INTERNATIONAL DESIGN CONFERENCE - DESIGN 2016
Dubrovnik - Croatia, May 16 - 19, 2016.

AN INTERDISCIPLINARY MODEL-BASED DESIGN APPROACH FOR DEVELOPING CYBERTRONIC SYSTEMS

M. Eigner, T. Dickopf and C. Huwig

Keywords: systems engineering (SE), model-based engineering, cybertronic systems, interdisciplinary collaboration

1. Introduction

Innovative and interdisciplinary engineering of consumer products and their production systems requires a rethinking of current design methodologies, processes, IT solutions, and the entire enterprise organization. Many of today's technical systems are interconnected by computer networks. Such systems, which communicate with each other, collect and distribute information, and are able to autonomously adapt their behaviour based on information available across different systems have been termed cyber-physical systems (CPS) [Lee 2008], [Broy 2010]. Main components of these CPSs are networked electronics with embedded software. However, a wide range of technical systems, is centered on mechatronics where mechanical and electrical engineering are the dominant disciplines. For systems which combine both properties (interconnected computing and mechatronics), the term cybertronic systems (CTS) has been coined [Eigner et al. 2014a]. This progress in mechatronic systems extends the capabilities of technical systems significantly: a cybertronic system is capable of adjusting its behaviour based on data collected from its environment which can also include other actively communicating systems.

Engineering design processes are constantly changing and pose new challenges. Rapidly changing market situations in a global economy and an increasing number of customer requirements have a significant influence on the design process. The growth in product systems complexity results from products of larger varieties serving multiple markets as well as from multidisciplinarity in product design and development, particularly due to the increased use of embedded computing in technical systems.

The above-mentioned challenges require interdisciplinarity of the methodologies. In order to define a design process suited for cybertronic systems, design methodologies of all involved disciplines – mechanical design, electronics and software engineering – need to be analysed with respect to their support of an interdisciplinary design process [Eigner 2013]. These existing methodologies have to be transferred into a common interdisciplinary method, process and IT solution approach. But existing methodologies do not support it.

Chapter two will give a short overview of current discipline-specific and interdisciplinary design methodologies and their disadvantages for an interdisciplinary model-based development of cybertronic systems. Chapter three will describe in detail two approaches from mechanical and software engineering and show their similarities based on the structural architecture of the methodologies. Based on partial results of the German BMBF research project mecPro², section four will present an interdisciplinary model-based approach for developing cybertronic. At least section five will conclude the interrogations and results of this paper.

2. State of the art

Based on functional modeling but without a formal modeling language, one of the first design methodologies of mechanical engineering was pro-posed in Europe in the late 1970s [Pahl and Beitz 1977]. Today, almost all modern design methodologies established in mechanical engineering based on four essential steps – planning and clarifying the task, conceptual design, embodiment design and detail design. In the concept phase these approaches define system functions and their realizations with (physical) solution principles [Andreasen 1980], [French 1999], [Malmqvist and Svensson 1999], [Ehrlenspiel and Meerkamm 2013], [Feldhusen and Grote 2013].

In Consequence of a wide range of different domains and levels of integration in electrical and electronical (hardware) design, a great diversity and variety among the design methodologies exist.

According to [Kümmel 1999] and [Stephan 2013] different design methodologies should be used in different application areas. A significant classification characteristic for the many design methodologies in hardware design is the level of independence of a certain technology. The "top-down", "bottom-up", and based on these the "yo-yo" design methodologies [Rauscher 1996], [Lüdecke 2003] are such technology-independent models. The Y-Chart approach developed by Gajski and Kuhn [Gajski et al. 2009] is one of the most common and widely used models in hardware design. The design process is classified into the three domains of behavioural, structural, and geometrical design and visualizes the different levels of abstraction in each domain as layers in the chart. The design methodology does not suggest a specific sequence of tasks (i.e. a specific process).

In software engineering, there is a large variety of design methodology models as well. In addition to the structure, the focus here lies on behavioural design [Schäppi et. al. 2005] rather than on functional design as in mechanical design methodologies. With the goal to make software development and support more productive and effective, several phase-oriented models supporting the software design process [Pomberger and Pree 2004], e.g., the waterfall, prototype or spiral model, have been developed. The V model adopted in design methodologies of all kinds of disciplines originated comes from software design [Boehm 1979]. With the UML (Unified Modeling Language) software engineering has a well-known modeling language that supports object-oriented and model-driven software and can be used in all stages of software design as well as for modeling design processes. Design processes which use this model-based potentiality are the Unified Software Development Process (USDP) [Rumbaugh et al. 2005] and the Rational Unified Process (RUP) [Kruchten and Versteegen 1999]. Design methodologies which focus on fast implementation and flexibility during the design process and do not based on specific design procedures are called Agile Software Methodologies [Beck and Andres 2005].

Design methodologies that integrate several disciplines typically address either mechatronics or Systems Engineering. Initially, mechatronics addressed the functional extension of mechanical systems with electrical or electronic as well as software components [Schäppi et al. 2005] Mechatronic design methodologies are based on a general understanding of the design process [Isermann 2008], on the mechanical design process by Pahl and Beitz [Gausemeier and Lückel 2000], or on variations of the V model [Bender 2005], [Anderl et. al. 2015]. The most prominent of all is the VDI 2206 design guideline [VDI 2206 2004] which describes a flexible design process based on three elements: "problem-solving cycle as a micro-cycle" [Daenzer and Haberfellner 2002], "V model as a macro-cycle" and "process modules for recurring working steps".

Systems Engineering, highly influenced by software, addresses the question of complexity and multidisciplinarity within engineering design using an integrative and interdisciplinary view of a product over its entire lifecycle from a system perspective. There are several standards defining the Systems Engineering process [IEEE 1220 1998], [ANSI/EIA 632 2003], [ISO/IEC 15288 2008]. In Model-based Systems Engineering (MBSE), an extension of the classical Systems Engineering approach, the collaboration between all involved disciplines and throughout all design phases should no longer be based on the use of documents but with centrally available digital models. [Estefan 2007] gives a detailed overview about existing Systems Engineering design methodologies.

In summary, we can identify a lack of methodology with regard to the design of cybertronic systems [Gausemeier et al. 2013]. Nowadays, discipline-specific methodologies are used to solve discipline-specific design problems. Current interdisciplinary design methodologies such as mechatronics and Systems Engineering do not evenly penetrate the fields of discipline-specific design but are rooted in

one discipline and expanded in an attempt to successfully cover the others as well. Further-more, there is currently no model-based formalism available that integrates discipline-specific methodologies in the context of interdisciplinary design problems along the design process of cybertronic systems.

3. Comparison

The similarities and varieties of an approach for mechanical engineering (VDI 2221) and an approach for embedded systems (SPES-Software Platform for Embedded Systems) should be illustrated in this section after a detailed introduction of these methodologies. These two approaches should represent the recent progress from mechanical products up to embedded systems with regard to equal artefacts in product development methodologies over the last decades. The required enhancement of existing discipline-specific methodologies by a model-based interdisciplinary approach should be also discussed after that by section 3.3.

3.1 VDI 2221

The Association of German Engineers (VDI) worked out a methodology for product development, called the [VDI 2221 1994]. This methodology is supporting developers across all industries, especially in mechanical engineering, in the development and construction of technical systems and products [Pahl and Beitz 1977]. This approach is divided into seven fundamental steps from understanding the development task to the termination of the construction. Each step delivers a certain work result at the end. The seven workflows are divided into task clarification, conceptual design, engineering design and development regarding the construction [VDI 2221 1994]:

- Task clarification: The task clarification contains phase 1. The requirements from the customer and the tasks from the product planning should be clarified and specified under condition of available information, information gaps and the addition of external and internal requirements. The result is a structured requirements specification document which accompanied the following workflows as an information resource. This resource should regularly be updated.
- Conceptual design: The conceptual section uses the requirements specification from phase 1 and includes workflow 2 and 3. The generation and structuring of system functions occurs in phase 2. The overall function is determined first. This function is classified into combined and grouped sub-functions. The resulting functional structure constitutes the basis of the following workflow. In phase 3, principle solutions to fulfil the functions are searched. The principle solutions consider selected effects (e.g. physical or chemical) and resultant structural definitions (geometry, movement or material used). The result of the conceptual phase builds up a working structure in which each principle solution is linked with the functional structure of workflow 2.
- Engineering design: Based on results from the conceptual design, an overall design of the system is developed. This design includes all commitments for the realisation of the product. The clarification of the principle solution in realizable modules occurs in workflow 4. The modular structure generates real groups and real elements including their interfaces. The realization of modules with scaled drawings from workflow 5 are combined with further detailed information to an overall design in workflow 6.
- Development instructions: The elaboration of all the necessary documents occurs in the last phase of the approach. A complete product documentation constitutes the final result of this methodology.

3.2 SPES 2020 Modeling Framework

The SPES Modelling Framework builds up on two concepts to create a two-dimensional development space. On the one hand, the system is described over the development process on different abstraction layers. With this method, the developer concentrates either on a specific problem in general or analyses the problem into detail. On the other hand, the second concept is generating another viewpoint for the developer on a specific aspect of the system into detail. In the context of this methodology, the system

development follows four viewpoints from solution-neutral requirements view into the specific technical solution [Pohl et al. 2012]:

- Requirements Viewpoint: This viewpoint inside the early development stage supports the generation of system level requirements by the choice, documentation, validation and management of these requirements. Therefore, the viewpoint contains a requirement artefact model. The main view of this model collects all the necessary requirements from the system context and delivers an instrument to structure these requirements over different abstraction layers.
- Functional Viewpoint: This view integrates a set of user functions into an extensive model of system functions and describes inside this model the system behaviour at each system boundary. For that purpose, user functions are realised in a black-box-model. The functionality of this user functions from the black-box-model is realised in an associated white-box-model. Requirements are integrated in a widespread system specification inside models from the functional viewpoint. The behaviour between different user functions is also specified inside this system specification. The functional hierarchy is summarised in reasonable functional groups to illustrate user functions in a certain abstraction layer.
- Logical Viewpoint: The logical Viewpoint describes a structured decomposition of the system to an architecture of logical components to realise the favoured functionality. A logical description of the solution with no technological boundaries is generated by the only examination of the black-box-models. Connections between logical components regarding data flows or data exchange are realised in the logical viewpoint via logical channels. The logical architecture of the system can represent the system from the logical component down to a very fine-grained layer. Subsystems are described on this layer as state machines for example.
- Technical Viewpoint: The technical viewpoint combines the system software with the hardware. A description of the physical architecture originates under the condition of time, usage of resources and redundancy. This physical architecture specifies the execution of software tasks from the logical viewpoints and also shows the possibilities where and how logical subsystems can be realised. The technical architecture includes beside hardware for data processing also sensors and actors to communicate with the system environment. The technical architecture realises the properties from previous development stages by physical components out of the logical viewpoints.

3.3 Discussion

The VDI 2221, evolved in 1993, and the SPES Modeling Framework from the year 2012 are having quite similar tasks by a deeper analysis. Both exemplarily presented methodologies are using the RFLP approach to develop products from different industrial domains. In both methodologies, requirements are refined and maintained in each following step, main functions are decomposed into sub-functions, principle or logical solutions are generated from these functions to propose principle solutions or physical components in a final task. The new methodology, suggested by chapter 4, proposes an interdisciplinary model-based approach by combining the strengths from the VDI 2221 and the SPES Modeling framework. By generating principle solutions, this new approach is oriented to the VDI 2221. The use of model-based techniques from the SPES Modeling Framework helps to establish the traceability from requirements to logical components. Figure 2 visualises the similarities between the 3 methodologies. The goal of all involved parties to create this interdisciplinary model-based approach, shown on an abstract level by Figure 2, is presented by section 4.

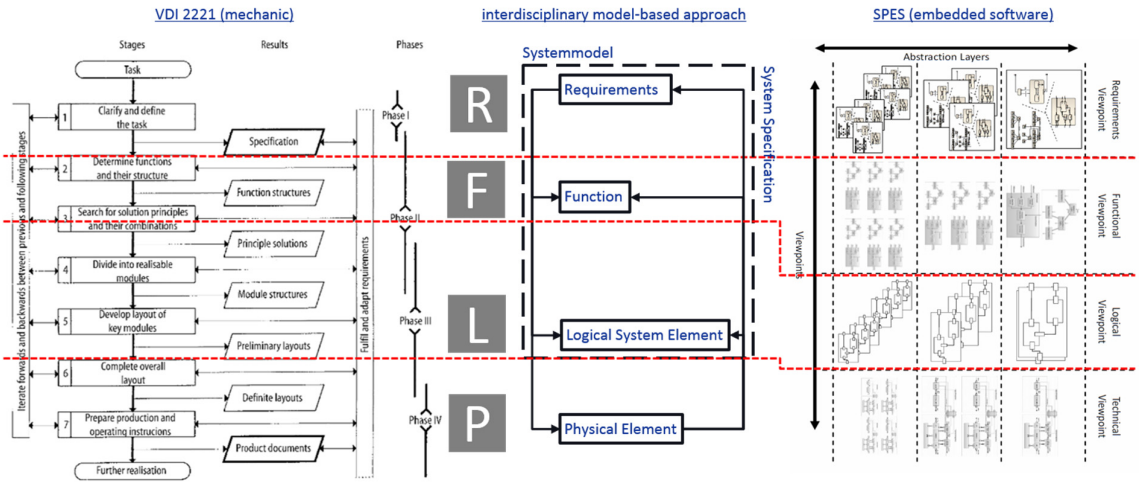

Figure 1. Comparison of VDI 2221 and the SPES method with RFLP

4. Interdisciplinary model-based design approach for cybertronic systems

The following interdisciplinary and model-based design approach for developing cybertronic systems is a result of the German BMBF research project mecPro² (model-based engineering of cybertronic products and production systems). The goal of the project is to increase efficiency in cybertronic systems development. Hereby, the focus lies on the integration of information and data from all relevant disciplines for developing a cybertronic system, particularly with regard to an integrated development process of products and production systems for a cooperative development of products, manufacturing equipment and associated production systems. That should be achieved through the support of concepts based on Model-based Systems Engineering and product lifecycle management.

4.1 The mecPro² model framework

The mecPro² Model Framework as part of a holistic model-based systematic specification concept describes a design approach for the development process of cybertronic systems in the early phase [Eigner et al. 2015]. This model-based approach is directly supported by the modeling language SysML. The design of the system takes place on several levels with increasing solution concretization. The model framework is based on various development methodologies in the fields of mechatronic, mechanic, electric/electronic, software and Systems Engineering analysed in the context of previous work packages of the mecPro² research project [Cadet et al. 2015].This approach use basic ideas from the two methodologies described and analysed above in chapter two as well as conceptual ideas from the extended V-Model by [Eigner et al. 2012, 2014b] (based on the VDI 2226 for developing mechatronic systems) and the so-called "Münchener Produktkonkretisierungsmodell (MKM)" by [Ponn and Lindemann 2011] (based on the VDI 2221 for developing technical systems and product). In detail the mecPro² model framework borrows the following aspects of the mentioned approaches:

- The RFLP approach from the extended V-model by Eigner et al. [2012, 2014b]
- The Viewpoints of the SPES Modeling Framework [Pohl et al. 2012]
- The consideration of principle solutions [VDI 2221 1993], [Ponn and Lindemann 2011]
- The Subdivision in requirement and solution space including the three axes detail, variation and concretisation of the MKM [Ponn and Lindemann 2011]

Figure 2 shows the general structure of the model framework. The axis "detail" comprises the accumulation of information without the restriction of solution space. For example, a system may be described in greater detail through use cases and use case activities without making any determinations of the internal structure of the system. At some point further detailing is no longer possible without a solution concretization. Then takes place the transition to a deeper level. This transition also occurs variation, because the solution concretization does not occur without the consideration of alternatives.

Each level describes a system from a structural and behavioural point of view by the use of the modeling language SysML. The following sections describe the levels of the model framework in detail.

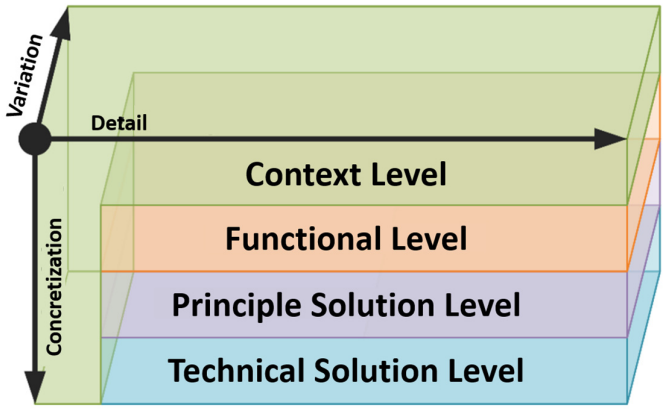

Figure 2. General structure of the model framework

4.1.1 Context level

System requirements arise from the context in which the system is being analysed. In the life cycle of the system varying contexts may occur. For example, a vehicle in the context of its assembly or a vehicle in the context of autonomous parking. Therefore, within the framework, each context is used to bundle certain requirements. In most of the cases, stakeholders will formulate their requirements in a natural language and pass it on to the system development. On the context level the transition and synchronisation between natural language system requirements and the model-based system description occurs. Whereas the SysML has a higher degree of formalization as natural language requirements, information gaps will emerge during translation. These gaps need to be supplemented by commentary or by assumed requirements. At the end of the translation process, the system requirements are available in two languages. As a natural language requirements and as model-based requirements in SysML. Then the information content of both descriptions should be identical. This approach has two big advantages. A maximum formalization of requirements which has great advantages for the target consistency of the development process and stakeholders can be answered in the language in which they have formulated their requirements - that may help avoiding misunderstanding. Figure 3 explains the translation process in a graphical way.

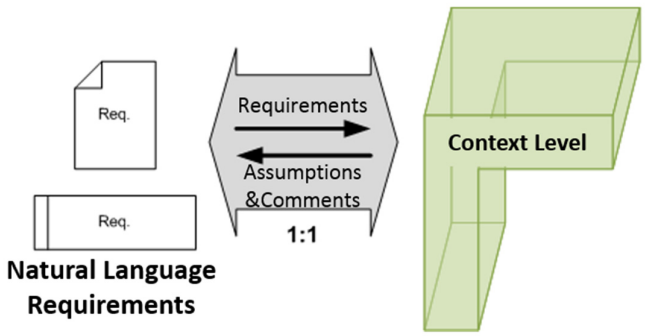

Figure 3. Translation process from natural requirements to a requirements model

The system is considered on the context level as a black box with interfaces to elements in its context environment. Here, the perceptible external behaviour of the system is described in detail by use cases, derived use case activities and state machines as well as by a sequence-based communication with elements of the environment.

4.1.2 Functional level

The functional level is the first concretization phase and it is strongly based on the model-based SysML approach of the FAS method [Lamm and Weilkiens 2010]. The aim of this level is a solution-neutral description of the system functionality based on the contexts in which it is used over its life cycle. Starting from the detailed descriptions of the expected behaviour at the context level, non-redundant functions are identified and placed in a hierarchical relationship by various heuristics. The level's final result is a functional structure that arose from the connectivity of system functions via material, energy and signal flows.

4.1.3 Principle solution level

On the principle solution level the technical aspects, which realize the desired function, are considered. Systems Engineering means, among other things, not to be satisfied with the first solution, but to gain a comprehensive overview of possible solutions [Haberfellner 2012]. This will be supported in the model framework by the principle solution level. Here solution variants should be systematically identified, analysed and evaluated to make an optimal selection with respect to the requirements. Selecting a principle solution can be based on various criteria, e.g. time behaviour, effectiveness, performance or cost. The evaluation and selection should be made in two stages: The degree of fulfilment of a function considered by a solution principle and the degree of fulfilment of possible principle solution structures, which are based on the functional structure of the functional level. Regarding to the analysation and evaluation of variability, this level describes a branch point for an early simulation based of defined test cases.

4.1.4 Technical solution level

On the technical solutions level the maximum concretization of a solution, for which an organizational unit is responsible for, is reached. The elements of the solution level will not be decomposed any further if a different organizational unit should concretize this element. The responsibility for these elements can be assigned clearly and without any overlapping to another organizational unit. A special case occurs when the elements can be classified discipline specific. In this case on the organizational interface the transfer of responsibility from system development to discipline-specific development is located as well. In this case the concretization of solutions and partitioning is on the same level.

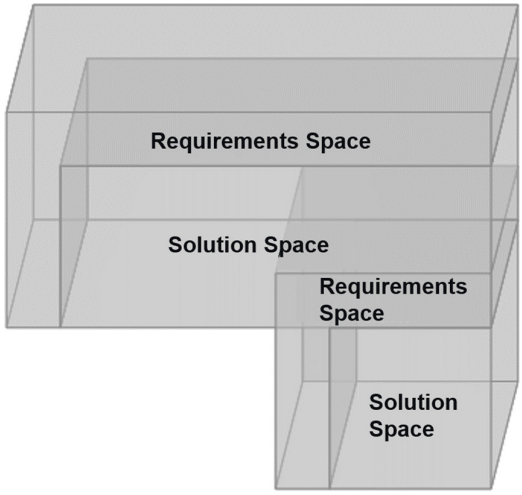

Figure 4. Relation between requirements space and solution space

At this level collaboration takes place necessarily. As seen in Figure 4, the solution space of the system under consideration and the requirement space of the elements overlap. The information of the technical solution level represents the sum of information of context levels of all the solution elements.
At this point can also be seen that, a continuity in model-based development process along the system breakdown to a discipline-specific partitioning can only be realized when the model is used as a

requirements specification. In case of a translation of the information of these level back into natural language requirements for the development process of the system elements, it would inevitably lead to a media break and the continuity of the model-based development process would not be achievable.
Generally, the technical solution level describes an abstract solution concept by the definition of logical system components, which realize the system functions and behaviour by applying the principle solution. The result of this phase is a logical system structure in which the individual system components are related to each other by their interfaces.

4.2 Distinction to the introduced methodologies

Regarding to the introduced methodologies in chapter 3, the context level of the mecPro² model framework combines the task clarification of VDI 2221 and the requirements viewpoint of the SPES modeling framework. Starting with natural-language requirements like in classical mechanical approaches, the mecPro² model framework use the concept of the SPES model framework and transform the requirements into a requirements model which contains the same information in a more formalized way. In classical mechanical methods this transformation process is not established.
For the required multidisciplinary system design, it is essential to generate functional solutions which can represented by all disciplines. In general the determination of solution-neutral functions is a fundamental step in the classical mechanical development process. In other disciplines and their methodologies, accept the SPES modeling framework, the term function is used rarely. But the concept of finding solution-neutral opportunities to implement the desired behavior is a very common way in Systems Engineering or mechatronic development approaches.
Regarding to the RFLP-Approach, in which each fumnnction is assigned to a solution element, the principle solution level describes an intermediate level of concretisation and detailing on the solution finding process. Identifying and evaluating principle solution or active principles is a normal step in the development process of mechanical products. In classical software engineering methodologies as well as in the SPES modeling framework this procedure is unusual.
The technical solution level describes a system solution based on logical system parts, which realize the system functions and behaviour by applying the principle solution, without limiting the solution by specific physical properties. To create a logical system structure is a usual step you can find in mechanical as well as in other disciplines.
Both of the introduced methodologies are using a physical level. While mechanical approaches use this layer for detailing there system elements with physical properties, the SPES modeling framework specifies the execution of software tasks from the logical viewpoints and also shows the possibilities where and how logical subsystems can be realised by combining the system software with the hardware. The mecPro² modeling framework doesn`t focus on this layer but enables an interface with the technical solution level.
Table 1 gives an overview, how far the mecPro² modeling framework as well as the methodologies it is based on, implement the interdisciplinary RFLP approach plus the extension of the consideration of principle solutions and the transformation of natural language requirements into a requirements model.

Table 1. Comparison of the introduced methodologies

		VDI 2221	Ponn Lindemann	V-Model Eigner	SPES	mecPro²
Req	natural language	●	●	●	◒	●
	model-based	○	○	○	●	●
Function		●	●	●	●	●
Principle Solution El.		●	●	○	○	●
Logical Element		●	◒	●	●	●
Physical Element		●	●	●	◒	◒

5. Conclusion and outlook

Based on aspects and concepts of existing design methodologies the mecPro² model framework tries to satisfy the requirements on a cybertronic design process by creating a system model along the four layers: context level, functional level, principle solution level and technical solution level. To allow collaboration and consistency from the specification of a system to the specification of their components, the mecPro² model framework use a model-based approach as well as the concept of requirement and solution space. Compared to the existing methodologies the essential improvement is that the requirements will be highly formalized based on the conversion from the natural language to a model-based approach. This allows a smooth transition from the requirements to the solution space.
The mecPro² model framework was developed together with OEMs and supplier of the German automotive industry and will be validated and verified by two application scenarios from the industry. The first scenario describes the development of the cybertronic system "intelligent autonomous parking" which is based on the interconnection between a car, a smartphone and the car park. The second scenario describes the development and production process of a cylinder head focussing on the model-based information exchange. Two demonstrators of the project involved system houses will show how the emerging information, by using the methodology (of the development and production process), will then be managed over the life cycle of the system. Planned follow-up projects should try, how the developed methodology could be used and integrated into the development process of these German OEMs and supplier.

Acknowledgement

In this paper background, objectives and results of the research project “model-based engineering of cybertronic Products and Production systems (mecPro²)” have been incorporated. This research and development project is funded by the German Federal Ministry of Education and Research (BMBF). The authors are responsible for the contents of this publication. The authors1 would also like to thank Tim Schulte and Marc Schneider for their contribution and research on the topic of Model-based Systems Engineering, and for their input, assistance and cooperation by creating these result. Further acknowledgments and information about the research project mecPro² you can find under: http://www.mecpro.de/static/kt1_de

References

Anderl, R., Nattermann, R., Rollmann, T., "Das W-Modell - Systems Engineering in der Entwicklung aktiver Systeme", Available at <http:www.plmportal.org/forschung-details/das-w-modell-systems-engineering-in-der-entwicklung-aktiver-systeme.html>, 2015, [Accessed 04.12.2015].

Andreasen, M. M., "Machine Design Methods Based on a Systematic Approach", PhD Thesis, Sweden, 1980.

ANSI/EIA 632, "ANSI/EIA 632: Processes for Engineering a System", 2003.

Beck, K., Andres, C., "Extreme programming explained", Addison-Wesley Amsterdam, 2005.

Bender, K., "Embedded Systems - qualitätsorientierte Entwicklung", Springer Heidelberg, 2005.

Boehm, B. W., "Guidelines for Verifying and Validating Software Requirements and Design Specifications", Euro IFIP 79. Proceedings of the European Conference on Applied Information Technology of the International Federation for Information Processing, Samet, P. A. (Ed.), London, 25-28 September, North-Holland Pub. Co. Amsterdam, 1979, pp. 711-719.

Broy, M. (Ed.), "Cyber-Physical Systems", Springer-Verlag Berlin, 2010.

Cadet, M., Meissner, H., Hornberg, O., Schulte, T., Stephan, N., Schindler, C., Aurich, J. C., "Modellbasierter Entwicklungsprozess cybertronischer Produkte und Produktionssysteme – Grundlagen, erste Ansätze und weiteres Vorgehen", Stuttgarter Symposium für Produktentwicklung, 2015.

Daenzer, W. F., Haberfellner, R., "Systems engineering", Verlag Industrielle Organisation Zürich, 2002.

Ehrlenspiel, K., Meerkamm, H., "Integrierte Produktentwicklung", Carl Hanser München, 2013.

Eigner, M., "Modellbasierte Virtuelle Produktentwicklung auf einer Plattform für System Lifecycle Management", Industrie 4.0, Sendler, U. (Ed.), Springer-Verlag Berlin, 2013.

Eigner, M., Dickopf, T., Schulte, T., Schneider, M., "mecPro² - Entwurf einer Beschreibungssystematik zur Entwicklung cybertronischer Systeme mit SysML", Tag des Systems Engineering 2015, Schulze, S.-O., Muggeo, C. (Eds.), Hanser Munich, Germany, 2015.

Eigner, M., Gilz, T., Zafirov, R. "Proposal for Functional Product Description as Part of PLM Solution in Interdisciplinary Product Development", Proc. of the Design Society International Design Conference 2012, Dubrovnik 2012.

Eigner, M., Muggeo, C., Dickopf, T., Faißt, K. G., "An Approach for a Model Based Development Process of Cybertronic Systems", Proc. of the 58th Ilmenau Scientific Colloquium,Technische Universität Ilmenau, 2014a.

Eigner, M., Roubanov, D., Zafirov, R. (Eds.), "Modellbasierte Virtuelle Produktentwicklung", Springer Vieweg Berlin Heidelberg, 2014b.

Estefan, J. A., "Survey of Model-Based Systems Engineering (MBSE) Methodologies", Incose MBSE Focus Group, 2007.

Feldhusen, J., Grote, K.-H., "Pahl/Beitz Konstruktionslehre", Springer Vieweg Berlin, 2013.

French, M. J., "Conceptual design for engineers", Springer London, 1999.

Gajski, D. D., Abdi, S., Gerstlauer, A., Schirner, G., "Embedded system design", Springer Dordrecht, 2009.

Gausemeier, J., Dumitrescu, R., Steffen, D., "Systems Engineering in der industriellen Praxis", Paderborn, 2013.

Gausemeier, J., Lückel, J., "Entwicklungsumgebungen Mechatronik", Heinz Nixdorf Institut, Paderborn, 2000.

IEEE 1220, "IEEE 1220: Standard for Application and Management of the Systems Engineering Process", 1998.

Isermann, R., "Mechatronische Systeme", Springer-Verlag Berlin, 2008.

ISO/IEC 15288, " ISO/IEC 15288: Systems Engineering - System Life Cycle Processes", 2008.

Kruchten, P., Versteegen, C., "The Rational Unified Process", Addison-Wesley München, 1999.

Kümmel, M. A., "Integration von Methoden und Werkzeugen zur Entwicklung von mechatronischen Systemen", Heinz Nixdorf Institut, Paderborn, 1999.

Lamm, J., Weilkiens, T., "Functional Architectures in SysML", Tag des Systems Engineering 2010, Maurer, M., Schulze, S.-O.(Eds.), Hanser Munich,Germany, 2010.

Lee, E. A., "Cyber Physical Systems: Design Challenges", Technical Report No. UCB/EECS-2008-8, University of California, Berkeley, 2008.

Lüdecke, A., "Simulationsgestützte Verfahren für den Top-Down-Entwurf heterogener Systeme", PhD Thesis, Duisburg/Essen, 2003.

Malmqvist, J., Svensson, D., "A Design Theory Based Approach Towards Including QFD Data in Product Models", Proceedings of the 1999 ASME Design Engineering Technical Conferences, American Society of Mechanical Engineers, New York, 1999.

Pahl, G., Beitz, W., "Konstruktionslehre", Springer-Verlag Berlin, 1977.

Pohl, K., Honninger, H., Achatz, R., Broy, M. (Eds.), "Model-based Engineering of Embedded Systems", Springer-Verlag Berlin, 2012.

Pomberger, G., Pree, W., "Software-Engineering", Carl Hanser München, 2004.

Ponn, J., Lindemann, U., "Konzeptentwicklung und Gestaltung technischer Produkte - Systematisch von Anforderungen zu Konzepten und Gestaltlösungen", Springer-Verlag Berlin Heidelberg, 2011.

Rauscher, R., "Entwurfsmethodik hochintegrierter anwendungsspezifischer digitaler Systeme", Pro-Universitate-Verlag Sinzheim, 1996.

Rumbaugh, J., Jacobson, I., Booch, G., "The unified modeling language reference manual", Addison-Wesley Boston, 2005.

Schäppi, B. Andreasen, M. M., Kirchgeorg, M., Radermacher, F. J. (Eds.), "Handbuch Produktentwicklung", Carl Hanser München, 2005.

Stephan, N., "Vorgehensmodell zur Unterstützung der interdisziplinären und föderierten Zusammenarbeit in der frühen Phase der Produktentstehung – Am Beispiel der Nutzfahrzeugindustrie", PhD Thesis, Kaiserslautern, 2013.

VDI 2206, "VDI 2206 - Design methodology for mechatronic systems", Verein Deutscher Ingenieure, Düsseldorf, 2004.

VDI 2221, "VDI 2221 - Methodik zum Entwickeln und Konstruieren technischer Systeme und Produkte", Beuth Verlag GmbH Berlin, 1994

Thomas Dickopf, Dipl.-Ing.
TU Kaiserslautern, Lehrstuhl für Virtuelle Produkt Entwicklung
Gottlieb-Daimler-Str. 44, 67633 Kaiserslautern, Germany
Email: thomas.dickopf@mv.uni-kl.de