Possibilities and challenges of an integrated development using a combined SysML-model and corresponding domain specific models

K. Kernschmidt*, G. Barbieri**, C. Fantuzzi**, B. Vogel-Heuser*

*Institute of Automation and Information Systems, Technische Universität München, Boltzmannstr. 15, 85748 Garching bei München (Germany) (e-mail: {kernschmidt, vogel-heuser}@ais.mw.tum.de)
**University of Modena and Reggio Emilia, Via Amendola 2, 42122 Reggio Emilia (Italy) (e-mail: {cesare.fantuzzi, giacomo.barbieri}@unimore.it)

Abstract: The development of a production plant, with multiple involved disciplines, requires many different models such as simulation models for the physical domain or models of the control-software. As all these models represent parts of the same system, it is inefficient to develop every model from scratch. Rather an integrated model, which considers all relevant aspects, should be set up first and from this the required domain specific models should be generated through model transformations. For the integrated model, the Systems Modeling Language (SysML) poses a suitable possibility to integrate the views of the different disciplines. Based on a literature research, existing model transformations from the integrated model to specific models as well as necessary extensions are presented. In this way a framework for an integrated development, where the SysML-model forms the integrating core model of the system, while the simulation of the different parts is carried out in domain specific simulation tools, can be established. Core benefits of this method are the data consistency of the models, and the fast and cost-efficient simulation and testing of the system at different points of the development cycle and with different levels of detail.

Keywords: system development, modeling, simulation, model transformation, SysML.

1. INTRODUCTION

Automated production plants nowadays are complex mechatronic systems and demand engineering solutions tailored to the customer’s needs. Thus, the development requires knowledge in mechanical, electrical, electronics and software engineering. Each discipline is specialized on a certain subsystem of the plant and uses during the development different domain-specific models, as for example during the development of the control software for the programmable logic controller (PLC) models of the software are made. All necessary models are created in specific modeling languages, which are well established in the respective domain, and with domain-specific methods and tools, which are optimized for the detailed modeling of the considered part of the system. Within a mechatronic system, however, there are many interdisciplinary dependencies and requirements which influence different parts of the system. In the conventional development method the necessary information of the system-elements, the requirements, dependencies etc. have to be added in every tool manually. Furthermore, if a change occurs during the development cycle, every partial model has to be checked manually in order to keep the information of the system consistently.

Thus, a more efficient way in the mechatronic system development is the usage of a joint model of the entire system, including all required information, and transforming subsets of the joint system model to domain specific models. The Systems Modeling Language (SysML), specified through the Object Management Group (OMG) [OMG (2012)], is a multipurpose modeling language, which fulfills the requirements to model multi-domain systems. The SysML is based on the Unified Modeling Language (UML) [OMG (2011)], which is a wide spread modeling language in software development.

At the beginning of the development process many parameters cannot be fully specified yet and the required functionalities of the plant can be implemented in alternative ways (e.g. a work piece could be moved by a conveyor belt or by a robot). These different implementation possibilities can be modeled in SysML and transformed to simulation models to analyze them therein. At this early step of the development only the standard course of action - the desired functionality - has to be simulated for decision making, e.g. the recognition of a workpiece with a barcode through a sensor, while the detailed modeling of exceptions, such as occurring errors (e.g. sensor does not recognize the workpiece), can be conducted after the decision for a certain system has been made. Through the automated transformation of the joint SysML-model to specific executable models a fast and cost-efficient simulation and testing of the system at different points of the development cycle and thus, with different levels of detail, becomes possible.

Based on a literature research, this paper presents existing possibilities for model transformations from an integrated SysML-model to specific models, the design of a framework for the development of mechatronic systems and further necessary model transformations.

The paper is organized as follows: After this introduction related work on concepts for integrated models and the possi-
bilities of existing model-transformations are shown. Then a framework for the development of production plants, using a joint SysML-model and transformations to models for simulation, is presented. Summarizing the paper, conclusions are drawn and an outlook on future research is given.

2. RELATED WORK

2.1 Integrated development of mechatronic systems

As described above the consistency of the system information is a big challenge, if every model is developed manually by different people. Product data management (PDM), which is usually implemented within product lifecycle management (PLM), is used to manage data and documents of the product development in a common database, to show relations between different documents and to keep the versions up to date [Saaksvuori and Immonen (2005)]. However, PDM does not focus on the development of a joint system-model and the transformation of this model to other modeling languages or tools.

Shi-Xiang and Sheng-Ze (2010) used the UML to model not only the software-part of a mechatronic system, but also the hardware. They address the design flow, including the specification of a system (with help of SysML Requirements Diagrams) and the development of its architecture. Furthermore, they state that the modeled architecture should be mapped to appropriate simulation models for a system analysis manually or through the APIs of the simulation tools.

An integrated SysML-based framework for mechatronic systems is presented by Chami et al. (2010). During the development of a mechatronic system the result of the collaborative design phase of the V-model is an integrated SysML system model. In order to link the integrated model with domain specific models it should include ‘requirements’, ‘elements’ e.g. blocks, and ‘parameters’ which are relevant in multiple, discipline specific models. After the development of the joint model the system is specified in domain specific tools, which are monitored by software agents. If a parameter in the specific model is changed the agents update the change also in the joint model.

Qamar et al. (2009) state the advantage of system modeling languages, especially the SysML, for specifying the system in one integrated model, and the need to link the system modeling tool to simulation tools such as Simulink. The integrated model should be used, as it is more comprehensive for the designers when investigating various design alternatives.

In Kawahara et al. (2009) and Sakairi et al. (2012) the necessity for an integrated SysML model and a corresponding Simulink simulation model for control systems, which include both, hardware and control software, is stated. The integrated model should use SysML for the specification, analysis, design, and verification of the control system. In the SysML parametric diagram the dynamic behavior of the system can be described though time derivative functions. However, a control system cannot be specified sufficiently by using only differential equations. Thus, for verifying the system with its discrete-time behavior and continuous-time behavior it should be simulated in specific tools such as Simulink or Modelica. As the simulation tools are not suited for a requirement analysis and architectural analysis, the authors advise to conduct the systems engineering processes before defining the detailed simulation models. For an efficient development process the Simulink model should be generated automatically from parts of the common SysML model.

Shah et al. (2010) present a multi-view modeling method for the engineering of embedded systems. As many domains are involved in the development, a joint model in SysML at a high level of abstraction should be developed, containing the relevant knowledge from the domain-specific views. Through profiles the domain-specific description of different disciplines in SysML is enabled. Thereby, each domain-specific view is a subset of the common model with a certain view of the system at a more detailed level. For the development in each domain, specific tools are used. Through model transformations the domain-specific views in SysML are mapped to the respective domain specific models.

2.2 Transformation of the integrated model to simulation models

The proposed framework focuses on a fast generation of simulation models from a joint SysML-model. Therefore the relevant parts from the SysML-model shall be transformed to according simulation tools. Some of the authors of this paper as well as other researchers have already developed transformation methods of specific SysML-diagrams. The existing transformation possibilities are explored in this section.

Existing transformations of the physical domain

The physical domain represents the used components of a system, their structure and relations. Existing transformation possibilities of the physical domain include in detail:

Parametric diagram - Simulink

Rösch et al. (2012) show a model transformation from the SysML parametric diagram to the Simulink block diagram, in order to create simulations of closed-loop controls. Each element of the parametric diagram is transformed to Simulink through defined mapping rules. Blocks and the part properties are transformed to subsystems in Simulink, ports to in- or outputs, and connectors to lines and branches. The value property of the parametric diagram is transformed to a combination of Data Store Memory Blocks, Data Store Read and Data Store Write Blocks. Finally constraint blocks are transformed to Simulink by using the Embedded MATLAB Function.

Cao et al. (2011) focus on systems with a hybrid behavior. Hybrid behavior characterizes the discrete switching between the continuous controllers, as it occurs often in mechatronic system, e.g. two running modes of a system which change through an event. Therefore the authors describe a possibility to extend the parametric diagram to model this hybrid behavior. Furthermore, the transformation between hybrid models in SysML and executable simulation models in Simulink are explored. The described method is based on the triple graph grammar formalism [Schürr (1994)]. Triple graph grammars execute the transformation through the defi-
nition of a third graph, next to the two graphs representing the meta-models of the source and target, containing the correspondence of the transformed elements.

**Structure diagrams - EPLAN, Modelica**

As described in section 2.1, Shah et al. (2010) propose a multi-view SysML-model of the mechatronic system. The specific SysML-views correspond to models in EPLAN (for the creation of production layouts) or Modelica (for the analysis of dynamic systems). In order to transform the common SysML model to the specific view and further to the simulation model story diagrams, as described by Fischer et al. (1998), are used for the definition of the transformation rules. The correspondence is defined in a metamodel. Thus, an input structural view in SysML is converted into the specific view and further to the simulation tool.

Paredis et al. (2010) give an overview of the SysML-Modelica transformation specification. Thereby, a SysML4Modelica profile is defined which represents the most common Modelica language constructs. During the system development first the integrated SysML-model is created. Then the part of the model that shall be analyzed by Modelica is selected and the SysML4Modelica profile applied. In a final step the subsystem can then be transformed to a Modelica model.

**Existing transformations of the logical domain**

The second part of the system is the logical domain. Here the behavior and the desired functions of the machine or plant are defined. In the following possibilities for the transformation of models of the logical domain to executable code are shown:

**State Chart diagram - PLC**

Witsch and Vogel-Heuser (2011) define a concept for PLC-statecharts, an adaptation of UML-statecharts (which correspond to the SysML state machine diagram), which can be used as a visual programming language for PLCs. Through the strict formal basis they can be transparently used in the context of IEC 61131-3 for the development of PLC software. The described approach was implemented as plugin for the industrial IEC 61131-3 programming environment CoDeSys V3, including full code synthesis, syntax checking, and online debugging, but could also be used in an integrated system model, such as proposed for the framework. As the UML statecharts were derived formally a code-generation from the model is easily possible.

Fantuzzi et al. (2011) present a design pattern to translate UML models into PLC code, which complies also to the IEC 61131-3 international standard. In the described pattern the machine or plant is divided into mechatronic objects including its control software, which is defined in a class stereotyped as “mechatronic” with input and output signals. The entire system is modeled then with composition links between the mechatronic classes. Each “mechatronic class” is associated to a Statechart diagram, which shows the control action of the module.

3. FRAMEWORK FOR AN INTEGRATED DEVELOPMENT

3.1 Architecture of the framework

The necessity for an integrated model of mechatronic systems has been described extensively in literature, as shown in section 2.2. In this contribution a framework is proposed which focuses on mechatronic production systems. Thereby especially two aspects have to be emphasized, which are described in the following:

First, for the development of mechatronic production systems two main parts have to be designed:

- The **logical domain** comprises the control software of the plant. In the final system it is usually implemented on a programmable logic controller (PLC).
- The **physical domain** contains the components (e.g. sensors, actuators, etc.), used for the execution of the desired functions of the system.

![SysML diagrams](image-url)

**Fig. 1.** SysML diagrams [according to OMG (2012)]. Diagrams, used in the framework, are highlighted.

The interaction of these two parts is analyzed nowadays in different ways through conceptual modeling and simulation, as for example through hardware in the loop simulation [Fantuzzi et al. (2012)]. Usually the logical part is considered therefore in its final implementation, including for example the human machine interface or alarm management. This approach however is inefficient, as simulations at an earlier point of time during the development, when only the main functions of the software are defined, would help to identify errors earlier and would enable the analysis of different implementation possibilities.
In order to carry out multiple simulations of the mechatronic production system during the development a fast creation of the simulation models is necessary. Therefore, the second aspect that is considered in the framework for an efficient and effective development is the automated generation of (parts of) the simulation models from an integrated model through model transformation.

Thus, the integrated model, including the logical and the physical domain should be created, using the SysML. In Fig. 1, the diagrams of the SysML are shown. Next to the requirements, which are modeled in the requirement diagram, the structure and the behavior of the system can be modeled through SysML. The division into behavior and structure corresponds to the logical domain (responsible for the behavior of the production system) and the physical domain (forming the structure of the system, e.g. defining the connections between the different elements). For the development of a system model in the proposed framework the following diagrams are especially useful (as highlighted in Fig. 1.): The steps of the control software should be modeled in the state machine diagram (stm), while the physical domain should be modeled in the block definition diagram (bdd) and the internal block diagram (ibd), as well as in the parametric diagram (par) for the dynamic behavior of the system’s components. Each part can be seen as a subset, or specific view, of the integrated model.

Fig. 2. shows the design of the proposed framework. As described above, the integrated model of the system is in the center of the development, including specific views of the physical domain and the logical domain. For the further analysis and simulation parts of the SysML-model have to be transformed to specific simulation tools. The different necessary domain-specific transformations are included in Fig. 2. The physical part of the production system can be simulated in tools such as Simulink, Modelica, or EPLAN. The SysML model of the logical domain should be transformed to Code for the PLC, which can be tested either on a real PLC or in a Soft-PLC tool, which implements the PLC on a computer. In order to analyze the interaction of the software with the developed physical system the PLC-program has to be connected to the real physical plant or to its simulation. This can be conducted through hardware in the loop simulation, if appropriate connection possibilities exist. This form of hardware in the loop simulation, with a simulated physical system and a real controller, is also conducted in the development of spacecrafts, as the system is very complex and tests with a real physical system would implicate enormous costs [Eickhoff, p.25f. (2009)].

Another way for a combined simulation is the transformation of the SysML-model of the logical domain into Stateflow. However, up to now there is no sufficient transformation of state machine diagrams to Stateflow (depicted through the dotted line in Fig. 2.). This transformation would be very helpful, because in this way a simulation model of the logical domain can be generated, which can be coupled directly to the simulation model of the physical domain in Simulink, as Stateflow and Simulink are both integrated into the Matlab environment. In this way a more comprehensive analysis and improvement of the system could be conducted already at an early stage of the development process, resulting in significant time and cost reductions.

3.2 Usage of the framework
In the previous section the design of the framework for the development of mechatronic production systems was described. Fig. 3. shows the process in the usage of the framework. As described by Bassi et al. (2011), the development of a system is carried out in several refinement steps, beginning at a high level of abstraction. Thus, the first step in the usage of the framework is the definition of overall system requirements. This step can be carried out with the help of SysML requirement diagrams. Based on the identified requirements general solution possibilities can be created. At this early point in the development process often several possibilities for the execution of a desired function exist (e.g. a function can be executed through hardware or software). Thus, several potential alternatives can be defined in this step (depicted as [Alternative 1] - [Alternative i] in Fig. 3.), which shall be analyzed further in specific simulation tools.

Therefore, for each alternative 1 to i, specific models are created. These specific models can be seen as subsets of the integrated model, as they are modeled in the same modeling language (SysML), but focus on a specific part of the system, namely the logical domain or parts of the physical domain. Thereby, only relevant aspects for decision making have to be considered, detailed aspects, as for example alarm management, do not have to be integrated in this stage. During the modeling process a mutual communication between the developers of the physical and the logical part is essential for an optimal system design and the development usually is carried out in several iteration steps. The transformation methods, described in section 2.2., require the development of SysML models according to very specific notation rules or design constructs. In order to create transformable models, the specific models of the mechatronic production systems have to meet these requirements. In order to reduce the development time further, it is beneficial to reuse parts from prior developments and integrate them into the models of the new system.

After the specific models have been developed for the logical and physical domain they are transformed automatically to models for the specific simulation tools, using the methods described in the literature, as specified in section 2.2. The simulation models are not coupled directly to each other, thus, changes that are made in a specific simulation model have to be implemented also in the according SysML-model, in order to keep the data of the system consistent. Depending on the used methodology, the changes in the SysML-model have to be edited manually, are changed automatically if a reverse transformation of the simulation model to the SysML model is conducted (as for example through the triple graph grammar approach described by Cao et al. (2011)), or are synchronized directly (e.g. through software agents as described by Chami et al. (2010)).

During the simulation of the developed models certain alternatives turn out to be inappropriate to fulfill the desired functions and meet the requirements. Thus, if the validation of an alternative failed, it is aborted and not developed further. If the validation of the alternative has been successful, it can be
considered for further development on a more detailed level. Thereby, further simulation and validation processes are necessary, leading to an iterative approach until a final decision can be made, which system shall be realized. The core benefit of the described method is the fast generation of simulation models already at an early stage of the development cycle. In this way more alternatives can be considered and validated, as the development of simulation models can be carried out earlier and with less costs.

4. CONCLUSIONS AND OUTLOOK

In this paper a framework for an effective and efficient development of mechatronic production systems was presented. The framework proposes to create a joint system model including the physical and the logical domain of the production machine or plant, using the behavior and structure diagrams of SysML. From the different subsets of the integrated model (e.g. the state machine diagram of the control software) simulation models should be generated automatically through model transformations. In this way development decisions can be verified or alternatives can be explored in a faster way. Existing transformation possibilities for the different specific models from parts of the common model were described. The missing transformation of the State machine diagram in SysML to Stateflow was identified, and poses the first step for further research. In a next step all identified transformation methods should be implemented in the development of a real mechatronic production system. In this way a reference model for the implementation of the proposed framework can be developed.

REFERENCES


Shi-Xiang, T., Sheng-Ze, W., 2010. The Conceptual Design and Simulation of Mechatronic System Base on UML. In: 2nd International Conference on Computer Engineer-