D2.1.1 – Modelica extensions for properties modelling

Part 0: Context, objectives and summary

WP2.1 – Properties modelling language
WP2 – Properties modelling and Safety

MODRIO (11004)

Version 2.0

Date 04/27/2016

Authors

Audrey Jardin	EDF
Daniel Bouskela	EDF
Thuy Nguyen	EDF
Wladimir Schamai	EADS
Eric Thomas	Dassault Aviation
Hilding Elmqvist	Dassault Systèmes AB
Martin Otter	DLR
Alfredo Garro	University of Calabria
Andrea Tundis	University of Calabria
Peter Fritzson	University of Linköping
Lena Buffoni	University of Linköping
Martin Sjolund	University of Linköping

Executive summary

The main goal of sWP2.1, entitled “Properties modeling language”, is to formalize system properties into Modelica and to use such properties models for validating system simulated behavior with respect to the expected one. It hence aims at automating and increasing the coverage of the tests an engineer can perform to validate the design he has in mind.

The objective of deliverable D2.1.1 is to produce*:*

(1) A new architecture for property modeling that integrates well into typical systems engineering life cycles.

(2) The specification of a new language for property modeling that is well adapted for operation engineers.

(3) The Modelica extensions for property modeling based on the new language specifications.

The specifications are illustrated using the complex case-study of the BPS (backup Power System).

Deliverable D2.1.1 is divided into 5 parts:

- Part 0: Context, objectives and summary (the present document)
- Part 1: Users motivation
- Part 2: Modeling architecture for property modelling
- Part 3: Specification of FORM-L a new language for property modeling
- Part 4: Modelica for property modelling

The prototypes developed in the scope of deliverables D2.1.3 through D2.1.6 (Dymola, SimulationX, OpenModelica, Wolfram SystemModeler) will be based on the specifications of deliverable D2.1.1.

The methodology and tools developed will be tested on industrial use-cases from the energy domain (deliverable D8.1.3) and aircrafts domain (deliverable D8.4.2).

Table of contents

Glossary

BPS	Backup Power System
FMI	Functional Mockup Interface
WP	Work Package
sWP	sub Work Package

1. Context and objectives

1.1. Context

General context of the MODRIO project

The objective of the ITEA2 MODRIO (Model Driven Physical Systems Operation) project is to extend state-of-the art modeling and simulation environments based on open standards to increase energy and transportation systems safety, dependability and performance throughout their lifecycle [1].

The project is focused on solving the following main technical issues:

- Properties modeling, to automate and increase the coverage of system verification and validation;
- Automatic generation of dependability and safety analysis models from Modelica behavioral models, including possible hybrid stochastic aspects;
- Hybrid system state estimation, to initialize as accurately as possible simulators from the real state of the system for diagnosis and prognosis, including the handling of uncertainties;
- Systems with multiple operating modes, to simulate models beyond normal operating conditions, and make it possible to explore the full chain of consequences from an initiating event on the system, starting from the best estimate of its actual state, and going through all possible modes, including dysfunctional ones;
- Modelling and simulation infrastructure and services, for model verification and debugging, new compilation techniques for parallel execution and multi-mode simulation, cloud computing simulation services for system dynamics, remote monitoring of assets.

General context of the Work Package 2

Work Package 2 is directly linked to the first two technical issues stated above as its structuring into two sub-Work Packages (sWPs) can testify.

The main goal of sWP2.1, entitled “Properties modeling language”, is to formalize system properties into Modelica and to use such properties models for validating system simulated behavior with respect to the expected one. It hence aims at automating and increasing the coverage of the tests an engineer can perform to validate the design he has in mind.

In parallel, the objective of sWP2.2, entitled “Safety analysis methods”; is to refine the granularity of safety and dependability analyses by relying on models that include more physics. It aims at using Modelica models to automatically generate Bayesian Networks and Fault Tree Analysis. It also investigates the extension of Modelica to represent hybrid stochastic models so as to perform dynamic dependability analyses.

These two research areas should be conducted with strong interconnections despite the fact that there applications occur at design stages that may appear now as rather separate in the V-cycle of the system development. They indeed should be well synchronized since:

(1) They both involve some language extensions that should not be conflicting.

(2) Properties can be used to describe the assertions on which safety and reliability analyses are based. For instance, a properties model may formalize the limits upon which system components are considered as dysfunctional.

(3) Safety and reliability analyses require the introduction of stochastic aspects to model the random failures that may occur in the system. Such stochastic aspects can also be used to state from which rate or to quantify how well a property should be considered as satisfied/violated.

At a first sight, WP2 seems then more related to design activities. Its results can however be applied to the domain of system operation, the reason being that the design requirements must fulfill the

operation requirements.

1.2. Links with other WPs

One of the major challenges for building a consistent assistance tool for operation lies in the possibility of predicting well the different consequences of the actions that a human operator as well as external or internal events may have on the system. In other words it deals with the capability of simulating systems with multiple operating modes. WP2 in combination with WP3 (state estimation) and WP4 (hybrid state machine) can help to structure such kind of simulations since:

(1) Properties can be used to define operation rules or the domain of each system operating mode.

(2) State estimation can be used to model system degradation such as wear.

(3) The hybrid state machine can be used to model the consequences of human actions, such as operation actions (determinist), or the consequences of system degradation, such as wear leading to possible breakage (stochastic).

1.3. Results from previous projects

First results have already been obtained in the course of EUROSYSLIB [2] and OPENPROD [6] projects.

EUROSYSLIB introduced the notion of property modeling for the formal expression of requirements, and the use of property models as automatic observers to perform automatic testing of system design vs. requirements. To that end, a first architecture was produced [4][5], but that was not sufficient to guarantee an efficient integration of property modeling into the systems engineering lifecycle.

OPENPROD made a thorough experiment on EDF power plant case-studies using ModelicaML for property modeling [7]. The result was encouraging but not conclusive [8].

OPENPROD also introduced the notion of uncertainties into Modelica models and produced requirements for Modelica extensions [9][10][11]. In the course of MODRIO, the requirements were used for the design of custom annotations.

1.4. Objective and structure of deliverable D2.1.1

The objective is to produce*:*

(1) A new architecture for property modeling that integrates well into typical systems engineering life cycles.

(2) The specification of a new language for property modeling that is well adapted for operation engineers.

(3) The Modelica extensions for property modeling based on the new language specifications.

Deliverable D2.1.1 is divided into 5 parts:

- Part 0: Context, objectives and summary (the present document)
- Part 1: Users motivation
- Part 2: Modeling architecture for property modelling
- Part 3: FORM-L specifications
- Part 4: Modelica for property modelling

To illustrate part 3, it might be worthwhile to also read the detailed case-study reported in *D8.1.3 - Part 1 - The Backup Power Supply (BPS)*.

The summaries of Part 1 though Part 5 are given in the second chapter of Part 0 (the present document).

A new methodology for using property modeling to test the compliance of system design vs. requirements will be presented in deliverable D2.1.2.

The new Modelica extensions proposed in Part 4 will be prototyped in the target tools listed in sWP2.1 (Dymola, SimulationX, OpenModelica, Wolfram SystemModeler) in the scope of deliverables D2.1.3 through D2.1.6.

2. Summary

2.1. Part 1: Users motivation

Facing ever increasing constraints in terms of safety, environmental or economical performances, systems become more and more complex. Therefore engineers need more powerful tools to take decisions rapidly and meet the “time-to-market” rule. Modeling and simulation tools at a system level (like Modelica environments) are now rather well adopted in the industries to assist engineers in that task. However, such tools focus until now only on efficient simulation of the behavior of the system. They do not propose any particular assistance for conducting verification tests. Usually the engineer has at his disposal one or several behavioral models that capture different aspects of the design choices for the system and runs several simulations to investigate whether the system requirements are satisfied or not. The relevance of the tests hence conducted depends on how well the models and test scenarios represent the system behavior w.r.t. the actions that may be performed on the system. The idea behind property modeling is that better model and scenario quality may be achieved when using formal requirements of the system. Experiment conducted at EDF R&D showed that formalizing requirements with property modelling indeed improves the quality of the requirements themselves by removing ambiguities, omissions or inconsistencies. It is expected that the property model of the requirements may be used as an observer to conduct the verification test automatically to detect possible violations in the requirements, and that it will be possible to generate automatically test scenarios from the property models. Automating the production of test scenarios and test runs should improve significantly the test coverage and therefore the demonstration that the system operates properly.

In fact, such approach will improve many engineering processes all along the system lifecycle, from its design to its operational phase. In particular, it will:

- Improve the specification phase by providing an explicit and unambiguous list of system requirements (including assumptions on the environment of the system and designer’s assertions regarding the system internal behavior). Formulating the requirements into formal statements will indeed enhance their legibility and avoid potential misunderstandings of their meanings. New static tests on the requirements model may also be imagined to automatically check the coherence of the system requirements or to point out their incompleteness;
- Ease the capitalization and the transmission of knowledge by having at a glance a functional point of view on how the system should behave (through system requirements, properties models document also operating domains and system functional constraints) whereas such expertise is rather long to develop over time especially for complex systems. As the requirements become executable under the form of property models when associated with a behavioral model, newcomers will indeed be able to virtually “play with the system” by running several simulations and then better understand the reason of each system requirement;
- Better keep track (and hence better assess the impact) of the evolutions of the system requirements (e.g. due to regulation evolutions, to changes in operational expectations …) simply by updating the appropriate property models;
- Improve the validation phase by automating the checking procedures. Such automation will indeed give more rigor to the tests (they will be less prone to human errors, the different items being examined systematically and not by hand anymore). It will also potentially increase the test coverage (more test scenarios will be simulated);

- Enable new engineering studies by supporting advanced modeling techniques like mode switching. Property models can indeed be used to describe for instance the limits beyond which the system enters a dysfunctional mode. It can thus help to simulate models beyond normal operating conditions and make it possible to explore the full chain of consequences from an initiating event on the system. This will be very helpful in the design phase to complement the classical penalizing scenarios for the sizing of the system with new scenarios that challenge the system beyond its normal operating conditions, and in the operation phase to better diagnose the system state.

2.2. Part 2: Modeling architecture for property modeling

Verifying that the design and operation of complex physical systems such as power plants satisfy the specified requirements may be quite difficult, due to the large number of requirements and test scenarios that must be considered to have satisfactory test coverage. The systems considered here exhibit a large number of dynamic continuous and discrete states representing respectively physical states and logical modes such as normal, dysfunctional and control system modes. A tool-independent modular modeling & simulation architecture is presented to automate as much as possible the testing phase for the verification of such systems.

Five types of models are considered: (1) requirements models to express formally the requirements, (2) architecture models to express the system design, (3) behavioral models to express the system dynamic behavior, (4) observation operators to observe dynamic variables from the behavioral model, and (5) binding models to connect together models (1) to (4) without modifying them in order to produce automatically the full verification model (cf. Figure 1).

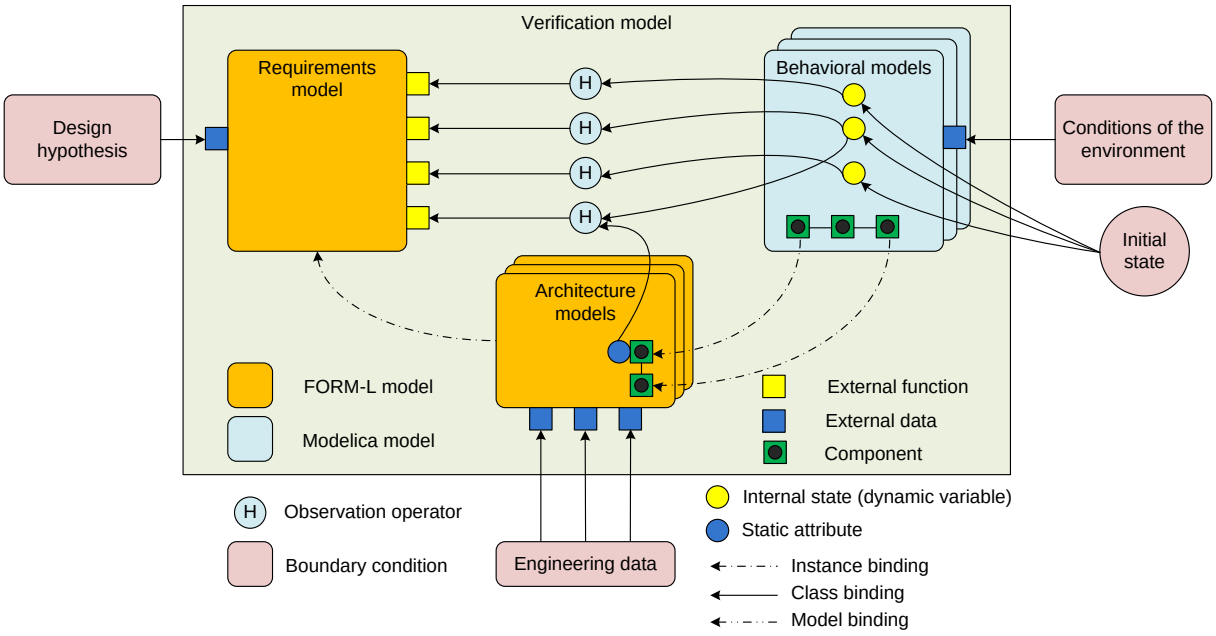

Figure 1: verification model complete with the boundary conditions and the initial state

The general principle governing the method is that the whole system encompassing the physical process and its control system should be considered, that requirements should be issued independently from design choices, and verified against several possible design architectures. Testing is performed by simulating the behavioral model together with the requirements model used as an automatic observer to detect possible violations in the requirements. Emphasis is put on the original aspects of the method, in particular the use of quantifiers to express generic requirements independently from design choices, the use of observation operators to allow the independent production of the requirements and behavioral models, and the use of 3-valued logic in order to

account for the fact that it cannot always be decided using simulation whether requirements are satisfied or not.

The methodological principles governing the different types of models are presented using the intermediate cooling system of a nuclear power plant as an example. Then a formal approach is used that is independent of the syntaxes of both FORM-L and Modelica languages to explain how FORM-L complements Modelica.

As a conclusion, two solutions are possible regarding the implementation of FORM-L: FORM-L as an extension of Modelica or as a separate language from Modelica. The first solution has been chosen at the WP2 workshop in Munich held on June 13, 2013.

This work is a revised an extended version of an original paper published in the proceedings of the ISA POWID 2014 symposium entitled 'Novel Open Source Modelling Architecture for the Design Verification against System Requirements using Simulation'.

2.3. Part 3: FORM-L specifications

This part presents the main notions and a syntax for the FOrmal Requirements Modelling Language (FORM-L) that is proposed to extend MODELICA. FORM-L is intended to support systems engineering methodologies. It is also intended to support the development and verification of operational aids. In this sense, its usefulness goes far beyond safety issues.

The main notions introduced by FORM-L are:

- Property models, modelling configurations.
- Properties, Requirements and Assumptions.
- Instructions and Actions.
- Continuous and Discrete Time Locators.
- Time Domains and Time Domain Interfaces.
- Events.
- Finite State Automata and Statecharts.
- Sets.

To illustrate the notions and the syntax proposed, extensive examples are given. Most are taken from a case study that has been used to develop the language, the Backup Power Supply (BPS) system. A brief overall description of that case study is given in the document to give a context to the examples and facilitate their understanding. A detailed description of the case study is given in document MODRIO D8.1.3 *Part 1 - The Backup Power Supply (BPS)*.

In the future, the concrete syntax proposed for the FORM-L language might be adapted to develop FORM-L based compilers (i.e. tool solutions independent from Modelica).

2.4. Part 4: Modelica for property modeling

This part presents a proposal on how to concretely implement "Properties modeling" with the Modelica language utilizing the FORM-L language design as inspiration. The following results have been achieved:

- Extensions to the Modelica language have been developed to achieve a more user-friendly, FORM-L like syntax for properties modeling in Modelica. Prototypes in Dymola and/or OpenModelica have been implemented for all these proposals.
- An open source Modelica Library Modelica_Requirements library has been developed (provided in directory D2.1.1\AdditionalMaterial) containing about 150 models/blocks and 60 functions to define requirements formally in Modelica, check them automatically when a model is simulated using blocks from this library and summarizing the result of the check at the end of a simulation. This

library has been evaluated with Dymola, OpenModelica and SimulationX.

- Part of the EDF use case for properties modeling, a Backup Power Supply system (D8.1.3-Part I), has been implemented and provided as example in the Modelica_Requirements library.
- Several aircraft specific examples from Dassault Aviation have been added to the Modelica_Requirements library.
- In order that formally defined requirements can be practically used, they must be associated in a convenient way to the corresponding behavioral model (the so called "mapping"). Within the MODRIO project several proposals have been developed, Modelica Change Proposals provided, and prototypes implemented. However, there is not yet a common agreement about the "best" approach. There seems to be agreement for one part of the "mapping" within a Modelica model: Extracting automatically all needed variable values from a model by casting a model instance to a record (containing these variables) and then utilizing this record in a requirements model, see (*MCP-0023, 2016*).
-

3. References

[1] Bouskela D., *MODRIO Full Project Proposal*, version 2.1, January 2013.

[2] *ITEA2 EUROSYSLIB Project*, information available at: http://www.eurosyslib.com/

[3] *ThermoSysPro library*, information available at: https://www.modelica.org/libraries/thermosyspro

[4] Jardin A., Bouskela D., Nguyen T., Ruel N., Thomas E., Chastanet L., Schoenig R., Loembé S., *Modelling of System Properties in a Modelica Framework*, in Proceedings of the 8th International Modelica Conference, Dresden, Germany, March 20-22, 2011.

[5] Jardin A., Nguyen T., Ruel N., *EUROSYSLIB Project – sWP7.1 – Properties modeling*, EDF technical report, H-P1C-2011-00913-EN, August 2011.

[6] *ITEA2 OPENPROD Project*, information available at: http://www.ida.liu.se/labs/pelab/OpenProd/

[7] Schamai W., Jardin A., Bouskela D., *Industrial Use Cases for Requirements Verification and Model Composition in ModelicaML*, 7th MODPROD Workshop on Model-Based Product Development, Linköping, Sweden, Feb. 5-6, 2013.

[8] Bruel. G, Jardin A., *OPENPROD Project – WP6 – I&C functional validation based on the modelling of requirements and properties: evaluation of ModelicaML*, EDF technical report, H-P1A-2012-03040-FR, February 2013.

[9] Bouskela D., Jardin A., *Handling of Uncertainties with Modelica*, 8th International Modelica Conference, Dresden, Germany, March 20-22, 2011.

[10] Jardin A., Bouskela D., *Handling of Uncertainties in OpenModelica : Data Reconciliation and Propagation of Uncertainties*, 5th OpenModelica Annual Workshop, Linköping, Sweden, Feb. 4th, 2013.

[11] Jardin A., Bouskela D., OPENPROD Project – WP6 – *Handling Uncertainties on EDF Models of Energy Systems, EDF technical report, H-P1C-2012-02254-EN, to be released.*

[12] MCP-0023 (2016): Model to Record. Modelica Change Proposal 0023. https://svn.modelica.org/projects/MCP/public/MCP-0023_ModelToRecord