Tests des logiciels

Jean-François Pradat-Peyre

2016

1 : Quelques aspects essentiels des tests logiciels

Quelques aspects essentiels des tests logiciels

- De la nécessité des tests
- Premières définitions des tests
- Les 7 principes généraux des tests
- Processus fondamentaux des tests
- Psychologie des tests

Importance du logiciel

- Exemples de défauts logiciels et leur cause
  - Convocation à l'école de personnes âgées de 106 ans
    - Cause : codage de l'âge sur 2 caractères
  - Mission Vénus : passage à 5 millions de Kms de la planète au lieu de 5 000 Kms
    - Cause : remplacement d'une virgule par un point
  - Passage de la ligne : au passage de l'équateur un F16 se retrouve sur le dos !
    - Cause : changement de signe de la latitude mal pris en compte
  - Station MIR : 2 jours sans courant (14 au 16 novembre 1999)
    - Cause : plantage d'un ordinateur qui contrôlait l'orientation des panneaux solaires
  - Hôpital : Décès d'un malade
    - Cause : erreur logicielle dans le programme de monitoring
  - Missile : En URSS, fusée pointant Hambourg au lieu du Pôle Nord
    - Cause : erreur de signe : d'où une erreur de 180° du système de navigation
  - Inondation de la vallée du Colorado en 1983
    - Cause : mauvaise modélisation du temps d'ouverture du barrage
  - Perte de Mars Climate Orbiter le 23 septembre 1999 après 9 mois de voyage : coût de 120 M$
    - Cause : confusion entre pieds et mètres
- Aujourd'hui le logiciel est partout : centrales nucléaires, avionique, appareils ménagers, etc. : Notion de systèmes complexes à logiciel prépondérant
  - "Il y a plus d'informatique dans la Volvo S80 que dans le chasseur F15" déclarait en janvier 2000 le Président d'Audi

Défauts logiciels

- Chaîne de l'erreur

  Erreur humaine (méprise) ➔ ***défaut*** logiciel ➔ ***défaillance*** (si le défaut est exécuté)

- Différence entre défaillance et panne
  - Défaillance logicielle
    - Le logiciel produit des résultats inattendus (calculs erronés par exemple)
    - Le logiciel ne délivre pas le service attendu (non enregistrement d'un article dans une base de données, par exemple)
    - Fonctionnement en mode dégradé
  - Panne logicielle
    - Le système devient inutilisable ➔ nécessité de redémarrer le logiciel/le système

- Fiabilité des logiciels
  - **MTBF** (Mean Time Between failure) : temps moyen de bon fonctionnement (entre deux défaillances ou pannes, temps de réparation compris)
    - Doit être le plus long possible
  - **MTTR** (Mean Time To Repair) : temps moyen de remise en route

Tester ?

- On ne sait pas, par construction, fabriquer des logiciels sans défaut
  - Toute activité humaine est imparfaite
- Limites théoriques
  - On ne sait pas, dans le cas général, construire un algorithme permettant de dire si deux programmes réalisent le même algorithme (problème indécidable)
    - Conséquence : on ne peut pas valider un programme par un autre qui réaliserait les même fonctionnalités
  - On ne sait pas prouver, dans le cas général, qu'un programme s'arrête (problème indécidable de la terminaison)
- Rôle des tests
  - En développement : découvrir le maximum de défauts avant la recette finale
  - En maintenance : vérifier les modifications, la non régression
  - Éventuellement, vérifier le respect d'exigences légales ou contractuelles
  - Éventuellement, certification du logiciel : domaine avionique par exemple

Quelques aspects essentiels des tests logiciels

- De la nécessité des tests
- Premières définitions des tests
- Les 7 Principes généraux des tests
- Processus fondamentaux des tests
- Psychologie des tests

Les tests ce n'est pas … (ou idées fausses sur les tests)

- Il faut éliminer tous les défauts
  - Doit-on tester le cas où l'automobile roule à 500 km/h ?
- L'amélioration de la fiabilité est proportionnelle au nombre de défauts corrigés
  - Il reste un défaut dans une fonction toujours utilisée …
- Je programme sans erreur, ce n'est pas la peine de tester …
  - Les environnements / méthodes de développement évolués diminuent mais n'élimine pas la présence de défauts
- Croire que c'est simple et facile
- Croire que cela n'exige ni expérience, ni savoir-faire, ni méthodes
- Se focaliser sur un seul aspect du logiciel

Les tests mettent en œuvre de nombreuses activités

- Tester ce n'est pas uniquement « passer des tests ».
- Il s'agit aussi (et surtout) de
  - Planifier les tests
  - Spécifier les tests
  - Concevoir les tests
  - Établir les conditions de tests
  - Définir la ou les plates-formes de tests
  - Définir les conditions d'arrêt d'une campagne de tests
  - Contrôler les résultats
  - Tracer les tests vis-à-vis des exigences (matrice de traçabilité)
- Les revues des documents clés du projet contribuent à la qualité des tests
- Les revues de code sont aussi très efficaces (1 heure de revue de code fait gagner de nombreuses heures de maintenance)

Les tests peuvent être menés selon différents niveaux de maturité (au sens CMM)

- **Niveau 1** : mise au point (déboguage) et tests ***ne sont pas différenciés***
  - Tester et déboguer sont des activités différentes
    - On débogue un programme qui sort « brut de fonderie » du codage
    - On teste un programme qui a été débogué
- **Niveau 2** : le but des tests est de montrer que ***le système fonctionne***
  - On aura tendance à tester que ce qui a été « codé »
- **Niveau 3** : le but des tests est de montrer que ***le système ne fonctionne pas***
  - mais la notion de risque n'est pas prise en compte
- **Niveau 4** : le but des tests est de ***réduire le risque*** de non-fonctionnement tel que perçu par l'usager à un niveau compatible avec la mission du système
- **Niveau 5** : la ***testabilité du système est complètement intégrée*** au processus de conception
  - Il est ***futile*** de concevoir ce qu'on ne saura pas tester

Les différents types (niveaux) de tests

- tests unitaires ou tests fonctionnels
  - tests « boîte blanche » ou « boites noires » de composants élémentaires du logiciel
  - Métrique : taux de couverture du graphe de contrôle
  - Contribution à la qualité : qualité du code
- Tests d'intégration
  - Tests « boîte noire » sur le code
  - Tests « boîte blanche » sur les interfaces entre modules/composants
  - Qualité des interfaces de communication entre les modules logiciels
- Tests de validation
  - Tests fonctionnels : « boîte noire »
- Tests non fonctionnels
  - fiabilité
  - utilisabilité
  - performance
  - portabilité
  - évolutivité
- Tests d'acceptation (recette)
  - Vérification de l'atteinte des exigences (systèmes, métier, certification)

Tests statiques et tests dynamiques

Tests statiques

- Ils reposent essentiellement sur le traitement des sources logiciels
  - Lecture croisée et inspection de code. Le contrôle est ici collégial
  - Recherche d'anomalies comme le typage impropre, l'incohérence des interfaces entre modules, etc.
- L'utilisation d'un langage fortement typé contribue à éliminer un grand nombre de fautes
- L'utilisation de règles de programmation, en fonction du langage utilisé, contribue à améliorer la qualité du code et à homogénéiser la programmation dans une équipe, un département
- De nombreux outils peuvent aider à la mise en œuvre de ces tests

Tests statiques et tests dynamiques (suite)

Tests dynamiques

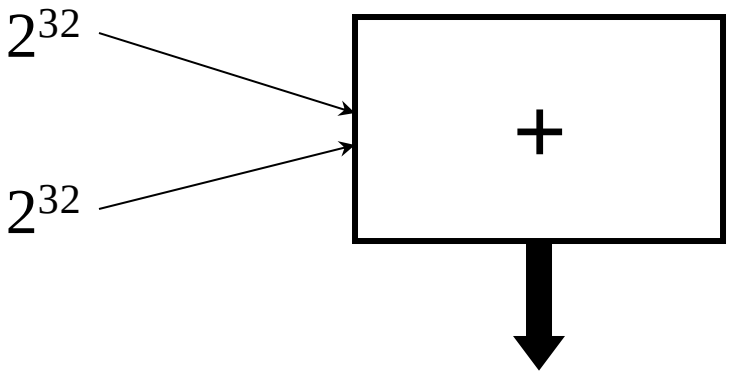
- Ils reposent sur l'exécution du code
  - Tests structurels reposant sur le graphe de contrôle et le flot de données
  - Tests fonctionnels en conformité avec les exigences et les spécifications
- Différentes méthodes existent pour déterminer les valeurs de test « pertinentes »
  - Tables de décision, partitionnement des domaines
  - Tests aux limites
- De nombreuses métriques existent pour quantifier la qualité des tests menés
  - Couvertures du code à différents niveaux
  - Couverture de la spécification
  - Taux de défauts découverts par unité de temps, d'équipe ou d'architecture

Quelques aspects essentiels des tests logiciels

- Nécessité des tests
- Premières définitions des tests
- Les 7 Principes généraux des tests
- Processus fondamentaux des tests
- Psychologie des tests

Principes généraux des tests logiciels (1/3)

- Principe 1 : les tests montrent la présence de défauts
  - Les tests ne peuvent pas prouver l'absence de défaut
  - Les tests réduisent la probabilité que des défauts restent cachés
  - Si aucun défaut n'est découvert, ce n'est pas une preuve qu'il n'en reste pas
- Principe 2 : les tests exhaustifs sont impossibles
  - Sauf pour des cas triviaux, la combinatoire d'un programme est impossible à explorer de façon exhaustive
  - Exemple « extrême » :



Tester tous les cas (2^{64})≅ >500 ans à raison de 1 milliard d'opérations/s

Principes généraux des tests logiciels (2/3)

- Principe 3 : tester tôt
  - Les activités de tests doivent commencer le plus tôt possible dans le cycle de développement du logiciel ou du système.
  - Elles doivent être focalisées sur des objectifs définis compatibles avec les risques et les exigences de qualité
- Principe 4 : Regroupement des défauts
  - En général, un petit nombre de modules concentre la majorité des défauts (règle des 80/20) => rôle des modèles d'erreurs (domaines spécifique)
- Principe 5 : paradoxe du pesticide
  - toute méthode de tests laisse un ***résidu d'erreurs*** contre lesquelles la méthode adoptée est inefficace
    - Corollaire : Le potentiel de détection des défauts d'une suite de test s'épuise
    - Il faut donc constamment renouveler les tests en changeant de point de vue (objectif et stratégie de test)

Principes généraux des tests logiciels (3/3)

- Principe 6 : les tests dépendent du contexte
  - Certains logiciels peuvent être utilisés dans des contextes très différents. Par exemple un système d'information pour un poste de commandement militaire peut être utilisé pour des types de mission très différents (maîtrise de la violence ou combat de haute intensité, par exemple)
    - Les objectifs et les types de tests ne seront pas forcément les mêmes dans ces différents contextes
  - Par exemple, un logiciel de commerce électronique ou un logiciel pilotant un radar seront testés de façon très différente
- Principe 7 : l'illusion de l'absence de défaut
  - Il ne suffit pas de corriger tous les défauts trouvés pour rendre un logiciel utilisable. le logiciel doit :
    - disposer d'une interface utilisateur conforme au métier des usagers
    - être facile à installer et à administrer
    - être facile à faire évoluer
    - etc.
  - En d'autres termes, il doit aussi satisfaire à des caractéristiques non fonctionnelles ISO9126 : Fonctionality, Usability, Reliability, Portability, Efficiency, Maintenability

Quelques aspects essentiels des tests logiciels

- De la nécessité des tests
- Première définition des tests
- Les 7 Principes généraux des tests
- Processus fondamentaux des tests
- Psychologie des tests

Activités principales du processus de test

- Planifier et contrôler
- Analyser et concevoir
- Implémenter et exécuter
- Évaluer les critères de sortie et informer
- Activités de clôture des tests

Ces différentes activités peuvent se chevaucher ou être concurrentes

Planifier les tests

- Activité qui vérifie la mission des tests, définit les objectifs de tests et spécifie les activités de tests de façon à atteindre les objectifs de la mission
- Les tâches majeures de planification des tests sont les suivantes
  - Identifier les objectifs des tests, en déterminer la portée et les risques
  - Déterminer les approches de tests
    - techniques
    - objets de tests
    - couverture
    - identification et interfaces avec les équipes concernées par les tests
    - outils de tests
  - Déterminer les ressources requises pour les tests
    - personnel
    - environnement de tests
    - plate(s)-forme(s) de tests
  - Implémenter la politique de test et/ou la stratégie de tests en liaison avec le plan qualité projet
  - Planifier les tâches d'analyse et de conception des tests
  - Planifier l'implémentation, l'exécution et l'évaluation des tests
  - Déterminer les critères de sortie des tests

Contrôler des tests

- Le contrôle des tests est une activité continue de comparaison de l'avancement par rapport au plan de tests et d'information sur l'état de cet avancement

- Les tâches majeures du contrôle des tests sont les suivantes
  - Mesurer et analyser les résultats
    - Métrique : taux d'échec
  - Mesurer et documenter l'avancement, la couverture des tests et les critères de sortie
    - Métriques
      - ratio nombre d'anomalies/nombre de corrections
      - nombre de problèmes en suspend
  - Décider des actions
  - Initier et suivre les actions correctives

Analyser et concevoir les tests

- Cette tâche recouvre les activités qui permettent de transformer les objectifs généraux du projet concernant les tests en spécifications précises sur les conditions de tests et leur conception
- Les tâches majeures d'analyse et de conception des tests sont les suivantes
  - Rassembler et réviser toute la documentation sur laquelle les cas de tests sont basés : exigences, architecture, conception, interfaces
  - Identifier les conditions de tests , les exigences de tests et les données de tests requises basées sur l'analyse des articles de tests (éléments devant être testés) : leurs spécifications, leur comportement, leur structure
  - Concevoir les tests
  - Évaluer la testabilité des exigences du système (repérer les « vœux pieux »)
    - Exemple de vœux pieux
      - « L'interface utilisateur doit être conviviale »
  - Décrire et concevoir l'environnement de tests et identifier les infrastructures et outils nécessaires : logiciels requis, plate(s)-forme(s) d'exécution des tests, etc.

Implémenter et exécuter les tests

- Cette tâche recouvre l'ensemble des activités qui transforment les conditions de tests en cas de tests et outillage associé (scripts, par exemple) et ou l'environnement de tests est mis en place
- Les tâches majeures d'implémentation et exécution des tests sont les suivantes
  - Développer et établir les priorités des cas de tests, créer les données de tests, écrire les procédures de tests, écrire les scripts de tests pour ceux qui sont automatisés
  - Créer des suites de tests (par famille), à partir des cas de tests pour optimiser l'exécution
  - Vérifier que les environnements de tests ont été mis en place correctement
  - Exécuter les cas de tests manuellement ou en utilisant des progiciels ou des scripts (pour les tests automatisés) en suivant l'ordre des priorités
  - Consigner les résultats de l'exécution des tests à des fins de suivi
  - Comparer les résultats actuels aux résultats attendus
  - Rédiger des rapports d'anomalie concernant les divergences ou les échecs et analyser leur cause (défaut dans le code, dans les données de tests, dans la documentation de tests ou dans la procédure de test)
  - S'il s'agit d'un défaut logiciel, effectuer la correction, r é-exécuter le test, ré-exécuter éventuellement des tests connexes de façon à s'assurer de la non régression

Évaluer les critères de sortie et informer

- Pour chaque niveau de test leur exécution est évaluée en fonction des objectifs qui ont été définis
- L'évaluation des critères de sortie des tests contient les tâches majeures suivantes
  - Vérification du ou des registre(s) de tests (test log) en fonction des critères de sortie spécifiés dans la planification des tests
    - Le registre de test consiste en un enregistrement chronologique des détails pertinents sur l'exécution des tests (IEEE 829 : Software Test Documentation)
  - Évaluer si des tests supplémentaires doivent être élaborés ou si les critères de sortie doivent être modifiés
  - Élaborer un rapport de synthèse sur les tests qui servira lors du bilan du projet ainsi qu'à l'amélioration éventuelle des procédures qualité concernant les tests
- Quelques mauvais critères d'arrêt des tests
  - Plus de budget
  - Plus de temps (« il faut livrer demain »)
  - Le nombre de défauts découverts est jugé « suffisant »

Clôturer les tests

- Cette tâche consiste à rassembler les données des activités de tests terminées de façon à consolider l'expérience
- Les activités de clôture des tests incluent les tâches majeures suivantes
  - Vérifier que les livrables prévus ont été effectivement livrés
  - Clôturer les rapports d'anomalie ou créer des demandes d'évolution pour ceux qui restent ouverts
  - Documenter l'acceptation du système
    - Dans le domaine militaire, la recette par le client (les armées) se conclut par un procès-verbal de qualification
  - Finaliser et archiver l'environnement et les infrastructures de tests pour une utilisation future
  - Fournir toute la documentation et l'outillage de test à l'organisation en charge de la maintenance
    - Le biseau entre le développement et la maintenance doit être géré et ne doit pas s'effectuer au dernier moment
  - Analyser les leçons apprises pour les versions et projets futurs ainsi que pour l'amélioration de la maturité des activités et procédures de tests
    - Rédaction, dans le bilan projet, de la partie concernant les tests à partir du rapport de synthèse sur les tests

Quelques aspects essentiels des tests logiciels

- De la nécessité des tests
- Première définition des tests
- Les 7 Principes généraux des tests
- Processus fondamentaux des tests
- Psychologie des tests

Qui fait quoi ?

- Tests unitaires
  - Ce sont des tests « boîte blanche », ils ne peuvent être conduits que par les développeurs. Le taux de couverture est mesuré durant cette activité
- Tests d'intégration
  - Ils devraient être conduits par une équipe indépendante car
    - Cela oblige les architectes/concepteurs et les développeurs à documenter les interfaces
    - La gestion des interfaces logicielles et matérielles n'est pas du ressort des développeurs
- Tests de validation
  - Ils doivent être conduits par une équipe indépendante car
    - Les tests de validation sont conduits avec une vue « client » (i.e. métier)
    - Ce sont des tests « boîte noire » (ils ne sont pas structurels)
    - Dans certains cas, ils peuvent être conduits par le client lui-même avec ou sans l'aide de la maîtrise d'œuvre (notion de version bêta)
      - Dans ce cas, il faut mettre en place un circuit court de remontée des anomalies/incidents
- Tests non fonctionnels
  - Devraient être conduits par un expert logiciel dépendant du service qualité mais rendant compte au responsable projet

Caractéristiques liées aux activités de test

- Mauvaise approche
  - Succès = pas de défaut mise en évidence
- Bonne approche
  - Tests = moyen d'assurance qualité
  - Corollaire : pas de défaut mise en évidence = échec
- Tout logiciel réel possède des défauts (même après livraison !)
- Les tests sont le dernier rempart contre les défauts résiduels
- Psychologie des managers
  - Difficulté à se convaincre de dépenser tant pour un résultat négatif (découverte de défauts) et de poursuivre les dépenses tant que des défauts sont mis en évidence
- Psychologie des développeurs/testeurs
  - Difficulté à se consacrer à une activité qui consiste à mettre en évidence que des erreurs ont été produites (malgré les procédures qualité de prévention des défauts)
  - Corollaire : les activités de tests sont conduites, en général, avec pour objectif (souvent inconscient) que le système est exempt de défauts ➔ les tests sont construits en conséquence

Conclusion

- Les tests sont une activité complexe mais nécessaire car le logiciel est omniprésent et contient (malheureusement) de nombreux défauts
- Il ne s'agit pas d'éliminer tous les défauts mais ceux pouvant conduire effectivement à une défaillance dans l'utilisation nominale du logiciel
- L'utilisation de méthodes et l'expérience sont primordiales car le risque est grand de ne pas rentabiliser ses efforts
- Les activités de test s'appuient sur une relation de confiance entre les différents acteurs