

Chapitre 3 : Tests et validation



Plan

2

- Introduction
- Problématique
- Objectifs
- Principes des tests et validation
- Types de défauts logiciels
- Catégorie de tests et validation
- Tests et cycle de vie logiciel
 - Niveaux de tests pour le cycle en V
 - Niveaux de tests pour le cycle itératif
- Méthodes de tests
- Types de tests
- Processus de tests
- Le testing, un métier, une éthique

Introduction

3

- «*Tester, c'est exécuter le programme dans l'intention d'y trouver des anomalies ou des défauts*» - G. Myers (The Art of Software testing)
- «*Testing can reveal the presence of errors but never their absence* » - Edsger W. Dijkstra. Notes on structured programming. Academic Press, 1972.

➔ Rechercher la présence d'erreurs pour les corriger

Problématique (1/3)

4

- Tester tous les comportements du système ➔ rarement envisageable et possible
 - Exemple : Microsoft Word compte 850 commandes et 1600 fonctions, ce qui fait un total de plus de 500 millions de conditions à tester (<http://fr.academic.ru/dic.nsf/frwiki/224491>)
 - Assurer la qualité des systèmes
 - Réduire les risques d'occurrence de problèmes dans l'environnement opérationnel
 - Respecter les exigences légales, contractuelles et des normes industrielles spécifiques
- ➔ Travail fastidieux

Problématique (2/3)

5

- Erreurs humaines lors des travaux :
 - de spécification
 - de conception
 - de programmation
 - de tests de logiciel
- Dues entre autre
 - Complexité grandissante des logiciels
 - Problèmes de communication
 - Manque de formation des ingénieurs
 - Pression des délais et des coûts durant les travaux d'ingénierie
 - ...

Problématique (3/3)

6

- Quelques bugs historiques :
 - L'échec du vol inaugural de la fusée Ariane 5 en 1996
 - ✦ Un dépassement de capacité provoque le crash informatique de l'appareil. Aveuglé, le pilote automatique perdit le contrôle de la fusée, et un dispositif de sécurité provoqua son auto-destruction quelques secondes après le décollage. (<http://fr.academic.ru/dic.nsf/frwiki/1718969>)
 - ✦ C'est le bug informatique le plus **coûteux** de l'histoire
 - Bug de l'an 2000 ou bug du millénaire, grande psychose sans trop de dégâts
 - Bug de l'an 2038 (systèmes à 32bits le 19 janvier 2038 à 3 h 14 min 7 s) (http://fr.wikipedia.org/wiki/Bug_de_l'an_2038)

Objectifs

7

- Processus de validation et vérification
 - S'assurer que le produit répond aux exigences
 - ✦ Conformité aux cahier des charges
 - ✦ Conformité aux documents de spécification
 - ➡ *Validation*
 - S'assurer que le produit est bien construit
 - ➡ *Vérification*
 - Améliorer la productivité des équipes

Principes des tests et validation (1/3)

8

- Les tests montrent la présence de défauts
 - Les tests peuvent prouver la présence de défauts, mais ne peuvent en prouver l'absence
 - ➔ Les tests réduisent la probabilité que des défauts restent cachés dans le logiciel
- Les tests exhaustifs sont impossibles
 - Tester toutes les combinaisons d'entrées et de pré-conditions n'est faisable que pour des cas triviaux
 - ➔ Utilisation de l'analyse des risques et des priorités pour focaliser les efforts de tests

Principes des tests et validation (2/3)

9

- **Tester tôt :**
 - Les activités de tests devraient commencer aussi tôt que possible dans le cycle de développement du logiciel et devraient être focalisés vers des objectifs définis
- **Regroupement des défauts**
 - L'effort de test devrait être fixé proportionnellement à la densité des défauts prévus et constatés dans les différents modules.
 - Un petit nombre de modules contient généralement la majorité des défauts détectés lors des tests pré-livraison, ou affichent le plus de défaillances

Principes des tests et validation (3/3)

10

- Les tests dépendent du contexte
 - Les tests sont effectués différemment dans des contextes différents. Par exemple, les logiciels de sécurité critique seront testés différemment d'un site de commerce électronique
- (D'après <http://www.istqb.org/>)

Types de défauts logiciels (1/4)

11

- Bug
 - Défaut de conception ou de réalisation d'un programme informatique, qui se manifeste par des anomalies de fonctionnement de l'ordinateur
<http://www.larousse.fr/dictionnaires/francais/bogue/10005>
 - Le mot anglais bug (insecte, bogue) vient du jargon des ingénieurs de matériel et représente les problèmes qui y survenaient
- Crash applicatif ou Deny of Service
 - Déclenchement d'un mécanisme à la fois matériel et logiciel qui met hors service le logiciel défaillant lors de la tentative de ce dernier d'effectuer des opérations impossibles à réaliser (exceptions : division par zéro, recherche d'informations inexistantes...) [http://fr.wikipedia.org/wiki/Bug_\(informatique\)\)](http://fr.wikipedia.org/wiki/Bug_(informatique)))

Types de défauts logiciels (2/4)

12

- Fuite de mémoire
 - Dysfonctionnement dû à un bug dans les opérations d'allocation de mémoire. La quantité de mémoire utilisée par le logiciel défaillant va en augmentant continuellement et gêne le déroulement des autres logiciels et les entraîne à des dysfonctionnements
- Vulnérabilité
 - Faiblesse dans un système informatique permettant à un attaquant de porter atteinte à l'intégrité de ce système, c'est-à-dire à son fonctionnement normal, à la confidentialité et l'intégrité des données qu'il contient. On parle aussi de faille de sécurité informatique.

Types de défauts logiciels (3/4)

13

- Faute de segmentation

- Dysfonctionnement dû à un bug dans des opérations de manipulations de pointeurs ou d'adresses mémoire.
- Lecture ou écriture des informations dans un emplacement de mémoire (segment) qui n'existe pas ou qui ne lui est pas autorisé.
- Détection des exceptions provoque alors la mise hors service du logiciel défaillant.

- Buffer Overflow

- Dépassement de tampon ou débordement est un bug par lequel un processus, lors de l'écriture dans un tampon, écrit à l'extérieur de l'espace alloué au tampon, écrasant ainsi des informations nécessaires au processus. Le comportement de l'ordinateur devient imprévisible. Il en résulte souvent un blocage du programme, voire de tout le système. C'est une faille de sécurité courante des serveurs informatiques.

Types de défauts logiciels (4/4)

14

- Deadlock ou inter blocage
 - Dysfonctionnement durant lequel plusieurs processus s'attendent mutuellement, c'est à dire qu'ils attendent chacun que l'autre libère les ressources qu'il utilise pour poursuivre. Les ressources restent verrouillées durant les attentes, ce qui peut bloquer d'autres processus et par effet domino bloquer l'ensemble du système. Un mécanisme de prévention provoque l'annulation de l'opération lorsque la durée d'attente dépasse le délai admissible (anglais *timeout*).

Catégories de tests et validation (1/2)

15

- Tests au cours du cycle de vie logiciel
 - Quel que soit le modèle de cycle de vie, plusieurs bonnes pratiques de tests s'appliquent:
 - ✦ A chaque activité de développement, correspond une activité de test.
 - ✦ Chaque niveau de test a des objectifs de tests spécifiques pour ce niveau.
 - ✦ L'analyse et la conception des tests pour un niveau de test devraient commencer pendant l'activité correspondante de développement.
 - ✦ Les testeurs doivent être impliqués dans la revue des documents aussitôt que des brouillons sont disponibles dans le cycle de développement.

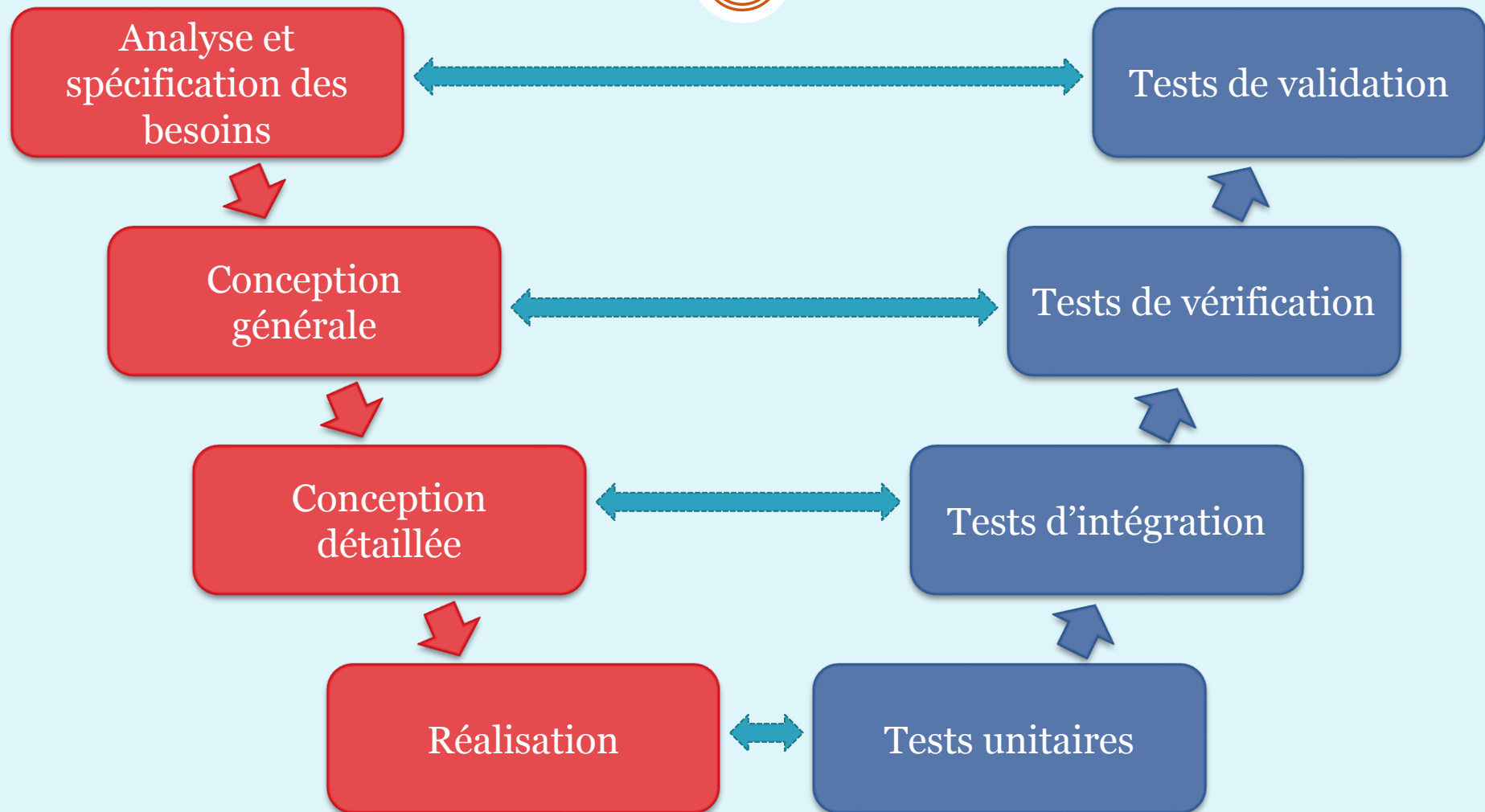
Catégories de tests et validation (2/2)

16

- Tests d'acceptation contractuelle et réglementaire
 - Les tests d'acceptation contractuelle sont exécutés sur base des critères d'acceptation contractuels pour la production de logiciels développés sur mesure
 - ✦ Les critères d'acceptation devraient être définis lors de la rédaction du contrat
 - Les tests d'acceptation réglementaires sont exécutés par rapport aux règlements et législations qui doivent être respectés, telles que les obligations légales, gouvernementales ou de sécurité

Tests et modèle en V

17



Tests et modèles itératifs (1/2)

18

- Le mode de développement itératif est une succession d'activités exécutées comme une série de petits développements: exigences, conception, construction et tests d'un système
 - Exemples : prototypage, développement rapide d'applications (RAD), Rational Unified Process (RUP) et les modèles de développement agiles
- Le système logiciel résultant (l'incrément) d'une itération peut être testé à plusieurs niveaux de tests à chaque itération

Tests et modèles itératifs (2/2)

19

- Un incrément, ajouté à ceux développés préalablement, forme un système partiel en croissance, qui devrait également être testé
- Les tests de **régression** sont de plus en plus importants sur toutes les itérations après la première
- La vérification et la validation peuvent être effectuées sur chaque incrément

Méthodes de tests

20

- **Méthode boîte noire**
 - Les essais tournent autour du fonctionnement externe du système
 - Les détails d'implémentation des composants ne sont pas connus
- **Méthode boîte blanche (ou transparente)**
 - Les essais tournent autour du fonctionnement interne du système
 - Les détails d'implémentation des composants sont tous connus
- **Méthode boîte grise**
 - Combinaison des deux approches précédentes

Types de tests (1/6)

21

- Test nominal / de bon fonctionnement
 - Les données entrées sont volontairement valides
- Test de robustesse / de défense
 - Les données entrées sont volontairement invalides
- Test de performance / d'endurance
 - Load test : test avec montée en charge
 - Stress test : soumis à des demandes de ressources anormales
- Test unitaire
 - Tester les fonctions (ou les modules) de code par les programmeurs

Types de tests (2/6)

22

- Test d'intégration
 - Valider le bon fonctionnement d'une ou de plusieurs parties (modules de code, bibliothèques, applications individuelles) développées indépendamment avec le reste de l'application
- Test de Compatibilité
 - Fonctionnement du logiciel dans une configuration spécifique du système (sous un système d'exploitation spécifique, dans un environnement de réseau particulier ..)
- Test de Recette
 - Confirmer que l'application répond d'une manière attendue aux requêtes qui lui sont envoyées

Types de tests (3/6)

23

- **Test Système**
 - Tester l'entière application par la méthode boîte noire dans l'environnement qui imite la situation réelle d'utilisation de l'application
- **Test de bout en bout (end 2 end)**
 - Similaire au test système. Met en jeu plusieurs composants en interaction : BdD, Réseau, autres système et application, composant hardware
- **Test de non régression (ou Tests liés au changement)**
 - Reprendre un ensemble de cas de tests après avoir fixé les bogues ou les modifications du logiciel ou de l'environnement
 - Les outils de test automatisé peuvent être extrêmement utiles pour ce type de test

Types de tests (4/6)

24

- **Sanity tests**
 - Jeu initial de test pour déterminer si une nouvelle version du logiciel présente des performances suffisamment acceptables pour entamer une campagne de tests majeur.
 - Par exemple, si le nouveau logiciel est un système qui plante toutes les 5 minutes, le logiciel n'est pas saint pour pouvoir aller plus loin dans des tests fonctionnels ou non fonctionnels
- **Test d'utilisabilité**
 - Valider si l'application est facile à utiliser
 - Test subjectif dépendant des utilisateurs finaux.
 - Les programmeurs et les testeurs ne sont pas impliqués

Types de tests (5/6)

25

- Test d'installation
 - Tester le processus d'installation / désinstallation intégralement, partiellement ou progressivement
- Test de sécurité
 - Détecter les intrusions et les failles de sécurité par le biais d'une vérification périodique de vulnérabilité de sécurité
- Test d'administration
 - Focalisation sur les aspects d'administration (tests des backups et restaurations; reprise après sinistre; gestion des utilisateurs; tâches de maintenance; chargements de données et tâches de migration..)

Types de tests (6/6)

26

- **Test libre / Aléatoire**
 - Déroulé par des utilisateurs connaissant (peu) le produit sans stratégie particulière ou un suivi de cas de tests
 - Ce genre de tests permet de chasser le maximum de bug au début d'une campagne et de vérifier l'acceptabilité du produit à la fin de la campagne
- **Test automatique / Test automatisé**
 - Génération automatique de jeu de tests utilisant le plus souvent des outils du Framework de développement
 - Les tests peuvent être des tests unitaires portant sur des fonctions ou des classes ou des tests fonctionnels via des scripts (batch)
 - L'analyse à froid des résultats de tests sera rendu disponible au testeur suite à une batterie en mode nuit, week-end, ...

Processus de tests (1/5)

27

- **Organisation des tests**
 - Dans le plan de management au lancement du projet et dans le plan d'assurance qualité produit
- **Rôles et responsabilités**
 - Les responsabilités et le profil des personnes intervenant dans le processus de test : Responsable de tests, concepteur de tests, testeur
 - Pas de développeur testant son propre code (ou une équipe Vs son Produit)
 - Les experts métiers sont souvent engagés dans des tests de recette

Processus de tests (2/5)

28

- Développement d'une stratégie de tests
 - Définir les moyens à mettre en place pour tester le logiciel
 - Décrire la stratégie de tests à mettre en place pour tester la première version, tester les versions suivantes, critères d'arrêt des tests
 - Rédiger les fiches de tests sur la base de l'analyse et conception des cas d'utilisation et des cas aux limites et détailler en fonction de la criticité des différentes fonctions
 - Passer en revue la stratégie, le plan et les cas de tests quant à leur testabilité, leur cohérence, leur couverture et leurs exigences de départ

Processus de tests (3/5)

29

- Planification et estimation de passage des tests
 - Chronométriser le passage des fiches
 - Ecrire un calendrier d'exécution des tests pour une série de cas de test donnés en tenant compte des priorités ainsi que des dépendances logiques et techniques
 - Planifier avec des outils appropriés, planifier le passage aux bancs de tests, prévoir parfois des équipes 3x8,...
- Critères de sortie
 - Critères d'acceptance et de qualification pour le passage à la livraison : Vecteur de Bugs (Critique, Majeur, Mineur, Evolution)
 - Activités de clôture des tests et activité relative de la production de rapports et synthèse des tests.

Processus de tests (4/5)

30

- Suivi et contrôle de l'avancement des tests
 - Le contrôle identifie
 - ✦ les déviations par rapport à ce qui a été planifié
 - ✦ les variations en termes d'atteinte des objectifs prévus
 - Le contrôle propose des actions afin d'atteindre ces objectifs
 - Juger la priorité des passages de tests par rapport aux dates de livraisons
 - Adapter le planning continuellement tout en entreprenant les actions nécessaires
- Gestion de configuration
 - Gestion des fiches de tests : Administration, Versionning, Traçabilité, partage entre les testeurs (et les développeurs)
 - Intégration avec la gestion des exigences

Processus de tests (5/5)

31

- Risques et tests

- Décrire un risque comme un problème probable qui peut compromettre l'atteinte des objectifs de projet d'un ou de plusieurs acteurs
- Se rappeler que le niveau de risque est déterminé par sa probabilité d'occurrence et son impact (dommages en résultant)

- Gestion d'incidents

- Rédiger un rapport d'incident couvrant l'observation d'une défaillance pendant le test
- Décrire les étapes de reproduction du scénario
- Etablir le rapport de synthèse de la campagne à partir des informations recueillies pendant le test et donner son avis quant au critère de libération

Coût des tests

32


- Les défauts logiciels coûtent en moyenne 3,61 \$ par ligne de code

(<http://www.lemagit.fr/technologie/applications/langages/2012/01/03/les-eacute-fauts-logiciels-ucirc-tent-moyenne-par-ligne-code/>)

- Le test représente 30 à 40% des coûts de développement d'un logiciel
- Le test représente en moyenne 1/3 du temps de développement

Le testing, un métier, une éthique (1/2)

33

- Le test des logiciels est un métier à part entière
- Activité dans le cycle de développement où l'on peut voir toutes les fonctionnalités d'un produit logiciel
- Le test logiciel est le maillon principal dans la chaîne d'assurance qualité produit
- Le test logiciel pourra avoir pour but de qualifier un logiciel ou certifier un produit
- Accès à des informations confidentielles et privilégiées par les testeurs
 Un code d'éthique est nécessaire

Le testing, un métier, une éthique (2/2)

34

- En référence au code d'éthique d'ACM et de l' IEEE pour les ingénieurs, ISTQB® définit le code d'éthique suivant :
 - PUBLIC – les testeurs de logiciels doivent agir en fonction de l'intérêt public
 - CLIENT ET EMPLOYEUR – les testeurs de logiciels doivent agir pour l'intérêt de leur client et de leur employeur tout en respectant l'intérêt public
 - JUGEMENT – les testeurs de logiciels doivent conserver leur intégrité et leur indépendance dans leur jugement professionnel
 - PRODUIT – les testeurs de logiciels doivent assurer que les fournitures qu'ils produisent (concernant les produits et les systèmes qu'ils testent) répondent le plus possible aux standards professionnels

Exemple d'une fiche de test

(35)

Test Sheet		By : mkh
Description : Importing the external reference data		Test sheet : Gen_DbInst_001
Tested modules : IRIS Database		
Context :		
Database service name initialised		
Iris user created on the oracle service name		
Dumped external data file is given		
Condition of success : All expected results are observed		
STEP	ACTION	EXPECTED RESULT
1	Launch Imp command and supply the irisName account login	Connection succeeded
2	Supply the path/name of the dumped external data And respond to all appearing questions	Import of the external data finished with no error
3	Connect as irisName/irisName	Connection succeeded
4	Select * from tab	Following entries are displayed: ALPS_B_NUMBER_GROUP TABLE ALPS_ENGINEERING_ROUTE TABLE ALPS_RD_CARRIER TABLE ALPS_RD_CID_DEFINITIONS_VIEW TABLE
Notes : this step is relative to XXX soft platform test		
For AZURE this step will be replaced by running a script to configure database links to external data in which case when starting: select * from USER_DB_LINKS; all needed data tables are displayed		