Validating UML models of Embedded Systems by Coupling Tools

Jozef Hooman
Embedded Systems Institute, Eindhoven
University of Nijmegen

Nataliya Mulyar, Ladislau Posta
Eindhoven University of Technology

The Netherlands

SVERTS, 11 October 2004
Overview

• Context, motivation; Boderc project
• Aim of this work
• Tool overview
  – Rose RealTime of IBM/Rational
  – Matlab/Simulink of The Mathworks
• Main concepts of the coupling
• Concluding remarks
CIP: Océ, produces document printing systems

Multi-disciplinary development of software-controlled machines

Typically (sometimes conflicting) requirements on throughput, reliability, costs, power consumption, temperature, noise, etc.
Multi-disciplinary development

Current development often rather sequential

Mechanical Engineering;
Electrical Engineering;
Software Engineering

- Decisions/changes in one discipline may cause problems in other disciplines
- Often leads to non-optimal solutions

BODERC project: project partners from different disciplines (M,E,S) work together on topics such as
- multi-disciplinary design space exploration
- predict consequences of design decisions asap

Main idea: use high-level models
Software modeled in Rose RealTime (Rose-RT)

- Executable UML models, code generation, based on specific run-time system
- Models can be executed, event flow can be presented as a Sequence Diagram
- At Océ: reusable software architecture in Rose-RT

Mechanical design partly in Matlab/Simulink

- Interconnected (discrete of continuous) blocks
- Behaviour can be simulated using solvers
- At Océ: used to model parts of abstracted mechanical behaviour
Combine tools of different disciplines to allow simultaneous simulation; here Rose-RT and Simulink

**Aim**

Challenges:
- **Conceptual:** e.g. different notions of time & data
- **Technical:** implement connection between tools
Capules: active objects, message-based communication via ports

transitions triggered by messages or time-outs

actions on transitions may send messages or set timers
Tool can generate code based on Service Layer, with e.g. Timing Service

Implementation of Service layer depends on target platform, e.g. Timing Service depends on granularity of underlying operating system

Two ways to test generated code:
1. Run executable model on intended platform
2. Execute model step-by-step to check reactive response to messages, correct timing is not guaranteed

Step: processing next message with highest priority
\[
F = m \cdot a \quad a = \frac{F}{m} \quad \text{motor position } \varphi = \int \int a
\]

\[\begin{align*}
\text{Gain} & \quad \frac{1}{s} \quad \frac{1}{s} \\
\text{Integrator} & \quad \text{Integrator 1} \\
\end{align*}\]

F = \text{reference signal} \quad \varphi : \text{actual position}
Matlab/Simulink

- **phi_M1**: Motor 1
- **gain_1*num_C1(s)/den_C1(s)**: Motion Controller 1
- **phi_M1**: Motor 1
- **phi_M1_ref**: Reference signal
- **paper position**: Paper position
- **transmission ratio**: Transmission ratio
- **pinch radius**: Pinch radius
- **motor position**: Motor position
- **pinch rotation**: Pinch rotation

Closed-Loop system 1
**Simulation:** compute at certain intervals, using a notion of *simulation time*, outputs, inputs and internal states (which may have discrete and continuous part) of all blocks

**Solver:** computes next states, for continuous part using certain numerical integration method

**Step size** (length of interval) can be fixed or variable
Main concepts

Notion of time

Timing of Rose-RT models:

• strongly coupled to timing service of operating system on which model is running
• not respected in step-by-step simulation

Decision: use notion of simulated time of Simulink

Assumption: information about execution time of transitions is available, reflecting assumption about underlying target platform
Decision: loose coupling via separate Multidisciplinary Coupling Tool (MCT)

- Interface via external ports
- Control steps in RRT get data and exec. time
- Time from Simulink
- Deal with timers (now 1)
TriggeRed SubSystem

MCTSfunction implemented by S-function
Concluding Remarks

First prototype, tested on very small examples

TO DO:

• Experiment with larger & industrial examples
• Test extension with multiple timers
• Investigate multiple threads in Rose-RT
• Experiment with various solvers
• Try simulation on multiple PCs
• Use TrueTime to model part of software within Matlab or network delays