# 7. Current state of Modelica

## 7.1 Language specification

While the first version of the **language specification document** released September 1997 focused on continuous-time systems, the specification was and is still continuously subject to improvement and enhancement, cf.

https://modelica.org/documents

The evolve of specification versions is motivated by

1. attempting a unique interpretation of the language among various tool vendors and
2. providing further improved and new features and capabilities

A sample of established **new features** in the past was for describing and supporting:

- discrete systems (versions 1.1, 1.2 and 1.4)
- external interfaces to C/Fortran routines (V1.2)
- array semantics (V1.3, V2.0)
- enumeration types (V2.0)
- hybrid paradigms and state machines (Ferreira and Oliveira 1999; M. Otter and Årzén 2005) (V2.2)
- stream connectors exhibiting bi-directional flow (Franke et al. 2009b) (V3.1)
- operator overloading (Olsson et al. 2009) (V3.1)
- synchronous controllers (Elmqvist, Otter, and Mattsson 2012) (V3.3)

The maintenance and progress of the language design among other issues are initiated

through well-defined procedures and are discussed in periodic **design meetings** organized by **the Modelica Association (MA)** typically 3-4 times a year, cf.

https://modelica.org/events/design_meetings

Decisions are officially electable by MA members in a transparent way. Organizational and individual membership of MA is practically open for any individual or organization under the conditions announced in

https://modelica.org/association

## 7.2 The Modelica standard library

An open-source **Modelica Standard Library (MSL)** was developed and continuously maintained and enhanced by MA. The library comprises a large set of various model components in many physical domains, cf. Table 7.1.

Table 7.1: Overview of some (sub)packages within MSL. An identifier `L` indicates that `Modelica.L` is a sublibrary of MSL at the top of hierarchy. The symbol `*.X` indicates that `X` is a subpackage of the previously mentioned sublibray. The symbol `*.*.Y` indicates that `Y` is a subpackage of the previously mentioned package.

| (Sub)Package | Description |
|---|---|
| `Blocks` | Basic components for building block diagrams |
| `*.Interface` | Connectors and abstract models for I/O blocks |
| `*.Math` | Mathematical functions as I/O blocks |
| `*.Continuous` | Basic continuous-blocks described by DAEs, e.g. Integrator, Derivative, PI & PID controllers, etc. |
| `*.Noise` (Klökner, Linden, and Zimmer 2014) | Random noise generators / statistical distribution |
| `*.Sources` | Signal generation blocks |
| `*.Tables` | Interpolation of one- & two-dimensional tables |
| Subpackages | `Logical`, `IntegerMath`, `MathBoolean`, `Nonlinear`, ... |
| `Electrical` | Diverse electrical components |
| `*.Analog` (Clauß et al. 2000; Majetta et al. 2009a) | Basic electrical components s.a. resistors, capacitors, inductors, transistors and diodes, heat storage and dissipation, ... (Cellier, Clauß, and Urquia 2007) |
| `*.Machines` (Kral and Haumer 2005) | Electric machine including loss effects from various sources (Haumer et al. 2009) with consistent thermodynamically established concept |
| `*.QuasiStationary` (Haumer et al. 2008) | Quasi-stationary analysis with sinusoidal excitation |
| `*.Spice3` (Majetta et al. 2009b; Majetta et al. 2011) | General devices (resistors, capacitors, inductors, sources) and semiconductor devices |

| | |
|---|---|
| *.*.Converters | Subpackages for converting ACDC, DCDC, ... |
| Subpackages | Digital, Multiphase, PowerConverters, ... |
| Fluid (Casella et al. 2006; Franke et al. 2009a) | Zero- and one-dimensional thermo-fluid flow components within subpackages: Vessels, Pipes, Machines, Valves, Dissipation, etc. |
| Magnetic | Magnetic components in couple of subpackages s.a. |
| *.FluxTubes | Electromagnetic devices with lumped magnetic networks |
| *.FundamentalWave (Kral and Haumer 2011b) | Rotating electrical three-phase machines relying on magnetic potential and flux fundamental waves |
| Math | Mathematical functions operating on matrices |
| Mechanics | Diverse one- and three-dimensional mechanical components within the subpackages: Multibody, Rotational Translation |
| Media (Elmqvist, Tummescheit, and Otter 2003) | Definitions for ideal gases, mixtures, water & air models incompressible & compressible media etc. |
| Thermal | Thermodynamical components within many packages s.a. |
| *.HeatTransfer | Heat transfer |
| *.FluidHeatFlow | Thermo-fluid pipe flow |
| SIunits | Type and unit definitions based on SI units |
| StateGraph (M. Otter and Årzén 2005) | A framework for modeling discrete event systems in a structured way via finite state machines based on Grafcet principles (David and Alla 1992) with capabilities for representing state charts (Årzén 1996). Earlier attempts were conducted (Elmqvist, Mattsson, and Otter 2001; Ferreira and Oliveira 1999; Mosterman, Otter, and Elmqvist 1998). |

MSL can be directly employed or extended by modelers for developing their own libraries

and applications. MSL, being developed by experienced modelers, can act as instructive guidelines for tackling modeling tasks using the state of the art Modelica-based design techniques.

MSL is subject to continuous maintenance and progress according to state-of-the-art software engineering techniques and intensive discussions. Regression testing is altered for every new contribution or a justified modification. The progress of the library state can be monitored through the github website

https://github.com/modelica/ModelicaStandardLibrary

Justified contributions in the form of pull requests are encouraged.

## 7.3 The functional mockup interface

Due to the increasing demand of model-based technologies in various application domains there is no need to follow an ad-hoc approach for involving Modelica-based components in external model-based applications, e.g. model predictive control, embedded systems and co-simulation among others.

In contrary, utilization of the so-called **the Functional Mockup Interface (FMI)** technology (Blochwitz et al. 2011) is the standard approach. This is achieved by exporting a Modelica models or a model specified in an arbitrary dynamical modeling tools to a simulation program written in the C programming language. This C-source code follows a unified API according to the FMI standard, cf.

https://fmi-standard.org/tools/

for an actual list of tools supporting FMI.

The FMI-compliant exported model together with a descriptive XML file is referred to as **Functional Mockup Unit (FMU)**, cf. Figure 7.1. The FMU can be independently simulated or integrated into other simulation platforms.

Advanced numerical solvers requiring **the Jacboian** of an ODE/DAE model for step-size adaptive numerical integration have been served since the second version of FMI specification (Blochwitz and al. 2012). This version recommends exported FMUs to provide function calls for evaluating **directional derivatives**. This even allows the evaluation of **dynamic parameter sensitivities** for model-based predictive control (Franke et al. 2015) among other possible applications.

The functionalities demonstrated in Figure 7.1 refers to **FMI for model exchange**. Another flavor is to allow the exportation of a bundled numerical solver within the FMU. This is referred to as **FMI for Co-simulation**.

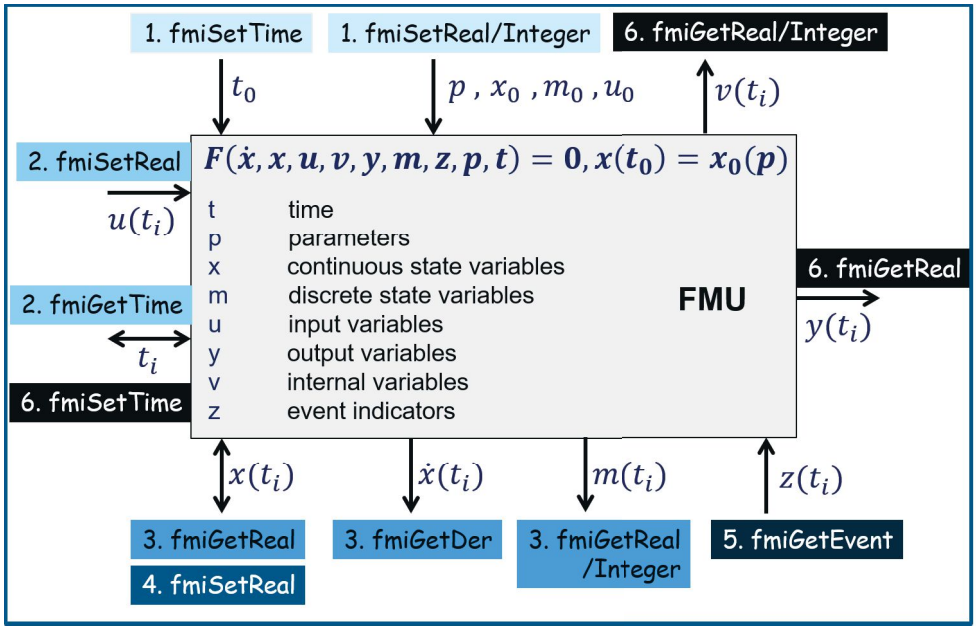


Figure 7.1: Functional mockup unit: Mathematical representation and some C-functions operating on them

Using FMI standards, generic FMU-based application-independent solutions and tools can be implemented, e.g. (Pfeiffer et al. 2012; Vanfretti, Bogodorova, and Baudette 2014). Particularly, the utilization of external design and optimization tool becomes a software engineering task e.g. (Vanfretti et al. 2016b).

Still there is a need for a further level of standardization for interacting FMUs, e.g. in power system modeling applications (Gómez et al. 2020). When describing a complex system comprising multi-physical phenomena, it is ideal to have the whole system described in only one language.

This is however almost impossible due to many reasons. First many other well-established long-rooted simulation tools optimized for specific domains exist. Their involvements in system modeling is highly beneficial. The in-depth accumulation of expertise with such domain-specific tools can not be excluded.

Thus, a practical solution that enables the collaboration of different working groups is to employ **co-simulation** where each tool exports its models as FMUs.

**Proposition 7.1** It is beneficial to design such an FMI-based co-simulation of a technically complex system in a standard manner allowing the orchestration among multi FMI-based simulations.

For this purpose, there are couple of FMI-related projects maintained by MA for standardizing the orchestration of FMI-based co-simulations.

#### System Structure and Parameterization (SSP)

SSP proposes an XML-based specification for describing the system FMU-based components and their interrelations (Köhler et al. 2016). Moreover, the specification covers also the overall parameterization in a consistent rather than isolated manner. This is particularly useful for MiL, SiL and HiL applications.

#### Distributed Co-simulation Protocol (DCP)

While SSP is more related to the modeling and descriptive part of a complex system, DCP is more related to the technical part of the co-simulation including interoperability aspects in a heterogeneous system (Krammer et al. 2019).

DCP follows a master-slave architecture along a finite discrete state machine characterizing the operation of the system. A standardized I/O data exchange model and description for communication protocol among slaves are provided.

## 7.4 Projects

The increasing significance of Modelica-based technologies can be best reflected by the amount of involvement in research projects (and participation in organized conferences). Table II lists some large-scale research industry-oriented projects primarily for developing and improving the Modelica language and its related technologies, cf.

https://modelica.org/external-projects

for an actualized list.

Table 7.2: Summary of some large-scale research projects around Modelica-based technologies.

| Project | Budget M € | Objective |
|---|---|---|
| (EUROSYSLIB 2007-2010) | 16 | Modelica libraries for embedded systems |
| (MODELISAR 2008-2011) | 27 | Developing FMI standards www.fmi-standard.org |
| (PEGASE 2008-2012) | 13.59 | Defining the most appropriate state estimation, optimization and simulation for power grid (Chieh, Panciatici, and Picard 2011; Villella et al. 2012) |
| (OPENPROD 2009-2012) | 11 | Integration of open model-driven M&S environments with industrial applications |
| (DYNCAP: 2011-2014) | | Dynamic operation of stream power plants with CO2 capture technologies |

| | | |
|---|---|---|
| (MODRIO 2012-2016) | 21 | Extending M&S tools from system design to system operation |
| (ITESLA 2012-2016) | 19.43 | Unified Modelica-based tools for pan European security assessments of large-scale power systems |
| (IEA EBC Annex 60 2012-2017) | | New generation computational tools for building and community energy systems |
| (TransiEnt 2013-2017) | | Efficient integration of renewable energies into urban energy systems |
| (ACOSAR 2015-2018) | | Advanced co-simulation interface for offline and online system integration |
| (OPENCPS 2015-2018) | | Verification/Validation and interoperability of key standard technologies (Modelica, SysML, FMI,...) |
| (DYNSTART 2015-2019) | 0.234 | Startup and shutdown processes of steam power plants with fluctuating share of renewable energy |
| (INDIGO 2016-2020) | 2.79 | More-efficient next generation of district cooling systems (Costa et al. 2020) |
| MISSION (2016-2024) | | Modeling and simulation tools for more-electrical aircraft design (Valdivia-Guerrero et al. 2016) |
| (EMPHYSIS 2017-2020) | 14 | New FMI standards for embedded systems, e.g. electronic control units and micro controllers |
| (ResiliEntEE 2017-2021) | | The resilience of coupled energy systems in the presence high share of renewable energy |
| (IBPSA Project 1 2018-2022) | | Modelica framework for building and community energy system design and operation (Wetter et al. 2019) |

## 7.5 Modelica simulation environments

Many mature Modelica simulation environments, commercial and open-source exist, a list of actual Modelica simulation tools is available under

`https://modelica.org/tools`

These environments are typically enhanced with additional services for model design and optimization, cf. (Fritzson et al. 2020, Section 4) for a compact coverage of tools associated

with OpenModelica.

In addition to FMI solution technology, cf. Section 7.3, simulation environments are usually associated with own and/or external interactive scripting capabilities. Interfaces to existing scientific computing languages s.a. Julia, Mathematica, Matlab or/and Python among other possibilities exist. With such interfaces Modelica-based simulation can be externally adjusted, performed, processed and analyzed, e.g. (Fritzson et al. 2020, Section 4.9).

## 7.6 Conferences and user groups

Despite the fact of Modelica could be viewed as an European product, the internationalization of Modelica through increasing activities in Asia and North America are remarkably present in the form of active user groups and periodic user meetings, cf.

https://modelica.org/users-groups

Meanwhile, there are periodically organized international conferences in three continents each every two years, cf.

https://modelica.org/events

Few facts include:

1. The 13th European version of the conference attracted couple of hundreds of participants largely from the industry with nearly one hundred peer-reviewed accepted papers
2. The first Japanese international conference was held in Tokyo May 2016
3. The first North American Modelica conference was held in Cambridge October 2018

Despite the actual state of COVID-19 pandemics, latest Modelica conferences were successfully held in a virtual manner. Participation was for free with no fees.

## 7.7 Modelica association membership

MA memberships are practically available for any interested party whether tool and/or modeling library providers or an end user after accomplishing transparent conditions published in the bylaws, cf.

https://github.com/modelica/MA-Bylaws/releases

The membership enables to take a part in the maintenance and progress of the Modelica language and its projects in a coordinated and organized manners according to the published bylaws. There are two forms of memberships:

1. **organizational memberships** for universities, research institutes and companies, cf.
   https://modelica.org/association
   for an actual list of organizations

2. **individual membership** for persons

The MA organization board taking care of the association common activities is periodically elected. Their members and duties are listed in the previous website.

## 7.8 Modelica newsletters

Newsletters on the Modelica language, tools, libraries, etc. are periodically published usually three times a year. The contents are collaboratively written by many parties through pull requests on github, cf.

https://github.com/modelica/newsletter

Older newsletters and email **subscription** can be obtained at

https://newsletter.modelica.org/

## 7.9 Books

Couple of **books on Modelica** in several languages exist, cf.

https://modelica.org/publications

Some of the materials in this e-book whether intentionally or not were influenced by other books. We are listing them here for interested readers who would like to explore some of the addressed topics in great details.

Definitely, the most comprehensive book providing a generous coverage of Modelica in more than a thousand of pages is provided by (Fritzson 2015). Many topics surrounding modeling and simulation is covered in this book s.a.

1. physical modeling principles and their underlying fundamental laws of Physics
2. equation-based compilation techniques
3. numerical integration methods for ODEs and DAEs
4. and many other active research topics

Some books that are not listed under MA website are still extremely useful. Particularly, (Cellier 1991; Cellier and Kofman 2006) were very useful for parts of this e-book regarding bond graphs, compilation and simulation techniques[1].

A remarkable book is **Modelica by Examples** written by Michael Tiller:

https://mbe.modelica.university/

[1] We are not doing any kind of advertisement. These are simply the books the first author have read during his PhD. There are definitely other useful books. We may provide more details in future as we manage to read more books.

The writing of the book was crowdfunded by the Modelica community. While the book is freely accessible online, printable/electronic versions can be purchased on the basis of "pay what you can or pay what you think this book deserves".

## 7.10 Education efforts

There are dozens of **shared lecture slides** prepared for many recognized universities, cf.

https://modelica.org/education

The OpenModelica simulation environment is equipped with interesting capabilities for preparing **interactive Modelica-based course materials**. These materials can be designed and embedded in interactive notebooks that include self-editable modeling exercises, executable simulations and plotting capabilities. Some of these simulations can be first executed after editing text-based Modelica models, cf. **OMNotebook** (Fernström et al. 2006).

Remarkably, all examples in (Fritzson 2015) are provided in an an OMNotebook-based called **DrModelica** (Sandelin et al. 2003) associated with the OpenModelica environment.

These technologies were adopted and generalized for large-scale interactive training and self-learning of Modelica in India via a so called **spoken tutorial** approach (Moudgalya et al. 2017). A spoken tutorial is a small modular piece of video that lasts for 7 to 15 minutes. It can be replayed and paused. The student needs to:

- reproduce what he just learned via OpenModelica and/or
- interactively edit text-based models and execute them

A series of well-prepared tutorials provides a decent step-by-step coverage of a particular topic. According to (Fritzson et al. 2020), more than 100000 students have been trained by the spoken tutorials on OpenModelica (and libraries). Additionally, there is also another spoken tutorial about the `OpenIPSL` power system library (Vanfretti et al. 2016b). These tutorials are accessible under

https://om.fossee.in/spoken-tutorial

Another helpful self-learning instrument is a cloud web-based generalization of OMNotebook called **OMWebbook**:

http://omwebbook.openmodelica.org

**Proposition 7.2** OMNotebook, DrModelica, OMWebbook and the spoken tutorial technologies provide excellent potentials and opportunities to design, prepare and spread further Modelica-based self-learning tutorials on diverse topics

The above initiative is associated with two further interesting projects for self-learners seeking more practice: Power system Simulation Project and Textbook Companion Project

(Moudgalya 2020).

In **Power System Simulation Project**, accessible under:

https://om.fossee.in/powersystems/pssp

, students and volunteers are asked to submit a power network model example implemented in OpenIPSL. The network example may come from any reliable source s.a. textbook, academic publications or a project report. A list of so far completed model examples is accessible under

https://om.fossee.in/powersystems/pssp/completed-pssp

In **textbook companion projects**, (a community of) students and volunteers are asked to crowdsource all relevant examples and exercises within a suggested / proposed existing textbook with Modelica models based on the OpenIPSL library. By the time of writing this Section (July 2021), modeling examples of (Stevenson 2014) and (Kothari and Nagrath 2014) have been accomplished, while those of (Gupta 2005) and (Nasar and Trutt 1998) are still under progress. Further complete set of examples for relevant books in related domains s.a. control, power electronics, differential equations, among 68 books exist. The set of completed books is accessible under

https://om.fossee.in/textbook-companion/completed-books