Integrating Formal Verification and Simulation of Hybrid Systems

Rodin Multi-Simulation Plug-in

Vitaly Savicks, Michael Butler, John Colley
Problem

• Traditional verification and validation methods are not sufficient for the high assurance of safety and reliability

• Rigorous analysis of multi-domain complex systems is difficult

• Formal methods are limited in modelling continuous domain

• Heterogeneous nature of hybrid systems makes it difficult to use a single development tool

• Different domain-specific tools for individual components are not integrated
Tool Integration

- Open languages for physical modelling
- Tool and platform-independent model exchange and co-simulation standards
- Automated formal analysis of discrete-event systems

Functional mockup interface for dynamic models
Event-B

• Simple modelling notation of set theory and first-order logic

• State variables, invariants and events

• Key features of abstraction and refinement

• Rodin open platform
  ‣ Automatic proof obligation generation
  ‣ Automated and interactive provers
  ‣ Plug-in extensions for requirements traceability, language extension, model-checking, UML modelling, code generation
Functional Mock-Up Interface

0 \quad t_{start} \quad t_{ci} \quad t_{ci+1} \quad t_{stop} \quad \text{simulation time}

\[ h_{ci} \]

Instances

Classes

Subsystem A
Solver Tool A
FMU (DLL)

Subsystem B
Solver
Simulation Tool
Multi-simulation Plug-in

- FMI v1.0 Java library for continuous model simulation
- ProB 2.0 for Event-B simulation and validation
- Generic master simulation algorithm
- Flexible mapping of Event-B models (timed or non-timed) to simulated components via *read* and *wait* events to support non-determinism and refinement
- Graphical component composition and simulation environment
Master Algorithm

1. Initial I/O

2. Evaluate $D_i$ every $h_{Di}$

3. Evaluate $C_j$ if connected to an evaluating $D_i$ that either reads the input or has the output changed

4. I/O at the end of evaluation

Component evaluations:

<table>
<thead>
<tr>
<th>time (s)</th>
<th>0</th>
<th>3</th>
<th>4</th>
<th>6</th>
<th>8</th>
<th>9</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

if changed output
Example: Voltage Control

Distribution voltage control system in Modelica

Event-B state machine of the OLTC controller
Component Diagram
Simulation Results

- Simulation time = 50s
- Step size = 0.1s
- Nominal V = 230V
- Deadband = 2V
- Detection t = 10s
- Mechanical t = 1s

Graphs showing:
- Distribution voltage
- Tap position
- OLTC controller state
Conclusions

• Generic solution for hybrid systems development that facilitates **formal verification**, tool-independent model composition and co-simulation

• **Generic master** algorithm based on FMI 1.0

• **Flexible mapping** of Event-B models (timed or non-timed) to simulation components that supports refinement

• Tool that enables **rigorous analysis** (using Event-B) of the **discrete** aspect of hybrid systems and the **simulation-based analysis** of interactions with the **physical** environment