MUTATION TESTING FOR DSLs

TOOL DEMO

P. Gómez-Abajo, E. Guerra, J. de Lara
Modelling & Software Engineering Research Group
http://miso.es
Universidad Autónoma de Madrid (Spain)

Mercedes G. Merayo
Design and Testing of Reliable Systems Research Group
http://antares.sip.ucm.es/testing
Universidad Complutense de Madrid (Spain)

DSM 2019
October 20th, Athens
WHAT IS MUTATION TESTING?

- Approach of software testing to assess the quality of test suites
- Injection of syntax changes in a program by using mutation operators
- The mutations introduced simulate common programming faults
- Facilitates to improve the quality of the test-suites and the mutation operators set
MUTATION TESTING FOR FA

Seed model

Test-suite

00 → ✓
01 → ✗
MUTATION TESTING FOR FA
MUTATION TESTING FOR FA

Seed model

Mutant model

Test-suite
00 → ✓
01 → ✗

Test-suite
00 → ✓
01 → ✗

The mutant is alive ✗
MUTATION TESTING FOR FA

Seed model

Test-suite

<table>
<thead>
<tr>
<th>Input</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>✓</td>
</tr>
<tr>
<td>01</td>
<td>✗</td>
</tr>
<tr>
<td>10</td>
<td>✓</td>
</tr>
</tbody>
</table>
MUTATION TESTING FOR FA

Seed model

Test-suite

<table>
<thead>
<tr>
<th>Input</th>
<th>Pass</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>✓</td>
</tr>
<tr>
<td>01</td>
<td>✗</td>
</tr>
<tr>
<td>10</td>
<td>✓</td>
</tr>
</tbody>
</table>

Mutant model

Test-suite

<table>
<thead>
<tr>
<th>Input</th>
<th>Pass</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>✓</td>
</tr>
<tr>
<td>01</td>
<td>✗</td>
</tr>
<tr>
<td>10</td>
<td>✗</td>
</tr>
</tbody>
</table>
MUTATION TESTING FOR FA

Seed model

Mutant model

Test-suite

Test-suite

The mutant is killed

✓
A SIMPLE WODEL PROGRAM

• Wodel: a DSL for model mutation

generate exhaustive mutants in "out/" from "model/"
metamodel "mm/fa.ecore"
description "Simple Wodel program"

with blocks {
    sts "Swap the symbols of two transitions" {
        t = select one Transition
        where {symbol <> null}
        modify one Transition
        where {symbol <> t->symbol}
        with {swap(symbol, t->symbol)}
    }
}

• Wodel tool extension for language independent MT  Wodel-Test
WODEL-TEST PROCESS

MT tool specification

Language support
- Metamodel
- M2T transf.
- T2M transf.

Mutation support
- Mutation operators (WODEL)
- Equivalence criteria

Execution support
- Program compilation
- Test execution

Generates

MT tool

Program under test
- Test cases

Tester

MT tool creator

Wodel-Test is also useful to build MT tools for common programming languages:

Wodel-Test for Java
CONCLUSIONS

• Wodel-Test: automates the generation of MT tools for arbitrary languages
• Eases the creation of MT tools for the MDE community
• Implemented 3 MT tools: Wodel-Test for FA/Java/ATL

FUTURE WORK

• We plan to create MT tools for other DSLs
• Reuse of generic mutation operators across similar languages (e.g., Java and C++)
• Support for the definition of functions in the Wodel DSL
Thank you!!

Wodel & Wodel-Test project websites:

http://gomezabajo.github.io/Wodel/
http://gomezabajo.github.io/Wodel/Wodel-Test/

Pablo.GomezA@uam.es
@GomezAbajo
WODEL-TEST ARCHITECTURE

2. WODEL-TEST
   - postProc

1. WODEL engine
   - Registry handler
   - syntactic equivalence
   - semantic equivalence

3. «interface»
   LanguageService Provider
   - projectToModel()
   - modelToProject()
   - annotateMutation()
   - compile()
   - run()

4. JAVA MT SPEC
   - «uses»
   - «includes»
   - WODEL programs
     (meta-model + mutation op.)