Test du logiciel

Module NI557-2006fev

Philippe Ayrault

Université Paris 6 et CNAM

Second semestre 2005-2006

Cours 4 : plan

- Les autres types de tests
- L'analyse des tests
- Les outils de tests
- Documentation associée aux tests

Il existe beaucoup de types de tests

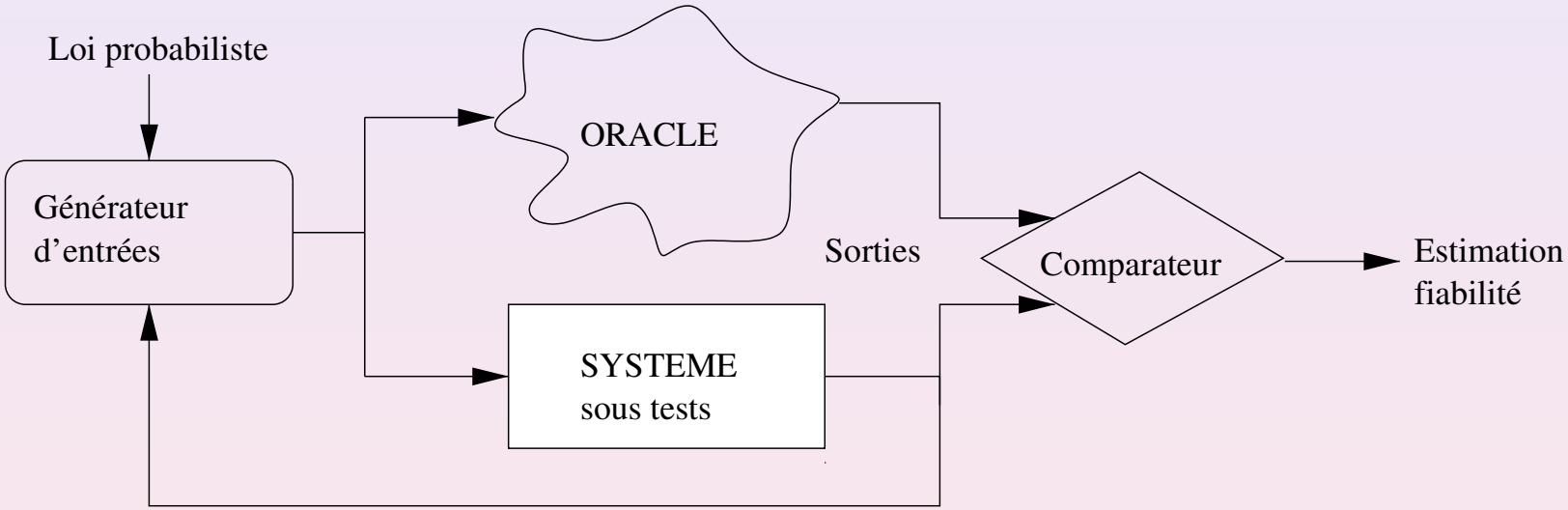
- *tests d'avalanche* permettant de vérifier le comportement du logiciel dans des conditions extrêmes
- *simulation de Monté-Carlo* permettant de simuler du bruit sur un signal ou de produire des erreurs de façon aléatoire afin de vérifier la robustesse du système logiciel
- *tests des temps de réponses et contraintes mémoire* permettant de vérifier le respect des contraintes imposées au système logiciel
- *error guessing* basé sur la connaissance acquise sur les systèmes antérieurs et sur l'expérience du testeur
- ...

Chaque type de tests à pour but de valider un aspect du système logiciel. Il faut donc bien les connaître afin de faciliter le processus de tests.

Les grandes familles des autres types de tests

- Les tests statistiques
- Les tests de non régression

Les tests statistiques : principe



But des tests statistiques

Évaluer la fiabilité d'un système plutôt qu'identifier des anomalies.
Ils sont généralement réalisés après la campagne de tests.
Ces tests consiste à :

- genérer les entrées du système suivant une loi probabiliste (distribution uniforme, distribution stochastique, . . .),
- puis à comparer les résultats fournis par le système avec les résultats d'un "oracle".

Cet oracle peut être un second système dans le cas d'une programmation en N-versions, ou un modèle du système dans un langage plus abstrait (Prolog, Asa, . . .).

Tests statistiques : avantages et inconvénients

Avantages

- tests automatiques. De nombreux modèles probabilistes industriels existent
- reproduction la plus réaliste possible de l'utilisation la plus fréquente du système.

Inconvénients

- choix et validité de la loi probabiliste choisie ainsi que de l'oracle
- la construction d'un oracle peut être aussi complexe que celle du programme à tester. on peut se poser la question de ce que l'on teste (l'oracle ou le système)

Tests de non-régression

- Revalidation d'un système logiciel suite à des modifications
- Identification des tests à réexécuter
- Identification des tests à ajouter

Revalidation

Les logiciels subissent continuellement des modifications (par exemple : maintenance corrective, maintenance évolutive). Il est par conséquent nécessaire d'effectuer une revalidation du logiciel.

Choix des tests

Pour des raisons de coûts et de délais, il n'est pas raisonnable de ré-exécuter l'ensemble des tests effectués sur la version initiale.

But des tests de non-régression

Optimiser le processus de validation d'un logiciel suite à ces modifications. C'est un processus de démonstration de la couverture de l'impact des modifications.

Quand fait-on des tests de non-régression ?

De la même façon que le processus de tests, les tests de non-régression se déroulent durant chaque phase de développement des modifications (ie de la spécification des modifications jusqu'au codage). Ils font partie du processus d'analyse d'impact des modifications du logiciel.

L'analyse des tests

- Justification de la couverture de tests
- Injection de fautes

But

- Assistance à la détermination et à la saisie des jeux de tests
- Détermination d'une mesure quantitative de la couverture de tests
- Détermination des parties du logiciel non atteintes par un ensemble de jeux de tests

Nombreux outils industriels existants : Attol, Devisor, C-Cover, ...

Détermination de la couverture de tests

Les outils de détermination de la couverture de tests permettent de déterminer plusieurs couverture de tests comme :

- la couverture des instructions
- la couverture de branches
- la couverture des décision à décision (LCSAJ)
- la couverture des chemins

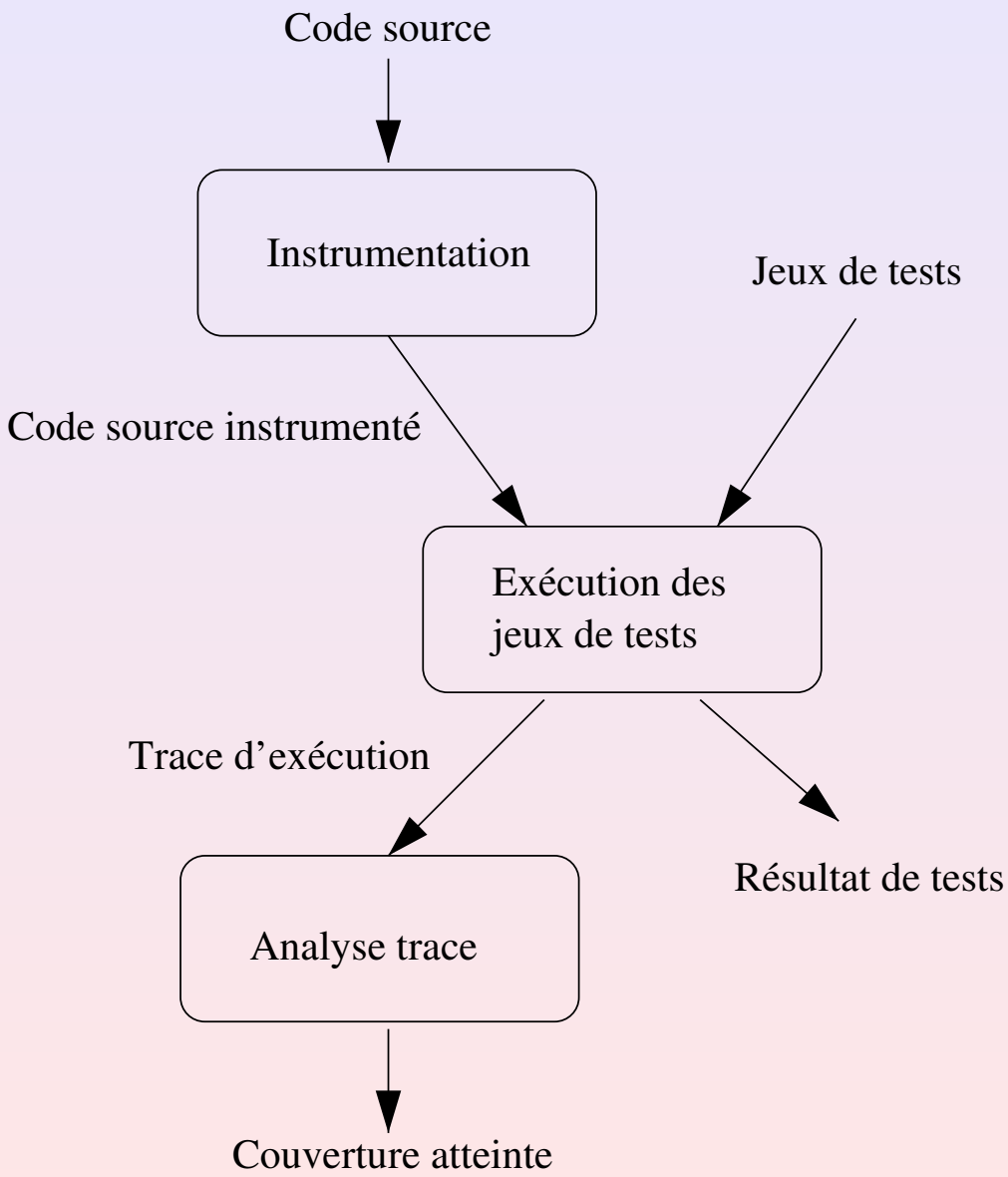
Ces outils ne permettent pas de vérifier le comportement désiré pour le logiciel. C'est-à-dire que la vérifiation des jeux de tests par rapport à la spécification reste à la charge du testeur.

Outils de tests industriel

Il existe de nombreux outils de tests industriel :

- Attol, le plus utilisé à ma connaissance
- Devisor développé par Dassault Electronique
- C-Cover spécialisé dans le langage C
- TestMate de Rational Software
- AdaTest de IPL
- . . .

Les outils de tests (2/2)



Outils de tests : principe

Ces outils fonctionnent en instrumentant le code source. C'est-à-dire qu'il ajoute des instructions de trace dans le code afin de connaitre les parties du code activées par un ensemble de jeux de tests. Ceci a deux désavantages :

- le code instrumenté est beaucoup plus gros que le code de départ (environ facteur 2)
- le comportement du code instrumenté peut être différent du code de départ.

Ceci impose donc de rejouer l'ensemble des jeux de tests sur le code source de départ.

Documentation associée aux tests

Comme le processus de développment, le processus de tests a sa propre documentation. Il existe deux documents à produire pour chaque phase de tests.

- Software Test Plan (STP)
- Software Test Report (STR)

L'ensemble des plans type de la DoD 2167A sont disponible sur le site du Département américain de la Défense.

Software Test Plan (STP)

- Scope
- Software test environment
- Test identification
- Test schedule
- Requirements tracability

Exemple : plan de tests du logiciel de la DoD 2167A

Afin de réaliser des tests, il est nécessaire d'écrire le plan de tests du logiciel. Le plan de tests du logiciel de la DoD 2167A :

- décrit l'environnement utilisé pour les tests,
- identifie des tests à réaliser
- décrit la planification des activités de tests.

- Le chapitre 1 permet d'identifier exactement le système à tester (son nom, son numéro d'identification, la version/release à tester. Il présente aussi une description succinte du système à tester ainsi que l'ensemble de la documentation applicable et de référence.
- Le chapitre 2 présente l'environnement de tests du logiciel.
- Le chapitre 3 présente les tests à réaliser.
- Le chapitre 4 présente la planification de la campagne de tests.
- Le chapitre 5 présente les moyens mis en oeuvre afin d'assurer la traçabilité des tests.

Software test environment

- Software items
- Hardware and firmware items
- Other materials
- Proprietary nature, acquirer's right and licensing
- Intallation, testing and control
- Participing organizations
- Personnel
- Orientation plan

Test identification

- Test levels
- Test classes
- General test conditions
- Test progression
- Data recording, reduction and analysis
- Planned tests

Software Test Report (STR)

- Scope
- Overview of the test results
- Detailed test results
- Test log

Bibliographie

- The art of software testing - G.J. Meyers, J. Wiley (1979, nouvelle édition 2004)
- Software testing techniques - B. Beizer, V.N. Reinhold (1992)
- Software Engineering - M.L. Shooman - Internationnal Student Edition (1985)
- Le test des logiciels - S. Xanthakis, P. Régnier, C. Karapoulios - Hermès (2000)
- Journal of Software Testing, Verification and Reliability - J. Wiley