Co-simulation Environment for Rodin: Landing Gear Case Study

Vitaly Savicks, Michael Butler, John Colley

University of Southampton, United Kingdom

Abstract. This work in progress presents a prototype multi-simulation environment for the Rodin platform that enables import, co-modelling and co-simulation of dynamic models and formal Event-B specifications, which can help in the design of mixed discrete-event/continuous-time systems. The proposed solution is based on the Functional Mock-up Interface standard and ProB animator for Event-B. The involved technologies and co-simulation semantics are explained, followed by a demonstration of preliminary results, obtained from a landing gear case study.

1 Introduction

Hybrid system models are a mixture of discrete-event and continuous-time components that can be domain specific and therefore require different development tools [1]. At the same time the complexity of systems, in particular within a safety-critical domain, demands the application of formal methods [2]. Both challenges can be addressed by integrating the existing technologies [3]. We propose a simulation-based collaborative approach that combines the Event-B [4] development in the Rodin platform [5] and co-simulation with tool-independent physical components via the FMI interface [6]. The approach is demonstrated on a landing gear example, modelled in Event-B and Modelica [7], and co-simulated in the tool.

2 Modelica and FMI

The Modelica language and the Functional Mock-up Interface standard for Model Exchange and Co-simulation are designed to facilitate tool integration and interoperability. While the Modelica language provides an object-oriented equation-based notation that is natural for describing physical processes in a structural way, the FMI interface enables the exchange and co-simulation of models from any tool that supports the standard by exporting them as a shared library (FMU) with a common interface and model description format. The co-simulation of exported FMI units is performed by the master algorithm that must be designed by the simulation host.

3 Rodin Multi-Simulation

Our co-simulation environment RMS (Rodin Multi-Simulation) provides a generic master algorithm and an extensible simulation component meta-model that currently implements Event-B and FMI components, which map to Event-B machines and FMI units, respectively. The environment allows diagrammatic composition of components via input/output ports and visualised simulation, coordinated by the master and executed in fixed-size simulation steps (communication points), which is a standard simulation approach. The simulation semantics of Event-B components is defined by the IO events, executed at the beginning of a simulation step to read the inputs, a sequence of proceeding events that master selects non-deterministically, and Wait events that mark the end of the step. The simulation of Event-B is performed via master by the ProB animator [8].

4 Landing Gear Experiment

RMS environment has been exercised on a landing gear system [9] that consists of a cockpit interface, a discrete controller and a continuous mechanical/hydraulic plant. The task of the system is to control the manoeuvring of landing gears based on the input from the cockpit and the sensors of the plant. The initial experiment uses a simplified version of the original specification, in a sense that it omits the triplication of each sensor and implements a single control module and a single manoeuvring sequence, i.e. gear extension. The plant (analogical switch, electro-valves and landing gear/door hydraulic cylinders) has been modelled in Modelica (Figure 1) and exported from the Dymola tool [10] as an FMU.

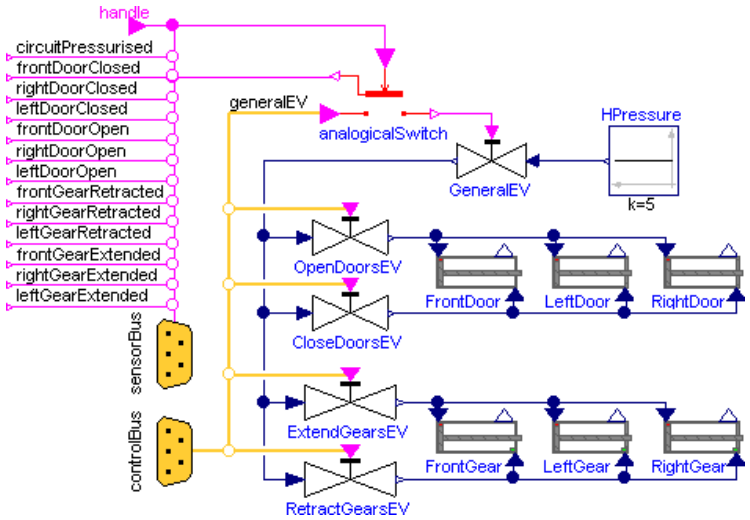

Fig. 1: Modelica model of the mechanical/hydraulic plant

In the model of Figure 1 there are Open, Close, Extend and Retract electro valves for the hydraulic supply to the doors and the gears, hydraulic cylinders for each of the doors and gears along with a source of hydraulic pressure, a general supply valve, and sensor and actuator ports. The behaviour of each component is defined by equations in Modelica. For example, an electro valve has the following specification:

```
model ElectroValve
  parameter Real closingTime = 1.0 "Closing duration";
  parameter Real openingTime = 3.6 "Opening duration";
protected
  parameter Real Rmax = 1.0 "Max opening";
  parameter Real dRcl = Rmax/closingTime "dR when closing";
  parameter Real dRop = -Rmax/openingTime "dR when opening";
  Real R(start = 0.0) "Current opening (0-open, 1-closed)";
  discrete Real dR(start = 0.0);
equation
  Hout = Hin*R;
  der(R) = dR;
algorithm
  // closing/opening event
  when E then
    dR := dRcl;
  elsewhen not E then
    dR := dRop;
  end when;
  // limiter of the R value
  when R <= 0 or R >= Rmax then
    dR := 0;
  end when;
end ElectroValve;
```

The controller was initially modelled in Modelica and StateGraph2 [11], and later in Event-B as a deterministic state machine (Figure 2) via the Statemachines plug-in for Rodin [12]. The *sExtending* state of the state machine comprises three parallel regions that model the state of the general electro-valve (top), doors (middle) and gears (bottom), and are synchronised by sensor-triggered transitions, such as *stopExtending*. The latter, for example, corresponds to the following Event-B event:

event *stopExtending*
where
$sDO = TRUE$
$sGE = TRUE$
$gearsExtended = TRUE$ `// gears are extended (from sensor)`
then
$sDO := FALSE$
$sGE := FALSE$
$sDelayDC := TRUE$

$sStopGE := TRUE$
$openEV := FALSE$ `// stop opening door`
$extendEV := FALSE$ `// stop gear extension`
$timerDC := 1$ `// start the 0.1s 'contrary order' delay (open/close door)`
end

Being the proof of concept the initial model does not yet incorporate safety and timing invariants and does not use the refinement. Refinement of state machines would allow us to refine the controller model towards an implementation and verify its correctness using the Rodin provers.

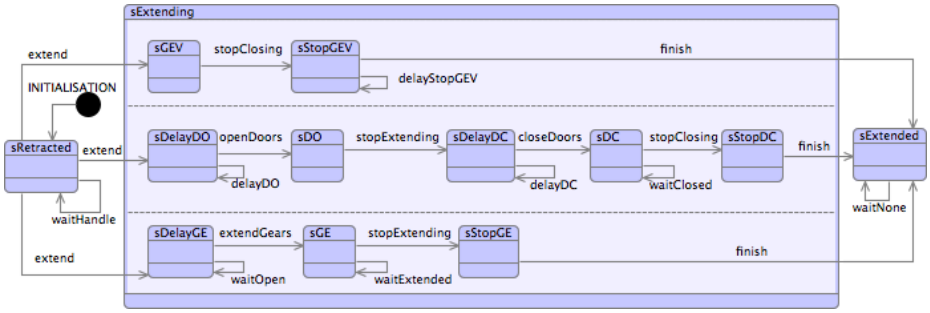

Fig. 2: State machine of the manoeuvring controller in Event-B

The cockpit and plant FMUs and the Event-B controller have been composed and successfully simulated in the RMS environment, demonstrating the expected behaviour when compared to a purely physical simulation in Modelica. The obtained results in terms of the controlled and monitored signals are illustrated in Figure 3, which shows the dynamic behaviour of the physical components (Open, Close and Extend electro valve output pressure, door and gear cylinder position) and how the dynamics changes in response to actuation signals from the Event-B controller. For instance, it is possible to observe a delay between the *generalEV* signal and the output pressure growth in the *OpenDoorsEV*, caused by the 0.8s transition duration from open to closed of the analogical switch.

5 Conclusion and Further Work

The presented work demonstrates the feasibility of a generic integration and co-simulation of Event-B formal models and multi-domain physical models that is aimed at combining formal verification and simulation-based analysis of hybrid systems. Our future R&D steps include a stronger formal analysis of the co-simulation semantics [13], development of an adaptive and deterministic master algorithm [14–16] and comparison of the proposed solution with traditional simulation approaches on a number of case studies, including the complete specification of the landing gear system.

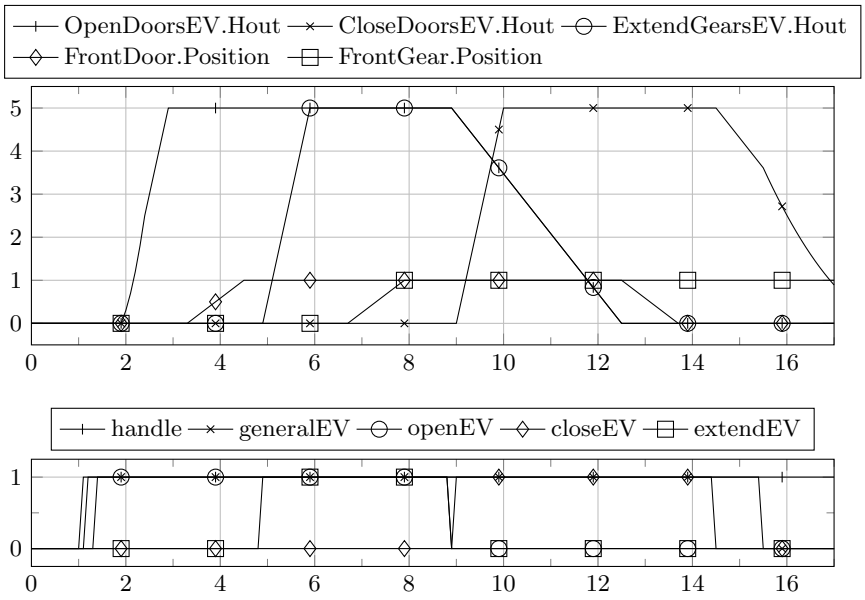

Fig. 3: RMS simulation results (time = 17s, step = 0.1s)

Acknowledgment: This work is part of the ADVANCE Project (Advanced Design and Verification Environment for Cyber-physical System Engineering) funded by the European Commission (`http://www.advance-ict.eu`).

References

1. Lee, E.A.: Cyber physical systems: Design challenges. In: International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing (ISORC). (May 2008) Invited Paper.
2. Gnesi, S., Margaria, T.: Formal Methods for Industrial Critical Systems. Wiley Online Library (2013)
3. Marwedel, P.: Embedded and cyber-physical systems in a nutshell. (2010)
4. Abrial, J.: Modeling in Event-B: system and software engineering. Cambridge University Press (2010)
5. Abrial, J., Butler, M., Hallerstede, S., Hoang, T., Mehta, F., Voisin, L.: Rodin: an open toolset for modelling and reasoning in Event-B. International Journal on Software Tools for Technology Transfer (STTT) **12**(6) (2010) 447–466
6. Blochwitz, T., Otter, M., Arnold, M., Bausch, C., Clauß, C., Elmqvist, H., Junghanns, A., Mauss, J., Monteiro, M., Neidhold, T., et al.: The Functional Mockup Interface for tool independent exchange of simulation models. In: Modelica'2011 Conference, March. (2011) 20–22
7. Fritzson, P., Engelson, V.: Modelica — a unified object-oriented language for system modeling and simulation. In: ECOOP'98—Object-Oriented Programming. Springer (1998) 67–90

8. Leuschel, M., Butler, M.: ProB: an automated analysis toolset for the B method. International Journal on Software Tools for Technology Transfer **10**(2) (2008) 185–203
9. Boniol, F., Wiels, V.: Landing gear system. `http://www.irit.fr/ABZ2014/landing_system.pdf` (2013)
10. Brück, D., Elmqvist, H., Mattsson, S.E., Olsson, H.: Dymola for multi-engineering modeling and simulation. In: Proceedings of Modelica. (2002)
11. Otter, M., Malmheden, M., Elmqvist, H., Mattson, S.E., Johnsson, C.: A new formalism for modeling of reactive and hybrid systems. In: Proceedings of the 7th International Modelica Conference, Linköping University Electronic Press (2009) 364–377
12. Savicks, V., Snook, C., Butler, M.: Event-B wiki: Event-B Statemachines. `http://wiki.event-b.org/index.php/Event-B_Statemachines` (2011)
13. Gheorghe, L., Bouchhima, F., Nicolescu, G., Boucheneb, H.: Formal definitions of simulation interfaces in a continuous/discrete co-simulation tool. In: Rapid System Prototyping, 2006. Seventeenth IEEE International Workshop on. (June 2006) 186–192
14. Hines, K., Borriello, G.: Dynamic communication models in embedded system co-simulation. In: Proceedings of the 34th annual Design Automation Conference, ACM (1997) 395–400
15. Schierz, T., Arnold, M., Clauß, C.: Co-simulation with communication step size control in an FMI compatible master algorithm. In: 9th International Modelica Conference. Munich. (2012)
16. Broman, D., Brooks, C., Greenberg, L., Lee, E.A., Masin, M., Tripakis, S., Wetter, M.: Determinate composition of FMUs for co-simulation. In: Embedded Software (EMSOFT), 2013 Proceedings of the International Conference on, IEEE (2013) 1–12