Tests unitaires

En programmation informatique, le test unitaire est un procédé permettant de s'assurer du fonctionnement correct d'une partie déterminée d'un logiciel ou d'une portion d'un programme (appelée « unité » ou « module »). On écrit un test pour confronter une réalisation à sa spécification. Le test définit un critère d'arrêt (état ou sorties à l'issue de l'exécution) et permet de statuer sur le succès ou sur l'échec d'une vérification. Grâce à la spécification, on est en mesure de faire correspondre un état d'entrée donné à un résultat ou à une sortie. Le test permet de vérifier que la relation d'entrée / sortie donnée par la spécification est bel et bien réalisée. Petit rappel de définitions : Test : il s'agit d'une vérification par exécution. Vérification : ce terme est utilisé dans le sens de contrôle d'une partie du logiciel. (Une « unité » peut ici être vue comme « le plus petit élément de spécification à vérifier»)

Il s'agit pour le programmeur de tester un module, indépendamment du reste du programme, ceci afin de s'assurer qu'il répond aux spécifications fonctionnelles et qu'il fonctionne correctement en toutes circonstances. Cette vérification est considérée comme essentielle, en particulier dans les applications critiques. Elle s'accompagne couramment d'une vérification de la couverture de code (évaluation de la couverture structurelle), qui consiste à s'assurer que le test conduit à exécuter l'ensemble (ou une fraction déterminée) des instructions présentes dans le code à tester. L'ensemble des tests unitaires doit être rejoué après une modification du code afin de vérifier qu'il n'y a pas de régressions (l'apparition de nouveaux dysfonctionnements). L'emploi d'une « stratégie de test » particulière peut limiter les tests à rejouer, par exemple : une analyse d'impact des modifications, corrélée à une preuve d'indépendance des modules, permet de cibler les cas de test unitaire à rejouer. Dans les applications non critiques, l'écriture des tests unitaires a longtemps été considérée comme une tâche secondaire. Cependant, la méthode Extreme programming (XP) a remis les tests unitaires, appelés « tests du programmeur », au centre de l'activité de programmation. La méthode XP préconise d'écrire les tests en même temps, ou même avant la fonction à tester (Test Driven Development). Ceci permet de définir précisément l'interface du module à développer. En cas de découverte d'un bug, on écrit la procédure de test qui reproduit le bug. Après correction on relance le test, qui ne doit indiquer aucune erreur.

Liens avec le référentiel SIO

SLAM4

- A1.3.1 Tests d'intégration et d'acceptation d'un service
- A4.1.6 Gestion d'environnements de développement et de test
- A4.1.8 Réalisation des tests nécessaires à la validation d'éléments adaptés ou développés
- A4.2.3 Réalisation des tests nécessaires à la mise en production d'éléments mis à jour

Savoirs associés :

- Techniques de test unitaire et d'intégration d'un composant logiciel

SI6 Savoirs associés :

- Typologie des tests

Librairies et frameworks

- java : JUnit
- php : PHPUnit

Références

Test Driven Development: By Example - Kent Beck
Scott W. Ambler : une introduction au Développement Guidé par les Tests (TDD)
Institut Agile : Developpement par les tests et présentation des principaux concepts liés