This is the published version of a paper published in *IEEE Transactions on Industrial Informatics*.

Citation for the original published paper (version of record):

*IEEE Transactions on Industrial Informatics*  
https://doi.org/10.1109/TII.2017.2785439

Access to the published version may require subscription.

N.B. When citing this work, cite the original published paper.

Permanent link to this version:  
http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-221663
CIM-Compliant Power System Dynamic Model-to-Model Transformation and Modelica Simulation

Francisco J. Gómez, Luigi Vanfretti and Svein Harald Olsen

Abstract—European regulations on information exchange have put new requirements on analysis tools, the main one being the adoption of the IEC Common Information Model (CIM) that may help improve interoperability across applications. This paper proposes the use of Model-Driven Software Engineering (MDSE) methods to meet these new requirements. Specifically, this paper shows how to apply Model-to-Model (M2M) transformations. The M2M method presented herein allows to work directly with the information and mathematical description and computer implementation of dynamic models, independent from specific analysis tools. The M2M method proposed requires the development of a mapping between CIM/UML and the Modelica language, which allows to derive Modelica models of physical power systems for dynamic simulations.

Index Terms— CIM, UML, SysML, Information Exchange, Information Modeling, Power Systems Dynamics, Modelica, OpenIPSL.

ACRONYMS

CGMES – Common Grid Model Exchange Standard
CIM – Common Information Model
DSO – Distribution System Operators
ENTSO-E – European Network of Transmission System Operators in Electricity
ERGEG - European Regulators’ Group for Electricity and Gas
EPRI - Electrical Power Research Institute
IEC – International Electro-Technical Commission
iTESLA – Innovative Tools for Electrical System Security within Large Areas
MDD – Model-Driven Development
MDSE – Model-Driven Software Engineering
M2M – Model-to-Model
M2T – Model-to-Text
OMG – Object Management Group
OpenIPSL – Open Instance Power System Library
RDF – Resource Document Format
SYSML – Systems Modeling Language
TISO – Transmission System Operators
T2T – Text-to-Text
UML – Unified Modeling Language
W3C – World Wide Web Consortium
XML – eXtensible Modeling Language

I. INTRODUCTION

A. Motivation

Efficient information exchange between Transmission System Operators (TSO), Distribution Systems Operators (DSO) and generation companies is required for network planning and power systems operations, e.g. real-time monitoring and steady-state analysis. These functions demand a high degree of coordination and consistency in data exchanges, which can be significantly streamlined through a common data exchange standard. The European Network of Transmission System Operators for Electricity (ENTSO-E) has adopted regulation EC 714/2009 for cross-border exchanges in the electricity network to ensure coordination of “data exchange and settlement rules, network security and reliability rules, interoperability rules, and transparency rules” [1].

To comply with these rules, software tools are one of the primary means to perform analysis and coordination tasks. The International Electro-Technical Commission (IEC) Technical Committee 57 (TC57) Working Group 13 and the Electrical Power Research Institute (EPRI), have been working on the development of the Common Information Model (CIM) to provide standard modeling semantics for power systems information exchange [2].

While the IEC CIM provides a solid basis for information exchange, it does not offer any guarantee on the model simulated response. The IEC CIM for Dynamic defines a relevant standard model with parameters for exchange. However, the physical dynamic behavior is only represented in a block diagram rather than in mathematical form (equations). In other words, the lack of a strict mathematical description defining each of the power system models is not exchanged (only parameters and pictorial diagrams). Thus, interoperability can only be unassailable though the comparison of several implementation results.

B. Previous works

The authors’ recent work has shown how the use of the Modelica language can preserve consistency of the behavior exposed by power system dynamic models when simulated in different tools [3]. In addition, the authors proposed in [4] a Model Execution Engine leveraging Modelica tools, showing that commercial Modelica compilers, such as Dymola, and open-source Modelica compilers, such as OpenModelica, can be used for power systems model simulation and analysis.

Open standard modeling languages for simulation, such as Modelica [5] can provide the mathematical description of physical systems, which is attractive to exchange user-defined...
A. languages. Section IV describes the results of the application describes in detail the mapping between the two modeling information exchange and make a detailed description of describe the state of the art on modeling languages for paper is organized as follows. In Section II the authors proposed model transformation methodology. Section III use of Object Management Group (OMG) standards. development paradigm based on models, which relies on the focus of MDD requires models and model transformations. These transformations require the development of mappings between a source model and a target model [7]. Table 1 shows a description of those transformation techniques.

C. Contributions
To address the issues outlined above, this paper provides three main contributions:
1) It proposes modeling process that CIM/UML and SysML to generated Modelica simulation models.
2) It defines a method to complement CIM Dynamics profile with equation-based component model definitions for physical modeling behavior, by using the Modelica language, for dynamic simulation analysis.
3) It defines of a scalable, modular and reusable mapping between different modeling semantics for Model-to-Model (M2M) transformation (UML/CIM, SysML/Modelica).

To show the detailed description of these contributions, this paper is organized as follows. In Section II the authors describe the state of the art on modeling languages for information exchange and make a detailed description of proposed model transformation methodology. Section III describes in detail the mapping between the two modeling languages. Section IV describes the results of the application of the proposed methodology, and Section V the authors present the conclusions of the work.

II. BACKGROUND
A. Model-To-Model Transformations

Model-Driven Software Engineering (MDSE) is a subset of Model-Driven Development (MDD) that consists of a development paradigm based on models, which relies on the use of Object Management Group (OMG) standards.

<table>
<thead>
<tr>
<th>Transformation Technique</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text-2-Text (T2T)</td>
<td>Transformation of origin data or input program code into documentation or program code, which can be executed by a machine.</td>
</tr>
<tr>
<td>Model-2-Text (M2T)</td>
<td>Transformation of an input model, implemented in a modeling language, such UML, into program code or documentation. This transformation can be also interpreted as code generation.</td>
</tr>
<tr>
<td>Model-2-Model (M2M)</td>
<td>Transformation of source model, defined with a modeling language, into a target model, defined in a different or the same modeling language, without compiling the target model into program code.</td>
</tr>
</tbody>
</table>

This work uses OMG standards such as UML and SysML for describing a MDSE methodology for model transformation and dynamic simulation. The software development process within the MDSE requires models and model transformations. These transformations require the development of mappings between a source model and a target model [7]. Table 1 shows a description of those transformation techniques.

Most power system analysis tools only implement T2T transformations, known in the power system domain as “parsers” or “data file format converters/filters”. Recently, with the goal of adopting the CIM standards, some power system analysis tools implement software interfaces, for M2T transformation, to read CIM files into the tools’ internal proprietary tool data format and representation, as illustrated in Fig. 1 (see [20] for detailed example of such approach).

In this paper, The M2M transformation is built on [8], with a focus on the UML and SysML semantics representations from CIM and the Modelica language. The workflow in Fig. 2 represents a transformation process between CIM and Modelica. As shown in Fig. 2 mapping rules are used for reading a source model in CIM, and the creation of a SysML Modelica-based class structure (as a meta-model). The meta-model is used to create the target model in Modelica that will be used for dynamic simulations.

B. UML and SysML

The Unified Modeling Language (UML) is one of the Object Management Group (OMG) ISO Standard specifications for system modeling [9]. UML plays an important role for MDD providing a standardized modeling language to create models for software development. The UML consists in a set of model elements representing an analysis of the properties and behavior of the system.

Those elements are classified in the following categories:
C. Standards in ENTSO-E; CIM and CGMES

The coordination of TSOs in planning and operation procedures such as system security, capacity calculation and outage planning was not sufficient to avoid a large system disturbance on 2006, affecting major parts of Europe [11]. After these events, the European Regulators’ Group for Electricity and Gas (ERGEG) defined a set of recommendations stating the need for compliance and consistency and highlighting the importance of the interdependencies and information exchange of the national implementations of grid models, which refers to both static and dynamic information upon which these models are implemented.

Within this context, the ENTSO-E has made large efforts in adoption of the Common Information Model (CIM) to improve the means for information exchange between TSOs. ENTSO-E has adopted different IEC CIM profiles to comply with the regulation and mandates [12], and thus the Common Grid Model Exchange Standard (CGMES) [13] was approved. Conformity of modeling, simulation and analysis tools to CGMES is carried out through Interoperability (IOP) tests [14]. The CGMES reflects current TSO requirements for accurate modeling of the ENTSO-E area for power flow, short circuit computations and dynamic simulations reflected within the following standards:

1) IEC 61970-552 CIM XML Model Exchange Format define language style and implementation rules of the data syntax of CIM,
2) IEC 61970-301 CIM Base defines a static information UML package containing UML semantics for the physical characteristics of the power network and electrical and non-electrical characteristics of equipment static models.
3) IEC 61970-302 CIM for Dynamics Specification and IEC 61970-457 CIM for Dynamics Profile, define a dynamic information UML package containing the UML semantics for the dynamic characteristics of equipment models.

D. CIM for power system dynamic analyses

The Common Information Model (CIM) covers the needs for power network analysis studies and is based on UML. The CIM semantic representation defines all the basic components and topology of the power network, with its steady-state behavior and a limited description on its dynamics. CIM uses UML’s structural semantics, defining sets of classes, attributes and binding rules among classes, to represent physical objects and their properties. The relation of classes is represented within different UML class diagrams that classify the network’s information into different packages. The set of these packages constitute the whole CIM canonical model.

The CIM Canonical model is represented under the IEC 61970 standard covering base power system models (IEC 619170-301) and base dynamic models (IEC 67190-302) [15].

The CGMES covers different subsets of this standard:
1) The Equipment profile (based on IEC 61970-452) that defines classes and attributes with basic information of equipment models
2) The Topology profile (based on IEC 61970-456) describes how a model is connected. It contains information of Terminal classes and their relation with ConductingEquipment and TopologicalNode classes.
3) The State Variable profile (based on IEC 61970-456) is the result from a topology processor and power flow calculations.
4) The Dynamics profile (based on IEC 61970-302 and IEC 61970-457) defines the classes and attributes for the behavior of the equipment.

The implementation of these profiles is performed by using Semantic Web Language Resource Description Framework (RDF) defined by World Wide Web Consortium (W3C) [16].

The serialization syntax is RDF/XML, which are extensions of the eXtensible Markup Language (XML). In this work, the RDF/XML implementation is exploited to select the most relevant model information.

E. Modelica for power systems

The modeling of power system equipment and networks does not end with the definition of attributes. The mathematical representation of these models must be developed to ensure unambiguous modeling, consistent exchange of the information to represent dynamic behavior and to perform consistent simulations in different tools. The Modelica language satisfies the requirements [3] and, in

1 These attributes will be used in this work to assign parameters to each of the mathematical model equations that are defined in Modelica components.
addition, it can be described with SysML semantics [6]. The CIM for Dynamics profile includes the definition of block diagrams of correspondent network component and their attributes. While this is very useful for humans to exchange knowledge, a computer-readable description would require a unique mathematical equation specification to guarantee consistency [3].

Modelica is an object-oriented equation-based programming and modeling language, which allows the representation of cyber-physical systems using a strict and openly standardized mathematical representation for dynamic modeling and simulation. The authors’ previous works in the FP7 iTESLA project has used Modelica for power system modeling [4][8][17-20], and Modelica-compliant tools have been used for simulations and compared with proprietary simulation tools [17-19]. One result of this project is the Modelica library containing dynamic for power systems component models [3], the OpenIPSL.

Modelica is suitable for unambiguous modeling and simulation [17]. A model in Modelica is totally decoupled from the mathematical solver that is used to provide a numerical solution of the model equations [3]. This characteristic guarantees an unambiguous way of modeling and simulation. Modelica models need to be provided with adequate starting guess values to perform simulations. OpenIPSL uses as starting guess the power flow solution values and Modelica tools solvers for the initial conditions of the entire dynamic model (for all algebraic and dynamic equations) to perform dynamic simulations. The challenges related to model initialization are discussed in [20][21].

III. CIM-TO-MODELICA MODEL TRANSFORMATION

To have a complete representation of a physical model, the use of the OpenIPSL Modelica components can complement the CIM/CGMES models. This work proposes the design of a M2M workflow to generate power system Modelica models from CIM, as shown in Fig. 2. The design includes the development of mapping rules between CIM/UML and Modelica syntaxes, and the design of a SysML-based meta-model for generating the resulting Modelica network model. Each of these mapping rules have been prototyped individually using XML, a portion of the code for one design is shown in Fig. 3 (others are omitted due to space constraints).

A. Binding Semantics: Mapping meta-model

The CIM semantics and Modelica language are implemented using different syntaxes, thus, it is important to identify the key features for binding both modeling languages. The workflow for the creation of a meta-model class structure based on the mapping rules is shown in Fig. 4.

The mapping matches the CIM classes and attributes names with the component names and their parameter names from the OpenIPSL library models. The action Mapping Rules defines the creation of matching criteria of CIM classes and OpenIPSL components using XML for each target component model.

The mapping is enhanced by including object’s references from the CIM model, to be used by the M2M transformation process for populating these IDs, to be used to identify relations between classes, as follows:

1) The rdf:ID attribute contains a unique ID, which is used to identify the object from the CIM model.
2) The rdf:resource is a pointer to another object within the CIM model. Its value is other rdf:IDs and that will be used to identify the other objects related to the current object

The Mapping Meta-Model action uses the mapping rules to create a class structure that encapsulates the mapping rules. These mapping rules are manually implemented for each new power system component to be modeled with CIM and the OpenIPSL (see Fig. 3). The Mapping Meta-Model action in Fig. 4 creates a new class from the XML definition of the new mapping rules using a JAVA parser for XML [22]. The result of this action is a mapping class structure, shown in Fig. 5.
which is used to populate the target model values from a CIM model, as follows:

1) The classes `ComponentMap` and `AttributeMap` contain the key attributes to identify the CIM classes and CIM attributes to read from the CIM model, and their correspondent key attributes in the Modelica component.

2) The values from the CIM model are populated within these two classes.

3) An additional class, `ConnectionMap` class, contains the references of the CIM classes `Terminal`, `ConductingEquipment` and `TopologicalNode`.

4) `ConnectionMap` is used to populate the references of these classes and to create the connections between components.

B. Building a power network’s SysML Meta-Model

A second workflow shows the process of identification of the target language syntaxes to be used to create the instances of the power system components models and power networks (see Fig. 6). This workflow considers basic characteristics from the Modelica language such as declaration of model parameters and declaration of components’ connection.

The `Network Meta-Model` action in Fig. 6, based on syntax definitions of the Modelica language, uses the Modelica class stereotypes `model`, `class` and `connector` to differentiate which component should be created and define its place into the model hierarchy for a network model (see Fig. 7). The `Network Class Structure`, considers a model hierarchy based on a high-level model e.g. the `network`, medium-level models, e.g. a `component`, and low-level models, e.g. a `connector`, and its implementation is depicted in Fig. 8.

The `Network Class Structure` complements the `Mapping Class Structure` and is used in the proposed M2M transformation to create the network model with the instances of the OpenIPSL Modelica library components and their connections, as follows:

1) Classes `moModel`, `moClass` and `moConnector` correspond the Modelica stereotypes `model`, `class` and `connector`.

   The meta-model structure includes two additional classes for defining high-level models within a network model.

2) The class `moPlant` encapsulates the components that are included in the model of a plant, e.g. machines and regulator components.

3) The `moNetwork` plant encapsulates both plant objects and component objects and defines the whole network structure.

4) `moClass` and `moConnector` will be used with values from the mapping to create the instances of the OpenIPSL objects. The class `moAttribute` stores the values of the parameters of each component.

IV. RESULTS: MODEL-TO-MODEL TRANSFORMATION AND SIMULATION

The main workflow of the proposed M2M transformation uses the structures defined within the `Mapping Meta-Model` and `SysML Meta-Model` workflows from sections III.A and III.B to automatically generate the target Modelica models from the source CIM model.

2 Because the words model, class and connector might be reserved keywords in different programming languages, we add the prefix mo in the name of the classes.
M2M transformation workflow

1) The values from the CIM model are read starting from a Terminal class and then from the classes related to that class.
2) With the rfd:ids and resource:ids attributes from CIM, the classes PowerTransformerEnd, ConductingEquipment, TopologicalNode and SvPowerFlow are identified.
3) In case of a ConductingEquipment class is found, EnergyConsumer, ACLineSegment, SynchronousMachine are identified.
4) For PowerTransformer, first the PowerTransformerEnd class is identified and then its related Terminal class.

The values from the Mapping-Meta-Model object are used to create the instances of the corresponding OpenIPSL components. The action Instantiate Network Objects creates the network structure adding the component objects and connections to the model, with the following steps:

1) The CIM Terminal class is transformed into an OpenIPSL Pin component, using the moConnector class in Fig. 8.
2) The specific CIM ConductingEquipment class is transformed into the corresponding OpenIPSL component, such as machine component or line component, using the moClass class in Fig. 8.
3) Following the instance definition of the components, the values of the CIM SvPowerFlow and SvVoltage classes are used to set the component instance’s parameters initial guess values.
4) The CIM TopologicalNode class is transformed into a OpenIPSL Bus component, with the use of the moClass in Fig. 8.
5) The connection between components are created using the moEquationConnect class in Fig. 8. It contains the component name and its pin and the name and pin of the associated bus. These workflow is executed within a loop, until the whole CIM model has been processed.

B. Simulation Model and M2M Results

To test the proposed M2M transformation process, the IEEE 9 Bus model [23] is used as input CIM model. The system is composed by 3 generators operating at 275 MVA and 16.5 kV in Generator 1; 320 MVA and 18 kV at Generator 2; 300 MVA and 13 kV at Generator 3, with a system base power of 100 MVA. This input model is implemented in four different XML/RDF files, which contain information from the Equipment profile (EQ), Topology profile (TP), State Variable profile (SV) and Dynamic profile (DY).

Using the class structures described in section III, the M2M transformation process workflow (Fig. 10) converts the CIM information model of the IEEE 9-Bus system into the corresponding Modelica mathematical model. The target network model is shown in Fig. 11, which shows the instances of OpenIPSL components and the topology of the network.
Fig. 11 SysML diagram of the 9-Bus system. In the bottom left corner there is the hierarchy of OpenIPSL power system components that form the network. In the bottom right corner, there is the Modelica instance implementation of Load and Machine components.

For simulation purposes, a fault has been added to the model to show better the dynamic behavior of the model. The fault is applied at Bus 8 in the system (see Fig. 13), and the buses voltages (V) and generators active power (P) are plotted in Fig. 13.

The reason to split the mapping into two different structures follows a modular implementation of the proposed methodology. The Mapping class structure of Section III.A can be reused to convert the CIM into another modeling language, while the Network class structure in Section III.B can be reused to generate Modelica code from a different source modeling language or from other simulation tools. The SysML representation of the network (see Fig. 11) shows model design independent from the modeling language or the power system simulation and modeling tool. The target model design can also be represented as a block diagram in Modelica (see Fig. 12).

V. CONCLUSION AND DISCUSSION

The complete process of binding the CIM syntax and Modelica syntax presented in this paper gives a mechanism to use mathematical modeling language as a complement for the CIM Dynamics information exchange profile, and its CGMES extension. The use of Modelica for the modeling of power systems dynamic behavior allows to comply with the application of EU rules for security, interoperability and transparency established by European regulations [8]. This work gives a proof of concept of how to follow Annex F in the new CGMES 2.5 standard guidelines [14], corresponding to the IEC 61970-600, to use Modelica models as a means to exchange user defined models defined in a Modelica library, and serves as an example of the feasibility of the approach recommended by this Annex F.

The results from the workflow process show that the proposed method is capable of transform a CIM/RDF classes and attributes of the network model into the OpenIPSL Modelica components and attributes, which contains the corresponding mathematical behavior of the network components. The resulting model can be used by different Modelica compilers, which compile the Modelica language into executable programing code. This implementation guarantees the scalability, reusability and modularity of the proposed methodology.

With this solution, dynamic analysis such as time-domain simulations, can be simulated by utilizing Modelica compilers that have proved to be applicable for this kind of analysis in different fields [24] and are showing promise for continental-level power systems simulations [25].
trips, other CIM profiles [26] need to be supported in the Modelica model. To automatically add events, such as line model has been implemented into the output behavior of the network with the presence of a fault, a fault systems network model with the use of CIM information and the results of this workflow show how to build a power Active Power (p.u.)

Fig. 13 Simulation results of the simulation generated model, IEEE 9 Bus model, first: Voltages (p.u.) at each bus; second: Active Power (p.u.) at each generator

The results of this workflow show how to build a power systems network model with the use of CIM information and OpenIPS library, to exchange power system components. To show the dynamic behavior of the network with the presence of a fault, a fault model has been implemented manually into the output Modelica model. To automatically add events, such as line trips, other CIM profiles [26] need to be supported in the proposed workflow.

REFERENCES


