INTERNATIONAL DESIGN CONFERENCE - DESIGN 2018
https://doi.org/10.21278/idc.2018.0138

THE KSCM AS PART OF A HOLISTIC METHODOLOGY FOR THE DEVELOPMENT OF CYBERTRONIC SYSTEMS IN THE CONTEXT OF ENGINEERING 4.0

T. Dickopf, M. Eigner and H. Apostolov

Abstract

With today's increased level of product complexity concerning the development of so-called cybertronic or cyber-physical systems, a rethinking of current design methodologies, processes, and IT solutions is required. After a short introduction of relevant work in the field of cybertronic systems design, this paper proposes a novel design methodology based on the MVPE Model and the newly introduced Kaiserslautern System Concretization Model for the model-based development in the context of Engineering 4.0.

Keywords: systems engineering (SE), cyber physical systems, engineering design, design methodology, model based engineering

1. Introduction

Statistics from recent years confirm a permanent change in the product development processes. These changes result from changed market conditions together with new requirements on the product imposed by customer needs. The increase in product complexity results, on the one hand, from a much stronger application of multi-market product design, derivative and variant diversity of products and, on the other hand, from the constant increase of electronic components and embedded software. The attribution of electronics and software to product's value has increased steadily in recent years and reached, for example in automotive, about 40 percent. When mechatronic components, products or systems communicate with each other, this is referred to as cyber-physical systems (Lee, 2008; Broy, 2010) or cybertronic systems (Eigner et al., 2017c). These new types of systems require a mindshift in the development methods, processes and IT infrastructure for the interdisciplinary product development process of the future. This mindshift is based on methodological, organizational and IT-infrastructural support along all engineering activities throughout the entire product lifecycle – from the early stages of requirements elicitation through to the end of life of the product – across all domains (mechanics, electrics/electronics, software, and services) and beyond the boundaries of companies. There is a lack of well-established, industrially-used methods, processes and procedural models for the interdisciplinary development of products and systems. The development methods and processes in the domains are very different, which have made engineers to be educated, think and work in discipline-oriented silos.
The present contribution is structured as follows: The following section describes briefly the research method used to generate the results of this paper. Section three gives a short overview over current discipline-specific and interdisciplinary design approaches and standards from the fields of mechanic, electronic, software, mechatronic and systems engineering. Section four introduces a selection of initial design approaches from the field of cybertronic system development. Section five introduces the newly developed, so-called Kaiserslautern System Concretization Model (KSCM), which adapts and merges

concepts from selected, existing design approaches and represents the methodical part of a new holistic methodology for the development today's systems in the context of Engineering 4.0 (Künzel et al., 2016). At last, section 6 concludes the results of this paper and gives an outlook to the ongoing evaluation of the results.

2. Research method

The development of the KSCM follows the CASE research cycle (clarification – analysis – synthesis – evaluation) after Müller (2013) (see Figure 1) and is still ongoing.

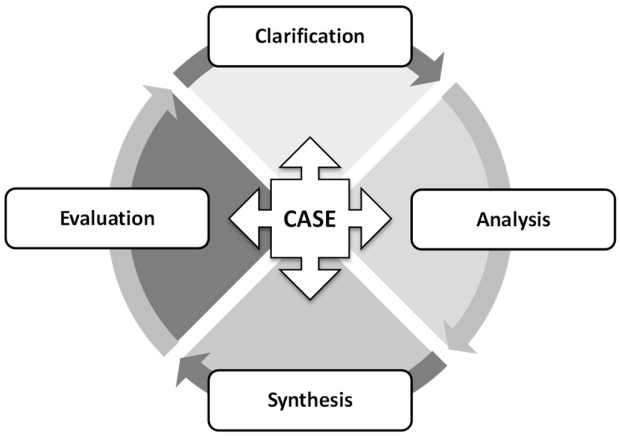

Figure 1. The CASE research cycle (Müller, 2013)

The following parts of this contribution provide insight into the results of the analysis phase (see Section 3 and Section 4) and the current state of the approach's synthesis in Section 5. The evaluation of the approach through a case study appertains to our future intends and will briefly addressed in the outlook section.

3. State of the art

Based on functional modelling, one of the first design methodologies in mechanical engineering was proposed in Europe in the late 1970s (Pahl and Beitz, 1977). Today, almost all well-established, modern design methodologies in mechanical engineering are based on four essential steps – planning and clarifying the task, conceptual design, embodiment design, and detail design. As seen as in Andreasen (1980), French (1999), Malmqvist and Svensson (1999), Ehrlenspiel and Meerkamm (2013), and Feldhusen and Grote (2013), the essential step of the conceptual design is the definition of functions and their realization by principle solutions. This concept is also adopted by the VDI guideline 2221 (1993). A different approach with a strong focus on the system requirements is given by Ponn and Lindemann (2011). The requirement space of the so-called Munich Model of Product Concretization is arranged in parallel to the different levels of the solution space, which implies that the requirements affect the entire development process.

In consequence of a wide range of different domains and levels of integration in electrical and electronic hardware design, a wide variety of diverse design methodologies exists. A significant classification characteristic for many design approaches in hardware design is the level of independence of a particular technology. The 'top-down', 'bottom-up', and the based on the previous two 'yo-yo' design approach are only a small number of examples for such technology-independent models (Rauscher, 1996; Lüdecke, 2003). One of the most common and widely used models in hardware design is the so-called Y-Chart developed by Gajski and Kuhn (Gajski et al., 2009). The hardware design according to the Y-Chart is classified into the three domains of behavioural, structural, and geometrical design represented by the three axes – the hands of the Y – in the chart. The abstraction levels in each domain are depicted as concentric layers around the origin of the three axes. A specific sequence of tasks (i.e., a specific process) is not given by the approach though.

With its focus on behavioural design (Schäppi et al., 2005) rather than on functional design as in mechanical design approaches, software engineering has as well a large variety of design approaches.

With the goal to make software development more productive and efficient, several phase-oriented models supporting the software design process (Pomberger and Pree, 2004), e.g., the waterfall, prototype model or the spiral model, have been developed. Boehm (1979) introduced a design approach in 1979, whose concept has been adopted in modern approaches of all of engineering domains – the V-Model. Due to the high effort of creating and maintaining sophisticated software, the development regarding the mentioned approaches takes place along a structured, strictly phase-oriented and highly formalized procedure where the development process is divided into manageable, temporally and content-limited phases. Through the development of object-oriented and model-driven concepts in software engineering, the UML (Unified Modeling Language) has been defined. Design processes, which use the potential of these concepts, are the Unified Software Development Process (USDP) (Rumbaugh et al., 2010) and the Rational Unified Process (RUP) (Kruchten, 2003). Design approaches, which focus on fast implementation and flexibility during the design process and are not based on specific design procedures, are called agile software approaches (Beck and Andres, 2005).

The term mechatronics was first used in 1969 by Ko Kikuchi (Harashima et al., 1996). Initially, the term only referred to the electro-technical and electronic functional extension of mechanical components and devices. Software has gained its importance in mechatronics much later (Schäppi et al., 2005). Mechatronic design approaches are based on a general understanding of the mechatronics' design process (Isermann, 2008), on the mechanical design process after Pahl and Beitz (Gausemeier and Lückel, 2000), or on variations of the V-Model (Bender, 2005; Nattermann and Anderl, 2010). The most prominent of all is the VDI 2206 design guideline (VDI 2206, 2004).

Other design approaches, which integrate different disciplines, can be assigned to the systems engineering domain. Systems Engineering (SE) and Model-based Systems Engineering (MBSE) address the issue of complexity and multidisciplinarity within engineering design using an integrative and interdisciplinary view of the product over its entire lifecycle from a system's perspective. Several standards defining the SE process (ANSI/EIA 632, 2003; IEEE 1220, 2005; ISO/IEC 15288, 2015) are available today. A detailed overview of existing design approaches in the field of systems engineering is given by Estefan (2008).

From the brief overview of relevant, well-established interdisciplinary and discipline-specific design approaches given by this section, one can conclude, that with regard to the requirements on new methodologies for the development of cybertronic systems in the context of Engineering 4.0 – i.e. support of interdisciplinary design processes, consideration of cybertronic characteristics of today's products, usable with agile and model-based development processes, etc. – none of the mentioned above can satisfy these sufficiently. A fusion of concepts of appropriate approaches could be an opportunity to close the gap of non-existent methodologies for the development of such complex, interdependent systems. In particular, the interdisciplinary approaches could form the basis of such a development method by the provision of aspects for domain integration. However, this foundation would need to be extended and refined by appropriate concepts from the discipline-specific approaches mentioned in the first pars of this section to meet all of the requirements. The following section will introduce three initial approaches selected as a basis for the methodology for development of complex cybertronic systems proposed in Section 5.

4. Related work in field of cybertronic systems design

Related work dealing with the development and integration of new methods, processes, IT solutions and organizational forms to allow a multi-disciplinary development of mechatronic or cybertronic systems and to ensure traceability throughout the entire system lifecycle is found from three major sources, which also influenced the methodology presented in the fifth section. An essential foundation therefore was given by the so-called MVPE (model-based virtual product engineering) Model (Eigner et al., 2012, 2014a, 2014b), which is an extension of the VDI V-Model for development of mechatronic systems as per guideline VDI 2206 (VDI, 2004). The MVPE adds Model-based Systems Engineering (MBSE) and System Lifecycle Management (SysLM) aspects to the original VDI V-Model. Additionally, focusing at the shift of today's technical products toward cyber-physical or cybertronic systems, two approaches stand out in particular – the mecPro² Model Framework (Eigner et al., 2015, 2016, 2017b) and the System Concretization Model after Pfenning (Pfenning, 2017). They both aim at handling the rising complexity of cybertronic systems and ensuring consistent, interdisciplinary and model-based

development process. The following subsections give a more detailed overview of the three models and compare them pointing out the similarities and differences between them.

4.1. MVPE Model for multidisciplinary product development

The MVPE Model is an extension of the V-Model – the macro-cycle for development of mechatronic systems in accordance to the VDI guideline 2206. The extensions focus on two essential points: the support of the interdisciplinary and discipline-specific design phases of the V-Model by MBSE methods; and on the seamless integration and management of data over the entire system lifecycle by a System Lifecycle Management Backbone. On the left wing of the V-Model, Eigner et al. (2012) identify three levels of modelling: modelling and system specification; modelling and first simulation; and discipline-specific modelling (see Figure 2).

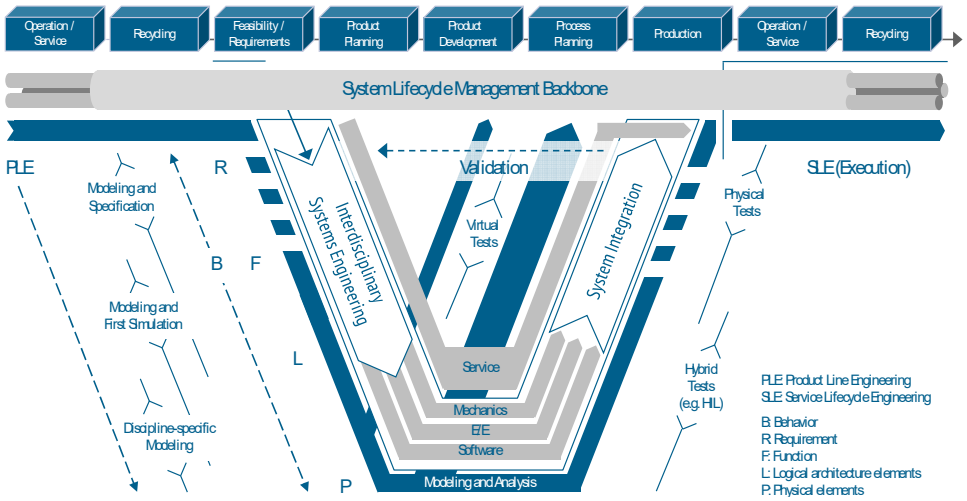

Figure 2. MVPE Model for multidisciplinary product development (Eigner et al., 2017a)

On the specification level, the system is described by models, which include the system requirements as well as the system architecture based on a functional and logical system structure. These models are descriptive and do not support simulation. The second level – modelling and first simulation – focuses on the creation and use of multidisciplinary simulation models (e.g. Matlab or Modelica). On the third level, the domains take over the development to conduct the detailed design of the system. Models of this level include discipline-specific aspects such as geometry and physical structure and are created by specific CAx tools. Parallel to these overlapping modelling levels, the left wing is segmented into the four abstraction levels of system specification: requirements (R), functions (F), logical architecture elements (L) and physical parts (P). Each level can include structural as well as behavioural system information. The MVPE suggest, in addition to the system development, the importance of a parallel development of services in the early phases as well as the non-neglecting of the topics Product Line Engineering (PLE) and Service Lifecycle Engineering during the lifecycle. (Eigner et al., 2017a) The newly developed methodology presented in the fifth section integrates with the MVPE Model providing the methodological guidance for completing the mentioned modelling tasks.

4.2. mecPro² Model Framework

The mecPro² Model Framework was developed within the German research project 'mecPro² – Model-based Development Process of Cybertronic Products and Production Systems' (Eigner et al., 2017c). This architectural framework was developed to facilitate the step-by-step description of the interdependencies between cybertronic systems and their elements in a coherent system model. The approach is directly supported by the Systems Modeling Language (SysML). As part of the so-called mecPro² Architectural Framework (Figure 3, left side), the Model Framework (Figure 3, right side) builds the methodological basis upon which a technical system can be formally specified in the early phase of its development.

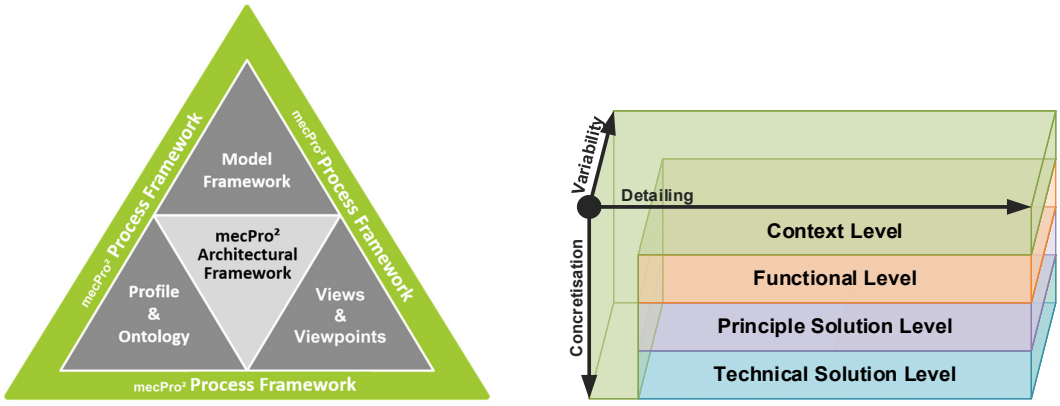

Figure 3. The mecPro² Architectural Framework and a detailed depiction of its Model Framework (Eigner et al., 2015)

The Model Framework is organized in two spaces and four levels, and has three dimensions. The solution concretization increases with the levels. The dimension Detailing represents the increased accumulation of information without the restriction of solution possibilities, until further detailing is no longer possible without the exclusion of possible solutions. When this point is reached, the transition to a lower level takes place. This transition also results in variability, because the solution concretization does not occur without the consideration of design alternatives. The system under development is described on each level from a structural and behavioural point of view (Eigner et al., 2015, 2016, 2017b).

The Model Framework is subdivided into a requirement and a solution space, whereby the Context Level corresponds to the requirement space. The Context Level facilitates the translation of natural language requirements into a model-based requirements specification and the synchronization in case of changes. The translation and synchronization process is performed in two main steps: a context-based differentiation of the system from its surroundings (this includes the environment, stakeholder, and external systems); and a detailed description of the system use cases in regard to its expected behaviour (Eigner et al., 2015, 2016, 2017b). The solution space is subdivided into three levels. The Functional Level is the first concretization step. The aim of this level is a solution-neutral definition of the functional architecture of the system based on the derivation of system functions from the expected behaviour of the Context Level. The purpose of the Principle Solution Level is to find the most appropriate solution elements to fulfil the defined system functions. On this level, principle solutions which can realize a function through their technical capabilities are analysed and evaluated. The different solutions vary in their basic principle idea. Analogically, the concretization of a chosen principal solutions to realize a function takes place at the Technical Solution Level. The aim of this level is a detailed white-box description of the system in terms of its structure and its tested behaviour. The end result of this process is the definition of the complete system architecture (which includes the form and function of the system) based on the requirements of the varying contexts in which the system will be used during its lifecycle as a part of one or more cybertronic systems (Eigner et al., 2015, 2016, 2017b).

4.3. System Concretization Model after Pfenning (2017)

While the mecPro² Model Framework deals with just the methodical, model-based specification of the system architecture and behaviour, the System Concretization Model (SCM) after Pfenning (2017) deals with this aspect on a macro-procedure level and thereby also provides an appropriate basis for the management of the model elements in a product lifecycle management (PLM) solution. It achieves this through the derivation of a coarse graph of relevant elements out of the system model elements and their relations, which are to be managed in the PLM solution. (Figure 4) These elements are called anchor elements and include requirements, functions, logical and physical elements, and test cases among others. Intermediate artefacts, which arise during the system development and support the solution-finding process, are not transferred to the PLM backbone (Pfenning, 2017).

Additionally, to the composition of the mecPro² Model Framework – i.e. subdivision into requirement and solution space; three dimensional character of the solution space along the axes of detailing,

concretization and variability; subdivision of the solution space into functional, logical and physical levels with a structural and behavioural aspect on each level – the SCM includes an additional verification space (Figure 4). Despite the use of formal modelling languages such as the SysML or UML in the solution space, the system under development is specified by descriptive, non-executable models. The verification space provides an environment for the implementation of executable simulation models (i.e. causal and non-causal 1D simulation). These simulation models are generated via model transformation from the descriptive models (Pfenning, 2017).

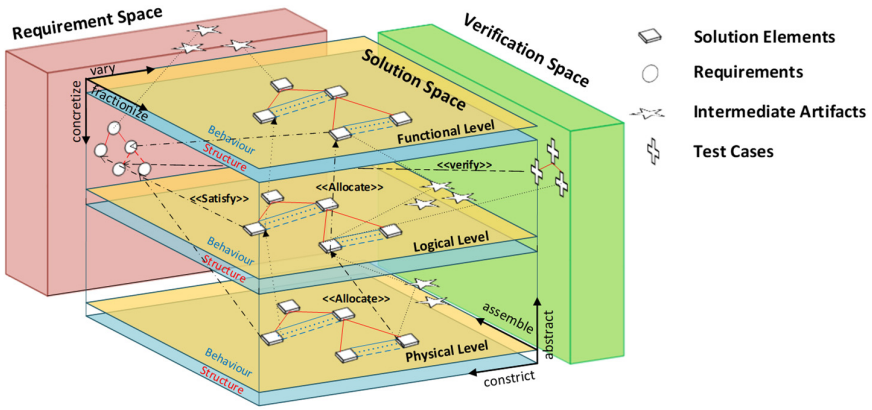

Figure 4. The System Concretization Model according to Pfenning (2017)

4.4. Comparison

While the MVPE Model describes a process model, the mecPro² Model Framework, as well as the SCM according to Pfenning, represent product models. As seen as in Figure 5, both product models have significant similarities. These results from their backgrounds, i.e. both approaches being based on concepts of existing, well-established design approaches from different disciplines. Essential aspects which are used in both the mecPro² Model Framework and the SCM are:

- The RFLP approach from the MVPE Model (Eigner et al., 2012, 2014a, 2014b)
- The viewpoints of the SPES Modeling Framework (Pohl et al., 2012)
- The subdivision in requirement and solution space including the three axes of concretization, decomposition and variation of the Munich Model of Product Concretization (Ponn and Lindemann, 2011; Lindemann, 2014)

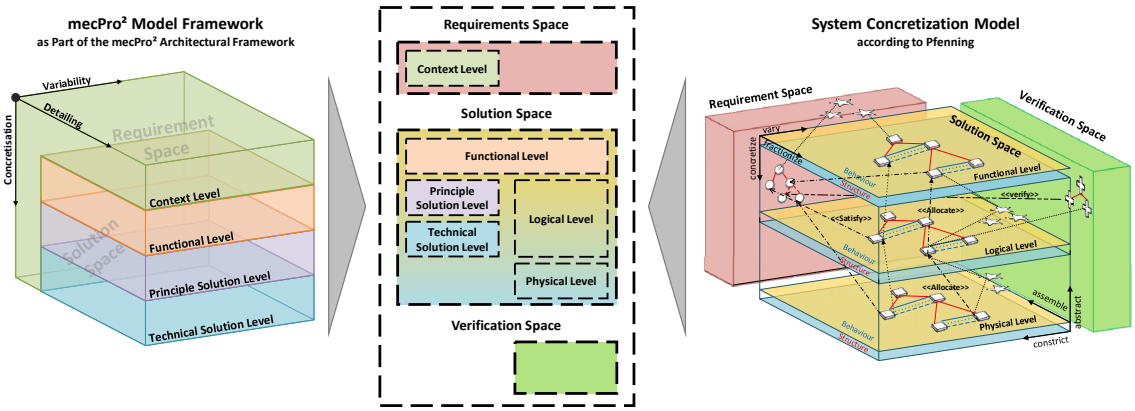

Figure 5. Comparison of the mecPro² Model Framework and the System Concretization Model according to Pfenning (2017)

It is evident that the Munich Model of Product Concretization builds the fundamental basis for both approaches. While the SCM adopts the subdivision into a requirement and solution space without further alteration, the solution space of the mecPro² Model Framework is covered by the requirement space

through the Context Level. The reason for this lies, inter alia, in the consideration of the transformation of natural language requirements into model-based requirements described by SysML elements and their interrelations (Eigner et al., 2015, 2016, 2017b). This concept is used in the SPES Modeling Framework as well (Pohl et al., 2012) and is explained in more detail by Rupp and SOPHIST (2013). The subdivision of the solution space in both models is inspired by the (R)FLP approach from the MVPE Model. Again, while the SCM adopts the concept one to one to describe the solution process, mecPro² extends the logical level by dividing it into two levels: the principle solution level and the technical solution level. Because mecPro² focuses only on the interdisciplinary system design with SysML, the physical level is not part of the Model Framework. A first concept for a model-based transition into the physical level of mechanical elements is given by Cadet et al. (2017). Both product models describe the system at each level of the solution space from a structural and a behavioural point of view. The SCM clarifies these aspects by two overlapping layers on each solution space level, while mecPro² addresses these aspects only in an indirect way through the detailed description of each level. In regard to the MVPE-Model, which defines the essential macro-development process of a system, both other approaches describe a method describing how the results of the development process steps should be realized. While mecPro² focuses on the interdisciplinary design on the left 'wing' of the V-Model, the SCM covers the whole V.

5. The KSCM as a methodical approach for the design of cybertronic systems

As a result of the high extent of overlapping aspects, the idea to merge the concepts of both approaches – mecPro² Model Framework and System Concretization Model – to one product model arose. Based on this idea, the Kaiserslautern System Concretization Model (KSCM) – a product model for the systematic development of cybertronic systems – was created (Figure 5, right side).
The fundamental structure of the KSCM is based on the SCM after Pfenning and is enriched by the essential concepts of the mecPro² Model Framework. In general, the KSCM is subdivided into the following four spaces, which will be described afterward in detail:

- requirement space
- solution space
- verification and validation space
- administration space

Like in the already existing approaches, the requirement and solution spaces are subdivided into specific abstraction levels of system specification. Because system requirements arise from the context in which the system is being analysed, the KSCM adopted the Context Level from the mecPro² Model Framework which extends the requirement space. That provides the possibility to transfer system information given by natural language requirements into a model-based system description with a higher degree of formality. This leads to two significant advantages compared against classical approaches: a maximum formalization of requirements furthers consistency and continuity in the development process; stakeholder can be answered in the language in which they have formulated their requirements, which may help to avoid misunderstandings. In general, on the Context Level the system will be regarded as a black box. However, it is possible that design decisions already exist in the form of requirements, which predefine the basic architecture to a certain extent. The essential step on the Context Level is to set the system boundary and define which external elements (external systems, stakeholder, and environmental influences) it interacts with in the different contexts. The interaction with external systems as well as the context-based expected system behaviour is bundled by predetermined system use cases. According to Crawley et al. (2016), these use cases represent the externally delivered functions of the system under development.
The solution space is adopted one to one from the SCM and consists of a Functional Level, a Logical Level, and a Physical Level. Consequently, the KSCM supports the interdisciplinary as well as the discipline-specific development processes. On the Functional Level, a functional architecture based on solution-neutral functions is defined. For that purpose, internal functions and relationships among these internal functions have to be identified. While the relationships between the internal functions are depicted by material, energy and signal flows in accordance to the VDI guideline 2221 (1993), the functions themselves are derived from the expected behaviour specified on the Context Level. Hence, the externally delivered functions of the Context Level are realized by the internal functions and relationships of the

Functional Level (Crawley et al., 2016). The Logical Level identifies the logical architecture – the form – of the system based on logical solution elements. Crawley et al. (2016) define form as the embodiment of a system, which is instrumental for the execution of the function. Form includes the entities composing the form and the relationships among those entities. Regarding the RFLP approach (Eigner et al., 2012), the entities of form are equivalent to the logical solution elements. In general, the task of a logical solution element is to realize the desired function. According to Haberfellner et al. (2012), an important aspect of the systems engineering is, among other things, not to be satisfied with the first solution. To gain a comprehensive overview of possible solutions, the identification process of the logical architecture is partitioned in two stages. The first stage comprises a systematical identification, analyzation, and evaluation of solution variants in order to enable an optimal selection of realizing elements in regard to the requirements and the degree of fulfilment of a function by the identified alternatives. The selection process in the second stage focuses on the degree of fulfilment possible logical architectures provide regarding the functional architecture defined in the previous level. If it is necessary, the Logical Level can be divided into the two sublevels, Principle Solution Level and Technical Solution Level (see Figure 4). The subdivision could be helpful, if parts of the system under development are predetermined by – for example – stakeholder requirements, and it is clear that they have to be realized by an element of specific kind, e.g. mechanical, electro-technical or software component. According to Crawley et al. (2016), with the completion of the Logical Level – and the concomitant mapping between the logical and functional architecture, respectively the mapping between form and function – the system architecture is specified. By applying discipline-specific methods and IT tools, the further detailing and discipline-specific implementation of the specified logical solution elements takes place on the Physical Level (Eigner et al., 2014).

The verification and validation space allows for the definition and execution of tests to check if the functions, logical or physical elements, the system as a whole or any of its subsystems are conform the system specification and fulfil the requirements and their intended purpose. The causal and non-causal 1D simulations Pfenning (2017) mentions in his SCM are as well included in this space. Different and new compared to the mecPro² Model Framework and the System Concretization Model is the administration space. It contains metadata such as unique identifiers and the element versions, which are necessary for the management of model elements over the entire system lifecycle. Each model element can be enriched with these metadata. While anchor elements will be managed in a System Lifecycle Management Backbone, the intermediate elements will be administered in the Team Data Management (TDM) systems of the authoring tools.

In regard to the MVPE Model, the requirement and solution spaces cover the left wing of the V, whereas the verification and validation space cover the right one and the room of virtual tests between the wings. The administration space covers, the whole MVPE Model and ensures the manageability of the model elements in a System Lifecycle Management Backbone (Figure 6).

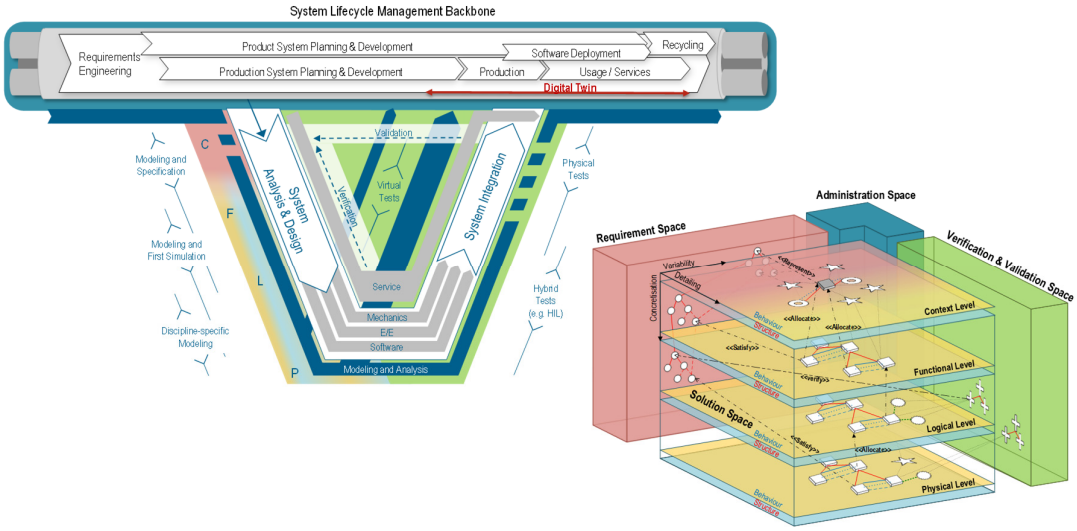

Figure 6. MVPE Model and Kaiserslautern System Concretization Model (KSCM)

The KSCM does not aim to replace the MVPE Model, instead, both models are complementary. Together they build a methodology for the development of cybertronic systems. According to Estefan (2008), a methodology can be defined as a collection of related processes, methods, and tools. While the process is defined as a logical sequence of tasks performed to achieve a particular objective by asking the question 'WHAT has to be done?' without specifying 'HOW' it should be done, the method answers this last question. A method consists of techniques to perform a task. However, each method is also a process by itself because of the sequence of tasks it contains, which have to be performed in that particular case. In other words, the 'HOW' at one abstraction level becomes the 'WHAT' at the lower level (Estefan, 2008). According to that, the MVPE Model and the Kaiserslautern System Concretization Model build together a macro-level methodology (for the development of cybertronic systems (Figure 6). While the MVPE, as a process model, defines what should be done, the KSCM, as product model, defines how it should be realized.

As mentioned before, each method includes tasks, which need to be performed during the application of this method (Figure 7). For the realization of the main tasks of the KSCM with the focus on the interdisciplinary system analysis and design (context (C), functional (F), and logical (L) level), a SysML-based methodology is in development. The so-called VPEmecPro²Methodology is a micro-level methodology for the solution process in the early development phases (Figure 7). Thereby, its process assumes the primary task and subtasks of the KSCM, while the included method defines how the process steps can be realized by using the Systems Modeling Language in order to define the system architecture based on context-based system requirements. The method includes the selection of the right SysML diagrams, elements and the definition of modelling rules.

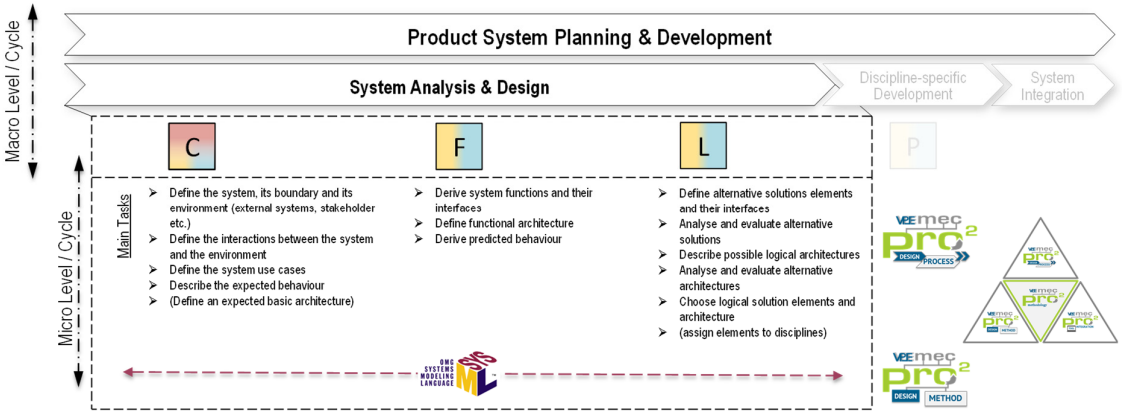

Figure 7. Micro-Level methodology for the model-based development along the KSCM

6. Conclusion and outlook

The interdisciplinary engineering of today's products and their production systems requires a rethinking of current design methodologies, processes, IT solutions, and the entire enterprise organization. Engineering design processes are continually changing imposing 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 global markets as well as from multidisciplinarity in product design and development, mainly due to the increased use of software in technical systems. The challenges mentioned above require interdisciplinarity of the design approaches. Based on aspects and concepts of existing design approaches briefly summarized in Section 2, two parallel and independent design approaches for the development and management of cybertronic systems have been identified and analysed. Because of the high level of similarities between them (see Figure 5) and the Munich Model of Product Concretization as the essential foundation of both approaches (Figure 8), an extended product model based on their merge has been developed – the Kaiserslautern System Concretization Model.

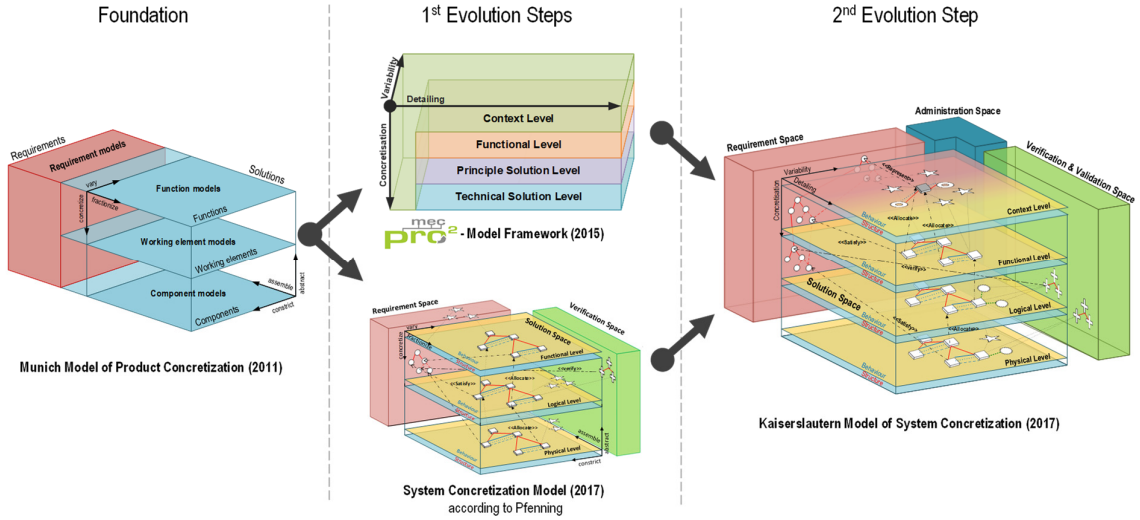

Figure 8. Evolution steps from the Munich Model of Product Concretization to the Kaiserslautern Model of System Concretization

The KSCM is integrated in the MVPE process model serving as its methodical supplement to form a holistic macro-level methodology – the VPE System Development Methodology – for interdisciplinary system development. Further, the micro-level, SysML-based methodology summarized by Figure 7 serves as a guide to help the engineer to perform the tasks in the solution-finding process in the early system development as part of their daily work. The elaboration of a case study aimed to facilitate the evaluation of the methodology based on a realistic scenario of an excavator as part of an autonomous construction area is ongoing and will be the subject of future publications.

Acknowledgments

The authors would like to thank Dr Michael Pfenning for his work, his contribution, and research on the topic of Model-based Systems Engineering and System Lifecycle Management and his additional input and assistance.

References

Andreasen, M.M. (1980), Machine Design Methods Based on a Systematic Approach, PhD thesis, Lund University.

ANSI/EIA 632 (2003), ANSI/EIA 632: Processes for Engineering a System, SAE International, USA.

Beck, K. and Andres, C. (2005), *Extreme programming explained*, Addison-Wesley, Amsterdam.

Bender, K. (2005), *Embedded Systems - qualitätsorientierte Entwicklung*, Springer, Heidelberg. https://doi.org/10.1007/b138984

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

Broy, M. (2010), *Cyber-Physical Systems - Innovation durch softwareintensive eingebettete Systeme*, Springer-Verlag, Berlin. https://doi.org/10.1007/978-3-642-14901-6

Cadet, M., Schulte, T., Dickopf, T., Stephan, N. and Eigner, M. (2017), "Modellbasierte Entwicklung cybertronischer Systeme in der frühen Phase des Entwicklungsprozesses - Ein Vergleich mit der klassischen Produktentwicklung", *Proceedings of the Stuttgarter Symposium für Produktentwicklung 2017, Stuttgart, Germany, June 28-29, 2017*, Verein zur Förderung produktionstechnischer Forschung e.V. and Fraunhofer IAO, Stuttgart.

Crawley, E., Cameron, B. and Selva, D. (2016), *Systems Architecture - Strategy and Product Development for Complex Systems*, Pearson, Hoboken, NJ.

Ehrlenspiel, K. and Meerkamm, H. (2013), *Integrierte Produktentwicklun*, Carl Hanser, München, Germany. https://doi.org/10.3139/9783446436275.fm

Eigner, M., Dickopf, T. and Apostolov, H. (2017a), "The evolution of the V-model: From VDI 2206 to a System Engineering based Approach for Developing Cybertronic Systems", *Proceedings of the PLM17 / IFIP 14th International Conference on Product Lifecycle Management, Seville, Spain, July 9-12, 2017*, IFIP, Laxenburg.
Eigner, M., Dickopf, T. and Huwig, C. (2016), "An Interdisciplinary Model-Based Design Approach for Developing Cybertronic Systems", *Proceedings of the Design 2016 / 4th International Design Conference, Dubrovnik, Croatia, May 16-19, 2016*, The Design Society, Glasgow, pp. 1647-1656.
Eigner, M., Dickopf, T., Apostolov, H., Schäfer, P., Faißt, K.G. and Keßler, A. (2014a), "System Lifecycle Management - Initial Approach for a Sustainable Product Development Process Based on Methods of Model Based Systems Engineering", *11th IFIP WG 5.1 International Conference - PLM 2014*, Springer, Heidelberg, pp. 287-300. https://doi.org/10.1007/978-3-662-45937-9_29
Eigner, M., Dickopf, T., Schneider, M. and Schulte, T. (2017b), "mecPro² - A holistic concept for the the model-based development of cybertronic systems", *Proceedings of the ICED17 / 21st International Conference on Engineering Design, Vancouver, Canada, August 21-25, 2017*, The Design Society, Glasgow, pp. 379-388.
Eigner, M., Dickopf, T., Schulte, T. and Schneider, M. (2015), "mecPro² - Entwurf einer Beschreibungssystematik zur Entwicklung cybertronischer Systeme mit SysML", In: Schulze, S.-O. and Muggeo, C. (Eds.), *Tag des Systems Engineering 2015*, Hanser, München, Germany, pp. 163-172. https://doi.org/10.3139/9783446447288.017
Eigner, M., Gilz, T. and Zafirov, R. (2012), "Proposal for Functional Product Description as Part of PLM Solution in Interdisciplinary Product Development", *Proceedings of the Design 2012 / 12th International Design Conference, Dubrovnik, Croatia, May 21-24, 2012*, The Design Society, Glasgow, pp. 1667-1676.
Eigner, M., Koch, W. and Muggeo, C. (2017c), *Modellbasierter Entwicklungsprozess cybertronischer Systeme*, Springer, Berlin. https://doi.org/10.1007/978-3-662-55124-0
Eigner, M., Roubanov, D. and Zafirov, R. (2014b), *Modellbasierte virtuelle Produktentwicklung*, Springer, Berlin. https://doi.org/10.1007/978-3-662-43816-9
Estefan, J. (2008), *Survey of Model-Based Systems Engineering (MBSE) Methodologies*. [online] INCOSE MBSE Initiative. Available at: http://www.omgsysml.org/MBSE_Methodology_Survey_RevB.pdf (accessed 22.10.2017).
Feldhusen, J. and Grote, K.-H. (2013), *Pahl/Beitz Konstruktionslehre*, Springer Vieweg, Berlin. https://doi.org/10.1007/978-3-642-29569-0
French, M.J. (1999), *Conceptual design for engineers*, Springer, London. https://doi.org/10.1007/978-1-4471-3627-9
Gajski, D.D., Abdi, S., Gerstlauer, A. and Schirner, G. (2009), *Embedded system design*, Springer, Boston. https://doi.org/10.1007/978-1-4419-0504-8
Gausemeier, J. and Lückel, J. (2000), *Entwicklungsumgebungen Mechatronik: Methoden und Werkzeuge zur Entwicklung mechatronischer Systeme*, Heinz Nixdorf Institut, Paderborn.
Haberfellner, R., Fricke, E., de Weck, O. and Vössner, S. (2012), *Systems Engineering - Grundlagen und Anwendung*, Orell füssli Verlag, Zürich.
Harashima, F., Tomizuka, M. and Fukuda, T. (1996), "Mechatronics – What Is It, Why, and How? An editorial", *IEEE/ASME Transactions on Mechatronics*, Vol. 1 No. 1, pp. 1-4. https://doi.org/10.1109/tmech.1996.7827930
IEEE 1220 (2005), IEEE 1220: Standard for Application and Management of the Systems Engineering Process, IEEE.
Isermann, R. (2008), *Mechatronische Systeme*, Springer-Verlag, Berlin. https://doi.org/10.1007/978-3-540-32512-3
ISO/IEC 15288 (2015), ISO/IEC 15288: Systems Engineering - System Life Cycle Processes, ISO, Geneva.
Kruchten, P. (2003), *The Rational Unified Process: An Introduction*, Addison-Wesley.
Künzel, M., Schulz, J. and Gabriel, P. (2016), Engineering 4.0 - Grundzüge eines Zukunftsmodells, Begleitforschung, AUTONOMIK für Industrie 4.0, VDI/VDE Innovation + Technik GmbH, Berlin.
Lee, E.A. (2008), Cyber Physical Systems: Design Challenges, Technical Report No. UCB/EECS-2008-8, University of California, Berkeley.
Lindemann, U. (2014), "Models of Design", In: Chakrabarti A. and Blessing L. (Eds.), *An Anthology of Theories and Models of Design*, Springer-Verlag. London, UK, pp. 121-132. https://doi.org/10.1007/978-1-4471-6338-1_6
Lüdecke, A. (2003), Simulationsgestützte Verfahren für den Top-Down-Entwurf heterogener Systeme, PhD thesis, Universität Duisburg, Essen.
Malmqvist, J. and Svensson, D. (1999), "A Design Theory Based Approach Towards Including QFD Data in Product Models", *Proceedings of the 1999 ASME Design Engineering Technical Conferences, Las Vegas, Nevada, USA, September 12-15, 1999*, ASME, New York, 1999.

Müller, P. (2013), *Integrated Engineering of Products and Services – Layer-based Development Methodology for Product-Service Systems*, Fraunhofer Verlag, Berlin.
Nattermann, R. and Anderl, R. (2010), "Approach for a Data-Management-System and a Proceeding-Model for the Development of Adaptronic Systems", P*roceedings of the IMECE2010 / ASME International Mechanical Engineering Congress and Exposition, Vancouver, Canada, November 12-18, 2010*, ASME, New York, pp. 379-387. https://doi.org/10.1115/imece2010-37828
Pahl, G. and Beitz, W. (1977), *Konstruktionslehre*, Springer-Verlag, Berlin.
Pfenning, M. (2017), Durchgängiges Engineering durch die Integration von PLM und MBSE, PhD thesis, Technische Universität Kaiserslautern.
Pohl, K., Hönninger, H., Achatz, R. and Broy, M. (2012), *Model-Based Engineering of Embedded Systems – The SPES 2020 Methodology*, Springer, Heidelberg. https://doi.org/10.1007/978-3-642-34614-9
Pomberger, G. and Pree, W. (2004), *Software-Engineering*, Carl Hanser, München, Germany. https://doi.org/10.3139/9783446227880
Ponn, J. and Lindemann, U. (2011), *Konzeptentwicklung und Gestaltung technischer Produkte – Systematisch von Anforderungen zu Konzepten und Gestaltlösungen*, Springer, Berlin. https://doi.org/10.1007/978-3-642-20580-4
Rauscher, R. (1996), *Entwurfsmethodik hochintegrierter anwendungsspezifischer digitaler Systeme*, Pro-Universitate-Verlag, Sinzheim.
Rumbaugh, J., Jacobson, I. and Booch, G. (2010), The Unified Modeling Language Reference Manual, Pearson Education.
Rupp, C. and SOPHIST GmbH (2013), *Systemanalyse kompakt*, Springer-Verlag, Berlin. https://doi.org/10.1007/978-3-642-35446-5
Schäppi, B., Andreasen, M.M., Kirchgeorg, M. and Radermacher, F.J. (2005), *Handbuch Produktentwicklung*, Carl Hanser, München.
VDI 2206 (2004), VDI 2206 - Entwicklungsmethodik für mechatronische Systeme – Design methodology for mechatronic systems, Beuth, Berlin.
VDI 2221 (1993), VDI 2221 - Methodik zum Entwickeln und Konstruieren technischer Systeme und Produkte, Beuth, Berlin.

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