Requirements for modeling dynamic function networks for collaborative embedded systems

Alexander Ludewig[1], Marian Daun[2], Ana Petrovska[3], Wolfgang Böhm[3], Alexander Fay[1]

Abstract: Through advances in information technology embedded systems have the capability to collaborate with one another and to merge into collaborative system groups (CSGs) thereby generating added value that a single system alone could not achieve. In order to be able to realize the development of these collaborative embedded systems (CESs) as well as the actual process of the collaboration, information models are used that describe different functions of the CESs with which they contribute to the CSG. The modeling of the individual functions of the CES and the connections of these different functions in the CSG as a function network forms the basis of the representation and realization of the overall function of the CSG with which the common added value is to be achieved. The aim of this paper is to present existing approaches of describing the functions in different domains and to present the requirements for the modeling of function networks in the context of CES and CSG. The requirements have been collected in close collaboration with industry partners from the CrESt project[4].

Keywords: function networks, functional modelling, model based systems engineering, collaboration, cyber-physical systems, embedded systems, function centered engineering

1 Introduction

Nowadays, collaboration within a group of embedded systems allows achieving complex high-value goals, which the individual systems are not able to achieve on their own. Consequently, embedded systems do no longer only sense their own environment by means of sensors and change their context by using actuators, but, furthermore, closely share information and collaborate with other embedded systems. Those embedded systems are also referred as collaborative embedded systems (CESs). This holds for various domains, although the kind of this interaction can vary greatly depending on the application domain (e.g., [KSL03], [VF13]). Due to the varying application scenarios of CESs, there is an urgent need for methods to master the complexity of development as well as the actual process of collaboration at runtime (cf. [BS14], [SP12]).

[1] Helmut-Schmidt-University Hamburg, Institute of Automation Technology, Holstenhofweg 85, 22043 Hamburg, ludewiga@hsu-hh.de, alexander.fay@hsu-hh.de

[2] University of Duisburg-Essen, paluno – The Ruhr Institute for Software Technology, Gerlingstr. 16, 45127 Essen, marian.daun@paluno.uni-due.de

[3] Technical University of Munich, Department of Informatics, Boltzmannstr. 3, 85748 Garching by Munich, petrovsk@in.tum.de, boehmw@in.tum.de

[4] The CrESt project aims at developing a consistent methodology to develop collaborative embedded and cyber-physical systems. It is publicly funded by the German Federal Ministry of Education and Research

To achieve such common goals by collaboration, the different CESs form a collaborative system group (CSG). For proper functioning of the CSG and achieving the CSG's common goals, it is necessary to describe the contribution of each single CES in that group with respect to the goals and functionality the CSG shall possess.

As functionality builds the basis of all CESs [ZH13], as well as the CSG's, it is important to evolve existing function-centered engineering approaches in such a way that they can take the needs of developing CESs into account. In this paper, we contribute a study of current functional modeling and analysis approaches, as well as general requirements for functional approaches to be used for the engineering of CESs. The requirements have been elicited by surveying industry needs from the automotive, robotics, automation, and energy domains.

The main objective of this paper is to systematically derive general requirements for function-centered engineering approaches for CESs. Therefore, Section 2 analyzes existing research approaches that focus on modeling of functions w.r.t. the ability to document collaboration aspects. Building upon this, we elicit the detailed requirements. Section 3 defines the methodological approach to requirements elicitation from industry. Subsequently, Section 4 discusses the results in terms of the final set of common requirements for functional modeling approaches. Section 5 concludes the paper and shows how requirements can be used to evolve an existing function-centered engineering approach.

2 Functional Approaches – State of the Art

In this section, we provide insight into current functional approaches and the objective of extending existing approaches. To do so, Sections 2.1 to 2.3 give a brief introduction to the foundations of functional modeling and analysis approaches. Thus, we also elaborate on the different uses of the term "function". Section 2.4 presents own previous work on function-centered engineering, which is based on the approaches of Sections 2.1 – 2.3 but does not take the engineering of CESs into account.

2.1 Functions in Systems Engineering and Mechanical Engineering

The effort to model functions as function networks has emerged as part of a framework from a model-based development approach [AL16]. This model-based development approach as part of systems engineering pursues the goal of supporting technically complex development processes through information models. These models should seamlessly replace the document-based exchange of information [FMS12].

With increasing technical complexity of the products and in particular their capability of collaboration, it has proven useful to use central information models already in early stages of development. This way, subsequent media breaks and redundancies shall be avoided and a common understanding of all actors involved in the development process as well as the requirements of the customers shall be achieved. In particular, the relevant depiction of dependencies between system elements is made possible by the use of models [KBD16]. The development of the model-based systems engineering according to [WA15] originated from a development approach which is mainly characterized by the disciplines of software and electrical engineering. The development of mechatronic systems, on the other hand, covers these disciplines as well, but originated from a mechanical point of view. Only by the increasing share of software and their influence on the capabilities of the systems, the development process according to [VDI2206] changed over time. The capability of communication and collaboration between different mechatronic systems through the possibilities of information technology has shaped another term called cybertronic systems (CTS) [ME14]. A clear differentiation between the terms CES and CTS does not yet exist. However, the demand for model-based development processes is clearly emphasized again.

Both within the existing development approaches of mechanical engineering as well as in the context of software development and systems engineering, the term ‘function’ is used and has an important role. Before considering function networks, a general differentiation of the functional concept is therefore necessary. Only with consistent understanding, a future application of function networks can take place in the different domains.

2.2 Foundations: Software Engineering

Across the literature, there are different perspectives when it comes to defining the term function, mainly coming from different standardized sources, varying between functions that can be user-centered and goal-centered, and, depending on the concrete context, considering functions as a part of the system and as a viable segment in many steps of the software development process. Accordingly to [ISO2476517] a function is a defined objective, or characteristic action of a system or component. Additionally, a function can be seen as a software module that performs a specific action, is invoked by the appearance of its name in an expression, may receive input values, and returns an output. [ISO2476510] and [ISO26514] give more user-centered function definition, as feature or capabilities of an application, seen by the user [ISO2476510] and as part of an application that provides facilities for users to carry out their tasks [ISO26514]. [ISO2476510] defines the term function as an aspect of the intended behavior of the system. The function’s input and outputs are considered in [ISO2476510], where the authors define the function as a transformation of inputs to outputs, by means of some mechanisms, and subject to certain controls, that is identified by a function name and modelled by a box. Function’s input is defined as data received from an external source, and as the entered data or the process of entering data into an information processing system or any of its parts for storage or processing. On the other side, function’s output is defined as data transmitted to an external destination, and the process by which an information processing system, or any of its parts, transfer data outside of that system or part.

When it comes to considering function as part of the software development process, the function itself is mostly related to the implementation/development phase of the development process, since mainly the system's functionality creation is being done in this phase. Nevertheless, the function has a deep impact in underpinning other software development phases. It has been related to the requirements and design phase while considering the project definition segment of the development process, and contrary, while focusing on verification segment, the function has been related to the testing phase. [ISO2476510] relates the function to the requirements phase of the software development process, and introduces the term, functional requirement, as a requirement that specifies a function that a system or system component must be able to perform. Functional design, referred to also as architectural design [ISO2476510] is the process of defining the working relationships among the components of a system and the result of that specific process. Functional testing [ISO2476510] can be considered as black-box testing that ignores the internal mechanism of a system or component and focuses solely on the outputs generated in response to selected inputs and execution conditions, or as performance testing that is conducted to evaluate the compliance of a system or component with specified functional requirements.

Functional modeling is used to model a wide variety of automated and non-automated systems. For new systems, it may be used first to define the requirements and specify the functions and then to design an implementation that meets the requirements and performs the functions. A functional model is a structured representation of the functions, activities or processes within the modeled system or subject area. It describes system functions (i.e. actions, processes, operations), functional relationships, and the data and objects that support systems analysis and design, enterprise analysis, and business process re-engineering [ISO2476510] requirements.

2.3 Foundations: Mechanical Engineering

The term function can also be found in development processes of mechanical engineering and mechatronic systems. In various guidelines, such as [VDI2221] and [VDI2222], the function is used differently, among other things as a result of a phase of development between the concrete definition of requirements or specific problem description and the finding of fundamental solutions for the system to be developed. According to [PA07], the main function of the system to be developed is initially set up, which represents a problem formulation by means of a noun-verb combination on an abstract level. According to [VDI2206], this problem formulation is also referred to as the target function for the required behavior under operating conditions. These input conditions are described as the input of the function by the flow quantities material, energy and information. The output of the function is represented here by the same flow variables with changed characteristics. The function can therefore be understood as a black box which is characterized by the ability to produce an output based on an input. In the further steps of the development, the target function is decomposed into sub-functions in order to reduce the complexity of the task to be solved. The individual sub-functions are connected with each other by the respective inputs and outputs, thereby creating the functional structure. Sub-functions can be further subdivided into basic functions to a certain degree. The functional structure is detailed so far until action principles and solution elements can be found to fulfil each basic function. Depending on the specific application domain, there are different guidelines that suggest appropriate subdivisions into basic functions. For mechatronic systems, a distinction is made between controlling, regulating, measuring and other functions. For example, [VDI3813] divides room automation functions into different classes. The common feature is the value-free, solution-neutral presentation, which does not indicate with which mechanisms, types of energy or information the actual implementation takes place.

2.4 Previous Work on Functional Modeling and Analysis of CES

In modern system development, functions are designed to be shared between the different embedded systems (e.g., [BP10], [JS00], [PBK07]). Hence, the SPES_XT modeling framework allows for separation between system functions and context functions within its extension for defining the functional design (see [AL16]) as well as within the SPES_XT context modeling framework (see [DA16]). The term system function refers to a logical function, which is part of the context subject. Context functions refer to logical functions, which are available to the context subject, but are part of context objects. The detailed meta model for functional modeling as specified by SPES_XT is given by Fig. 1. As can be seen, functions are structurally connected and have a specific function behavior.

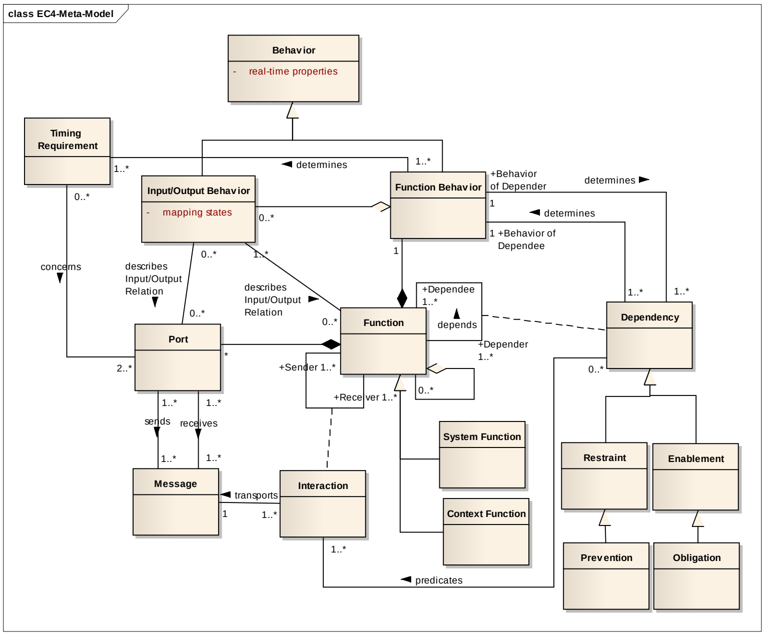

Fig. 1: Meta model for functional modeling

The SPES_XT functional modeling framework combines concepts from the current state of the art (see Section 2.1 - 2.3) to allow for continuous function-centered engineering of embedded systems. For instance, it allows distinguishing between system functions and context functions, which is, among others, needed to define which functions are in the scope of the current development project and which preexisting functions are used from other systems embedded in the same super systems. For example, to allow defining that an Adaptive Cruise Control makes use of basic functions provided by the Electronic Stability Control to determine the car's current speed and for emergency braking to prevent rear-end collisions. Furthermore, functions can be defined across different abstraction levels making use of composition and decomposition. In addition, the engineer is provided with the possibility to define the functions behavior on the respective level of granularity needed for a certain situation. For example, in early stages of requirements engineering the basic functional behavior to be implemented is defined, while in later phases a detailed port behavior of the function can be defined and checked against the original defined high-level behavior on the requirements level.

3 Methodological Approach

To meet the objectives for extending functional modeling approaches for coping with the collaborative nature of modern CES, we elicited industry's requirements. To achieve this, we relied on a three-step approach involving surveys, workshops and interviews. This section summarizes the methodological approach by detailing the single steps:

- *First Step.* In the first step, requirements for an engineering methodology for CESs have been defined by industry professionals while being supported by academia. To ensure wide applicability of the elicited requirements, requirements were elicited in six categories: flexibility, dynamicity, and adaptiveness of architectures, as well as openness, uncertainty, and awareness of context changes. To ensure domain independent applicability of the elicited requirements, requirements were exemplarily instantiated for four different industrial engineering examples from the automotive, robotics, industrial automation, and energy domain. For this first step, we basically rely on requirements elicitation conducted among industry and academic partners of the CrESt project. The elicitation was mainly conducted in several workshops for each of the six categories.

- *Second Step.* After general requirements for the engineering of CESs had been defined in Step 1, we inspected these requirements in a second step, identifying requirements that to, at least, some extend impact functional modeling and analysis approaches. Identification was conducted by the authors. Therefore, specified requirements of the six categories were reviewed and all requirements potentially relevant were identified. Subsequently, these requirements were discussed among the authors to achieve agreement on inclusion or exclusion. The final decision of inclusion and exclusion of requirements to the function relevant set of requirements was made after a further discussion, where cases of doubt have been explicitly discussed in more detail.

- *Third Step.* Based on the identified set of requirements that was relevant for defining a function-centered engineering methodology, function-specific requirements were derived by the authors, neglecting the not-relevant parts of the requirements. These requirements were grouped and aligned to avoid duplicates and redundancies. Again, agreement among the authors was achieved by various discussions. In discussions, for instance, the question how fine-grained requirements shall be defined, for trading off between the easy understandability of overall requirements and the threat to include non-function related parts within the requirements, was considered. For example, when it comes to collaborating systems the CSG commonly exhibits a varying functional behavior, depending on the current members of the CSG. However, in single systems engineering there are far more reasons for variability. The first impression to exclude such common variability as out of scope was rejected after identifying the further need to also consider variability-intensive CES partaking in a CSG.

Thus, the final set of requirements for extending the function-centered engineering methodology from Section 2.4 was defined, which will be introduced in Section 4.

4 Requirements

In the following chapter, the mentioned groups, into which the requirements have been divided, will now be discussed in more detail. The groups claim to be as general as possible for all considered domains and are therefore general in their description. By this representation, misinterpretations shall be avoided.

The first group of requirements **R1** includes the aspect of modeling the overall function. When different CESs collaborate, they contribute their individual functions, and a higher overall function can emerge. For modeling aspects, these distribution needs to be taken into account as well as the overall function that results from the interplay. Depending on the domain, a certain type of orchestration is required to make this overall function manageable. One possible way is to use a central instance that performs the orchestration tasks. Since this central instance must be assumed to be dynamic and not constant, depending on the application scenario, there is a need to appoint a CSG leader in the context of the collaboration. If the process of determining the leader is understood as a function, this functionality must be taken into account within the modeling. In general, the capabilities of influence that individual systems have on each other should be able to be mapped.

The subject of collaboration between individual systems involves the possibility that the CESs automate just this collaboration. To carry out the collaboration, the information exchange between the individual systems is required. **R2**, therefore, includes the requirements associated with the communication of functions between CESs. The function describes the specific contribution which the individual CES can provide within the framework of the collaboration. For this, the ability to formally communicate this unique function, as well as its boundaries, is required. This furthermore requires a semantically correct description of all the functions of the CESs.

The next group of requirements **R3** includes the requirements arising from the dynamic and open connections between the CESs. The number of CESs within the CSG may vary over time. CESs can leave and join the CSG. These changes in the composition of the CSG must be taken into account in the modeling, as well as the influence of the variable composition of the overall function. It should also be kept in mind that the CES within the CSG may be subject to change. Among other things, these changes may be due to the fact that the CES changes due to variability. Since the topic of variability has a large scope, it should not be considered here. It should be noted, however, that not only the overall functions of the CSG can change due to the varying number of CESs, but also on different levels, the CES itself.

Collaboration between individual CESs serves to achieve a common goal that the single system alone could not achieve. The exercise of functions of CESs serves to achieve the goals. Depending on the application domain, there may also be different conflicting goals during the collaboration. In the fourth group **R4**, therefore, relationships between functions and goals in the modeling have to be considered. In particular, the mapping of priorities of individual CES as well as the relation of priorities to the overall function must be taken into account.

The fifth group of requirements **R5** includes various aspects for modeling errors and failures in relation to function. Among other things, it requires the execution of different functionality to compensate for errors and failures. Corresponding functionalities have to be considered for future realization in the modeling. A simple example at this point is the unplanned failure of a function of a CES within the CSG. If this failure affects the overall function of the CSG, then a compensation is required. Among other things, this can be that another CES, which can also provide this function, compensates for the failure.

In group **R6**, requirements have been compiled concerning the compatibility between functions of individual CESs as well as requirements for the correctness of the execution of functions during collaboration. In order to be able to assess the reliability of the overall function of the CSG and the correctness through appropriate functionality, a modeling is required which represents both the respective contributions to the overall functions as well as planned and unplanned emergent effects.

In the last group **R7,** modeling requirements have been collected concerning the functionalities of the restructuring of the CSG. While requirements for modeling variable functional structures at different levels have been collected in **R3**, **R7** includes the concrete modeling of the functionality to perform these changes. An explicit modeling of this function is necessary to implement corresponding dynamic structures of the CSG at runtime.

5 Outlook

The capability of embedded systems to collaborate can only be achieved if the various requirements directed to these capabilities are taken into account in the early stages of development. Model-based development approaches can be used as a basis for formally unambiguously describing and interlinking all required information. Due to the early modeling of required functional properties of the systems to be developed, these functional properties can be verified already at very early stages of development. The modeling of functions during the development of the CES can find crucial application beyond this development process at runtime. The basis of the collaboration is that the capabilities and limitations of CESs are formally described as having a unified understanding of system functions and context functions within the CSG. Not only does this constitute the essential condition for the provision of a higher added value which the CSG seeks to achieve, but it can also be used to carry out appropriate analyses on the adequacy of the functions performed on the basis of function networks. In addition to the analysis of desired functions, unwanted functional interactions can be detected. Fulfilling the mentioned requirements presented in Chapter 4 is an essential aspect of developing future CES. For this purpose, appropriate methods are being researched in the project CrESt.

Acknowledgements. This research has partially been funded by the *German Federal Ministry of Education and Research* under the grants no. 01IS16043A, 01IS16043U, and 01IS16043V.

6 References

[Al16] K. Albers et.al.: System Function Networks. In: Pohl, Broy et al. (Hrsg.): Advanced model-based engineering of embedded systems: Extensions of the SPES 2020 methodology. Cham, Switzerland: Springer, S. 119–144, 2016.

[BP10] S. Brinkkemper, S. Pachidi: Functional Architecture Modeling for the Software Product Industry. In: Ali Babar, Gorton (Hrsg.): Software architecture: 4th European conference, ECSA 2010. Berlin: Springer, S. 198–213, 2010.

[BS14] M. Broy, A. Schmidt: Challenges in Engineering Cyber-Physical Systems. In: IEEE Computer Volume: 47, Issue: 2, S. 70–72, 2014.

[VDI3813] VDI GUIDELINE 3813. Building automation and control systems (BACS) - Room control functions (RA functions), 2011.

[Da16] M. Daun et.al.: SPES XT Context Modeling Framework. In: Pohl, Broy et al. (Hrsg.): Advanced model-based engineering of embedded systems: Extensions of the SPES 2020 methodology. Cham, Switzerland: Springer, S. 43–57, 2016.

[VDI2206] VDI GUIDELINE 2206. Design methodology for mechatronic systems, 2004.

[FMS12] S. Friedenthal, A. Moore, R. Steiner: A practical guide to SysML: The systems modeling language. 2nd ed. Waltham, MA: Morgan Kaufmann, 2012.

[JS00] A. Jantsch, I. Sander: On the roles of functions and objects in system specification. In: CODES 2000 (Hrsg.): Proceedings of the eighth international workshop on Hardware/software codesign, S. 8–12, 2000.

[KBD16] L. Kaiser, C. Bremer, R. Dumitrescu: Exhaustiveness of Systems Structures in Model-Based Systems Engineering for Mechatronic Systems. In: 3rd International Conference on System-Integrated Intelligence (SysInt 2016), 2016.

[KSL03] G. Karsai, J. Sztipanovits, A. Ledeczi: Model-integrated development of embedded software. In: Proceedings of the IEEE - Volume:91, Issue: 1, S. 145–164, 2003.

[Me14] H. Meissner et.al.: Model-Based Development Process of Cybertronic Products and Production Systems. Advanced Materials Research, Vol. 1018, 2014, S. 539–546.

[VDI2222] VDI GUIDELINE 2222, BLATT 1. Methodic development of solution principles, 1997.

[Pa07] G. Pahl et.al.: Konstruktionslehre: Grundlagen erfolgreicher Produktentwicklung; Methoden und Anwendung. 7. Aufl. Berlin, Heidelberg: Springer, 2007.

[PBK07] A. Pretschner, M. Broy, I.H. Kruger: Software Engineering for Automotive Systems: A Roadmap. In: Future of Software Engineering, 2007. FOSE '07: IEEE, S. 55–71, 2007.

[SP12] K. Sampigethaya, R. Poovendran: Cyber-physical integration in future aviation information systems. In: Digital Avionics Systems Conference (DASC), 2012 IEEE/AIAA 31st: IEEE, 2012.

[VDI2221] VDI GUIDELINE 2221. Systematic approach to the development and design of technical systems and products, 1993.

[ISO26514] ISO/IEC 26514:2008. Systems and software engineering — Requirements for designers and developers of user documentation, 2008.

[ISO2476510] ISO/IEC/IEEE 24765:2010. Systems and software engineering - Vocabulary, 2010.

[ISO2476517] ISO/IEC/IEEE 24765:2017. Systems and software engineering - Vocabulary, 2017.

[VF13] A. Vogelsang, S. Fuhrman: Why feature dependencies challenge the requirements engineering of automotive systems: An empirical study. In: Requirements Engineering Conference (RE), S. 267–272, 2013.

[Wa15] D.D. Walden et.al. (Hrsg.): Systems engineering handbook: A guide for system life cycle processes and activities; INCOSE-TP-2003-002-04, 2015. 4. ed. Hoboken, NJ: Wiley, 2015.

[Zh13] L. Zhang: Specifying and Modeling Automotive Cyber Physical Systems. In: Chen (Hrsg.): IEEE 16th International Conference on Computational Science and Engineering (CSE), 2013: 3 - 5 Dec. 2013, Sydney. Piscataway, NJ: IEEE, 2013.