INTERNATIONAL DESIGN CONFERENCE - DESIGN 2014
Dubrovnik - Croatia, May 19 - 22, 2014.

A SYSTEM INTEGRATION METHOD FOR THE CONTINUOUS SUPPORT OF THE DESIGN PROCESS

K. Paetzold and J. Kößler

Keywords: model-based systems engineering, system integration, data- and information- flow, interfaces

1. Introduction

Today's system development is faced with an increasing complexity. This results not only from the increasing functionality that must be met by technical systems but also from a greater consideration of the product life cycle. In addition, the behavior of complex technical systems must often meet the demands of very different stakeholders. One of the major challenges in the future will lie in the fact that technical systems are characterized by a greater openness to the environment; the product functionality must be increasingly ensured in the context of an unstructured environment, which is difficult to be fully described by the designer. Complexity ultimately results from the fact that already the components or parts of the technical system require a high knowledge of certain specialist fields, which can nowadays only be limited provided by companies. As a result, component development is already being outsourced to suppliers. This division of labor also shifts the core competencies of the companies whose know-how is no longer in the detailed solution but in the integration of components from different suppliers.

Aspects of system integration get more and more importance to ensure the product quality, as the technical integration has been ranked among the top four challenges in systems engineering [Sage and Lynch 1998]. Based on this fact, methods of systems engineering are available that postulate a holistic perspective not only on the technical system but also on the associated project management. For this, a variety of approaches from the product development can be used, but generally these support only partial aspects of a solution-finding process. The system integration is nowadays mainly processed at the end of the development process and, in addition, not being adequately supported.

Therefore the research questions are:

- How should the data and information flows in the product development process be organized in order to get a higher significance of system integration in development process?
- How can existing methods and tools from product development be used to support the system integration.

These thoughts led to the hypothesis that nowadays in case of the strong division of work, the data- and information flow will be continued and concretized in a number of parallel streams with view to concurrent engineering. The challenge will be the coordination and integration of these streams of data and information flow. Therefore, in this contribution an approach will be presented, which supports the system integration by determining interface descriptions based on the system requirements and which can be adapted and completed during the development process in the context of product concretion.

2. Model-based systems engineering as initial basis

The Starting point of the considerations is the question how data and information flows can be better matched and coordinated in a distributed development. Today, product development is primarily done document based. This means that existing data and information of a technical system are generally available. Nevertheless, they are very heterogeneous in their processing as they offer different views on the technical system. In addition, there are indeed link-ups and relations between documents but rarely between data. These aspects make it difficult to derive direct relationships between the data and information from several different development teams.

The basic idea of model-based systems engineering (MBSE) is to formalize models for product description so that these support the entire product life cycle concerning the determination of system requirements as well as steps of synthesis and analysis, starting with the conceptual phase [INCOSE 2007]. It is postulated that complex technical systems are represented by a system model, which not only supports the entire development process [Weilkins 2008] but also helps to develop balanced system solutions in response to the needs of various stakeholder [Anderl et al. 2012]. To describe a technical system, two kinds of perception have to be generally considered: a structural-based viewpoint describes the system architecture based on the segmentation into subsystems and components and a functional-based viewpoint which explains the behavior of the system by coupling input/output-relations. Both viewpoints are necessary for system integration. The coupling of subsystems occurs on the basis of input/output relationship and contributes to the understanding of the behavior of the overall system. The segmentation into components is normally connected with the operational structure and the definition of design-teams in product development. At the same time the structural view is manifested in the product structure which is implemented in used in corrent product data management systems (pdm-systems).

In this context, different approaches are being pursued, e.g. MBSE approaches of NASA [2007], OOSE [2006] and Vitech [Estefan 2007]. All approaches have in common that they are focusing on a complete system description based on functional considerations. The approaches of NASA and also the approach by VITEC permit a modelling of system states based on a behavioral description. Material, energy and information flows of technical systems can be modelled. The fundamental considerations associated with these approaches seem to be purposeful, since they may be regarded as domain-independent and thus contribute to a general understanding of the system by all parties.

In the following, the approach of OOSE which ends in the modelling approach of SysML serves as a basis. The concept of SysML offers the possibility to couple a structural description of the technical system with the functional description. Thus, a fundamental requirement of systems theory is picked up; for a complete description of the technical system both a functional and a structural point of view are required [Patzak 1982].

However, SysML is currently (ultimately like all MBSE approaches) a stand-alone solution. This means that on the one hand this tool offers the possibility to describe the technical system considering function and structure, to detail it during the development process in terms of a concretion and on the other hand, there are also methods for a SysML-based interface description [Fosse and Delp 2012]. In addition methods have been in the past developed to specify the system description using SysML.

But there are no links to standard tools and methods of product development. First approaches are currently being developed [Augenbaugh and Paredis 2004], [Eigner et al. 2012], but they only offer selective links and are not yet mature. A holistic integration of the approaches and ideas of SysML in data and information flows is required. Major challenges in the development of complex technical systems are the description, specification and in-process control of interfaces within the technical system. Again, there are first approaches in SysML, but these are only usable within the tool and do not have a sufficient link to the data and information flows in the development process.

In the following, a concept will be presented, which is based on a requirement description in SysML, thus, using a product model which combines both architectural and functional view, adds interfaces in technical systems. Starting point for the interface definition is the idea that interfaces represent additional input values for the component development, which can affect their behavior or could also arise as a result of component development. Here, the output values are of particular importance. The behavior of the components is not of particular relevance, but can be essential input factors to other

components as they significantly influence their behavior. Structural factors, that influence space conditions for example, may be considered as interfaces as well. These are also important for system integration but play only a minor role in the following.
With this interface-description, it is also possible to describe interfaces between design teams. It can be further used to support development projects and to control the product maturity management of a technical system. The presented approach considers adaptive design as well as variant design, whereas an assumption is made on the fact that basis product structures are known and become apparent in the development process, namely with the allocation of sub-functions to components.

3. Methodology for interface description of complex technical systems

3.1 Background

The description of interfaces in a technical system is based on a holistic systemic view, whereby technical systems on the one hand are described by its functions or sub-functions, but on the other hand, the structure description specifies and possibly also completes the sub-functions.
The basic idea of this approach to be described is summarized in the following. The development of technical systems follows the logic that functions and subordinate sub-functions are initially defined according to the system's purpose. The functional view is supplemented by a structural view by considerations concerning adequate physical solution principles. If, for example, an actuator has to fulfill a defined path-time characteristic (function), mechanical (gear) or electrical (stepping motor) solution principles are possible. Both points of view, function and structure, are interdependent and determine subsequent development steps.

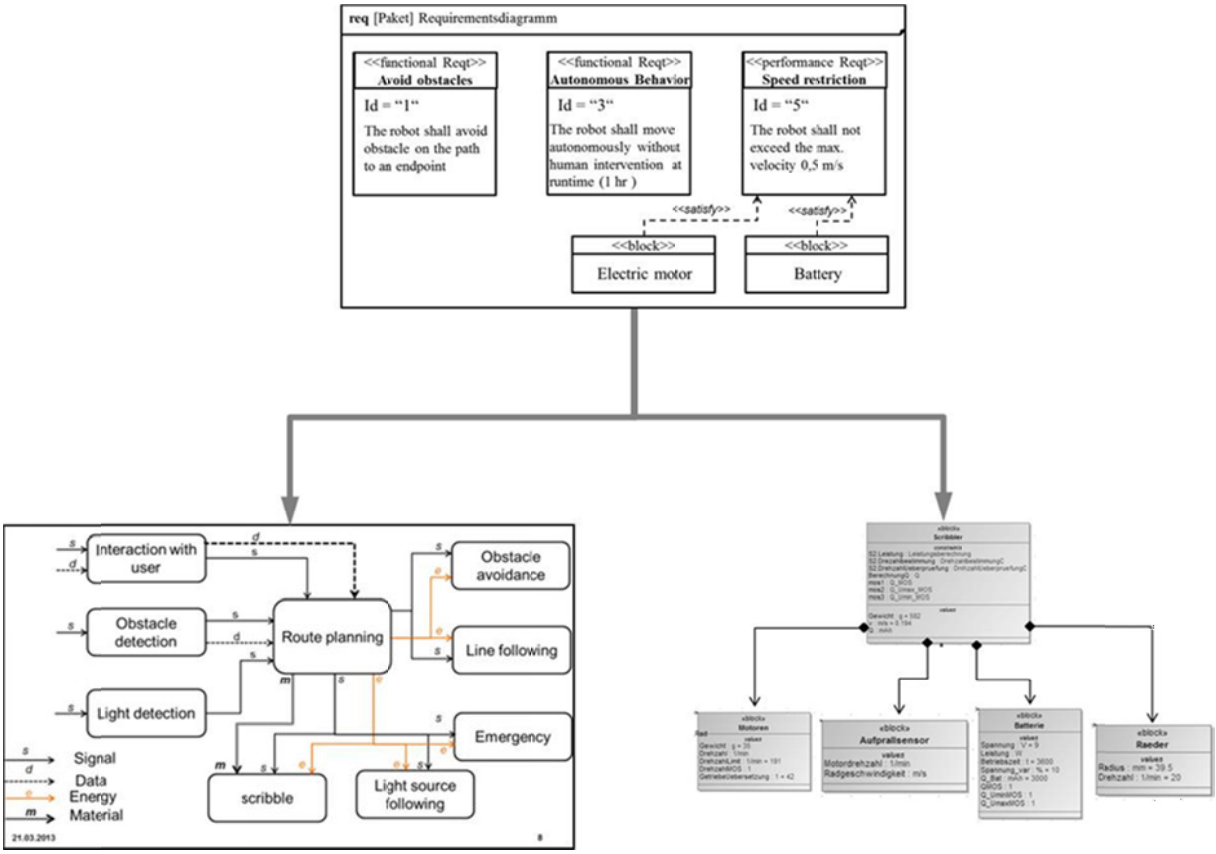

Figure 1. Functional and structural representation of a technical system by SysML

Product development starts with the definition of requirements. For the approach, which will be explained now, SysML will be used for capturing and processing with the requirements. Thereby the focus in the description of the goal-system is laying on both, function and architecture. The

identification of the requirements of a technical system by approaches of model-based systems engineering follows the logic according to Figure 1. These requirements can be functionally represented by transferring them into block diagrams and links by input-output relations. This system models correspond to the parameter-based modeling approaches in which the wirk-structure is reduced to the essential parameters which can be used for a simulation [Panreck 2002]. This basically allows to identify the system behavior in the early phases, thus within requirements engineering. For this purpose, a transfer of the functional model of SysML into an executable model (e.g. by Matlab/Simulink) is required (for example [Debbabi et al. 2010]).
In parallel, the requirements can also be transferred in a system structure. This does not correspond to the detailed structure description as in later phases of product development is defined, for example, in the modeling of technical systems (geometry in mechanical engineering, circuits in electrical engineering, source code in software development). Nevertheless, the component description may be considered as a valid simplification for the early stages of product development. An assignment of components is based on this structural description which nowadays determines the process-oriented organization of many companies.
Both forms of representation can be linked, so in the phase of requirements clarification it is already possible to determine which partial functions are fulfilled by the components. Dependencies between functions are identified by the functional description using block diagrams; consequently, the dependencies between sub-functions as well as dependencies between components can be identified. Based on this, the project will be fixed as responsibilities for the development of the individual components are assigned. This information can supplement the architecture descriptions.

3.2 Description and complementation of the system's behavior

As explained in the previous chapter, models can be derived based on the functional system description, which can be executed in tools like Matlab/Simulink or Dymola. Therewith an executable model of the goal-system is available. The result of the simulation can be used to assess whether the behaviour of the system coincides with the expected behaviour. This corresponds to the verification of the completeness of the requirements. Figure 2 shows a behavioral model for a gear, which initially describes only the speed and torque conversion. The consideration of the tooth meshing (in the form of coverage) or the tooth base stiffness also allows drawing conclusions on the dynamic behavior.
Tools like Matlab/Simulink can not only represent time-continuous systems but discrete event systems are also integrated by the link-up with state charts. This way, software, as a crucial component of today's complex technical systems, can also be considered when modeling the system's behavior. By evaluating the simulation results, not only the system's behavior can already be analyzed from the requirements. It can also be analyzed whether the interests of various stakeholders on the technical system are considered sufficiently with the designed system's behavior. One result of the analysis of the system's behavior in this phase of the product development process is also used to specify and complete the requirements by identifying the need of additional functions as well as to derive restrictions and extensions of ranges of values or parameters, which contributes significantly to reduce development risks.
The assignment of sub-functions to the structure description (architecture in SysML) determines a first component assignment, which can be transferred to the behavioral model. Although behavioral models in Matlab/Simulink are not really accurate concerning the topology, this association guarantees the link-up between model elements and partial functions or components. Consequently, behavior-based interfaces between components become apparent, which are not only described qualitatively. Rather, sensitivities in the dependencies between functions and components can be identified based on parameter variations.
Thereby a description of interfaces based on the requirements is available, which can be used to complete the data- and information-flow. In terms of the development-related system integration, the interface description should be adapted to the product maturity or the development progress. With each stage of concretion in the development process (e.g. associated to milestones or stage gates), the behavioral model can be detailed to specify the sub-components or sub-functions with respect to the new findings from the synthesis process. An example, the concretion of a gear, is shown in Figure 2.

While only gear ratio and power are known in the early stages, first, the overall size can be determined by the setting of the number of teeth and module. Concerning the properties of the gear, the tooth geometry has to be determined which in turn determines the tooth rigidity and the coverage, consequently, statements concerning the dynamic behavior and the noise emission are possible.

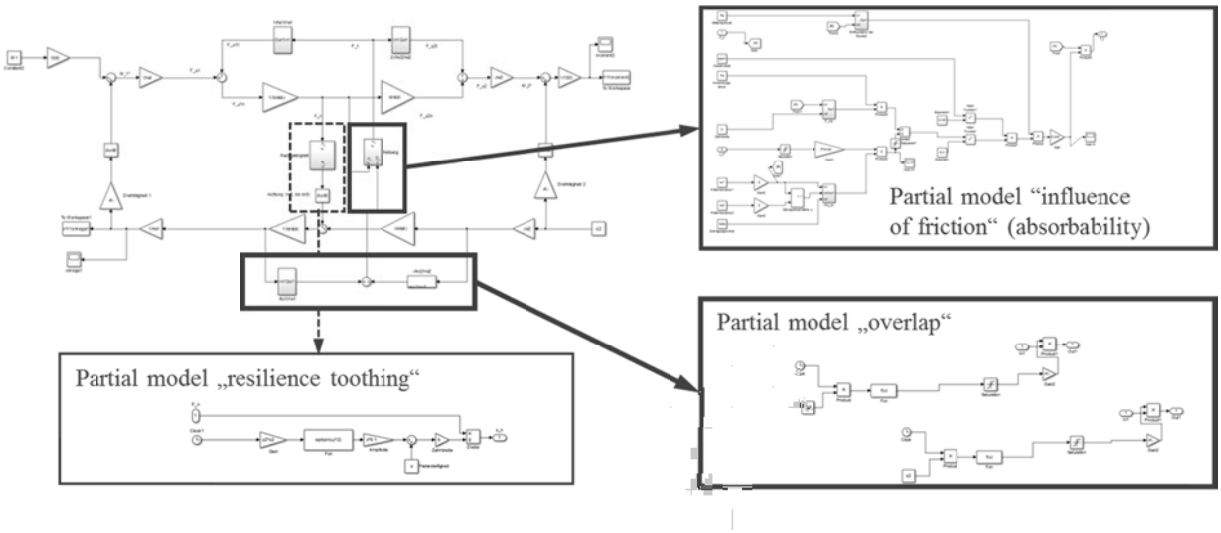

Figure 2. Detailing of a behavior-based model using the example of a gear

The subsequent simulations show, whether and how the determination of characteristics (see section 4) affect the overall system's behavior in the synthesis steps. This in turn allows to holistically concretizing the pre-defined interfaces between components and sub-functions.

To make this activity accompanying to the development process, the detailing of the behavior model based on the product specification is required. An increasing concretion of the structure of the technical system or the sub-components is connected to this. This structural description is, as already mentioned above, dependent on the specific domain (in contrast to the described behavior) [Reitmeier and Paetzold 2010]. To use this more concrete structure in the behavioral models, a reduction of the data on the parameters required for the overall performance and the integration is necessary. Two difficulties have to be considered here:

- Firstly, there is a need of estimating which concretion can actually have an impact on the overall system or only serve to optimize a functional differentiated component.
- Secondly, it is to be feared that specialists from the component development make this reduction from a point of view, which focuses on the important aspects with respect to the function of the component. Aspects that are important for the integration of the component into the overall system but not from a technical point of view are neglected.

The Simulink-models and the simulation results will be analyzed with respect to these aspects. To avoid this effect, a systems engineer is required for data reduction when identifying the progress or the product maturity, as he focusses more on the overall context of the technical system.

Here, it is a major responsibility of the system engineer to focus on such technical functions which play only a minor role in the component-specific detailing, but, represent quite important factors for the overall performance respectively other components. It is his responsibility to decide which level of detailing for the overall model is necessary. This can be generally based on a broad experience from the current product range.

The result of the model-analysis will be concretized with the interface description. Therewith interfaces description coupled to the development progress are available and can be used to support decision situations. This clarifies that it is not the objective of a behavioral simulation to validate single properties. The aim is merely to describe the interfaces and complete them accompanying to the process. The level of concretion of the behavioral model is closely connected to this objective. The

model refinement is only to the extent necessary so that relations between components and sub-functions can be clarified.

4. Integration of the approach in the data and information flows of the product development process

To efficiently provide this approach of interface description, an effective integration in the product development process in general and in project management in particular is required. For this purpose, a general analysis of the data and information flows is required. The description of the goal system provides the basic for data- and information-flow in the design process. Project-specific milestones help to analyze the product maturity.

Starting point of any development are the requirements that can be according to WEBER understood as desired properties to be fulfilled by the technical system [Weber 2005]. The product development process itself can be represented by the alternating steps of synthesis and analysis. Here, the product maturity increases with each step. The result of synthesis steps is an increasing detailing which requires adequate tools of analysis to validate the properties. It is crucial that the characteristics as a result of a step of synthesis are determined that way so that the required properties are achieved. These characteristics in turn are the input parameters for steps of analysis to identify the realized properties. Accordingly, developers have to deal with two categories of data: characteristics and properties [Weber 2005]. The distinction helps in the process of detailing the behavior model for the overall system. The realized properties, which are defined with milestones or stage gates, determine the refining of the model.

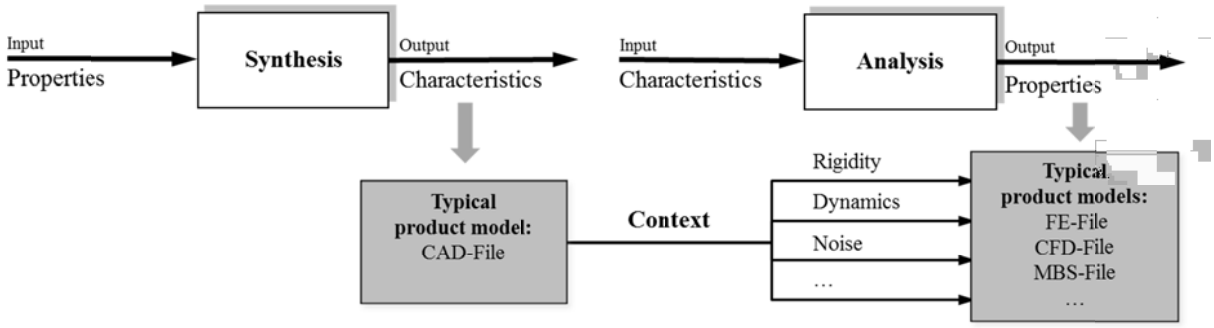

Figure 3. Data flow between steps of synthesis and analysis

Product data management (PDM) systems serve as a key element to support data and information flows. Assuming that this approach to the interface description focusses on adaptive design or variant design, the structure of a technical system is nowadays represented by these tools, which is normally based on a component-oriented view. In the PDM system mainly characteristics as a result of synthesis processes are stored in a structured way. Properties (e.g. results of simulations) are included as well as in a document-based form are stored document-based and, therefore, not directly linked with the characteristics is possible to only a limited extent.

It has proven advantageous that modern PDM systems provide a workflow functionality, so that they support the provision of data as well as the development process in the project itself, as workflow elements like milestones or stage gates, minimum requirements for the product maturity, responsibilities or release mechanisms are included and linked to the data processing. These workflows support in principle to coordinate not only the work of development teams but also between them. Thus, the necessary process-related conditions are provided to closer connect interface descriptions to the project management.

To display the extended interface description from the simulation of the technical system and thus the interface description between development teams, it is helpful to establish a link-up between characteristics and properties. For this purpose, a matrix-based representation seems reasonable

(Figure 4). By using matrices, not only the link-up between characteristics and properties (i.e. a Domain Mapping Matrix, DMM) but also the connection between characteristics and properties itself (i.e. a Design Structure Matrix, DSM) can be displayed [Luft et al. 2013]. In addition, these matrices can be supplemented be additional information, such as data quality or sensitivities [Reitmeier and Paetzold 2013], which in turn completes the interface description.

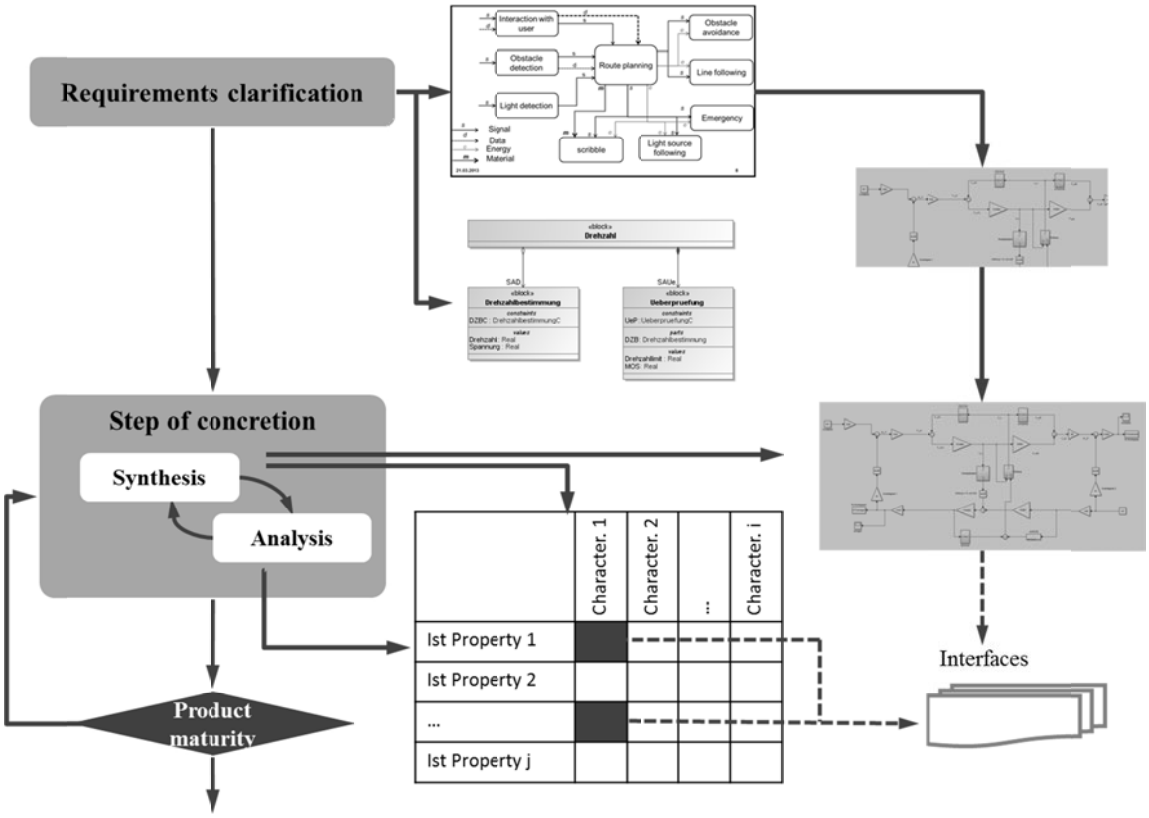

Figure 4. Integration of the interface description in the data and information flow

Summing up, the distinction between data classes occurring in the product development process and their matrix-based processing provides a significant increase of the information content, which may be used for the interface description and updating in the sense of a process-related system integration throughout the entire development process. The integration into a PDM system also allows the link-up to the development workflow, which also ensures a procedural and organizational consideration of the interfaces.

5. Conclusion and Outlook

Systems integration will be one of the most important challenges to handle systems complexity. The goal of the explained project is to develop methods which support the process of systems integration beginning with the early phases of product development. The basic consideration is to use data and information flow in the product development. These data and information flows will be extended with information on interfaces in the product as well as in the design process. This information will be generated with simulations.

The illustrated procedure supports the specification of interfaces between system components or sub-functions of a technical system based on known methods. The explained approach to concretize interfaces management in systems integration is based on simulations of the overall system behaviour with the help of parameter-based modelling approaches. The simulations to be executed are not

intended to identify specific properties but merely to specify input and output values of individual sub-functions in more detail, namely to the extent that the expected or at a certain product maturity realized system behaviour is specified. Nevertheless, these models can be used especially in the early stages to validate whether the realized system behaviour meets the required system behaviour or additions in the description of requirements or partial functions are necessary.
The refinement of the models which is necessary to identify the product maturity requires a statement about which factors of component development influence individual functions of other components in such a way that they are relevant for the system's behaviour. For this purpose, it is helpful to evaluate the experience and knowledge to a product group. Typical faults or weaknesses in the system behaviour provide indications where to focus on. The big challenge in the application of the approach described is to what extent it is possible to integrate the generated data and information in the product development process and to make these available for the relevant stakeholders. Therefore, one of the next key steps is the implementation of the new data model in a product data management system and the link-up thereof with developing workflows.
To identify interfaces based on simulations in our example we used Matlab/simulink. Other simulation tools are possible. Important is, that a parametric modelling approach will be used. The challenge is to couple the description of interfaces with a pdm-system. This is necessary to make the information usable in the design process to support decisions.
A weakness of this approach to interface description needs to be seen in the effort for modelling, with view to the modelling depth. The additional efforts with respect to real conditions within a project appear to be very high. However, assuming that similar components or sub-functions that are similar in major properties are used within a product range, it seems reasonable to establish model libraries for basic functions as well as for their detailing to reduce the modelling effort within a project. At the same time the expert knowledge can be stored in the model libraries, so it is possible even for inexperienced developers to better understand influences on the overall system behaviour and to take into account in their own work.

References

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

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

Debbabi, M., et al., "Verification and Validation in Systems Engineering: Assessing UML/SysML Design Models", Springer Verlag Heidelberg, 2010.

Eigner, M., Gilz, T., Zafirov, R., "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 (CVT 2012)"

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

Fosse, F., Delp, C. L., "Systems Engineering Interfaces: A Model Based Approach", California Institute of Technology, IEEEAC Paper, 2012.

INCOSE, "Systems Engineering Vision 2020", Technical Operations International Council on Systems Engineering (INCOSE), 2007.

Luft, T., Krehmer, H., Wartzack, S, "An advanced procedure model for property-based product development.", Weblink, In: Lindemann, U.; Venkataraman, S.; Kim, Y. S.; Ion, W.; Malqvist, Y. (Hrsg.): Proceedings of the 19th International Conference on Engineering Design 2013, 19.-22. August 2013, Seoul, Korea, ICED13. 2013, ICED13/305, 2013.

NASA/SP-2007-6105, "NASA Systems engineering Handbook", Rev., 2007.

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

Panreck, K., "Rechnergestützte Modellbildung physikalisch-technischer Systeme", Fortschritt-Berichte, VDI-Verlag, Düsseldorf, 2002.

Patzak, G., "Systemtechnik – Planung komplexer innovativer Systeme, Grundlagen, Methoden, Techniken", Springer, Berlin, 1982.

Reitmeier, J., Paetzold, K., "Process-integrated analysis of the development situation for an efficient simulation planning", Proceedings of the 19th International Conference on Engineering Design – ICED 2013, Lindemann, U., Venkataraman, S., Kim, Y.S., Ion, W., Malqvist, Y., (Eds.), Seoul, 2013.

Reitmeier, J., Paetzold, K., "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, 2010.

Sage, A. P., Lynch, C. L., "Systems Integration and Architecting: An Overview of Principles, Practices and Perspectives", Journal of Systems Engineering, Vol. 1, Issue 3, 1998, pp. 176-22.

Weber, C., "CPM/PDD - An extended theoretical approach to modeling products and product development processes", in: Proceedings of the 2nd German-Israeli Symposium on advances in methods and systems for development of product and processes. TU Berlin / Fraunhofer-Institut für Produktionsanlagen und Konstruktionstechnik (IPK), 07.-08. Juli 2005, Stuttgart, Fraunhofer-IRB-Verlag, 2005, pp.159-179.

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

Prof. Dr.-Ing. Kristin Paetzold
Universität der Bundeswehr München, Institute of Technical Product Developement
Werner-Heisenberg-Weg 39, D-85577 Neubiberg, Germany
Telephone: +49-(0)89-6004-2814
Telefax: +49-(0)89-6004-2815
Email: kristin.paetzold@unibw.de
URL: http://www.unibw.de/lrt3