21ST INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN, ICED17
21-25 AUGUST 2017, THE UNIVERSITY OF BRITISH COLUMBIA, VANCOUVER, CANADA

TOWARDS CROSS-LINKED DEVELOPMENT OF HIGHLY COMPLEX PRODUCTS

Toepfer, Ferdinand; Naumann, Thomas
Daimler AG, Germany

Abstract

Within the distributed development of highly complex products, complexity among involved individuals is a major issue. In order to allow transparency and consistency of information in between individuals and their Engineering Objects (EOs), Parameter Management is discussed as an approach which functions as an intermediator of information on basis of engineering parameters. Based on case studies in the automotive industry, the paper describes the underlying framework of the approach, addresses differences to Model-Based Systems Engineering (MBSE) and describes structures used to purposefully organize and cross-link parameter information.

Keywords: Complexity, Systems Engineering (SE), Parameter management, Design practice, Engineering parameters

Contact:
Ferdinand Toepfer
Daimler AG
Mercedes Benz Cars Research and Development
Germany
ferdinand.toepfer@daimler.com

Please cite this paper as:
Surnames, Initials: *Title of paper*. In: Proceedings of the 21st International Conference on Engineering Design (ICED17), Vol. 2: Design Processes | Design Organisation and Management, Vancouver, Canada, 21.-25.08.2017.

1 INTRODUCTION

This paper refers to the approach described by Toepfer and Naumann in "Parameter Management, a novel Approach in Systems Engineering" (2017), which is currently implemented at Daimler AG to support complex engineering processes based on engineering parameters. The initially mentioned article focused on a close-up description of the approach - including the significance of engineering parameters as format-neutral, product- and process-specific Engineering Objects[1] (EOs). Parameter Management was conceptualized upon the understanding of product development as a highly complex, interdisciplinary and distributed action- and process-system with a multi-criteria goal system (Ropohl 1978, Patzak 1982, Willke 1991, Ehrlenspiel 2013, Naumann and Koehler 2014). Based on the comprehension of systems in engineering design and their system models as omnipresent concepts of representation, parameter management can be seen as a holistic approach aiming for the management of complexity by supplying diverse model based engineering processes with model-neutral information. Parameter Management allows for more transparency and consistency among EOs and is not limited by heterogeneous EO formats. This contribution particularly focuses on the description of the underlying framework which is used to structure contextualized parameter information in order to make formalized knowledge retrievable for stakeholders of the development process, including engineers of various disciplines, designers, managers and many others. The article aims at fostering the understanding of challenges in the conceptualization of cross-linked development in order to support the management of complexity in the development of highly complex products. Based on the concept of so called "Active Chains" (Vester 2011) an approach is suggested which allows for more transparency, more consistency and more efficiency in communication among stakeholders of a highly distributed, interdisciplinary and complex development process.

2 CURRENT CHALLENGES IN THE AUTOMOTIVE DEVELOPMENT

Managing complexity is certainly one of the biggest challenges in the domain of automotive product development. The term complexity is most commonly referred to as a system property which is characterized by the quantity, variety and dynamic of system elements and their interrelations (Ehrlenspiel and Meerkamm 2013, Lindemann et al. 2009). It is a matter of system comprehension to accordingly identify sources of complexity in product development. Considering the fact of constantly increasing part numbers and buildable variants, it can well be stated that variance is in fact one key driver of complexity of technical systems. According to the principle of equivalence (Ropohl 1978, Naumann 2005) this technical complexity is consequently compensated by a rise in social complexity. Social systems work through the cooperation of individuals bound into multiple formal and informal, intra- and interdisciplinary structures. Interaction and communication lead to the forming of social structures which allow for an exchange of information. A rising number of individuals, subsequently leads to organizational complexity as coordinative efforts rise (Naumann et al. 2011).
Further drivers of technical complexity are modularization concepts which aim for the product-series spanning use and re-use of components (Baldwin and Clark 2000, Schuh 2005). This concept - effectively in use to reach scale effects on components - requires the standardization of interfaces and components and thus additionally limits degrees of freedom in the layout process. Most critical, changes in product-series spanning architectures affect more than one product-series and all their product variants. In the development process changes in the architectures lead to very complex situations as changes have to be validated in many product variants and disciplines. A shorter time to market in order to realize faster innovation cycles additionally leads to an increase in the parallelization of processes and decreases reaction time, leading to higher system dynamic. Combined with variant-rich and highly crosslinked products, design decisions can have huge impacts on components, overall functions, behaviour, and interconnected systems.

[1] EOs are referred to as the entity of structured information represented by heterogeneous graphical formats, in partial models of an integrated product- and process- model (Bitzer et al. 2007, Faisst and Dankwort 2007). In the context of this paper, e. g. requirements, functions, elements, concept elements, systems as well as parameters are considered as EOs.

While the description of the current challenges in automotive development is self-speaking in its urge for appropriate tool support, it is rather a question of how this high-level complexity can be approached.

3 UNDERLYING CONCEPTS OF SYSTEMS AND MODELS

The need for interdisciplinary and coherent approaches towards managing complexity in engineering design lies in the diversity of involved disciplines, their distinct perspectives of knowledge and their underlying system models (Patzak 1982). It is thus required to formulate an approach which is able to describe and unite the resulting manifoldness of system representations in order to manage complexity among stakeholders in a holistic manner (Naumann et al., 2011). An approach to do so lies in the concepts of systems theory.

3.1 Systems

Modern systems theory has its roots in the works of Bertalanffy (1949) and Wiener (1948) and led to the development of model concepts to unitedly depict formal similarities in different realms of reality. Ropohl (1978, 2009) accordingly describes general systems theory as an "exact model theory", based on three underlying system concepts, which even-handedly describe a system (Figure 1).

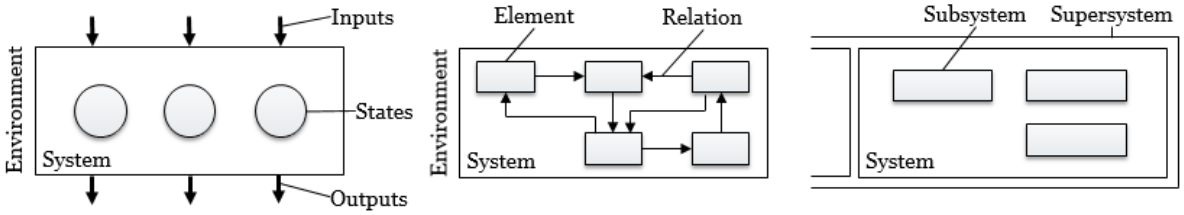

Figure 1. Functional, structural and hierarchical concept of a system (Ropohl 2009)

- **Functional Concept:** describes a system in the context of Inputs, States and Outputs. Inputs and Outputs cross the system boundary which mark-off the system from its environment.
- **Structural Concept:** describes a system as interrelated elements which -based on their properties - lead to a certain system behaviour. The elements of a system are not examined separately but in their overall interrelation.
- **Hierarchical Concept:** describes a system as a whole which is described through the presence of a superior and an inferior system. It allows for an examination on different levels of hierarchy.

Due to its formal abstractness, general systems theory allows for an adaption onto random realms of reality (Hubka 1973). It is thus that Naumann et al. (2011) use this concept to describe product development as a sociotechnical system on basis of a Meta-Model (Naumann and Koehler 2014) in order to describe complex phenomena in product development free from disciplines specific restrictions.

3.2 Models

The underlying concept of systems and their manifold representations is realized through models. According to Stachoviak (1973) models are characterized by the three characteristics of

- **Representation**
- **Reduction**
- **Pragmatism**

Models are thus always representations of natural or artificial originals. They are reduced to certain aspects and attributes of the original which are relevant for a certain creator or user of the model and they are designed for certain purposes and operations. Models are omnipresent concepts in engineering design and are used throughout the development process, from the first specification until the final validation of a system, strongly varying in abstraction, graphical format and complexity (Vajna et al 2009). The management and integration of resulting EOs is realized within an integrated Product- and Process-data-model serving as a Meta-structure (Gausemeier et al. 2001). In order to transform information towards a fully developed digital representation of the product, individuals drive forward the maturity of EOs as representations of manifold partial models. These EOs are highly interrelated among each other, processed by many individuals in different disciplines and finally have to be synthesized in a coordinated manner. PDM systems manage the handling of EOs, however do not

contribute to the exchange of their information content (Ostermayer 2001, Pavkovic et al. 2011, Koehler and Naumann 2014). The management of EOs and Engineering Object Relations (EORs) information however is crucial for the functioning of a social system in product development to drive forward the information transformation process of product development and to allow for traceability (IEEE 1990, Storga 2004, Aizedbush-Reshef et al. 2006,). This especially concerns representations of geometry generated in Computer Aided Design (CAD) tools and calculation models which are among the most commonly used model representations in automotive development.

4 SYSTEMS ENGINEERING, A QUESTION OF METHODOLOGY

According to INCOSE (2010) Systems Engineering is an "interdisciplinary approach…to realize successful systems". Weilkiens (2006) further outlines the methodological and tool-supported aspect in Systems Engineering in order to manage complex systems. Considering product development as a highly complex sociotechnical system that is based upon highly distributed tasks in many different disciplines, it is hardly possible to coherently describe the different forms of complexity inherent in all areas of product development. Dependent on the case, problems in complexity may be of different character and be dominated by different aspects such as uncertainty (Danilovic and Sandkull 2002), dynamic (Gomez and Probst 1997), variety (Malik 2003), intransparency, connectivity (Ehrlenspiel and Meerkamm 2013) or tolerance (Puhl 1999). The question can thus be asked whether there is a methodological silver bullet for handling complexity in product development. A methodology which is
suitable for one complex problem might not necessarily be suitable for a different or even similar problem.

4.1 Problem Statement

Based on the understanding of models as pragmatic representations of one or multiple system concepts, a major problem arises with the interoperability in between the multitude of partial models of an integrated product model. Amongst others Königs et al. (2012) address resulting challenges in communication, data-transparency and consistency, especially due to heterogeneous toolsets used. Rudolph (2006a) describes this problem based on mathematical analysis (Figure 2).

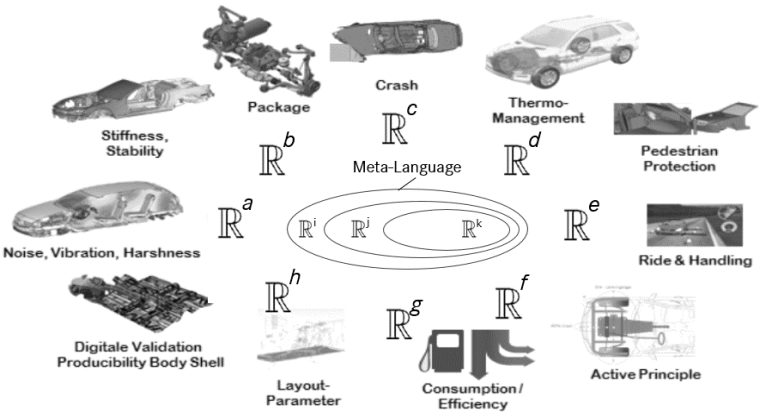

Figure 2. Allowing comprehensive Information exchange between domain specific models through a high dimensional Meta-Language (Rudolph 2006) (Toepfer and Naumann 2016)

As models in different domains possess different design spaces with different dimensionality (IR), there is no bijective mapping possible which would allow the mapping of design representations in between the domain specific models. In order to depict model spanning interdependencies Rudolph (2006a) outlines the necessity of a high dimensional meta-language which contains design information of multiple design representations.
However, the concept of using a meta-language is restricted to the design languages' own dimensions. Design representations outside these dimensions cannot be included in the languages' design representation.

4.2 Model-Based Systems Engineering Approaches

The concept of a meta-language is also used in the domain of MBSE. The Systems Modelling Language (SysML) was developed for use in the Systems Engineering domain in order to support the specification, analysis, design and validation of systems through machine readable modelling languages with a sematic foundation (Dickerson 2013, OMG 2017). Grown out of the Unified Modeling Language (UML) as a graphical language in software development, SysML is an extension for the application in Systems Engineering and has its roots in object orientated problem solving (Dickerson 2013). Based upon the object orientated programming (OOP) paradigm, UML was initially conceptualized to improve the creation of object orientation applications. According to Rudolph (2006b) the paradigm of OOP allows for creating viewpoints known from systems thinking including object hierarchies. Diagrams allow for depicting certain viewpoints of a system including objects, relations, conditions and activities by using different diagram types. Only the whole of diagrams eventually describes the system consistently and precisely and allow its use for problem solving (Rudolph 2006b). The resulting system model is an internal network of elements and their attributes which is graphically displayed through diagrams. Diagrams however are a graphical representation about parts of the model, but not the model itself (Eigner et al. 2014). It requires the definition and the modelling of all EOs and their EORs within a diagram based language (diagrams) and consequently the gradual incorporation of EOs.
It is questionable however, if it is always purposeful to explicitly model interdependencies among EOs who's EORs are implicitly known to designers anyway. On the contrary, the creation and maintaining of a system model requires high efforts (Koehler and Naumann 2014, Königs et al. 2012), expert knowledge and consequently additional roles of system engineers (Weilkiens 2006).

4.3 Parameter Management Approach

According to Toepfer and Naumann (2017) a parameter describes a quantifiable and model-specific characteristic. A parameter is a structured information about a model representation and can thus be considered an EO itself. The information a parameter represents, lies within the model representation of an EO. A parameter thus is an EO within an EO. Unlike the hosting EO, the parameter is independent of a graphical format. Parameters consistently describe characteristics, and their quantities in arbitrary EOs and can thus be used for consistent information exchange between heterogeneous EOs.
For instance: The parameter "Position of Engine" appears in multiple EOs such as from CAD and digital Mock-up models used for design and packaging. In the domain of driving dynamics, it is used in multiple-body simulation models for driving behaviour calculation. In Noise, Vibration and Harshness calculations the parameter is used in MATLAB models to allow for the layout of engine mounts.
Although the resulting EOs are not necessarily compatible among each other due to different dimensionality (see 4.1), they all share a common and format neutral element – the Engineering Parameter.
As outlined by Frankenberger (1997), Dörner (2000), Hacker (2002) individuals are limited in their psychological capacity. Changing, replacing or adding additional processes, models and model representations tools or organisational structures is perceived as difficult and challenging to many stakeholders. New approaches and tools are thus most often categorically rejected. Accordingly, Toepfer and Naumann (2016) state that stakeholders prefer to work in the context of their partial models and are mostly dismissive with new and particularly complicated tools. For an approach to be accepted by stakeholders, the following criteria have been identified as necessary to be met:

- Has to be easily understandable
- Has to require a minimum effort for the user to learn and use
- Should be integrated in the known workflow of stakeholders and use existing data
- The benefit has to demonstrably outrun the required efforts

The approach and corresponding tool which is currently implemented at Daimler AG to manage complexity amongst EOs was comprehensively described in (Toepfer and Naumann 2017) and repeatedly validated in different use cases such as the support within the process of validating a consistent vehicle architecture (Toepfer and Naumann 2016). Figure 3 gives an overview over the key aspects of organizing, managing and further processing parameter information in the Parameter Management Approach.

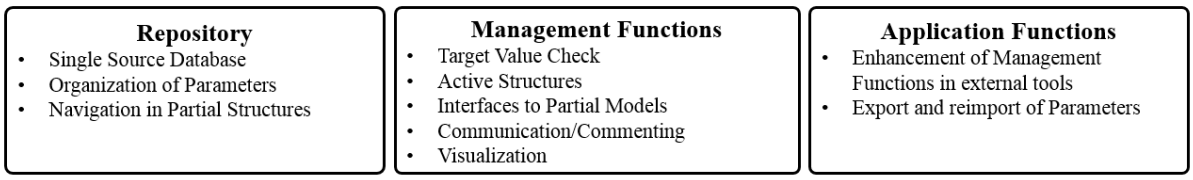

Figure 3. Concepts of Parameter Management to store, link and further process parameter information (According to Toepfer and Naumann 2017)

The following section particularly focuses on the underlying navigation structures in the repository (see Figure 3) which are used to organize and structure parameters, in order to make information and related knowledge retrievable for stakeholders of all processes. The repository addresses the concept of a database integrated into a complex IT-infrastructure and its multiple interfaces with author- and back-up systems which accesses subscripted parameters. In the repository, all information of parameters and their related EOs are organized, allowing for traceability among these parameter-hosting EOs. Parameters thus become the information hub for all authorized stakeholders and their EOs.

5 STRUCTURAL ORGANIZATION OF ENGINEERING PARAMETERS

As mentioned before parameters are integral parts of EOs as they are always contextualized through their hosting EOs and allow for a mapping of their structures and processes. A parameter itself is an EO which usually occurs multiple times in different partial models of an integrated product- and process-model. Parameters can thus be represented in different partial models and they can have different forms of presentation when used in different formats of the same partial model. Instances of parameters in both representation and presentation can differ which is normal as they undergo multiple changes in the transformation process of design. However, the challenge remains, to make changes transparent, for that individuals are informed when changes occur and to make information consistent so that individuals have access to up-to-date information (Königs et al. 2012).
The following sections address the concept of structuring information in different partial structures as parameter are always context specific information. The structures are subsequently used as filter criteria for stakeholders in order for them to retrieve required information.

5.1 Contextualization of Parameters

Parameters are contextualized in many different ways. As a quantifiable characteristic of any system, system element or function, parameters are parts of partial models which depict certain system aspects. It is thus crucial to understand and categorize the context of parameters to consistently manage their information content.
It is to be kept in mind that real-life scenarios are very complex and that the organization of parameters has to be sophisticated in order to make information retrievable and purposefully presentable for stakeholders of various disciplines. Hence navigation structures are used to relate the parameters to their systemic context. In order to make information –independent from domains- comprehensively categorizable and retrievable, the navigation structures are based upon the comprehension of systems (Section 3) and the proficiency of stakeholders. In total, five structures are used in the approach of Parameter Management to organize parameters. Three structures are used to assign parameters according to their systemic characteristics which are:

- Component structure (structural concept) (Section 5.3)
- Function structure (functional concept) (Section 5.4)
- System structure (combination of structural, functional and hierarchical concept) (Section 5.5)

Additionally, the two following structures are used:

- Product Configuration Structure (Section 5.2), to embed parameters in a product specific context explicitly taking variance driven complexity (see. Section 2) into account
- Active Chain Structure (Section 6) to be able to connect interrelated parameters in so called active chains in order to create a network of model spanning dependencies

In terms of modelling-efficiency the structures used in the Parameter Management Approach bear the following benefits:

- Structures are known to stakeholders from their working environment
- Consequently, minor efforts to understand the structures and the tool
- Modelling effort to contextualize parameters is reduced to a simple drag & drop mechanism
- Automatisms update parameter information automatically in structures

The concept of contextualizing parameters in navigation structures uses both pre-defined and self-customizable structures. While pre-defined structures offer a consistent way of handling information, self-customizable structures offer flexibility for problem-specific application of stakeholders.

5.2 Product Configuration Structure

The product configuration structure forms the product specific context of every parameter. Consequently, a parameter is always assigned to a product configuration. A product configuration thereby represents exactly one buildable variant. The concept of managing product configurations is a well-known concept in PDM systems (Ohl 2000, Eigner and Stelzer 2009). However, PDM systems do not necessarily manage product configurations in separate structures. Configurations can be configured through code-rules which function as a filter in an aggregated product structure. This product structure contains all components necessary to build every possible variant. With the application of code-rules, all components necessary for one configuration are selected (Ohl 2000). As the underlying framework of parameter management foresees a coupling of EOs from the PDM system into the data repository to keep parameters up-to-date, configuration specific EOs have to be retrieved and imported, so they can be mapped on a corresponding parameter instance defined through a specific configuration.
Depending on the type of the PDM structure used in a specific development process it can be necessary to use code-based filtering criteria. However, the number of possible variants (10^{53} for a current product-series) are not singularly developed and validated in the development process. For this number of variants only special, most difficult variants and variants determinative for certification are explicitly validated and monitored in special structures. Managing these variants within the product configuration structure has also been proven as a possible way of mapping relevant data from the PDM system to the Parameter Repository. Further structures especially loosely defined structures in pre-development vaults have to be managed carefully as their context is not easily to be mapped on the configuration structure.
In the Parameter Management approach architectures depict the highest level in the configuration structure (Figure 4) as architectures are product-series spanning concepts which standardize certain geometrical and functional characteristics for all variants below (Schuh 2005).

Figure 4. Product Configuration Structure for product-specific contextualization

Further levels of hierarchy between architecture and configuration are product-series and product-variant. This concept was chosen as parameters can always be broken down to a certain configuration, target values of parameters however are often specified on higher levels as they are valid for multiple configurations.

5.3 Component Structure

A product structure in the sense that it is used in PDM systems to derive product variants, is both an aggregated and composed structure which manages all component variants (Groll 2008). As the Parameter Management Approach does not use the concept of managing variants (products) by underlying code rules, the structure is simply referred to as component structure. This especially emphasizes the pragmatic comprehension of components as system elements.
The component structure is set up according to a particular product structure used in the PDM system at Daimler AG (Figure5). This is due to the fact that it is a well-known structure to the vast majority of developers. The retrieval of parameters via the selection of components is an easy task for most developers as they are familiar with the layout of the structure. The lowest level of this structure is the component level and the only level that parameters can be assigned to. It is due to the fact that higher

levels depict redundant information in an assembly context and that assemblies to some extent represent a system description.

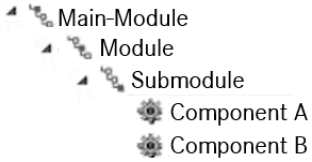

Figure 5. Component Structure for Component-specific contextualization

The component structure is an aggregation-structure on every level and only exists singularly for all products. This is different from PDM concepts which manage component structures for certain products with an underlying vault systematic. Due to the resulting quantity of possible component variants in the context of all products - built and to be built - it has been found the most efficient way to only include components in a type describing manner. This significantly reduces the number of components in the structure. The instantiation of component parameters is then realized via the relation of parameter and product configuration, as the product configuration by default describes its set-up.

5.4 Function Structure

Functional parameters are organized in the function structure (Figure 6). It is a composition of sub-functions which describes functional contributions towards one or more functions. Unlike the component structure which only carries parameter information on the lowest level, the function structure can be instantiated on every level and is not limited in the number of sub-function levels.

Figure 6. Poly-hierarchical Concept of Function Structure

Furthermore, it is possible to assign functional parameters to different functional trees (Figure 6). This poly-hierarchical concept is chosen as sub-functions can contribute to the realization of multiple functions.

5.5 System Structure

As described above, a system incorporates functional, structural and hierarchical aspects. In order to be able to unite a subset of parameters which describe these aspects of a particular system, the system structure operates as a configurable structure which represent the system concepts defined by Ropohl.
In the system structure a system can be composed of functions or sub-functions, assemblies or single parts. This concept is necessary as the function- and component- structure cannot be mapped on one another. It is because a parameter either describes a function or a component, however not both. Hence, they are either assigned to a function or a component. A relation can thus not be established through parameters. The system structure uses any object defined in the function- and component- structure and unites them underneath a system-knot in the system structure. System-knots can again carry multiple system-knots themselves allow for integrating subsystems into a system (hierarchical concept). In order to describe system characteristics themselves, parameters can also be assigned to system-knot, forming a system parameter. As part of the research conducted within the design of the tool it was also investigated if the system structure concept could be generalized to be applicable to different variants. This however did not turn out to work as a system is exactly one instantiation of a component- and function-structure. The use of a different component or function would automatically mean a change in the component-function mapping and describe a different system and system behaviour.

6 TOWARDS A CROSS-LINKED DEVELOPMENT ON BASIS OF ENGINEERING PARAMETERS

This article addressed the request for appropriate approaches and tools to manage complexity in product development. As future development will become even more complex in terms of technical, procedural and organizational aspects, stakeholders have to be granted access to tools which allow for a reduction of perceivable complexity. This will be particularly achieved by increasing transparency and consistency among EOs, to allow a more efficient communication among stakeholders in the information transformation process of product development.
Unlike MBSE Approaches which use graphical modelling languages as a meta-language to describe systems with their heterogeneous EOs and EORs, the underlying approach of this contribution uses existing models in engineering design and manages their parameter information as format independent information in order to establish a comprehensive and domain independent information exchange among system models. To establish a cross-linked development the coupling of EOs to explicitly "model" EORs is realized through active chains in the tool. Active chains are simple aggregations of parameters and can be created without restrictions by stakeholders to monitor parameters of interest. As parameters are hosted by EOs, the cross-linkage of parameters within in the active chain automatically represents a relation between the corresponding EOs. Active chains function as an intermediator of information between stakeholders and their models of a distributed development process and allow for traceability among EOs. However, for stakeholders it does not require the explicit modelling of interrelations between partial models with modelling languages. The concept allows for the creation of self-customizable interdependency networks. As there are no restrictions through modelling conventions, stakeholders can depict qualitative dependencies by simply dragging parameters of interest on active chains. Active chains can then further be managed within a self-customizable active chain structure. It allows for the aggregation and arrangement of active chains on multiple hierarchical levels to form active structures as a larger network of interdependencies. Upon this interdependency network further functionalities of the approach allow for an automated information flow when changes of parameters occur. Parameters that undergo critical changes trigger an information flow to all stakeholders who monitor the respective parameter. This concept is also part of the active chains. When a parameter - which is bound within one or multiple active chains- undergoes a change, all cross-linked parameters are informed about the change and trigger the flow of information. This concept is used to reduce communication efforts among individuals of a sociotechnical system. Parameter Management thus significantly contributes to the transparency and consistency of information as well as to the efficiency of communication.

REFERENCES

Aizedbush-Reshef, N.; Nolan, B.T.; Rubin, J. (2006), Shaham-Gafni, Y, "Model Traceability", *IBM Systems Journal* 45.

Baldwin, C.Y.; Clark, K. B. (2000), Design Rules: *The Power of Modularity*, MIT Press, Vol. 1, Cambridge, MA.

Bertalanffy, L.von (1972), Zu einer allgemeinen Systemlehre (1949), in Bleicher, K.: *Organisation als System*, Betriebswirtschaflicher Verlag Gabler, Wiesbaden.

Bitzer, M.; Eigner, M. (2007), Langlotz, M., "Management Decision Support by PLM Solutions", *Proceedings of the 16th International Conference on Engineering Design*, Paris.

Danilovic, M.; Sandkull, B. (2002), "Managing Complexity and Uncertainty in a Multiproject Environment", Pro*ceedings of 5th International Conference of the International Research Network on Organizing by Projects*, Rotterdam.

Dickerson, E. Charles; Mavris, D. (2013), "A brief history of Models and Model Based Systems Engineering", *IEEE Systems Journal*, Vol. 7, No.4, December 2013.

Dörner D. (2000), Die Logik des Misslingens, Rowohlt Verlag, Reinbeck.

Ehrlenspiel K.; Meerkamm H. (2013), *Integrierte Produktentwicklung*, 5th Edition, Carl Hanser Verlag.

Eigner, M.; Stelzer R. (2009), *Product Lifecycle Management*, Springer-Verlag, Berlin Heidelberg.

Eigner, M.; Roubanov, D.; Zafirov R. (2014), *Modellbasierte virtuelle Produktentwicklung*, Springer Verlag, Berlin, Heidelberg.

Faisst, K. G. (2007), Dankwort, C. W., "New Extended Concept for the Usage of Engineering Objects and Product Properties in the Virtual Product Generation Process", Pro*ceedings of the 16th International Conference on Engineering Design*, Paris.

Frankenberger, E. (1997), *Arbeitsteilige Produktentwicklung*, Fortschrittsberichte VDI, VDI Verlag, Düsseldorf.

Gausemeier, J., Ebbesmeyer, P., Kallmeyer, F. (2001): *Produktinnovation – Strategische Planung und Entwicklung der Produkte von morgen*; Carl Hanser Verlag, München.
Gomez, P.; Probst, G. (1999), *Die Praxis des ganzheitlichen Problemlösens*. 3rd Edition, Bern: Paul Haup.
Groll, M. (2008), *Verbindungsdokumentation – Die Verbindungsbasierte Produkt- und Prozessdokumentation*, Dissertation.
Hacker, W. (2002), *Denken in der Produktentwicklung*, Rainer Hampp Verlag.
Hubka, V. (1973), *Theorie der Maschinensysteme*, Springer-Verlag, Berlin Heidelberg.
IEEE (1990), *IEEE Standard Glossary of Software Engineering Terminology*, Institute of Electrical and Electronics Engineers, New York, USA.
INCOSE (2010), *International Council on Systems Engineering*, INCOSE Systems Engineering, Handbook, v.3.2, Seattle, WA, USA.
Koehler, N.; Naumann T. (2014), "Supporting the Modeling of Traceability Information", *Proceedings of the 13th International Design Conference*, Dubrovnik.
Königs, S.F.; Beier, G.; Figge, A. (2012), Stark, R., "Traceability in Systems Engineering", *Advanced Engineering Informatics* 26.
Lindemann, U. (2009), Maurer, M.; Braun, T., *Structural Complexity Mangement*, Springer.
Malik, F. (2003), *Strategie des Managements komplexer Systeme*, Haupt Verlag, Bern.
Naumann, T. (2005), *Adaptives Systemmanagement, Dissertation*, Universität Magdeburg.
Naumann, T., Tuttass, I., Kallenborn, O., Königs, F. (2011), "Social Systems Engineering", *Proceedings of the 18th International Conference on Engineering Design*, Lyngby/Copenhagen.
Naumann, T.; Koehler, N. (2014), Meta-Model of Sociotechnical Systems: "Derivation, Structure and Content"; *Proceedings of 10th International Symposium in Tools and Methods of Competitive Engineering*, Budapest Hungary.
Ohl, S. (2000), *Prognose und Planung variantenreciher Produkte am Beispiel der Automobilindustrie*, VDI-Frotschrittsberichte, VDI-Verlag, Düsseldorf.
OMG, Object Management Group, *OMG Systems Modeling Language*, Available at http://www.omgsysml.org/ (Visited on 01/02/2017).
Ostermayer, R. (2001), *Pragmatisch-Situative Wissensrepräsentation – ein Baustein für das Wissensmanagement*, Dissertation, Universität Karlsruhe.
Patzak, G. (1982), *Systemtechnik, Planung komplexer innovativer* Systeme, Springer-Verlag, Berlin, Heidelberg, New York.
Pavkovic, N., Bojcetic, N., Franic, L., Marjanovic, D. (2011), "Case Studies to Explore Indexing Issues in Product Design Traceability Framework", *Proceedings of the 18th International Conference on Engineering Design*, Copenhagen.
Puhl, H. (1999), *Komplexitätsmanagement*, Dissertation, Universität Kaiserslautern.
Ropohl, G. (2009), *Allgemeine Technologie, Universitätsverlag Karlsruhe*, 1st and 3rd Edition, 1978.
Rudolph, S. (2006), "Know.How Reuse in the Conceptual Design Phase of Complex Engineering Products", *Proceedings Integrated Design and Manufacture in Mechanical Engineering*, Grenoble, France.
Rudolph, C. (2006), Trans*formation eines technischen Systemmodells auf Basis der Systemtheorie der Technik nach Ropohl*, Dissertation, Universität Duisburg-Essen.
Schuh, G. (2005), *Produktkomplexität managen*, Strategien – Methoden – Tools, Carl Hanser Verlag, 2005.
Stachoviak, H. (1973), *Allgemeine Modelltheorie*, Springer Verlag, Wien; New York.
Storga, M. (2004), "Traceability in Product Development", *Proceedings of the 8th International Design Conference*, Dubrovnik.
Toepfer, F. (2016), Naumann T.; "Management of Vehicle Architecture Parameters", *Proceedings of the 14th International Design Conference*, Dubrovnik.
Toepfer, F. (2017), Naumann T.; "Parameter Management, a Novel Approach in Systems Engineering", *Proceedings of the 6th International Conference on Research into Design*, Guwahati.
Vajna, S. (2009), Weber, C.; Bley, H.; Zeman, K., *CAx für Ingenieure*, Springer, Berlin Heidelberg.
Vester, F. (2011), *Die Kunst vernetzt zu denken, Ideen und Werkzeuge für einen neuen Umgang mit Komplexität*, Dt. Taschenbuch Verlag, München.
Weilkiens, T. (2006), *Systems Engineering with SysML/UML*, Morgan Kaufman OMG Press, Amsterdam.
Wiener, N. (1948), *Cybernetics*, MIT Press, Cambridge, MA.
Willke, H. (1991), *Systemtheorie*, 3rd Edition, Gustav Fischer Verlag, Stuttgart, New York.