HowTo : Fonctionnement du Framework de mutation.

Le framework de mutation est disponible sur github à l'adresse suivant : https://github.com/CorentinHardy/DevOps12

Prérequis :

Afin d'utiliser notre frameworks, il faut dans un premier temps :

- avoir un projet maven avec des tests fonctionnels
- ajouter le plugin spoon dans le pom.xml de votre projet avec la syntaxe suivante :

<plugin><groupId>fr.inria.gforge.spoon</groupId><artifactId>spoon-maven-plugin</artifactId><version>2.1</version></plugin>

- ajouter la dépendance vers le projet de mutation dans le pom.xml:

<dependencies><dependency><groupId>DevopsMutation</groupId><artifactId>DevopsMutation</artifactId><version>1.0-SNAPSHOT</version></dependency>

Nous conseillons d'appliquer des timeout sur les tests unitaires que vous avez afin de ne pas rester bloquer si une mutation venait à induire une boucle infinie par exemple.

Lancement du test par mutation :

Afin de lancer le test, il faut ouvrir un terminal shell puis après s'être mis dans le répertoire framework, lancer le script de lancement des tests par mutations avec la commande suivante :

./devops.sh PATH_VOTRE_PROJET/

Lors du lancement de ce script, vous aurez à préciser le pourcentage d'éléments qui seront transformés par chaque mutation (ou un élément est un bout de code que cible la mutation en question).

Plus ce pourcentage sera élevé, plus il y aura de chance que les mutants soit "mort-nés", c'est-à-dire qu'ils ne passent pas compilation.
Ce pourcentage fait office de sélecteur pour l'ensemble des mutations.
Il doit être entré sous la forme :

pourcentage de mutation a appliquer par processeurs :
0.1
Par la suite, les mutations seront appliquées.

Les mutations disponibles dans ce framework sont :

Processeur BinaryOperatorMutator dans lequel on trouve les transformations suivantes :

les + sont transformés en - et inversement
les / sont transformés en * et inversement
les == sont transformés en !=
les >= sont transformés en >
les =< sont transfomés en <

Processeur BinaryOperatorMutator dans lequel on trouve les transformations suivantes :

les --i sont transformés en ++i
les i++ sont transformés en i--

Rapport de test sur les mutants :

Une page HTML dénommée report.html est disponible dans le répertoire racine du framework. Celle-ci comporte un graphe et un tableau résumant les résultats des tests.
On y trouve : le nombre de codes mutants ayant été générés au total, le nombre de codes mutants ayant été tués à la compilation, le nombre de codes mutants ayant été tués par les tests et le nombre de codes mutants ayant survécu face au test.

En cliquant sur les liens dans le tableau, il est possible d'afficher des informations supplémentaires :

-Si vous cliquez sur le nombre de mutants ayant été générés au total vous pourrez voir tout les processeurs différents et si le code généré associé est mort né, tué par les tests ou a survécu.
-Le nombre de mutants morts nés pointe sur une page qui affiche le nom des processeurs associés aux mutants morts nés.
-En cliquant sur le nombre de mutants tué, vous arriverez sur une page affichant le nom de chaque processeur correspondant à une version du code ayant été tué par les tests. De plus pour chaque processeur vous aurez un tableau qui montre le résultat de chaque classe de test en affichant si les tests ont été réussis, ou échoués. Nous indiquons également si les l'échec à eu lieu à cause d'un test "failed" ou d'une erreur dans les tests.

-Le nombre de mutants ayant survécu au test vous amène sur une page qui affiche les processeurs ayant engendré un code mutant ayant survécu au test. Vous pouvez également pour chaque processeur cliquer sur un lien vous listant les lignes et les classes dans lesquelles des modifications ont été effectuées (si la page n'est pas trouvable, c'est qu'aucune modification n'a été effectuée dans le code source, et vous pouvez donc ignorer ce mutant).