INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN, ICED15
27-30 JULY 2015, POLITECNICO DI MILANO, ITALY

SUPPORT OF THE SYSTEM INTEGRATION WITH AUTOMATICALLY GENERATED BEHAVIOUR MODELS

Kößler, Johannes; Paetzold, Kristin
Universität der Bundeswehr München, Germany

Abstract

The development of technical systems is faced with an increasing complexity. A reason for this is the growing number of domains working together for a final product. Thereby the system integration is getting more important for companies while the development of single components is often outsourced to other companies. This leads to the need of detailed interfaces descriptions between these components to support the product development process and the data and information flows. In this paper the initial steps to automatically generated behavioural simulation models are presented. The fundament for the generation of these models is model-based system engineering (MBSE). The available diagrams of MBSE are used to generate the main structure of a behaviour model. In addition partial models have been developed describing the behaviour of machine elements. Combining the MBSE models with the partial behaviour models allows the generation of a behaviour model of a complete system. A first validation of the concept of automatically generated models is presented. Based on simple MBSE and CAD models of a gear unit, different behaviour models are generated for the use with Matlab/Simulink.

Keywords: Complexity, Functional modelling, Systems engineering (SE), Model-based systems engineering (MBSE), Interfaces

Contact:
Johannes Kößler
Universität der Bundeswehr München
Institut für Technische Produktentwicklung
Germany
johannes.koessler@unibw.de

Please cite this paper as:
Surnames, Initials: *Title of paper*. In: Proceedings of the 20th International Conference on Engineering Design (ICED15), Vol. nn: Title of Volume, Milan, Italy, 27.-30.07.2015

1 INTRODUCTION

The development of technical systems is faced with a growing complexity of requirements. One reason is the increasing functionality the systems must meet. In addition the development must consider not only the requirements at the beginning of the development process, but also the complete product life cycle and the interests of different stakeholders as well. This necessitates consideration of different domains during the entire development process to find successful solutions.

1.1 Problem Description

The development of components needs a high knowledge of certain specialist fields to meet the complex requirements of technical systems. Therefore the component development is often outsourced to suppliers. The core competence of the company shifts to the integration of components instead of developing them. This change causes the need to describe interfaces between components to successfully integrate the components into the final product.

Consequently methods of system integration are required enabling companies to develop systems that can meet the complex requirements. An important method is the product validation in early stages of the product development process. Due to the small amount of information a detailed simulation of the complete system is not possible. Hence behaviour based simulation models help simulating the complete system based on parameterized models. These models can be used to validate the complete system behaviour and evaluate interfaces between the components.

1.2 Objective

A method that supports the system integration during the design process has been introduced by Paetzold and Kößler (2014). This method enables the system engineer to validate the products based on MBSE and SysML models with the help of behaviour simulations. The aim of these simulations is the identification of interfaces and their evaluation of importance. A key step of this method is the generation of the behaviour simulation based on an MBSE model. While MBSE models are primarily used to describe the structure und the functionality of a system, their functionalities are not satisfying to simulate complex behaviour models. For these complex behaviour models a linkup to external tools like Matlab/Simulink or Dymola is required. To improve the development process the generation of this models should be supported by an automation that reduces the necessary work manually done by a system engineer. The step of the automated generation of behaviour models is focussed in this paper. Therefore the research questions are:

- Which information is needed to create a behaviour simulation?
- How can the creation of the behaviour models be automated?

2 STATE OF THE ART AND RELATED WORK

2.1 Development process

The development process starts with the requirements that must be met by the technical system. There are many different approaches that describe the product development process. Naming all of them would go beyond this paper. Only some of them are mentioned, for example the concept of Hubka and Eder (1992), the Axiomatic Design Theory by Suh (1990, 2001) or the characteristic-property-modelling (CPM) approach by Weber (2005a). The CPM-approach describes the development process by using two alternating process steps. These steps are named synthesis and analysis (Figure 1). The synthesis step is the main process step to create a solution based on the requirements. The analysis step is the process step of validation. The results created during the synthesis step are validated during the analysis step. These steps are repeated until the results of the analysis steps meet the requirements with the given deviation.

Beside the process steps the CPM-approach also describes the data that is generated and needed for the development process. This data is named properties and characteristics. The properties describe the system in terms of the behaviour and cannot directly be influenced by the developer. The characteristics on the other hand describe the system in terms of structure, shape, material and consistence which can directly be influenced by the developer.

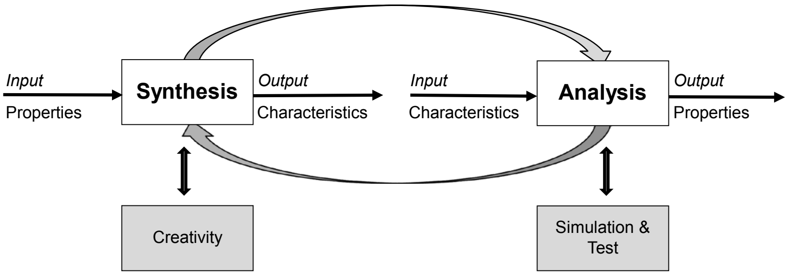


Figure 1. CPM-approach process steps (Reitmeier and Paetzold, 2010)

Initially the requirements to a system serve as the input parameter to the synthesis process. They describe the properties that the system must realise. During the synthesis phase the developer has to find a solution that should fulfil the requirements and also find characteristics that describe this solution. During the analysis phase the characteristics, which are the result of the synthesis process, serve as input parameter to the analysis phase and the developer has to evaluate actually achieved properties. This can be done with experiments or simulations. The occurring differences between the requirements and the actual achieved properties attend as additional input parameters for the following iterations. Because of this process description of the development process the CPM-approach is used for this paper.

2.2 Model-based systems engineering

The development of products is done within a very heterogeneous environment. Many domains are working together and each domain has its own methods and tools. Because of a missing standardization of interfaces between developing tools the results of each method and tool are often stored document based.

Model-based system engineering (MBSE) tries to formalize models for the product description. The goal is to support the entire product life cycle starting with the requirements as well as the actual development process (INCOSE, 2007). It also tries to integrate different domains that work together to develop a product. It is postulated that complex technical systems are represented by a system model. This does not only support the entire development process (Weilkins, 2008), but also helps to develop balanced system solutions in response to the needs of various stakeholders (Anderl et al., 2012). Two perceptions of a system description have to be considered, which are the structural-based and the functional-based view. Figure 2 shows these two different views and initial requirements that lead to these views.

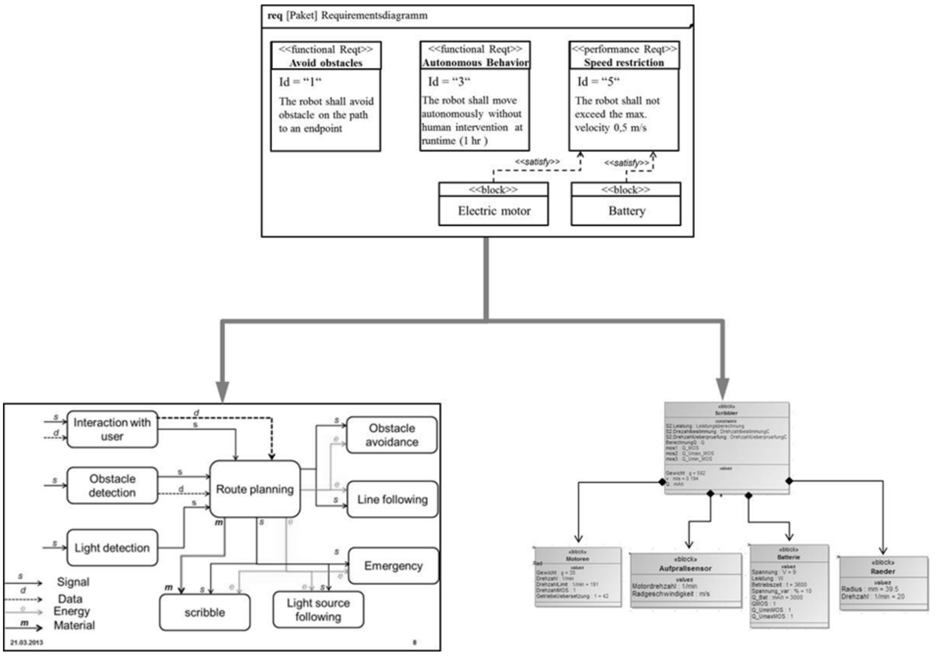


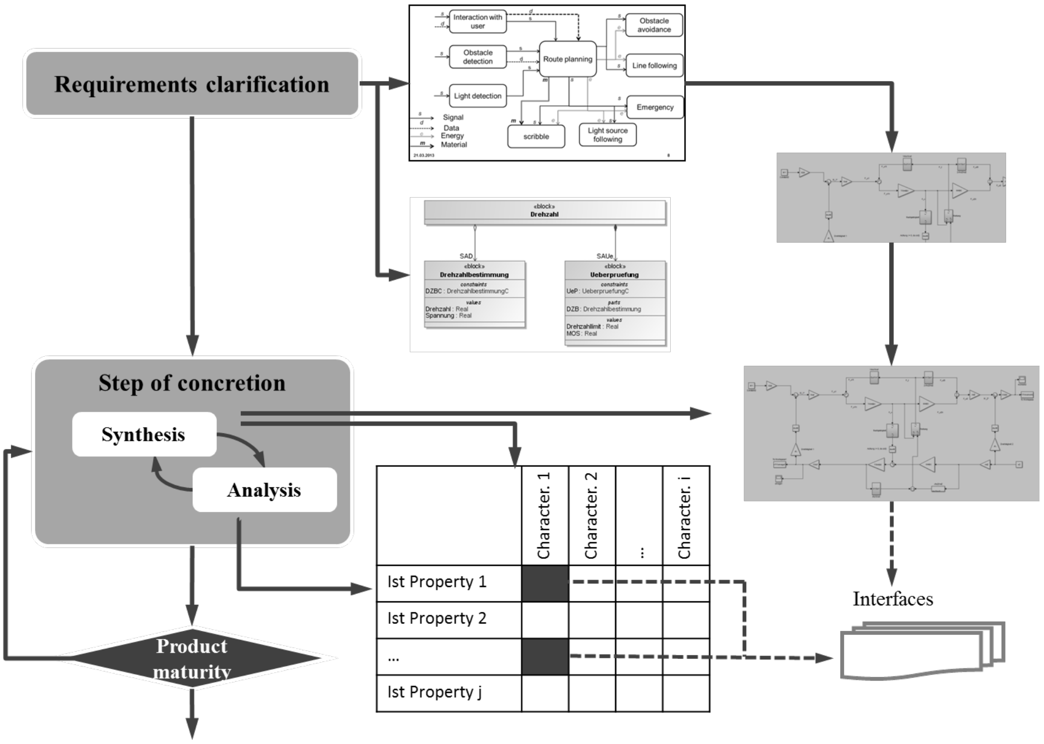
Figure 2. Functional and structural representation of a technical system with SysML (Paetzold and Köẞler, 2014)

The structural-based view describes the system based on the segmentation into assemblies, subsystems and components. These segments are combined hierarchically to represent the complete system. The functional-based view describes the behaviour of the system based on functions that are needed to fulfil the requirements to the system. The combination of functions is generated by connecting their input and output parameters. By using the functional description the system is represented in terms of a flow model.

Many different approaches are existing to implement the MBSE, e.g. MBSE approaches of NASA (Nasa, 2007), OOSE (OOSEM, 2006) and Vitech (Estefan, 2007). All these approaches are focusing on the description of the complete system based on functional considerations. In the following the approach of OOSE provides the basis. This approach ends in the modelling approach of SysML. SysML serves as a modelling language which allows to combine structural and functional descriptions of technical systems. It enables the description of a technical system concerning structural and functional views and also supports the detailing of these descriptions during the development process. So far SysML is still not completely integrated into the development processes and the development environments. First approaches to integrate SysML are currently being developed (Augenbaugh et al., 2004; Eigner, 2012), but only a few links to tools exist. However the SysML definition of a system can be used to generate parameterized behaviour models. While the structural view provides the components of the system, the functional view affords their relations. Each view can contain additional information which can be used for parameterizing the models.

2.3 Integration in the development process

As already mentioned the development process has to deal with the two process steps synthesis and analysis. The challenge is to integrate these steps into a continuous development process. In this context product data management (PDM) systems serve as a key element. They are supposed to support the data management as well as the data and information flows. Currently first steps of an approach are being developed that describe how properties and characteristics, which are part of the CPM-approach, can be stored in PDM-Systems to support the integration (Kößler et al 2014). With the use of SysML to describe the system additional data occurs that has to be integrated into existing PDM-Systems (see Figure 3).



	Character. 1	Character. 2	…	Character. i
Ist Property 1	■			
Ist Property 2				
…	■			
Ist Property j				

Figure 3. Integration of the interface description in the data and information flow (Paetzold and Kößler, 2014)

This figure represents a simplified process of the CPM-approach. This simplification does neither show the dependency of the development process to external conditions nor the temporal change of characteristics and properties. However the actual steps of concretion and additional sources of information like behaviour models are shown. Many different tools are available and are used in the industry which leads to the need of integrating these tools into the development process and support the data and information flows.

3 GENERATION OF BEHAVIOUR MODELS

The system integration is getting more and more important for companies. Beside the components the interfaces between the components and their descriptions are focused more intensively during the development. Therefore a holistic system view to the product is necessary. A simulation of the complete system behaviour in early stages helps to improve the development process. The improvement is achieved by the reduction of iterations with the use of simulations to evaluate interfaces. To support this, there are two different possibilities to describe the system:

- The structural view
- The functional view

The structural view tries to represent the architecture of the system. With the use of SysML as the modelling language, this view is mainly created by the generation of a block definition diagram (bdd). These diagrams consist of blocks whereby each block is the description of a component or an assembly of the complete system. In particular during the early stages of the product development process these structures do not represent the exact structure of the final product, but a valid simplification.

The functional view on the other hand tries to visualize the functions and sub-functions that are necessary to satisfy the requirements. The combination of both views allows identifying which functions are met by the different components. As a result the dependencies between different functions and components can be identified.

In Paetzold and Kößler (2014) a method has been described that uses SysML as the modelling language and behaviour models to simulate the behaviour of a system. This method is based on the deviation of executable models from SysML block definition models and functional models. The models help to verify the behaviour of the designed system in comparison to the expected behaviour which is also modelled using SysML. This comparison supports the evaluation of the completeness of the requirements. With the use of tools like Matlab/Simulink, time-continuous systems as well as discrete event systems can be represented. This allows to integrate different domains like mechanical engineering or software modelling into the behaviour model. Thereby it enables the verification of interests of different stakeholders on the technical system. The models also represent the assignment of functions to the components of the system, which fulfill these functions. Also these models are not very accurate concerning the topology, they allow the identification of interfaces between these components and the evaluation of their sensitivity by varying their descriptive parameters. The description of the interfaces supports the development process related to the integration and is an important information for controlling the development process (Figure 4).

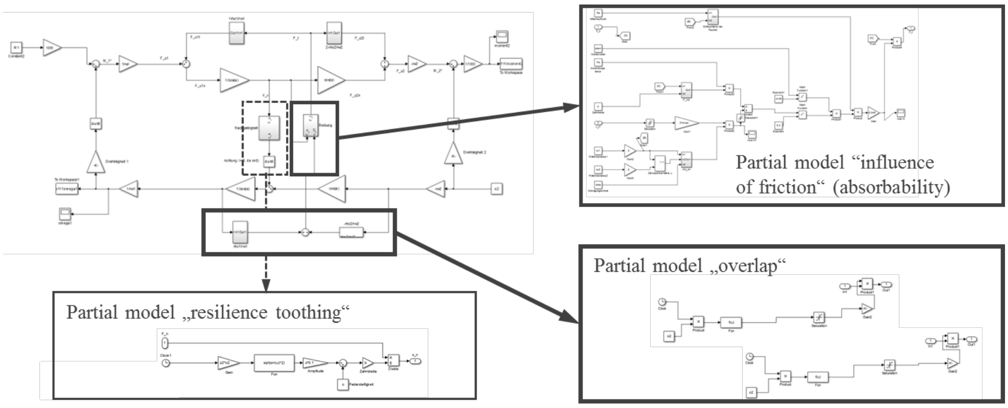


Figure 4. Detailing of a behaviour-based model using the example of a gear (Paetzold and Kößler, 2014)

The model in Figure 4 shows the behaviour model of a gear unit. In early stages only the conversion of speed and torque based on the requirements is known. During the development process additional information can be assigned to the SysML models and thereby be used to generate a more detailed simulation model. In the given example these are the influence of friction, the overlap and the resilience of the toothing. By detailing the simulation models the impact of the interfaces concerning the complete system behaviour can be evaluated. By changing the parameters that describe these interfaces the sensitivity of the parameters as well as the sensitivity of the complete interfaces can be assessed.

To support the generation of simulation models the knowledge of systems engineers is needed. To reduce the dependency of experts it is reasonable to provide a library that offers models for common components and functions that are used in many different products. A possible description of these models has been provided by Weber (2005b). Weber has shown how the behaviour of machine element can be modelled by the use of the multi-pole theory as known from electrical and control engineering (Figure 5).

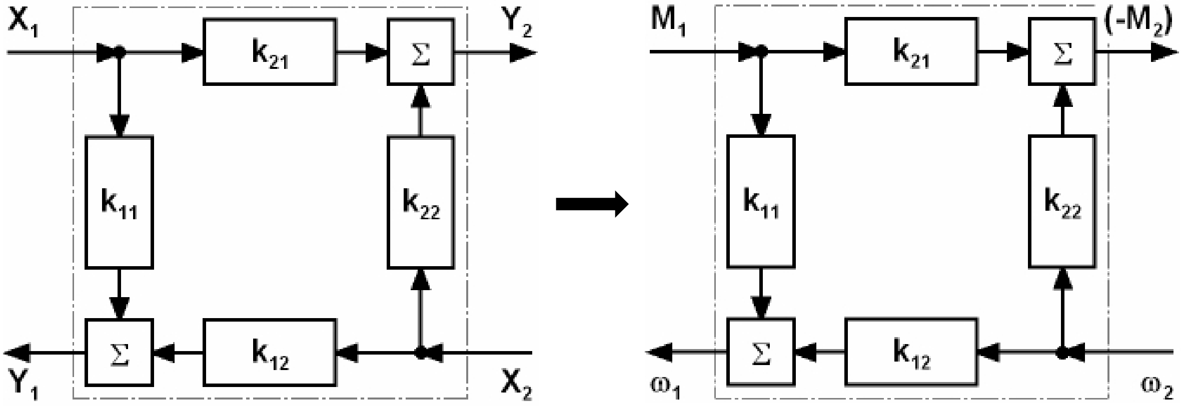


Figure 5. Multi-pole theory used for machine element (Weber, 2005b)

This figure shows the general representation of the multi-pole theory and the adaptation for mechanical system. A gear unit is presented which converts torque and speed. To generate a more complex system it is necessary to find representations for different machine elements and find a possibility to combine the elements to create a complete system. These representations of machine elements should be stored in a library to make them available for the generation of a complete system model. The given example describes a gear unit which could consist of two gear wheels. The simple conversion of torque to another torque can be described just by one parameter the gear transmission ratio. With the representation of the gear unit with two gear wheels the first gear wheel just converts a torque to a force while the second gear wheel converts the force to a torque. The more detailed system is described by two parameters the diameter of each gear wheel. Therefore the representation of a single gear unit consists of the input parameter torque, the output parameter force and the parameter diameter. This description can be used as a partial model and by combining different partial models a complete system could be generated. The behaviour of a gear wheel concerning the speed is similar and can also be added to the partial model. These models can be generated for different machine elements depending on the forces and torques they gather and release.

To automatically generate a behaviour model of a system partial models for each component are needed. They must contain definitions for the input parameters (e.g. torque), output parameters (e.g. force) and descriptive information. The input and output parameters are necessary to allow the combination of partial models to a complete system model. Currently they only need the information concerning their unit and direction. The descriptive parameters allow a more detailed description of the behaviour. Using the gear wheel as an example the diameter is required to actually describe the conversion of torque to force. But additional information like material or the geometry of the teeth of a gear wheel can be used to generate a more detailed model as shown in figure 4.

4 IMPLEMENTATION OF THE MODEL GENERATION

Based on the approach described in chapter 3 a functionality and a tool that implements this functionality has been developed to support the generation of executable behaviour models. Beside the generation of models partial models of machine elements have also been developed. The partial models are necessary to create a complete system model. Currently only some partial models for basic machine elements are existing (e.g. gear wheels, shafts), but the model library is still further developed. The models are generated to be used with Matlab/Simulink, but any other tool could be used. In addition we are working on an integration of these models into a PDM-System. The basic process of automated model generation is shown in Figure 6.

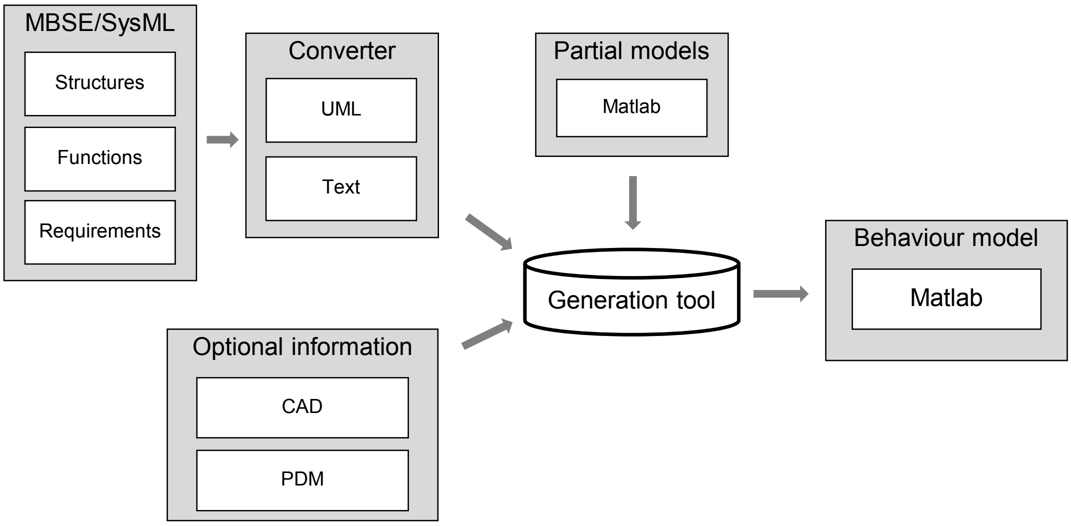


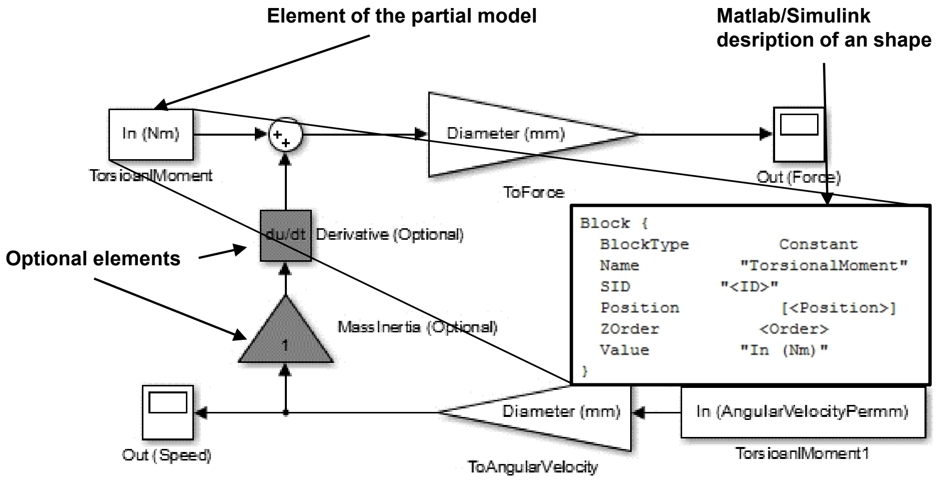
Figure 6. Basic concept of simulation generation

As shown, three different sources of information are connected for the generation of the simulation model. These sources are the actual SysML definition of the system, the library of partial models and additional optional information.

The main source of information for the behaviour model is available within the structural and functional representation of the system in SysML. As a result of the development process this information is offered by a block definition diagram. By using default export functionalities of the SysML tools these models can be exported to a UML-model, which is basically a structured text file. These files can be parsed by the tool to extract the information that is required for the simulation model. This information is:

- Data of the blocks
- Each block offers the possibility to add information to describe it (e.g. the diameter of a gearwheel, assigned partial models). This data is used to identify components that are used within a behaviour model. Additionally the values of each block are extracted to make them utilizable for the partial models.
- Connections between the blocks
- The combination of the blocks or the functional structure defines the architecture of the behaviour model. Each connection needs the information concerning the direction of flow and the unit. This describes the type of the connection (e.g. force, torque or speed). Within the model the connections can be defined by using different types of diagrams. (e. g. Block-Definition-Diagrams or Activity-Diagrams)
- Information concerning the partial models
- To actually create a behaviour model for a particular tool, it is necessary to create a relation between a block of SysML and the particular tool. This relation is created by assigning a partial model of the simulation tool to a block of SysML.

The next source of information is the library of partial models of the simulation tool. This paper addresses the simulation tool Matlab. Therefore the library only contains partial models of Matlab, but it is also possible to integrate other simulation tools. Each partial model consists of different types of information. This information is the header information on the one hand and the body information on the other hand. The header information consists of a name to identify the model, the simulation tool and the input and output parameters of the model. By comparing this data with the specified data of the SysML blocks it is possible to identify the correct partial model, which will be used for the behaviour model. The body information consists of a detailed definition of the simulation model data. This definition allows the actual creation of a complete behaviour model that is executable with the chosen simulation tool. It also offers additional optional parameters (e.g. mass inertia) to create a more detailed model. If this information is not available during the generation of the behaviour model, the optional elements that use this data are removed (Figure 7).



```
Block {
  BlockType         Constant
  Name          "TorsionalMoment"
  SID         "<ID>"
  Position           [<Position>]
  ZOrder          <Order>
  Value         "In (Nm)"
}
```

Figure 7. Partial model “Gearwheel” of simulation tool “Matlab/Simulink”

The third source of information are additional sources like CAD-files or data that is stored in PDM-systems. During the very early phase of the development process this information is not available. However with the progress of the developing more and more data is stored in other systems instead of using SysML. Therefore the generation tool offers interfaces to add this data and make it available for the behaviour model generation.

Figure 8 shows two examples how the generation tool converts a SysML block definition into a behaviour simulation model.

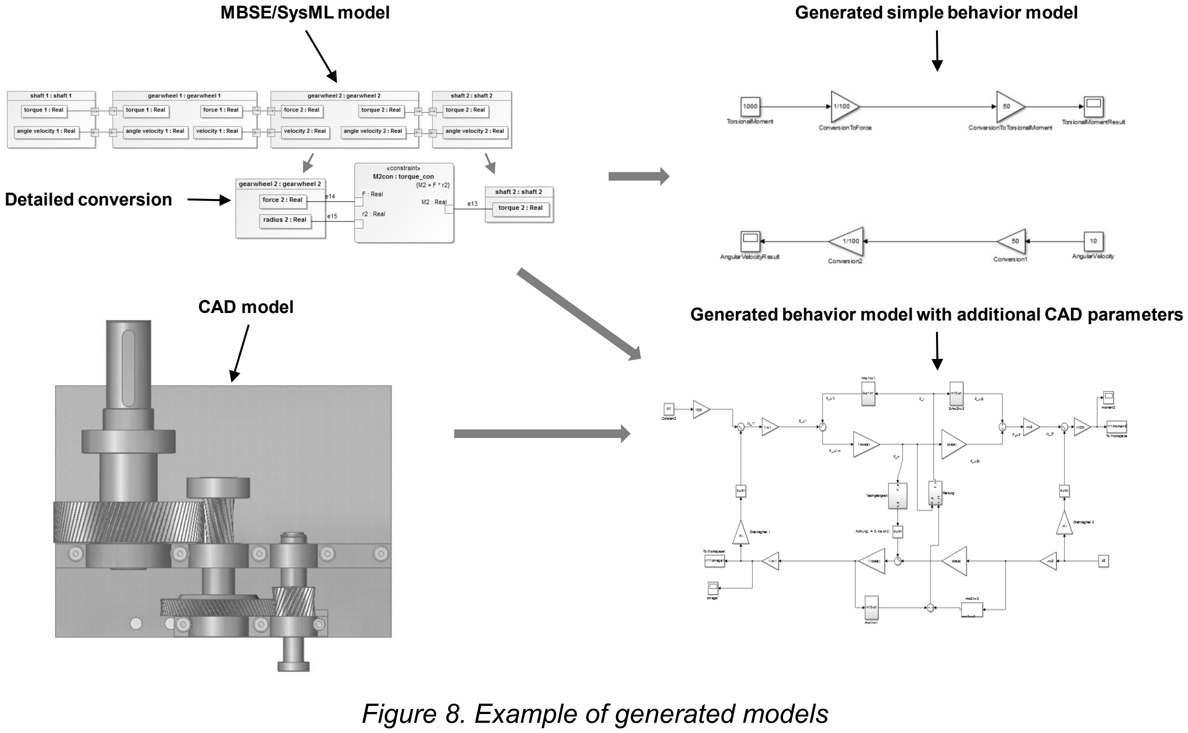


Figure 8. Example of generated models

The aim of this model is the simulation of a gear unit. This simple gear unit basically consists of shafts that convert an incoming torque and speed by the use of two gearwheels. This system is modelled within SysML and the essential information for the simulation generation is added. The first generated behaviour model with the name "Simple behaviour model" just uses the SysML model for the generation of the simulation model. As a result of the simple SysML model the generated simulation model only consists of the conversion of torque and speed.

The second behaviour model with the name "Behaviour model with additional parameters" uses the SysML and additional information that is provided by a parameterized CAD file. With the use of this additional data a more complex simulation model is generated. This model consists of the conversion of torque and speed and also considers mass inertia and stiffness.

5 CONCLUSION & OUTLOOK

In this paper the initial steps to automatically generate behavioural simulation models have been presented. As a basic definition for this models block definition diagrams are used that are defined using SysML. Based on the multi pole theory a library has been created that contains partial models for some machine elements. These partial models can be assigned to blocks in SysML. These partial models and the block definition diagrams are the key information needed to create a behaviour model.

In addition a tool has been developed and presented that uses the SysML definition of a system and automatically generates an executable behaviour model for Matlab. This tool utilises a UML representation of the SysML block structure, the containing values of the blocks (as defined in SysML), the assigned partial models and the connections between the blocks for this generation. It is also able to use additional information stored in other systems like PDM-Systems or CAD-files. Beside that a library has been generated providing partial Matlab models that can be used for the generation. These models contain definitions of necessary and optional information required for the generation. Based on the available data the level of detail of the behaviour model is changing. Thereby the behaviour models can be used in early stages as well as in later stages of the development process.

The functionality of this tool has been shown with an example of a gear unit. Based on the results of this example the following tasks came up. The library of partial model has to be enhanced to allow the integration of more machine elements into the behaviour models. Also more complex models have to be generated to evaluate the functionality and the amount of support of this tool during the development process. In addition this method has to be integrated into the development process. Currently we are also working on the integration of SysML as well as the integration of behaviour models into a PDM-System. This integration should support the development process with the reduction of interfaces between different development tools.

REFERENCES

Anderl, R., Eigner, M., Sendler, U., Stark, R. (2012) Smart Engineering – Interdisziplinäre Produktentstehung – acatech DISKUSSION.

Aughenbaugh, J.M., Paredis, C.J.J. (2004) The Role and Limitations of Modeling and Simulation in Systems Design, ASME International Engineering Congress 2004, Anaheim, USA.

Eigner, M., Gilz, T., Zafirov, R. (2012) Neue Methoden, Prozesse und IT Lösungen für die virtuelle disziplinübergreifende Produktentwicklung, in: Berns, K.; Schindler, Christian; Dreßler, K; Jörg, B.; Kalmar, R.; Zolynski, G. (Hrsg.): "Proceedings of the 2nd Commercial Vehicle Technology Symposium 2012"

Estefan, J. A. (2007) Survey of Model-Based Systems Engineering (MBSE) Methodologies, Technical report, California Institute of Technology, May 25 2007.

Hubka, V., Eder, W.E. (2007) Einführung in die Konstruktionswissenschaft. Springer, Berlin-Heidelberg, 1992.

INCOSE "Systems Engineering Vision 2020", Technical Operations International Council on Systems Engineering (INCOSE).

Kößler, J., Reitmeier, J., Paetzold, K. (2014) Implementation and automation of the characteristic-property-concept within a PDM-system, Proceedings of NordDesign 2014, Espoo Finnland, August 27th – 29th 2014.

NASA/SP-2007-6105 (2007) NASA Systems engineering Handbook

OOSEM (2006) "Object-oriented Systems Engineering Method", OOSEM Descriptive Qutline for INCOSE SE Handbook Version e, Annoted update.

Paetzold, K., Kößler, J. (2014) A system integration method for the continuous support of the design process, In: Marjanovic, D., Storga, M., Pavkovic, N. and Bojcetic, N. (eds) 13th International Design Conference - DESIGN 2014, Dubrovnik, University of Zagreb/The Design Society.

Reitmeier, J., Paetzold, K. (2010) Property and behaviour based product description - component for a holistic and sustainable development process, In: Marjanovic D., Štorga M., Pavkovic N., Bojcetic N. (Hrsg.): Proceedings of the 11th International Design Conference DESIGN 2010, Dubrovnik.

Suh, N.P. (1990) The Principles of Design. Oxford University Press.

Suh, N.P. (2001) Axiomatic Design. Oxford University Press.

Weber, C. (2005a) CPM/PDD - An extended theoretical approach to modelling products and product development processes, Proceedings of the German-Israeli Symposium on Advances in Methods and Systems for Development of Products and Processes, Bley, H., Jansen, H., Krause, F.-L., Shpitalni, M. (Eds.), Fraunhofer Stuttgart.

Weber, C. (2005b) Simulationsmodelle für Maschinenelemente als Komponenten mechatronischer Systeme, Proceedings of 50. Internationales Wissenschaftliches Kolloquium, Technische Universität Illmenau, 19.-23. September 2005, Illmenau: Scharff, P.,

Weilkins, T. (2008) Systems Engineering mit SysML/UML: Modellierung, Analyse, Design, dpunkt Verlag, 2nd. Edition.