d'instruction
   L'inverse n'est pas vrai
3. **Tests basés sur l'expérience**
   3.1Tets basés sur l'estimation d'erreur (données de projets précédents, erreurs que les développeurs ont tendance à les commettre, etc)
   3.2 Tests exploratoires (peu de spécification, deadline serré, charte de test)
   3.3Checklist : liste de cas de test déjà préparée ou mise à jour ou créée


Types de test
Techniques de test
Dynamique
Fonctionnels
Non Fonctionnels
Boites Blanches
Liés aux changements
Confirmation
Regression
Boites noires
Blanches
Basés sur l'expérience
Partitions d'équivalence
Analyse de valeurs limites
Table de décision
Transition d'états
Cas d'utilisation
Couverture d'instructions
Couverture de décisions
Estimation d'erreurs
Exploratoires
checklist

Chapitre 3 : Tests statiques

Les tests statiques :

- **Revue : Exigence**
- **Analyse statique : Code**

Bases de test pour les tests statiques : Spécification (exigence, user stories, etc), modèle, code , contenu page web , guide d'utilisateur

1- Processus de revue
- **1.1 Planification : Estimation temps/ effort, technique et type de revue**
- **1.2 Lancement : distribution du produit d'activité**
- **1.3 Revue individuelle**
- **1.4 Communication et analyse des problèmes (défauts, clarifications)**
- **1.5 Correction et production de rapport**

2- Responsabilités dans une revue
- **2.1 Manager : planification, affecte budget/ personnel, etc**
- **2.2 Responsable de la revue : gère la revue**
- **2.3 Réviseur : Identification de défauts (Expert domaine …)**
- **2.4 Auteur : crée le produit d'activité et corrige si nécessaire**
- **2.5 Scribe : Collecte et recueille les défauts**
- **2.6 Modérateur (facilitateur) : fait la médiation, assure le bon déroulement des réunions**

3- Types de revues

Degré de formalisation ↓

- **3.1 Revue informelle : entre binômes / Collègues**
- **3.2 Relecture technique :**
  - *** Améliorer le produit d'activité**
  - *** Présence obligatoire du scribe**
  - *** Réunion de revue gérée par l'auteur**
- **3.3 Revue technique :**
  - *** Obtention d'un consensus**
  - *** Scribe n'est pas l'auteur**
- **3. 4 Inspection (degré de formalisation le plus élevé)**
  - *** Rôles bien définis**
  - *** Présence d'un modérateur formé**
  - *** checklist**

Revue par paire

4. **Techniques de revues**
   - **4.1 Adhoc :**
   - **4.2 Checklist**
   - **4.3 Essais à blanc**
   - **4.4 Basées sur les rôles : jouer le rôle d'un utilisateur (enfant, personne agée, etc)**
   - **4.5 Basées sur les perspectives : adapter des points de vue de différentes parties prenantes**

Chapitre 5 Gestion des tests

1- Organisation des tests

1.1Independence des tests

Les testeurs indépendants se sentent isolés

1.2Testeur / Test Manager

- **Test manager: planification , sélection de l'équipe, stratégie, lance les activités de test, contrôle, rapport d'avancement et de synthèse, choix de l'environnement**
- **Testeur : contribue au plan de test, révise les exigences, conçoit les cas de test, les exécute, les automatise, configuration des environnements, rapport de défaut**

2- Planification des tests

2.1 Plan de test : document qui contient le planning des activités de test, les environnements, les risques, les ressources, les métriques

2.2 Critères d'entrée/ Critères de sortie

- Critère d'entrée : de quoi doit on disposer pour lancer les tests, disponibilité des exigences, données de test , environnement, etc

- Critères de sortie : quand arrêter les tests, couverture, nombre de défauts trouvés, etc

Cas de test	Priorité	Dépendance technique avec:	Dépendance logique avec:
TC1	Haute	TC4	
TC2	Basse		
TC3	Haute		TC4
TC4	Moyenne		
TC5	Basse		TC2
TC6	Moyenne	TC5	

Quelle séquence d'exécution prend LE MIEUX en compte les pri

a) TC1 – TC3 – TC4 – TC6 – TC2 – TC5 **Tc1 ne doit pas etre avant Tc4**

b) TC4 – TC3 – TC1 – TC2 – TC5 – TC6

c) TC4 – TC1 – TC3 – TC5 – TC6 – TC2 **TC5 ne doit pas etre avant TC2**

d) TC4 – TC2 – TC5 – TC1 – TC3 – TC6 **TC3 a une priorité haute en la comparant avec TC2**

2.3 Estimation des tests

- basée sur les métriques : burndown chart

- basée sur l'expertise : wideband delphi, planning poker (agile)

3. Gestion de configuration = contrôle de version

4. Risque et test

Risque= probabilité+ impact

-Risque produit

- Risque projet

Chapitre 6 Outils de support aux tests

Aide à la gestion des tests	Aide aux tests statiques	Aide à la spécification des tests
• Outils de gestion des exigences • Outils de gestion des cas de tests • Outils de gestion d'incidents • Outils de gestion de configuration	• Outils de support de revues • Outils d'analyse statique (D) • Outils de modélisation (D)	• Outils de conception des tests (= outils de génération automatisé des cas de test) • Outils de préparation des données de tests

Aide à l'exécution et à l'enregistrement des tests	Support de performance et de surveillance	Support pour les besoins de tests spécifiques
• Outils d'exécution des tests • Harnais de test (D) • Comparateur de tests • Outils de mesure de couverture (D) • Outils de test de sécurité	• Outils d'analyse dynamique (D) • Outils de tests de performances / de tests de charge/ de tests de stress • Outils de surveillance	• Evaluation de la qualité des données • Test de l'utilisabilité