Synthèse de recherche documentaire

Développement Dirigé par les Tests

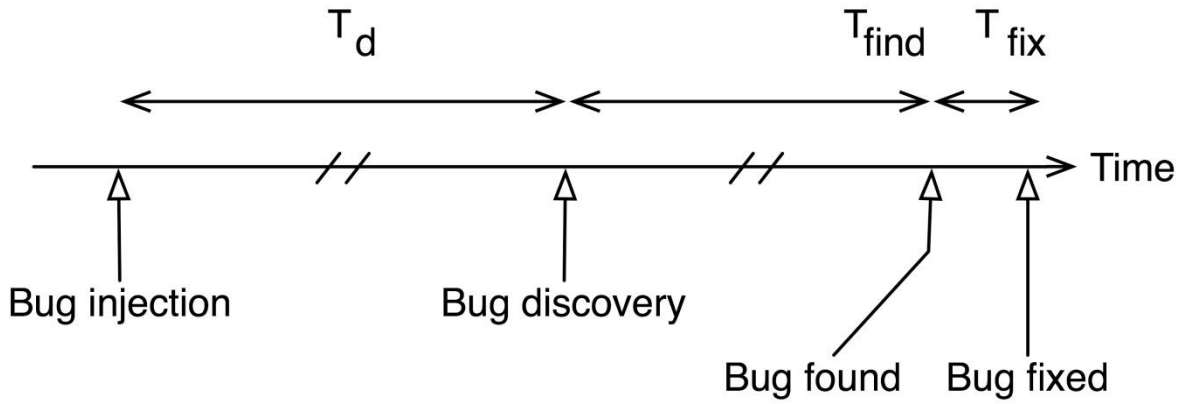
La technique de développement dirigé par les tests implique une méthodologie particulière dans la conception de logiciels ou de composants informatique et ce quel que soit le langage utilisé.

Intérêts d'une telle approche

En effet, la conception des tests dans un premier temps permet plusieurs choses.

La première chose à faire est de déterminer **un comportement exhaustif** du composant que nous développons. Cela se fait en considérant **les tests** comme des **spécifications**. Il est plus facile « à froid » de penser aux cas possibles d'échecs ou d'exceptions qui peuvent survenir avant, plutôt que lorsque nous avons fini le développement. Cela tiens compte de l'esprit humain (revenir sur un travail que l'on pensait fini est quelque chose de difficile) et du moral.

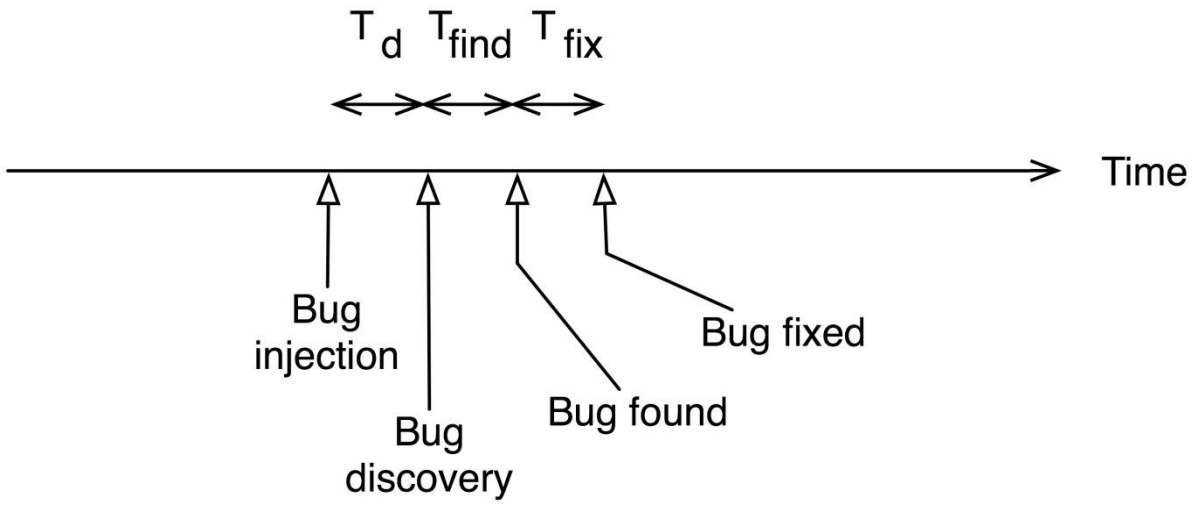
Physics of Debug Later Programming



D'autres parts la conception des tests en premier lieu permet d'**anticiper** au plus juste le **comportement du composant**. Ainsi il permet un meilleur enchaînement dans la conception et l'intégration du programme auquel le composant est lié. Cela se rapproche d'avantage d'un « modèle en V » que d'un de ses prédécesseurs, le « modèle en cascade » qui présentait justement de nombreux défaut suite à l'impossibilité de modifier a posteriori des spécifications ainsi que du code.

Cette approche réduit également le nombre d'erreur du composant et de son programme.

Physics of Test Driven Development

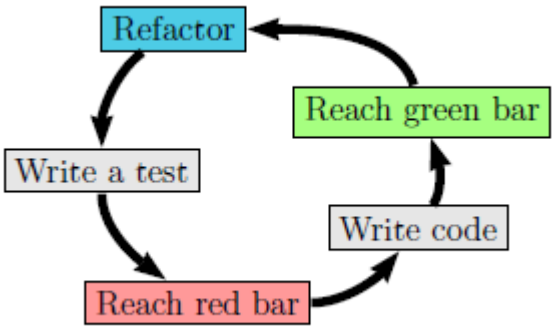


En effet, par cette approche les erreurs (syntaxe/code) sont perçues directement et sont donc de tailles minimes compte tenus du nombre d'itérations importantes en terme de test. C'est donc un **traitement préventif** qui **permet un gain de temps** sur le **long terme.**

Enfin en procédant à des tests unitaires un problème est **directement identifiable** et n'est donc pas lié à un code provenant d'une autre source. L'identification de bugs est donc atomique, on garantit ainsi donc une portabilité et compatibilité de manière sûre par ce procédé.

La méthode elle même

Le **TDD** présente un **cycle itératif** qui se décompose en 5 **étapes.**



- Au préalable on écrit notre fonction, ainsi qu'une seconde qui sert de test. Ensuite on écrit **un test** qui est **destiné à échouer** car le corps de notre composant est vide, ou complété par un bouchon.
- Dans un second temps on **exécute** le test.
- On écrit ensuite le **code minimal** afin de satisfaire le test.
- On **exécute le test** ensuite et vérifie qu'il **satisfait les conditions**.
- Et enfin il s'agit **d'améliorer le code** pour gagner en performance tout en conservant l'intégrité de celui-ci pour son bon fonctionnement.

[Cf. Annexe 1]

Ressources :

Site web :

http://www.labri.fr/perso/fmoranda/it306/td4.pdf

Ouvrage : « Test-Driven Development for Embedded C » by James W.Grenning