Co-funded by the Erasmus+ Programme of the European Union

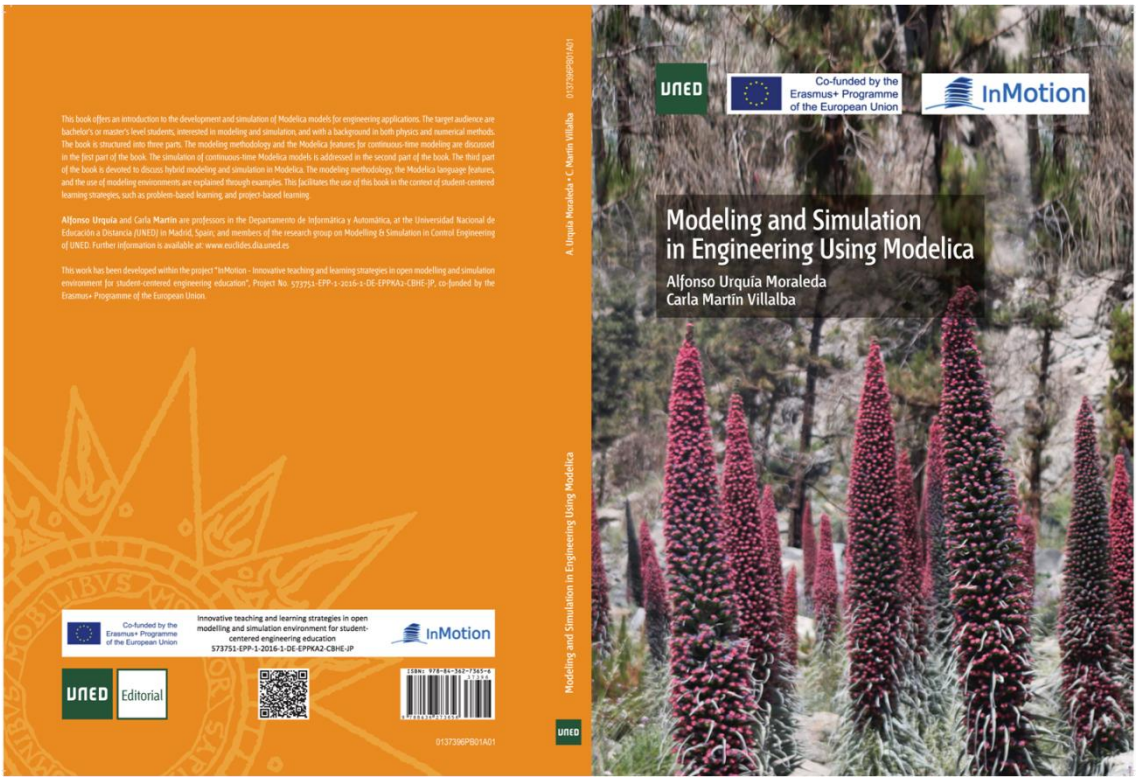
Innovative teaching and learning strategies in open modelling and simulation environment for student-centered engineering education Projekt-Nr.: 573751-EPP-1-2016-1-DE-EPPKA2-CBHE-JP

InMotion

Modeling and simulation in Engineering using Modelica

Alfonso Urquía and Carla Martín

Textbook abstract



This book offers an introduction to the development and simulation of Modelica models for engineering applications. The modeling methodology, the Modelica language features, and the use of modeling environments are explained through examples. This facilitates the use of this book in the context of **student-centered learning strategies**, such as problem-based learning, and project-based learning. The **target audience** are bachelor's or master's level students, interested in modeling and simulation, and with a background in both physics and numerical methods.

The book content has been structured into the following three parts: continuous-time modeling, simulation of continuous-time models, and hybrid system modeling and simulation. Each part is composed of three lessons. The learning objectives of each lesson are summarized in Tables 1, 2 and 3, which are placed at the end of this document.

The **first part of the book** is devoted to explain the physical modeling paradigm, the object-oriented modeling methodology, and the Modelica features for continuous-time modeling, covering the development in Modelica of atomic and composed models, and model libraries.

The physical modeling paradigm and the object-oriented modeling methodology are discussed in Lesson 1. The basics of Dymola and OpenModelica modeling environments are also explained in the first lesson. Students are encouraged to replicate by themselves, using any of these modeling environments, the model simulations described in the subsequent lessons.

The description in Modelica of atomic models, this is, models that are not composed of smaller components, is introduced in Lesson 2. The system behavior is represented in these models using equations and algorithms. System modeling examples in the electrical, mechanical and thermal domains are used for illustrative purposes.

Model reuse by hierarchical model composition and inheritance, and implementation and use of model libraries, are discussed in Lesson 3 through a series of examples. For example, a rectifier circuit that was described as an atomic model in Lesson 2 is now modeled from a different perspective: a Modelica library of electrical components is developed, and the circuit is composed instantiating and connecting model classes from the library. Other model libraries are developed in the mechanical and thermal domains, and applied to analyze longitudinal vibrations and heat conduction in a bar. The implementation of the object-oriented modeling methodology in Modelica is illustrated using a tank system with level and temperature control loops.

The **second part of the book** addresses the simulation of continuous-time models. The aim is to explain the minimum key concepts whose knowledge is required for understanding the messages generated by Modelica modeling environments during the model translation and simulation. Readability has been favored over mathematical rigor. Students are introduced to the model analyses and symbolic manipulations automatically made by the Modelica modeling environments, and the numerical methods for differential-algebraic equation (DAE) systems. Addressed topics include computational causality assignment, index of DAE systems, selection of state variables, and numerical methods for DAE systems.

After studying the first three lessons of the e-book, students know that model behavior can be described in Modelica using equations and algorithms. They also know that the Modelica modeling environments automatically determine the variable to evaluate from each equation, and also the evaluation order of the model equations and algorithms. The concepts behind these analyses of the model computational structure are explained in Lesson 4. For the sake of simplicity, incidence matrices are employed in this discussion, instead of bipartite graphs.

A step in the computational causality analysis described in Lesson 4 is to check whether the model is structurally singular. Some structurally singular models are incorrect. However, as explained in Lesson 5, this is not always the case, existing different approaches for solving this kind of systems. The basics of the approach implemented by Dymola and OpenModelica are explained. Also, the definition of DAE index is given, and its relationship with the numerical properties of the DAE system is discussed. Other

topics addressed in Lesson 5 are the initialization of DAE systems, and the selection of state variables. Dynamic selection by the modeling environment and static selection by the model developer are both considered.

Once the model has been transformed into an efficient solvable form, applying (among others) the methods explained in Lessons 4 and 5, numerical algorithms are applied to simulate the model evolution over time. Some basic concepts related to the to these numerical algorithms are explained in Lesson 6. The discussion is structured into three parts: solution of algebraic loops, numerical integration of ordinary differential equations, and numerical integration of algebraic-differential equations. Only those concepts that are useful for using the Modelica modeling environments and understanding their diagnosis and error messages are introduced.

The **third part of the book** is dedicated to hybrid modeling and simulation with Modelica. In this context, hybrid models are those that combine continuous-time behavior with events. A formal specification of hybrid models and its relationship with the simulation algorithm on one hand, and with the Modelica description on the other hand, are discussed in Lesson 7. The initialization of Modelica models, this is, the calculation of the model variables at the initial time, is also explained in this lesson.

A model is said to have a variable structure if its mathematical description can change during the simulation run. Models of this type can be in different modes, being each mode described by a particular system of equations. During the simulation run, transitions between modes are taking place according to predefined conditions, producing the corresponding changes in the model mathematical description. Modelica supports certain type of models with a variable structure. This is explained in Lesson 7.

Numerical methods for detecting and handling events are described in Lesson 8, favoring simplicity over mathematical rigor. The use by Modelica modeling environments of crossing functions for event detection is also explained. The discussion offers the minimum knowledge required for understanding the issues associated to the event description in Modelica, providing insight into how to design and implement Modelica models that can be simulated efficiently, without generating errors.

The phenomenon known as chattering is described and illustrated by means of an example in Lesson 8. A simulation exhibits chattering if the number of state events executed during the simulation is large in comparison with the number of integration steps. Chattering is problematic as it significantly slows down the simulation. Modelica allows to avoid chattering in some cases. In other cases, the only way to avoid chattering is to modify the modeling hypotheses.

Lesson 9 is devoted to hybrid modeling practice using Modelica. The actions that Modelica allows to perform in an event are basically of the following three types: generate a change in the model mode; update the value of discrete-time variables; produce abrupt changes in the value of continuous-time state variables. Models containing these different types of event are described in Lesson 9. Among the examples

of variable structure models are electric circuits containing electric switches or diodes, mechanical systems exhibiting dry friction, reversible direction of liquid flow in a tank system, and reversible direction of heat transfer through a wall.

Table 1: Lessons and learning objectives of Part I.

Part	Lesson	Learning objectives
I. Continuous-time modeling	1. Modeling methodology and tools	– Relate the following concepts: physical modeling paradigm; equation-based modeling languages; computational causality of the model; non-causal modeling; causal modeling; object-oriented modeling methodology; object-oriented modeling language. – Discuss features of the Modelica modeling language. – Discuss the sequence of stages performed by the Modelica modeling environments for translating the Modelica model into executable code. – Discuss the simulation algorithm for hybrid DAE systems implemented by the Modelica modeling environments. – Use Dymola and OpenModelica for editing, debugging and translating Modelica models, experimenting with them, and analyzing the simulation results.
	2. Continuous-time atomic models	– Declare scalar and array variables in Modelica. – Relate basic types and attributes. – Use the Modelica.SIunits and Modelica.Constants packages of the MSL. – Declare and call functions in Modelica. – Describe the continuous-time behavior using equations, algorithms and functions, involving scalar and array variables. – Develop continuous-time atomic models in Modelica.
	3. Model libraries	– Design model libraries applying the object-oriented modeling methodology. – Define connectors, components and compound models in Modelica. – Use class inheritance in Modelica. – Use replaceable classes in Modelica. – Describe in Modelica models with a regular structure, using both arrays of components and arrays of variables. – Define and use record classes in Modelica. – Use the if-clause and for-clause of Modelica. – Use the inner/outer construct of Modelica. – Develop libraries of continuous-time models in Modelica. – Develop models by inheriting and instantiating classes defined in Modelica libraries.

Table 2: Lessons and learning objectives of Part II.

Part	Lesson	Learning objectives
II. Simulation of continuous-time models	4. Computational causality	– Relate the following two concepts: equation-based modeling language, and assignment of computational causality. – Analyze manually the computational causality of small-dimension models. – Analyze the computational structure of overdetermined and underdetermined DAE systems. – Write the simulation algorithm of small-dimension, non-singular DAE systems.
	5. Index and initialization of DAE systems	– Discuss how high-index DAE systems are manipulated to reduce their index by Modelica tools such as Dymola and OpenModelica. – Calculate the index of small-dimension DAE systems. – Discuss the difficulties associated to the numerical solution of high-index DAE systems. – Formulate the initialization problem of small-dimension DAE systems, making explicit (if any) the hidden constraints. – Manipulate small-dimension DAE systems for selecting the state variables. – Select the state variables in Modelica. – Discuss the principles of the dynamic state selection supported by Dymola.
	6. Numerical methods	– Relate modeling hypotheses with algebraic loops, and stiffness. – Discuss difficulties associated to the symbolic manipulation and the numerical solution of non-linear algebraic loops. – Perform tearing of small-dimension algebraic loops. – Classify ODE integration methods attending to the following criteria: explicit and implicit; single-step and multi-step; order; and fixed and variable step size. – Discuss the principles of the DASSL integration algorithm, and the inline and mixed-mode integration methods.

Table 3: Lessons and learning objectives of Part III.

Part	Lesson	Learning objectives
III. Hybrid system modeling and simulation	7. Hybrid system specification	– Formulate hybrid-DAE models according to the OHM formalism. – Given the description of a hybrid-DAE model according to the OHM formalism, formulate the simulation algorithm of the model. – Describe events in Modelica using when and if clauses. – Describe variable structure models in Modelica. – Describe model initialization conditions in Modelica.
	8. Event detection and handling	– Discuss how simultaneous events are handled in Modelica. – Discuss how event chains are executed by Modelica modeling environments such as Dymola and OpenModelica. – Discuss the event detection mechanism based on crossing functions that is implemented in Modelica modeling environments such as Dymola and OpenModelica. – Discuss how the trigger time of events is calculated by Modelica modeling environments such as Dymola and OpenModelica. – Use the noEvent operator of Modelica. – Detect and avoid chattering.
	9. Hybrid modeling practice	– Develop multi-mode models in Modelica. – Design, develop and use hybrid model libraries in Modelica.

The textbook written by Alfonso Urquía and Carla Martín (UNED).
Editorial UNED (Madrid, Spain), 2018, ISBN 978-84-362-7365-6