A Framework For Automating The Generation Of Machine Simulation Models

Uwe Schob, Fraunhofer Institute for Machine Tools and Forming Technology IWU

Abstract—During development, systems to be created are partitioned into several aspects, which are managed as separate documents. Their reuse for deriving other documents, like simulation models, currently involves mostly manual interpretation of the given information.

This paper targets the process of creating simulation models, which are semantically derived of already existing information. Based on the steps taken by a simulation expert to create a model, a general approach for transforming data models is presented.

A self developed software framework enables the definition of various transformations. Their results are, but not limited to, behavioral simulation models containing kinematic and kinetic associations. As an example, a model is automatically created and used as a machine emulation to perform control software tests.

Keywords—Simulation, virtual commissioning, machine modeling, model reuse.

I. Introduction

THE increasing global product diversification demands more flexible and thus more complex manufacturing plants. Developing and deploying them in less time is a key factor in being competitive. The design of automated machines requires specialists from different domains. Plans for the mechanical construction, the electrical diagrams as well as control programs are created under pressure of time followed by the assembly and startup phase.

To enable a manufacturer to meet the mentioned requirements, several research projects were performed. The introduction of paradigms to restructure the development process itself may be considered as the most future-oriented approaches among them as seen in [1], [2] and [3].

Another approach to reduce the time-to-market is to increase the products quality in an early development-stage. As part of the digital factory [4], simulation is the method of choice to verify early design decisions and help finding flaws [5]. A common practice is to simulate machines or parts thereof.

Such virtual machines are often used to verify control programs and thus increase their overall quality in an early stage of the development process. This method is called virtual commissioning and may be performed without real machine hardware, leading to a reduction of the time needed for real commissioning as shown by Zäh in [6].

A positive outcome may only be achieved, if the time needed to create simulation models does not exceed the expected gain at a later phase. In most cases, the generation of simulation models is too time consuming and thus regarded as additional effort to the normal development with few benefits. This problem is stressed repeatedly as in [7] and [8].

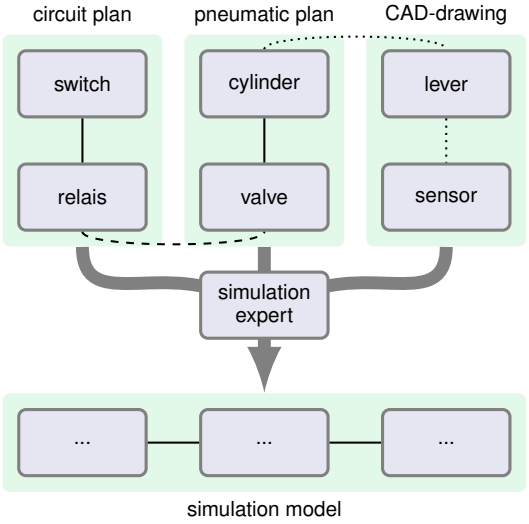

Fig. 1. Interpreting documents for simulation models

As stated above, the idea of virtual commissioning lacks optimizations allowing its application to broader areas. To meet the problem of the time consuming model generation task, one has to be reminded that most of the required information already exists. To create simulation models, an expert helps himself by using various documents as shown in figure 1. Construction and manufacturing plans as well as circuit and cycle time diagrams are interpreted and migrated into a model emulating the machines behavior.

The goal of this work is to develop a procedure that automates the generation of simulation models. The developed method is generic to consider different sets of input documents and allows variable interpretation thereof. A software framework in the form of reusable libraries was developed to enable the definition of several transformations. Each of them may target a specific kind of simulation representing an aspect of a machine.

II. Related work

Concerning the development of machines, simulations thereof and also the modeling of virtual machines, several works have been performed.

Targeting the optimization of the development process itself, a trend towards more abstract description paradigms can be

Uwe Schob is with the Department of Automation, Fraunhofer Institute for Machine Tools and Forming Technology IWU, Chemnitz, Germany, e-mail: uwe.schob@iwu.fraunhofer.de.

Manuscript received June 30, 2010; revised October 15, 2010

observed. In [2], Bathelt extends the existing methods of the V-model-based design in [1] to bridge the gap between mechanical construction and control software. The mechanical construction in the form of a function structure is enriched by additional information modeling the information within a machine. These enriched documents can later be exported as a basic control program containing most of the machines sequences.

The technology of the virtual commissioning is widely known among manufacturers. Its benefits are well documented, as in [6]. Various software tools are established, that support defining virtual machines and their inner logic. These definitions may be achieved through different modeling means. They range from proprietary logical linked signal flows over modeling standards like Modelica [9] even to the integration of compiled C-code. Similar to the means of modeling, the resulting detail level and model fidelity vary as well. A higher level of detail in the modeled system or the simulation results requires more effort in creating the models.

To meet this effort, Reinhart proposes a layered way of modeling virtual systems [10]. Primitive modeling elements available in the simulation tools library are grouped to generic mechatronic modules. Their composition to actual machine models is done in the more abstract layer of an engineering tool, which also uses libraries. Although this method allows creating models by connecting few sophisticated modules rather than many primitive ones, it relies on neatly designed and maintained libraries.

Another approach, that inspired the work of this paper was presented by Juhasz in [11]. Kinematic and geometric informations of a CAD-assembly model are exported. The two aspects are then combined into a multi-body representation within a self-developed application. By setting various parameters and adding of components of the electrical domain, e.g. drives, a multi-domain simulation model is acquired. This is specified by the means of the Modelica language [9].

Although the concept of models is common knowledge, formal definitions or specifications are rarely found in the field of machine simulation. Looking into the field of software technology, models itself are set as main focus. The Model Driven Architecture (MDA) is an approach to enable the consequent use of models in defining key aspects of software applications [12]. Despite the fact, that its main intention is to formalize the application development, it may also be applied to other fields of research. For this work, the MDA was used as a guideline to generalize the process of generating simulation models.

III. General approach

As stated above, the work of a simulation expert is the transformation of several documents of an automated system into a simulation model. He is not just able to perform this task for one specific machine, but for classes of machines. He combines the given information, interprets them and finally expresses them in another syntax.

Looking at this principle from the view of the involved software tools, each document is just a specific data model conforming to a meta model. The meta models define, which kinds of object types, attributes and relations between them are allowed. The meta models them self conform to individual domains. As an example, a CAD-drawing of a machine is a concrete data model, which instantiates elements defined in the meta model, e.g. components and constraints. The meta model is a formalization of the discipline of mechanical engineering or parts thereof.

The MDA describes the principle of the model-to-model transformation as a way to substantiate a general application model [12]. However in the context of this work, its general meaning is used. As seen in figure 2, a transformation might be described as a mapping of elements of a meta model M_A to elements of another meta model M_B. By using this method, each element or group of elements of a source model A_1 is translated into one or more elements of a target model B_1. Once specified mapping rules may be applied to every other source model A_i to derive a target model B_i.

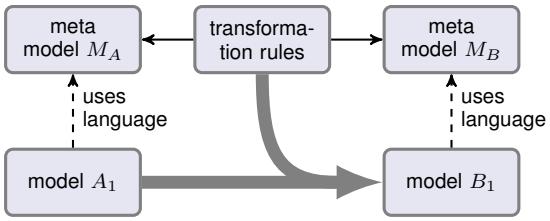

Fig. 2. Model-to-model transformation on meta model level [12]

As seen in figure 1, the interpretation process often requires more than one document. To being able to transform all of them in a uniform way, one has to be reminded, that they all represent aspects of the same system. So defining one mechatronic meta model, where each documents' meta model is part of, allows the general model-to-model transformation to be applied. Figure 3 shows the application with three exemplary documents.

A special case occurs, if even the target's meta model is a part of the mechatronic meta model. In this case, the transformation rules may only need to target objects within one language space. Additionally, the storage of the information is simplified as only one file format is needed, thus leading to a slim implementation.

Using this general approach, different transformation sets might be derived. The necessary steps to define a specific set according to the schema shown in figure 3 are as following:

1) Definition of a requirement profile of the desired simulation model.
2) Decision, which of the available documents containing which machine aspects will be used.
3) Selection of a common mechatronic meta model, which may contain all the information of the source and target documents.
4) Formulating transformation rules for the required object types.

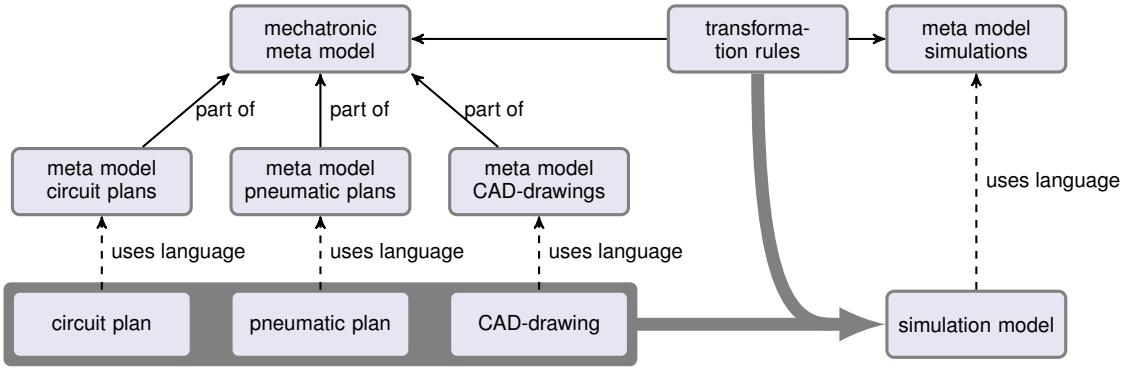

Fig. 3. Transformation of construction documents through meta model mapping

A. Requirement profile

Defining a requirement profile is equivalent to knowing, what a resulting simulation model should accomplish. The more effects and details the simulation should consider, the more complex the model will become. The model's complexity may be caused by a large number of elements or a high complexity of the element's internal structure. Additionally, more physical effects also need more parameters to be set accordingly. An example is the stick-slip effect in a pneumatic cylinder moved by a slight pressure. For achieving accurate results, a simulation model considering these effects requires accurate measuring of the several friction coefficients. Often, this task is too time consuming to be performed.

Similar to defining the simulations level-of-detail, its working platform needs a closer evaluation. The working platform defines, whether it is a standalone simulation or some kind of distributed simulation system. The latter requires additional communication functionality and synchronization mechanisms. Additionally, a distributed simulation needs further detailing regarding the kind of simulation, e.g. software-in-the-loop or hardware-in-the-loop. If embedded systems, like programmable logic controllers (PLC), are included in the simulation, more strict time constraints may apply.

B. Model identification

Taking the general requirement profile into consideration, it is now necessary to analyze, where the different modeling aspects may be acquired. During the development of automation systems, various documents are created. They are results of the numerous design steps and often used as base of the following steps. Their style and formalism may differ from domain to domain as well as between different users.

Documents conforming to a strict formalism are more suitable for the application of this approach, because they simplify the algorithmical utilization. However, it has to be reminded, that not all information may be available in well formalized documents. Textual descriptions as well as implicit associations between documents are common and difficult to interpret. The associations between objects can be differentiated into three types, as seen in figure 1. According to [13] associations can be defined as followed:

- formal (___) ones have a strict syntax and semantic,
- semi formal (_ _ _) associations have a strict syntax but no unique meaning and
- informal (.....) associations follow neither a strict syntax nor have a defined semantic and are often given in natural language.

In order to create an application understandable simulation model, formal associations between objects are required. In addition to the transformation mapping itself, a preliminary step of identifying all relevant semi formal or informal associations is necessary. They need to be translated into formal associations as well.

An example of an informal association is the positioning of a sensor "at the end of a conveyor belt". The information might be found as a side note in a CAD-document or a circuit diagram. How this information is coded, may depend on the software tool or the engineer describing it.

An example of a semi formal association is the electric trigger of a pneumatic valve, which can be found two times, in the circuit diagram as well as in the pneumatic plan. Both objects represent the same physical component and have identical marks. However, identical marks of objects might also mean, that one physical component consists of several functional units, that are spread across several circuit diagram pages.

Regarding the above mentioned conditions, documents need to be selected, that most information as stated in the requirement profile can be acquired. The following list contains common exemplary document types available in the development process.

- Cycle time diagrams are early documents mostly created during the process planning. Semantic associated functions of a machine or a process are shown together in their timely sequence. The diagrams can be used as an early base of commanded machine behavior, although they often occur as mere informal documents.
- CAD-drawings contain the mechanic geometric representation of single functional elements of a machine. They can be extracted from CAD-software and can be used to acquire component masses, geometric extensions and used materials.
- CAD-assemblies describe, how elements are combined to composites. Through various constraints, the spatial

positioning of the elements are calculated. This can be used as base for multi-body systems.
- Circuit diagrams show, how an automation system is powered and the mechanical parts a driven. They also represent the interconnection between mechanics and control, as the available control and sensor signals are defined. A large part of a machines inner logic is defined through the circuit diagrams.
- Fluid plans are similar to circuit diagrams as they provide auxiliary, e.g. hydraulic or pneumatic, power to a machine. They often have associations to elements of the circuit diagrams.

C. Selection of a meta model

In section III the necessity of a mechatronic meta model for a uniform transformation mechanism is described. The selection of such a meta model is a decisive factor concerning the transformation, as this defines how mapping rules might be formulated. Additionally it implies by which means the source documents can be integrated into the meta model.

Although the digital factory [4] requires collaborative software tools, the current practice shows another picture of a highly heterogeneous world of applications. Exchange of data is mostly done by exports and imports accepting the possible loss of information. Plenty of different formats exist, claiming to be a holistic mechatronic model. Due to the amount of available formats, with each having advantages and disadvantages, and the multitude of possible transformation sets, a general choice for the optimal meta model can not be presented.

However, the following paragraphs present a small number of suitable formats, though their usage has to be evaluated with regard to the defined requirement profile.

- Standard for the Exchange of Product Model Data defines neutral mechanisms to describe products during their life cycle [14]. Among others, parts of the standard are the definition of components and composites of products, protocols for the exchange of data, bindings to several languages and abstract test beds. A broad usage is found mainly in the field of CAD-tools, where it covers a export format for geometric information. Although the standard covers a vast area of aspects, the development of automation systems may only be viewed separated from the produced objects. Additionally, its pure extent makes its handling an expensive task, thus rendering it less useful for smaller projects.
- Automation Markup Language is a neutral data format designed to represent aspects of a manufacturing plant [15]. The format regards the fact, that during the development information are created and edited in different software tools. It is an integration format connecting several other standards and includes information of CAD-information, circuit diagrams as well as control logic. Due to its relatively novelty, architectural specification published in April 2009, a broad market spread can not be assumed.
- Self developed models are an alternative to existing data formats, if a fitting one can not be found or the integration is too expensive. It may only be a suitable mean, if designed for a very specific transformation set. Later extensions might render it less maintainable or require the reinvention of aspects already done in other formats.

D. Formulating transformation rules

Having realized the above described steps, the remaining one for completing the definition of a transformation set is formulating the mapping rules. Independent of a specific technology, the task of the transformation is to identify patterns in the source data models. A pattern can be a specific type of object, an object with a special set of attributes and/or values as well as groups thereof. For each such defined pattern, an object or group of objects in the target model needs to be defined. A general mapping should be able to handle associations of the form $1:1$, $1:n$, $n:1$ and $n:m$.

An example of a $n:1$ transformation rule is shown in figure 4. Note that the physical component of a valve has a visual representation seen in the pneumatic plan that decomposes to several raw data objects. These need to be grouped to a single simulation element.

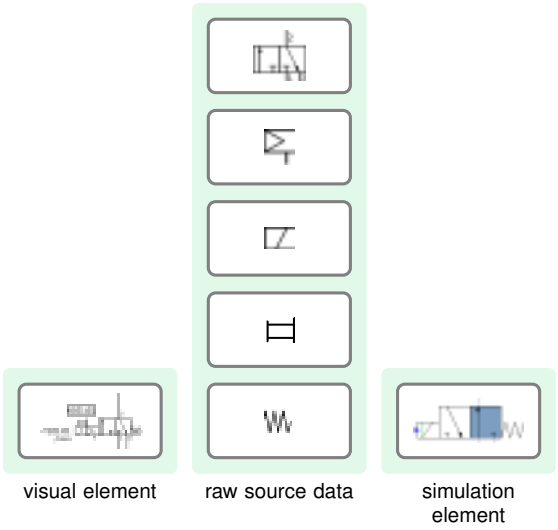

Fig. 4. Grouping of elements during transformation

Presenting all possible technologies for implementing such mappings could result in a rather long list, exceeding the scope of this paper. Therefore only three exemplary means are addressed.

- Query/View/Transformation is a language specified in the context of the MDA [16]. Its main purpose lies in the field of software technology, where it can be used to substantiate UML class models into a platform specific model of an application. Although it is intended for code model transformations, its general character allows other fields of usage.
- Extensible Stylesheet Language Transformation as described in [17] emerged as web-technology. The rules are formulated in an XML-document, whose processing may transform an input XML-document into another XML-dialect.

- Custom algorithms can be used alternatively to existing means. They should be favored if the existing means do not allow enough versatility or their integration would require tremendous efforts.

E. A framework as tool support

Apart from the theoretical base for a transformation set, a software framework supporting their definition is presented. The Automatic Model Generation (AMG) framework provides the means to substantiate a general transformation idea to a concrete set of software modules performing this task automatically.

Therefore, the term of an adapter was introduced. The interoperability between software systems is a matter of designing appropriate adapters, which translate one set of data elements into another one [18]. An Adapter represents a software module containing the algorithms to read the content of a specific type of document and deliver it to a common data model. Vice versa, an adapter may also collect data from the common data model and transform it to a specific document. A special kind of adapter exclusively handles data within the common data model, thus realizing the model-to-model transformation.

Figure 5 shows the general work flow within the AMG framework. It starts with the skeleton of an unspecific meta model, which defines the general structure of a common data model. The meta model is described using an XML schema [19]. A complete transformation set is acquired through the following three steps:

1) The extension of the skeleton by one or more sub models defines a meta model specific for one transformation set.
2) Through automated code generation, the framework creates code skeletons for every defined sub model. Each code skeleton represents an adapter already capable reading and writing from and to the common data model.
3) The final step is adding custom algorithms for understanding a specific type of document or performing model mappings. The adapter may directly use a file or indirectly access the necessary information through an application programming interface (API) of the respective software tool.

The adapters share a common basic definition by which the AMG framework can execute them as a sequence. Each adapter may require additional information for its execution. For example a reference to a document or a number of transformation rules might be needed. If all these information is already entered, e.g. during a previous run, the sequence can be executed fully automated, thus leading to one or more desired target documents.

IV. Application for the virtual commissioning

A transformation set for the use case of the virtual commissioning was created using the AMG framework. As stated in section I, the virtual commissioning is used to verify control programs, thus leading to a scenario of a hardware-in-the-loop-simulation, where the hardware is the machines PLC. The simulation needs to resemble the inner behavior of the machine at the interface of the control inputs and outputs as presented in [20]. The following sections give an overview of the simulation aspects taken into consideration and the software tools involved. The data flow for this transformation set is shown in figue 6.

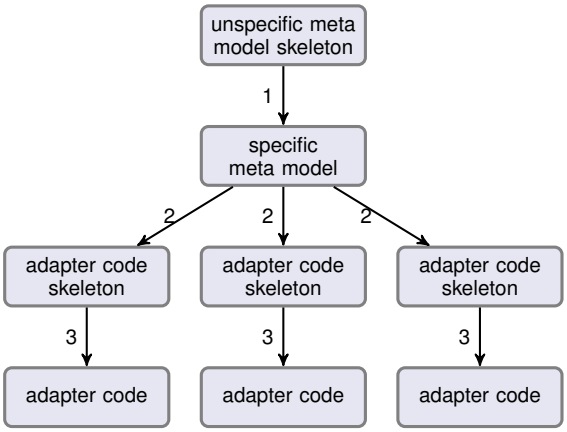

Fig. 5. AMG framework for specifying transformation sets

A. Aspects of a machine

For this transformation set, the following documents can be used as a source:

- CAD documents are used to acquire the geometric information of each part. Additionally, the assembly information is taken into consideration to derive the machines kinematic.
- Electric & pneumatic plans are analyzed for the inner machine behavior and the connection to the control hardware. These plans define, which signals of the control trigger which actuators of the machine as well as which sensor signals are available. Due to the usage of a software integrating electric and pneumatic engineering, these documents are used in conjunction.

An analysis of the available documents revealed several aspects usable for the simulation model. The main part of the machines behavior is defined by the electric and pneumatic plans. They rely on objects like switches, relays, valves and cylinders which are connected through wires or tubes. The elements are composed of several symbols, which need to be grouped to resemble a physical component. The grouping is performed through the symbols reference marks. Additionally, the electric plans define, at which addresses a signal is delivered to a control.

The CAD documents should contain the parts of a machine, which are driven by the control. Actuators affected by electric or pneumatic connections often have an association to a part in a CAD document. These associations are not regularly defined and thus need to be entered manually. Similar, a sensor signals association to a position of a mechanical part needs to be added. As sensors are often found at the output side of a mechanical system, the kinematic chain starting at the input side has to be considered as well. The inclusion of the

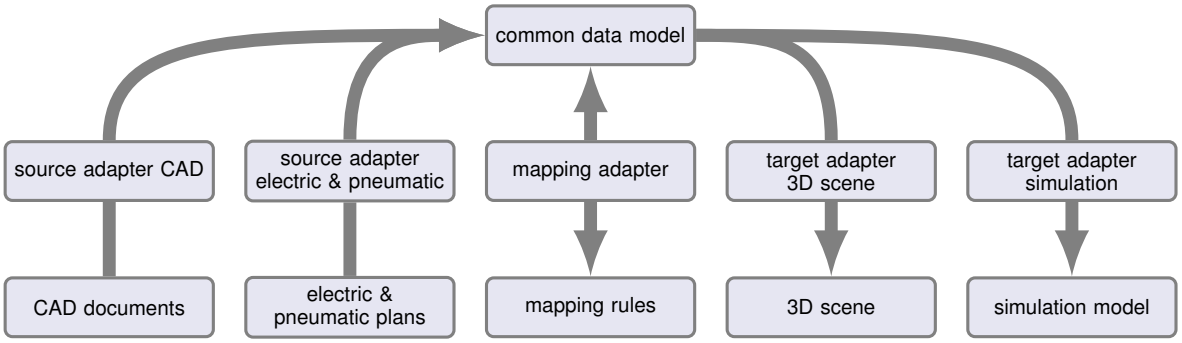

Fig. 6. Using the AMG framework for creating virtual machine models

kinematic systems allows an exact virtual measuring of the sensor signals.

B. Usage of adapters

As shown in figure 6, five adapters were implemented for this transformation set. Each of them was created as a code skeleton by the AMG framework and afterwards filled with specific algorithms.

- The adapter CAD attaches itself to the software Autodesk Inventor 2009, which was used for managing the CAD documents. It is a commonly used application, which is by default delivered with an API [21]. It was used to access the runtime data model of the application, thus ignoring the drawbacks of focusing on a single file format. All formats loadable by the application might also be used by the adapter. It mainly extracts the kinematic system and the 3D objects needed for a visualisation. The available assembly constraints were analyzed according to Kim [22] to derive kinematic chains and the full multi-body-system of the machine.
- The adapter electric & pneumatic connects to EPlan electric P8 and fluid through its integrated API as described in [23]. It accesses loaded documents in a runtime format, which mainly consists of pages, symbols and connections. As the basic elements of the documents are symbols and not physical components, a sophisticated grouping algorithm was necessary.
- The mapping adapter consists of the two parts multi-body-system mapping and behavior mapping. The former includes rules to map found constraints to different kinds of general joints. The supported joints depend on the simulation system used. The latter grouped elements with identical reference marks and created appropriate elements in the simulation system. In addition to the mapping rules, the user was able to enter additional non available cross reference information through this adapter.
- The adapter 3D-scene is used to export the volume models of the CAD software into a surface format for a 3D viewer application. The market holds various of such applications and file formats as presented by Ball in [24]. In favor of a time-saving implementation, the open source format Collada was choosen [25], as its specification and XML schema are freely available. Additionally, an open source application was available, which was extended for the dynamic presentation of moving 3D scenes.
- The adapter simulation provides the means to create an actual simulation model out of the common data base. This is done by instantiating all defined elements in the software SimulationX [26] and connecting them similar to the real wiring. The model includes communication elements to dynamiclly transmit signal information from and to the PLC as well as to the 3D viewing application. The simulation models conform to the language Modelica [9].

The task of the automated model generation is done after the target models have been created. To perform an actual virtual commissioning, the simulation model needs to be run from the appropriate application. Similar, the 3D viewer application needs to be started as well as the real control hardware. If any of the source documents were changed, the whole transformation loop may simply be rerun to acquire updated models. Changes to the transformation set are only necessary, if the source or target meta models changed.

V. Conclusion

This paper has presented a general approach for automating the generation of machine simulation models. The approach is based on a model-to-model transformation on a meta model level. The source models refer to documents that are created during the normal development process. The target models refer to one or more simulation models representing different aspects of a machine. Extendable sets of rules define, how source elements are transformed into target elements. Although the approach aims to be fully automated, the level of automation depends on the available sources. Information available in a formal way can be used without restrictions. Unclear or undocumented information, like associations between an actuator and its driven mechanical part, require manual user input.

Applying the approach for a specific set of source and target document types leads to a specific transformation set. It consists of several adapters which mediate between the specific documents and a common data model. A once defined transformation set itself may then be applied to various instances

of the defined source document types, i.e. the descriptions of different machines.

The creation and automated execution of transformation sets is supported by the the self developed AMG framework. It allows the extension of the common data model by the means of XML schema [19]. Additionally, code skeletons with already included access to the common data model may be provided by the framework as adapters to speed up the development of transformation sets.

Despite the addressed drawbacks, the work has still shown, that it is possible to reuse existing documents created during the development of automation systems. By this means, the effort in creating virtual machine models can be reduced significantly. Design flaws and software errors can be uncovered in an early stage, thus increasing the overall software quality before the system's startup.

VI. Outlook

An immediate task in this field of work should be the extension of the given example for the virtual commissioning of larger automation systems. Therefore the aspect of material flow within a system needs to be taken into consideration. Almost all machines handle or modify product parts and assume sensory feedback of them. A virtual machine without virtual product parts may only be tested to a small degree.

It needs to be remarked, that the process of model transformation may require manual user input if not all requisite information are documented or formalized. In addition, the transformation rules will only cover, what was anticipated by the developer during their formulation. The rules may fail, if source documents or users do not strictly stick to an assumed formalism. Furthermore, the transformation requires simulation couterparts for each element in the source documents. Considering the vast amount of commercial available components, a complete library of simulation elements might remain difficult to fulfill.

To ease the above mentioned restrictions, further work should focus on the formulating of mapping rules in a broader scope. It is necessary to specify rules, that are more flexible in handling not strictly formalized source documents. The more sophisticated the transformation rules are, the more flexible a transformation set could be used. This would allow end users to continue their own style of documenting without losing the benefit of the automated model generation.

References

[1] "Vdi 2206 - entwicklungsmethodik für mechatronische systeme," Verein Deutscher Ingenieure, Jun. 2004.

[2] J. Bathelt, "Entwicklungsmethodik für sps-gesteuerte mechatronische systeme," Ph.D. dissertation, Eidgenössische Technische Hochschule Zürich, 2006.

[3] F.-L. Krause, T. Tang, and U. Ahle, "ivip - integrierte virtuelle produktentstehung - abschlussbericht," online, 2002. [Online]. Available: http://www.ivip.de

[4] "Vdi 4499 - digitale fabrik - grundlagen," Verein Deutscher Ingenieure, Feb. 2008.

[5] *Anlaufoptimierung durch Einsatz virtueller Fertigungssysteme*, online, Universität Hannover - Institut für Fertigungstechnik und Werkzeugmaschinen, 2006. [Online]. Available: http://ramp-up-halbe.de/

[6] M. F. Zäh, G. Wünsch, T. Hensel, and A. Lindworsky, "Feldstudie - virtuelle inbetriebnahme," *wt Werkstattstechnik*, vol. 96, pp. 767–771, 2006.

[7] M. Bergert and C. Diedrich, "Durchgängige verhaltensmodellierung von betriebsmitteln zur erzeugung digitaler simulationsmodelle von fertigungssystemen," *Automatisierungstechnische Praxis*, vol. 7, pp. 61–66, Jul. 2008.

[8] R. Drath, P. Weber, and N. Mauser, "Virtuelle inbetriebnahme - ein evolutionäres konzept für die praktische einführung," in *Automation*, 2008.

[9] "Modelica - a unified object-oriented language for physical systems modeling," online, Modelica Association, Sep. 2007. [Online]. Available: http://www.modelica.org/documents/ModelicaSpec30.pdf

[10] G. Reinhart, T. Hensel, A. Lindworsky, and M. Spitzweg, "Teilautomatisierter aufbau von simulationsmodellen," *wt Werkstattstechnik*, vol. 97, pp. 663–667, 2007.

[11] T. Juhász and U. Schmucker, "Automatic model conversion to modelica for dymola-based mechatronic simulation," in *6th International Modelica Conference*, 2008.

[12] J. Miller and J. Mukerji, "Mda guide version 1.0.1," online, Jun. 2003. [Online]. Available: http://www.omg.org/cgi-bin/doc?omg/03-06-01.pdf

[13] M. Frappier and H. Habrias, *Software Specification Methods: An Overview Using a Case Study*. Hermes Science Publishing, 2000.

[14] "Iso 10303-1 - industrial automation systems and integration – product data representation and exchange – part 1: Overview and fundamental principles," International Origanization for Standardization, 1994.

[15] B. Grimm, L. Hundt, A. Lüder, and J. Peschke, "Universelles datenaustauschformat," *A&D-Kompendium*, vol. 2008/2009, pp. 266–268, 2008.

[16] "Meta object facility (mof) 2.0 query/view/transformation specification," online, Object Management Group, Apr. 2008. [Online]. Available: http://www.omg.org/spec/QVT/1.0/PDF/

[17] "Xsl transformations (xslt)," online, World Wide Web Consortium, Nov. 1999. [Online]. Available: http://www.w3.org/TR/xslt

[18] U. Schob, *Mechatronische Modellbildung von Produktionsanlagen*. VDM Verlag Dr. Müller, 2007.

[19] "Xml schema part 0: Primer second edition," online, World Wide Web Consortium, 2004. [Online]. Available: http://www.w3.org/TR/2004/REC-xmlschema-0-20041028/

[20] U. Schob and T. Blochwitz, "Virtual start-up of automation systems," in *12th ITI Symposium*, 2009.

[21] *Autodesk Inventor API*, Inventor 11 ed., Autodesk Inc., 2006.

[22] J. Kim, K. Kim, J. Lee, and J. Jeong, "Generation of assembly models from kinematic constraints," *International Journal of Advanced Manufacturing Technology*, vol. 26, pp. 131–137, 2005.

[23] *EPLAN API 1.0*, EPLAN Software & Service GmbH & Co. KG., 2006.

[24] A. Ball, L. Ding, and M. Patel, "Lightweight formats for product model data exchange and preservation," in *PV 2007 Conference*, 2007.

[25] M. Barnes, "Collada - digital asset schema," online, Sony Computer Entertainment Inc., Jun. 2006. [Online]. Available: http://www.khronos.org/files/collada_spec_1_4.pdf

[26] "Simulationx," online, Gesellschaft für ingenieurtechnische Informationsverarbeitung mbH, 2010. [Online]. Available: http://www.iti.de/simulationx.html

Uwe Schob finished his studies of Applied Computer Science at the University of Technology Chemnitz in 2007. Since then, he is employed as scientific staff member with the Department of Automation of the Fraunhofer Institute for Machine Tools and Forming Technology. His tasks consist of performing and managing industry funded research projects to enhance current CNC and PLC solutions. Other fields of work are machine simulation and the Virtual Commissioning.