APPLICATION OF MODEL-BASED TESTING TO DYNAMIC EVALUATION OF FUNCTIONAL MOCKUP UNITS

Cláudio Gomes, Romain Franceschini, Nick Battle, Casper Thule, Kenneth Lausdahl, Hans Vangheluwe, Peter Gorm Larsen
COSIMULATION: OBJECTIVE

A technique to combine simulators to approximate ideal solution [5].

Case Study by Kristof Berx, Davy Maes, and Klaas Gadeyne, from Flanders Make

http://overturetool.org/

http://www.blensor.org/

https://nl.mathworks.com/products/simulink.html

INTERFACES

Overture

MATLAB SIMULINK

Controller

Body

Env.

Control Simulator

Model

Solver

Controller

Body Simulator

Model

Solver

Body

Env. Simulator

Model

Solver

Env.

setIn(...)

getOut(...)
doStep(..., H)
MOTIVATION

Barriers in FMI adoption (ordered by importance):
- Lack of transparency in features supported by FMI tools.
- Insufficient documentation and a lack of examples, tutorials, etc.
- It is difficult to implement FMUs.
- There is a lack of tools that sufficiently support FMI.
- Difficulties in practical aspects, like IT-prerequisites in cross-company collaboration.
# RELATED WORK: GOAL

<table>
<thead>
<tr>
<th>Contribution</th>
<th>Static Test FMU</th>
<th>Dynamic Test FMU</th>
<th>Test Importing Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>FMU Compliance Checker [1]</td>
<td>✅</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FMU-SDK [2]</td>
<td></td>
<td>✅</td>
<td></td>
</tr>
<tr>
<td>Reference FMUs [3]</td>
<td></td>
<td>✅</td>
<td></td>
</tr>
<tr>
<td>Battle et al. [4]</td>
<td>✅</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FMI Cross Check Repo [5]</td>
<td></td>
<td>✅</td>
<td></td>
</tr>
<tr>
<td>FMI MOBSTER [6]</td>
<td></td>
<td>✅</td>
<td></td>
</tr>
</tbody>
</table>

APPROACH OVERVIEW

Model Based Testing (MBT): the use of a model of the System-Under-Test (SUT) in order to guide test case generation.

- yEd Graph Editor: https://www.yworks.com/products/yed
- Graph ML Syntax: Brandes, Ulrik, Markus Eiglsperger, Jürgen Lerner, and Christian Pich. Graph Markup Language (GraphML), 2013.
EDGE IMPLEMENTATIONS

class FMIGraphModel extends ModBatGraphModel:
    // SUT and Time
    val instance = IFmiComponent
    var t = 0
    ...

    // Edge Methods
    def e_Instantiate() = |
    instance = instantiate(fmu, getGuid(fmu))
    }
    def e_SetInit() = |
    // Pick a random var from the
    // INI set of variables
    // Pick a value (e.g., nominal value), and
    // invoke the corresponding
    // instance operation.
    setVar(Variable(randomElement(INI))
    }
    def e_Terminate() = |
    val s = instance.terminate()
    assert(s == Fmi2Status.OK)
    }
    def e_Step() = |
    // Choose a step size according to
    // FMU Capabilities
    var H = chooseSelf()
    // Execute the step, and
    // check if the step was carried out
    val res = instance.doStep(t, H, true)
    assert(res == Fmi2Status.OK)
    t = t+H
    }
    def e_Free() = |
    instance.freeInstance()
    }
    ...
}
RELATED WORK: APPROACH


THE DISTRIBUTED CO-SIMULATION PROTOCOL FOR THE INTEGRATION OF REAL-TIME SYSTEMS AND SIMULATION ENVIRONMENTS

Martin Krammer, Martin Benedikt
VIRTUAL V3HCLE Research Center
Graz, Austria
{martin.krammer,martin.benedikt}@esi-iti.gmbh

Torsten Blochwitz, Khaled Alekeish
ESI-ITI GmbH
Dresden, Germany
{torsten.blocchwitz,khaled.alekeish}@esi-group.com

Nicolas Amringer
4SPACE GmbH
Paderborn, Germany
namringer@4space.de

Christian Kater
Leibniz Universität Hannover
Hannover, Germany
kater@sim.uni-hannover.de

Stefan Materne, Roberto Ruvoloču
TWT GmbH
Stuttgart, Germany
{stefan.materne,roberto.ruvolocu}@twt-gmbh.de

Klaus Schacht, Josef Zehetner
AVL List GmbH
Graz, Austria
{klaus.schacht,josef.zehetner}@avl.com

Micha Damm-Norwig
Ks.MicroNova GmbH
Kassel, Germany
micha.damm-norwig@micronova.de

Viktor Schreiber
University of Ilmenau
Ilmenau, Germany
viktor.schreiber@uni-ilmenau.de

Natarajan Nagarajan
ETAS GmbH
Stuttgart, Germany
natarajan.nagarajan@etas.de

Isidro Corral
Robert Bosch GmbH
Stuttgart, Germany
lcorral@de.bosch.com

Tommy Sparber
Spant Microw Electronic Design GmbH
Graz, Austria
t.sparber@meds.at

Serge Klein, Jakob Andert
RWTH Aachen University
Aachen, Germany
{klein_se, andert}@vka.rwth-aachen.de
SYNTACTIC SUGAR

Address FMI-Specific Challenges:

- Many edges between same src and trg states
SYNTACTIC SUGAR

- Edges shared by many src states
SYNTACTIC SUGAR

- Repeat operation a finite number of times
SYNTACTIC SUGAR

- Decomposition: Edge and State merging
EXPERIMENTS: OVERVIEW

169 FMUs

FMU Supplier

FMU

FMI Cross Check Repository

Our Tool

GraphML

FMIOps.scala

Reductions

EFSM

Modbat

Results

Model based on FMI State machine

EXPERIMENTS: MODEL

Number of tests per FMU: 1000
Self-loop limit: 10

Select variables at random.
Set value to nom/1

Rollback functions
Step size={0.001, 0.01, 0.1}

Select variables at random.
Set value to nom/1
EXPERIMENTS: STATISTICS

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>FMUs passing all tests.</td>
<td>77</td>
</tr>
<tr>
<td>FMUs failing at least one test.</td>
<td>55</td>
</tr>
<tr>
<td>Total FMUs tested (sum above two).</td>
<td>169</td>
</tr>
<tr>
<td>Failures analyzed</td>
<td>102</td>
</tr>
</tbody>
</table>
EXPERIMENTS: COMMON FAILURES

1. The FMU does not recognize the value reference for a variable declared in its model description. Such a variable is set during initialization mode.
2. After an FMU is terminated, it fails when a variable belonging to the set X is queried.
3. The getRealStatus operation is not supported after an instance is terminated.

Figure 11: Calling sequence of Co-Simulation C functions in for
EXPERIMENTS: COMMON FAILURES

4. During stepping mode, a tunable parameter (i.e., a scalar variable with causality="parameter" and variability="tunable") is changed. The FMU then logs a message that it cannot be changed, and returns an error.
5. URI has multiple possible formats for the absolute path of a file, and some FMUs only support one. This causes a failure in the instantiation of the FMU.

6. The outputs are queried after a change in the inputs, without a doStep in-between, causing the FMU to return an error.

7. The reset operation (mandatory in the standard) is not implemented.

8. Some FMUs do not isolate instances in the sense that one failed operation in an instance of an FMU will affect the outcome of other operation calls in a different instance of the same FMU.

9. A variable was set with a value that is outside the scope of an FMU (even when picked respecting the boundaries set by the FMU).
LIMITATIONS

- FMI Cross Check repository is a moving target.
- 1000 Tests per FMU cover 95% of all cases for that FMU (including possible variables set and get).
- Most tests do not run co-simulations until the end (we have FMU Compliance Checker for that).
- Choice of default values for parameters can cause tests to fail for correct FMUs. Seldom happened (14 out of 14733 tests).
- Asynchronous operations not considered.
LESSONS LEARNED

- FMUs mix numerical with model parameters
  • Better documentation on what these do,
  • Facilitate configuration of co-simulations
- FMUs do not disclose constraints on variables
  • Co-simulations crash and it’s hard to know why.
LESSONS LEARNED

- Different interpretations of feedthrough: the outputs are queried after a change in the inputs, without a doStep in-between, causing the FMU to return an error.
  
  • Can be tested per FMU: \( \text{get}_c(\text{set}_c(s_c, u_c, v_1), y_c) \neq \text{get}_c(\text{set}_c(s_c, u_c, v_2), y_c) \).

7 (out of 113) FMUs implement feedthrough.
Some tools changed approach in between versions.

After discussions in FMI Steering committee, FMUs (for FMI 2.0.1) should not support feedthrough.

Feedthrough is important for physical couplings:
SUMMARY

FMI Cross Check Repository

Lessons learned
Most common failures

Results

1000 tests per FMU

https://msdl.uantwerpen.be/git/claudio/FMIMOBSTER
THANK YOU!

Department of Engineering - Cyber-Physical Systems
Åbogade 34
building 5341
8200 Aarhus N
Denmark
claudio.gomes@eng.au.dk