Tools for course sw verification and testing

Angelo Gargantini

March 4, 2020

Testing di codice (2)

- Per l'esecuzione dei test usiamo JUnit `http://junit.org/`. E' già integrato con eclipse.
- Per la copertura usiamo:
  - 

codecover `http://codecover.org/` perchè ci aiuta nella valutazione del MCDC
    - la versione "ufficiale" non è aggiornata, per questo abbiamo una versione @unibg: `https://github.com/fmselab/codecover2`
  - anche consigliato eclemma `http://www.eclemma.org/`
- per la generazione di casi di test, abbiamo visto:
  - randoop: `https://code.google.com/p/randoop/`
  - evosuite: `http://www.evosuite.org/`

- randoop: la cosa migliore è mettere il tutto in una

New Randoop Launch Configuration

Launch Configuration Parameters

Output folder does not exist and will be created on launch

Output Folder: testrandoop

Package Name:

Class Name: RandoopTest

Stopping Criterion Stop test generation after:

Randoop has generated 100000000 tests, OR

Randoop has generated tests for 10 seconds

Test Output Parameters

Output tests that: Pass or Fail

Code verification (3)

- JML
- OpenJML `http://openjml.org/` con il plugin di eclipse
  - OpenJML provides a standard Eclipse plug-in update site at `http://jmlspecs.sourceforge.net/openjml-updatesite`.
- ci sono anche altri tool ma sono sconsigliati
- Key per JML
  - key base può essere trovato qui `http://www.key-project.org/`
  - per scaricare il plugin eclipse (per kepler): `http://www.key-project.org/download/releases/eclipse/kepler`
  - nota che c'è anche un plugin per editing di contratti JML in KEY
  - altro materiale: quicktour

Formal verification (4)

- model checker NuSMV: `http://nusmv.fbk.eu/`
- plugin per eclipse (unibg): `http://nuseen.sourceforge.net/`

Model Based testing (5)

- per il combinatorial testing usiamo CitLab: `https://code.google.com/a/eclipselabs.org/p/citlab/`
- per il testing basato su FSM usiamo modelJUnit: `http://sourceforge.net/projects/modeljunit/`