A Method to Manage the Transition from the Principle Solution towards the Controller Design of Advanced Mechatronic Systems

zur Erlangung des akademischen Grades eines
DOKTORS DER INGENIEURWISSENSCHAFTEN (Dr.-Ing.)
der Fakultät für Maschinenbau
der Universität Paderborn

genehmigte
DISSERTATION

von
M.Sc. Cheng Yee Low
aus Pahang, Malaysia

Tag des Kolloquiums:	23. März 2009
Referent:	Prof. Dr. –Ing. Jürgen Gausemeier
Korreferent:	Prof. Dr. –Ing. habil. Ansgar Trächtler

Foreword

The Heinz Nixdorf Institute is an interdisciplinary research institute in the field of information technology. We develop new technologies, innovative applications as well as methods for the design of technical systems of tomorrow. Special emphasis lies on mechatronics and especially the corresponding design methodology.

The design of mechatronic systems is still a challenge. Established design methodologies, e.g. of conventional mechanical engineering, are no longer adequate – especially in the early design phase "conceptual design", which results in the so-called "principle solution". The principle solution determines the basic structure and the operation mode of the system in a domain-spanning way. Such a principle solution forms the basis for the subsequent concretization in the various domains of mechatronics, i.e. mechanics, electric/electronics, control engineering and software engineering. To ensure a consistent development process, domain-specific concretization tasks need to be extracted systematically from the principle solution.

Against this background, Mr. Low has developed a novel method to manage the transition from the principle solution towards the concretization in the domain of control engineering. The basis is a specification technique for the description of the principle solution, developed at my research group. Mr. Low defined, how to specify the key control concepts within the principle solution and how to extract all information, relevant for the subsequent domain-specific controller design. He evolved his method in the context of the Collaborative Research Center 614 and successfully validated it by means of a self-optimizing motor drive and an autonomous railway vehicle.

The work of Mr. Low is a significant contribution to the advancement of design methodology for mechatronic systems. Indeed he has made an important step towards realizing our vision of a new school for the design of technical systems of tomorrow.

Paderborn, March 2010 *Prof. Dr.-Ing. J. Gausemeier*

Acknowledgements

This work “A Method to Manage the Transition from the Principle Solution towards the Controller Design of Advanced Mechatronic Systems” is the outcome of three years research at the Heinz Nixdorf Institute, University of Paderborn.

First and foremost, I am eternally grateful to Prof. Dr.-Ing. Jürgen Gausemeier for giving me the chance to conduct my doctoral research in his working group. I really appreciate his strategic advices and patient supervision in the previous years.

I am very grateful for the co-advisorship of Prof. Dr.-Ing. Ansgar Trächtler. I am also grateful to Prof. Dr.-Ing. Detmar Zimmer as the chairman of the board of examiners and to Prof. Dr.-Ing. Rainer Koch as the assessor of the board of examiners.

I thank the International Graduate School of Dynamic Intelligent Systems at the University of Paderborn for its generous fellowship and many other supports.

Regarding my work, I am particularly indebted to my current and former team leaders Dipl.-Wirt.-Ing. Sascha Kahl and Dr.-Ing Ursula Frank for their dedicated guidance and constructive criticism. Besides that, I am truly grateful to Dipl.-Ing. Bernd Schulz and Dipl.-Ing. Christian Henke for their valuable inputs with respect to the application examples in this work.

I would like to thank all current and former colleagues, especially the team Design Methodolgy for Mechatronic Systems, for their acceptance and tolerance in all my endeavors in the team. I thank Dipl.-Wirt.-Ing. Ingo Kaiser, Dipl.-Ing. Roman Dumitrescu, Dipl.-Wirt.-Ing. Sebastian Deyter, Dipl.-Wirt.-Ing. Jörg Donoth, Dipl.-Math. Herbert Podlogar, M.Sc. Lydia Lackmann, Dipl.-Info. Sebastian Pook, and Dipl.-Wirt.-Ing. Andreas Warkentin.

I also have to thank Sabine Illigen and Alexandra Dutschke as well as the team of Dipl.-Ing. Karsten Mette for their kind assistance all the time.

I must acknowledge Dr.-Ing. Hua Chang who assisted me during the early period of my time in the working group. I also thank M.Sc. Madhura Purnaprajna for her proof reading of this work. I am indeed grateful to all friends for their encouragement and motivation.

Last but not least, I thank my family and relatives, especially my parents, for their unconditional care from the other side of the globe.

Paderborn, October 2008 *Cheng Yee Low*

List of Published Partial Results

[GFL+07a] GAUSEMEIER, J.; FRANK, U.; LOW, C.; HENKE, C.: From Domain-Spanning Conceptual Design to Domain-Specific Controller Design of Self-Optimizing Systems. In: Proceedings of the Systems Engineering for Future Capability, February 12-13, Loughborough, UK, 2007

[GFL+07b] GAUSEMEIER, J.; FRANK, U.; LOW, C.; HENKE, C.: Synergistic Impact of Domain-Spanning Conceptual Design on Control of Self-Optimizing Systems. In: Proceedings of the 1st IEEE International Systems Conference – SysCon 2007, April 9-12, Honolulu, USA, 2007

[GKL+08a] GAUSEMEIER, J.; KAHL, S.; LOW, C.; SCHULZ, B.: From the Principle Solution towards Controller Design of Self-Optimizing Systems. In: Proceedings of the 7th International Heinz Nixdorf Symposium, February 20-21, Paderborn, Germany, 2008

[GKL+08b] GAUSEMEIER, J.; KAHL, S.; LOW, C.; SCHULZ, B.: Systematic Development of Controller Design Based on the Principle Solution of Self-Optimizing Systems. In: Proceedings of the 10th International Design Conference – DESIGN 2008, May 19-22, Dubrovnik, Croatia, 2008

[GLS+08] GAUSEMEIER, J.; LOW, C.; STEFFEN, D.; DEYTER, S.: Specifying the Principle Solution in Mechatronic Development Enterprises. In: Proceedings of the 2nd IEEE International Systems Conference – SysCon 2008, April 7-10, Montreal, Canada, 2008

A Method to Manage the Transition from the Principle Solution towards the Controller Design of Advanced Mechatronic Systems

Contents Page

1 Introduction

1.1 Problem Statement

Various kinds of advanced technical systems have been developed nowadays for applications in space exploration, national security, transportation, industrial automation, and health care. Such systems rely on the close interaction of mechanics, electric/electronics, control engineering, and software engineering, which is aptly expressed by the term mechatronics. The symbiosis of the diverse domains of mechatronics enables these technical systems with novel functionality that was infeasible before in any of the individual domains.

Such beneficial potential of mechatronics is obtained from the innovation potentials of the technologies and by functional and spatial integration of the technologies [VDI2206, p. 18]. The current trend in mechatronics led by the conceivable development of information technology which will enable such systems with inherent partial intelligence. They will be able to learn, to communicate, and to optimize their behavior autonomously in response to environmental or operational changes [Gau02] [Gau05].

Given the interdisciplinary nature of mechatronics, numerous methods and tools are deployed from the conception of ideas until the prototyping of mechatronic systems. Along the development flow, the more problems can be resolved during the early development phases, the fewer problems will have to be resolved later by costly investigations based on real prototypes [Trä07, p.4]. In current practice, the domain-specific approaches adopted by the engineers and their scattering design concepts create a chaotic situation for those who try to develop breakthrough solutions in mechatronics. Despite the enormous coordination effort between the various domains of mechatronics, there are still time-consuming and costly design changes which hinder the overall development progress of such systems [GLS+08].

In order to fully utilize the beneficial potentials of mechatronics, engineers have to adopt an interdisciplinary and integrative approach to design mechatronic systems. Such an approach requires competences that are not confined to a single engineering domain. They need to be capable of maneuvering and communicating across domains and seamlessly integrate the domain-specific work which were conventionally done in an independent fashion. It is a must rather than a preference for engineers to comprehend not only the behavior of constituent elements of a mechatronic system but how they act together to form the overall behavior of the whole system.

A significant milestone for the development of advanced mechatronic systems lies at their conceptual design phase. During the conceptual design phase, engineers elicit the needs and the system characteristics desired by their clients, and subsequently conceptualize the basic structures and the modes of operation of the system to be developed. At this stage, the initial basic concepts from the various domains of mechatronics have to be integrated in order to constitute an overall concept of the system to be developed. Such a domain-spanning concept determines the principle solution of mechatronic systems.

In current practice, the principle solution of mechatronic systems is specified as per the competence, experience, and creativity of the engineers in charge. It is the responsibility of the engineers to define the basic concepts from each of the domains of mechatronics and subsequently integrate them together. In such a circumstance, there can be a risk where the concepts from a certain domain of mechatronics become very dominant or the concepts from another domain is not well taken up during the conceptual design phase [Fra06, p. 44]. In this context, the dominant concepts can be over considered, while the other concepts can be unrecognized or unexpressed. The consequence can be that the beneficial potential of mechatronics is not fully utilized and results in a suboptimal principle solution.

As such, the basic concepts from the various domains of mechatronics that have to be taken into consideration during the conceptual design phase have to be defined. Furthermore, systematic approaches to ensure a seamless development flow from the domain-spanning conceptual design phase towards the domain-specific concretization phase are necessary, but remain hitherto a challenge to the design community.

1.2 Objective

The objectives of this work are the following:

- An approach to specify the basic control concepts within the domain-spanning principle solution of advanced mechatronic systems.
- An approach to manage the information extraction from the principle solution for the controller design of advanced mechatronic systems.

Both approaches constitute a method to manage the transition from the principle solution towards the controller design of advanced mechatronic systems. The method is exemplified and validated using the demonstrators of the Collaborative Research Center 614 “Self-Optimizing Concepts and Structures in Mechanical Engineering”. The demonstrators consist of a self-optimizing motor drive and an autonomous railway convoy.

1.3 Approach

This section describes the structure of the dissertation.

Chapter 2 starts with the definition of terms. It is followed by the principles of advanced mechatronic systems. In this section, the paradigm shift from mechatronics to self-optimization is explained. After that, a brief introduction into the application examples is given. Subsequently, the development of advanced mechatronic systems is addressed. After defining the problems to be resolved, the requirements for managing the transition from the principle solution towards the controller design of advanced mechatronic systems are outlined.

Chapter 3 analyses the state-of-the-art for this work. It covers domain-spanning design methodologies and specification techniques for advanced mechatronic systems as well as the domain-specific design methodologies and specification techniques for controller design. At the end of the chapter, the call for action in fulfilling the requirements outlined in Chapter 2 is described.

Chapter 4 starts by describing an approach to specify the basic control concepts within the domain-spanning principle solution of advanced mechatronic systems. Further, an approach to manage the extraction of information from the principle solution for the controller design of advanced mechatronic systems is described. Subsequently, the concretization of controller design is described.

Chapter 5 further elaborates the application examples introduced in Chapter 2. In this chapter, the method presented in Chapter 4 is exemplified by a self-optimizing motor drive and an autonomous railway convoy. At the end of the chapter, the method is evaluated against the requirements outlined in Chapter 2.

Chapter 6 summarizes the work done and subsequently proposes the future work.

Supplementary remarks regarding the application examples described in **Chapter 5** are attached in the Appendix.

2 Problem Analysis

First and foremost, Section 2.1 defines the fundamental terminologies used in this work. This is followed by Section 2.2 which describes the principles of advanced mechatronic systems. A brief introduction into the application examples is provided in Section 2.3. Section 2.4 then describes the development flow of such systems. Subsequently, Section 2.5 defines the problems encountered during the transition from the conceptual design phase towards the controller design phase of such systems. Finally, the requirements to be fulfilled in this work are outlined in Section 2.6.

2.1 Definition of Terminologies

There are numerous definitions of a **system**. The International Council on Systems Engineering (INCOSE) deploys the following consensus about the definition of a system.

> *"A system is a construct or collection of different elements that together produce results not obtainable by the elements alone. The elements, or parts, can include people, hardware, software, facilities, policies, and documents; that is, all things required to produce systems-level results. The results include system level qualities, properties, characteristics, functions, behavior and performance. The value added by the system as a whole, beyond that contributed independently by the parts, is primarily created by the relationship among the parts; that is, how they are interconnected [Tec04]."*

As pointed out above, the value added by the system as a whole depends on how the parts are interconnected. As such, the added value has to be assured during the **design** of a system. Design can be understood as follows.

> *"The conceiving of a whole, a solution concept, the identifying or finding of the solution elements required for this and the intellectual, model-based joining together and connecting of these elements to form a workable whole [DH02, p. 158]."*

This idea also includes the so-called conceptual design. With the increasingly complex functionality of the technical systems nowadays, the aforementioned added value has to be assured as early as during the **conceptual design** phase of the systems. The conceptual design of technical systems is understood as follows.

> *"Conceptual design is achieved by abstracting the essential problems, establishing function structures, searching for suitable working principles and then combining those principles into a working structure. Conceptual design results in the specification of a principle solution (concept) [PBF+07, p. 131]."*

The outcome of conceptual design is the **principle solution** of the system to be developed.

> *"The principle solution is the fundamental solution for a development task, specifying the basic aspects of the physical operation, the type of components, and their arrangement but without defining them in detail [PB96]".*

Design is accordingly a process which, starting from the requirements, leads to the concretization of a technical system [VDI2206, p. 113]. As the design of technical systems is getting more challenging than ever before, a comprehensive **design methodology** is essential. A design methodology can be defined as:

> *"Design methodology is a concrete course of action for the design of technical systems that derives its knowledge from design science and cognitive psychology, and from practical experience in different domains. It includes plans of action that link working steps and design phases according to content and organisation. These plans must be adapted in a flexible manner to the specific task at hand. It also includes strategies, rules and principles to achieve general and specific goals as well as methods to solve individual design problems or partial tasks [PBF+07, p. 9]."*

As pointed out in the definition above, different domains and different design phases are involved during the aforementioned course of action for the design of technical systems. In current practices, different kinds of **specification techniques** are used to assist the engineers who engaged in the development project.

> *"Specification techniques provide the basis for the formulation, description and documentation of development outcomes. They consist of signs and symbols as well as the rules governing their usage. Specification techniques can be informal, semiformal or formal. Informal specification techniques are imprecise and interpretable (as text or sketches). Formal specification technique can be processed by computers and are therefore very precise but not interpretable. Semiformal specification techniques can be classified in between: they are neither as precise as the formal*

> *specification techniques nor as interpretable as the informal specification techniques [Fra06, p. 8].”*

The other terms and definitions will be included within their respective contexts throughout this work.

2.2 The Principles of Advanced Mechatronic Systems

The term “mechatronics” refers to the symbiotic cooperation of mechanics, electric/electronics, control engineering, and software engineering in order to improve the behavior of a technical system. Due to the advancements in information technology and integrated microprocessors, future mechatronic systems will include subsystems with inherent partial intelligence. They will be able to learn, to communicate, and to optimize their behavior autonomously in response to the changing environmental or operational conditions. The overall behavior of the system will be characterized by the communication and cooperation between these intelligent subsystems. The term self-optimization characterizes this perspective [GZF+07]. Taking into consideration of this paradigm shift from mechatronic systems to self-optimizing systems, the following subsections first describe the principles of mechatronics and then the paradigm of self-optimization.

2.2.1 Mechatronics

Apart from mechanics and electronics, modern technical products encompass a high degree of information and communication technology. This is aptly expressed by the term “mechatronics”. A widely accepted definition for mechatronics is given by the Association of German Engineers, as the following.

> *“Mechatronics is the synergetic integration of mechanical engineering with the electronic and intelligent computer control in the design and manufacturing of industrial products and processes [VDI2206, p. 14].”*

The symbiotic cooperation of mechanics, electric/electronics, control engineering and software engineering opens up fascinating perspectives for the development of future technical products. As shown in Figure 2-1, mechatronic systems can be classified into two categories.

- The first category of mechatronic systems is based on the spatial integration of mechanics and electronics. The aim is to achieve a high density of mechanical and electric functions within the available space. The main potentials of such integration are miniaturization, lower production costs and higher reliability. The focus of this category is placed on

the assembly and connecting technologies, e.g. MID (Molded Interconnect Devices).

- The second category of mechatronic systems deals with the controlled movements of multi-body systems. The aim of this category is to optimize the behavior of a technical system. Sensors collect information about the environment and the system itself. The system utilizes this information to derive optimal reactions. The reactions are realized by the system's actuators. Thus, these systems are able to react to changing environmental conditions, to identify critical operating conditions and to optimize their activities by applying the principles of control engineering.

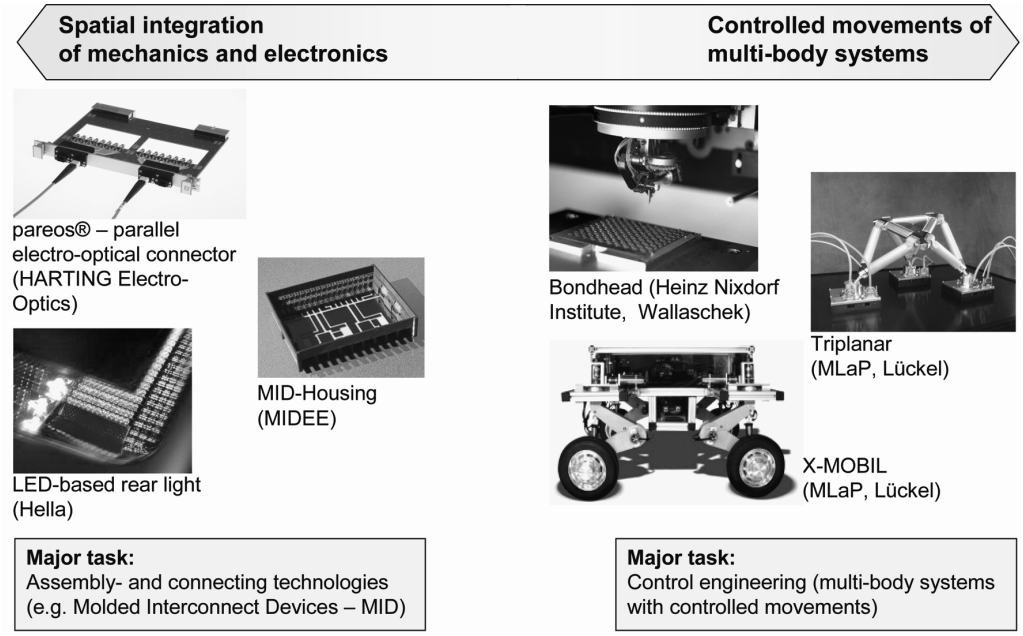


Figure 2-1: Categories of mechatronic systems [GKP08, p. 5]

Usually mechatronic systems of the second category have to handle more than one control task which are hierarchically interdependent and thus have to be coordinated. To cope with this complexity, LÜCKEL [LHL01] uses a hierarchical structure of three levels as shown in Figure 2-2. The basis of this hierarchical structure is provided by the so called mechatronic function modules (MFMs), consisting of a basic mechanical structure, sensors, actuators and a local information processor containing the controller. A combination of MFMs, coupled by information technology and mechanical elements, constitute an autonomous mechatronic system (AMS). Such systems also possess a controller, which deals with higher-level tasks such as monitoring, fault diagnosis and maintenance decisions as well as generating parameters for the subordinated information processing systems of the MFMs. Similarly, a number of AMSs constitute a so called networked mechatronic system (NMS), simply by cou-

pling the associated AMSs via information processing. In the context of railway vehicle technology, a spring and tilt module would be a MFM, a RailCab would be an AMS, and a convoy would be a NMS.

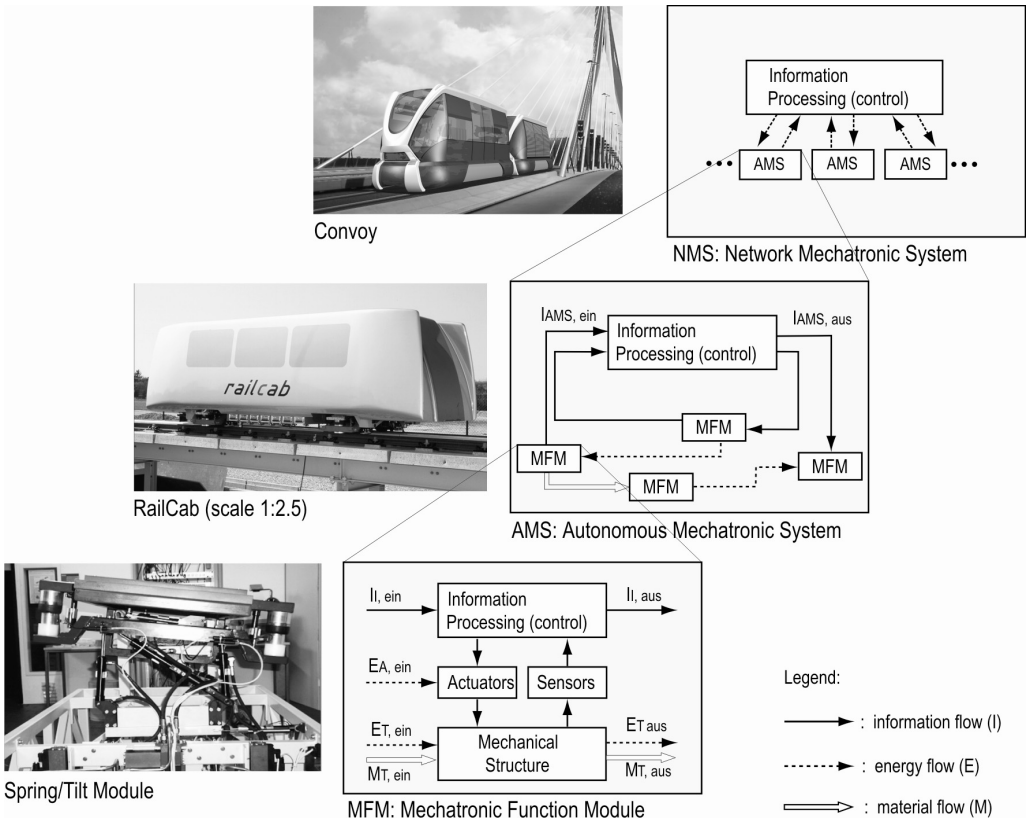


Figure 2-2: Hierarchical structure of mechatronic systems by LÜCKEL [LHL 01]

2.2.2 Self-Optimization

The conceivable development of information technology will enable mechatronic systems with inherent partial intelligence. Such a development leads to the perspective of self-optimization. The collaborative research centre CRC 614 "Self-Optimizing Concepts and Structures in Mechanical Engineering" defines the term "self-optimization of a technical system" as follows.

> *"The self-optimization of a technical system is understood to be the endogenous adaptation of the system's objectives to changing influences and the resultant purposive autonomous adaptations of its parameters, possibly also its structure, and thus its behavior. Self-optimization thus goes considerably beyond the familiar rule-based and adaptive strategies; self-optimization facilitates systems with inherent "intelligence" that are able to take action and react autonomously and flexibly to changing operating conditions [SFB01]."*

In order to have a clear structure for the self-optimizing systems, the hierarchical structure of mechatronic systems suggested by LÜCKEL is adopted and extended to include the aspect of self-optimization. The result is shown in Figure 2-3. On each hierarchical level, the controllers are enhanced by the functionality of self-optimization. Thus the system elements (that means MFM, AMS and NMS) receive an inherent partial intelligence. The behavior of the overall system is characterized by the communication and cooperation between these intelligent system elements. From the point of view of information technology, these distributed systems are considered as cooperative software agents.

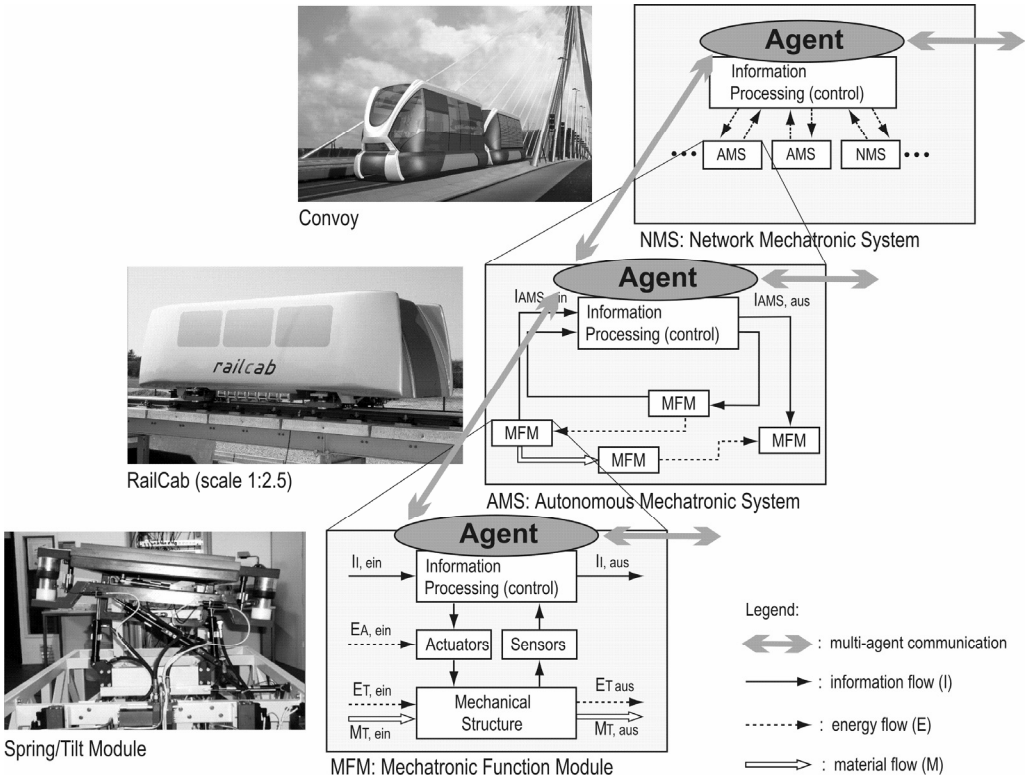


Figure 2-3: Hierarchical structure of a self-optimizing system

As illustrated in Figure 2-4, with the closed-loop control as the basis, self-optimization goes considerably beyond the well known rule-based and adaptive control strategies. Self-optimizing systems are more superior due to the fact that while the adaptive control strategies optimize the control parameters concerning to the fixed objectives, self-optimizing systems can even adapt their system of objectives.

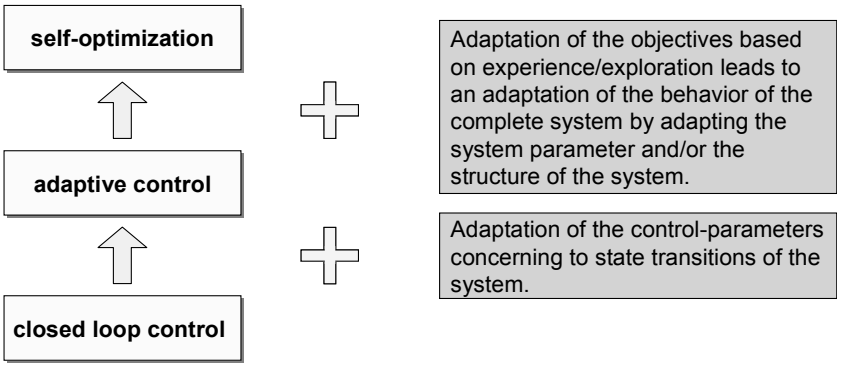


Figure 2-4: The current trend in mechatronic research: from closed-loop control towards self-optimization

The key aspects and the mode of operation of a self-optimizing system are illustrated in Figure 2-5. The self-optimizing system determines its currently active objectives on the basis of the encountered influences. There are three types of objectives: external objectives, inherent objectives, and internal objectives. The external objectives affect the system extraneously (e.g. by the user or the other systems) whereas the inherent objectives reflect the intrinsic purpose of the system and guarantee the system's functionality. The internal objectives are generated from the external and inherent objectives for performing an optimization. These system objectives are adapted autonomously. Adapting the objectives means, that the relative weighting of the objectives is modified, new objectives are added or existing objectives are discarded and no longer pursued. In the following sections, the term objective refers to the inherent objectives unless otherwise specified.

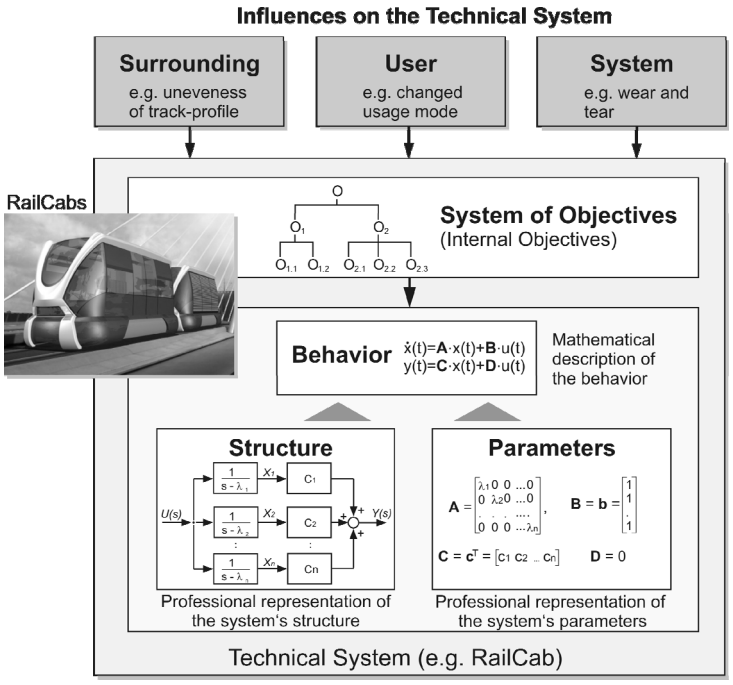


Figure 2-5: Key aspects of a self-optimizing system [GFD+08a]

Adapting the objectives in this way leads to the adaptation of system behavior. That is achieved by adapting the parameters and where necessary the structure of the system. The term parameter adaptation means modifying a system parameter, for instance, changing a control parameter. Structural adaptations affect the arrangement of the system elements and their relationships. In case of structural adaptation, it can be either a reconfiguration or a compositional adaptation. Reconfiguration means to modify the relationships of a fixed amount of elements of the structure. Compositional adaptation means to add new elements and/or remove actual elements from the structure.

Self-optimization takes place as a series of three actions that are generally carried out repeatedly. This sequence of actions is called a **self-optimization process** [FGK+04]**.**

1. Analysis of the current situation. Here the self-optimizing system acquires all relevant data about its actual state and its environment. Besides the observations previously stored, observations can also be made by communicating with the other systems indirectly. The performance of the objectives pursued is a main aspect of the analysis.

2. Determination of the system objectives. The objectives can be determined by selection, adaptation or generation. Selection refers to be the selection of alternative objectives from a finite amount of possible objectives. Adaptation refers to the gradual modification of existing objectives. Generation refers to the addition of new objectives from the existing ones.

3. Adaptation of the system behavior. Adapting the objectives in this way leads to adaptation of the system behavior. This is achieved by adapting the parameters and where necessary the structure of the system.

From a given initial state, the self-optimization process goes on, on the basis of specific influences, into a new state, i.e. the system undergoes a state transition. The process can be carried out on each hierarchical level of the system, as shown in Figure 2-3. Obviously, the realization of such a process will demand enormous information processing. For this purpose, an adequate concept to structure the information processing is needed. Therefore the concept of **Operator Controller Module (OCM)** was developed [SFB04] [OHG04]. From the information processing point of view, it is considered to be an agent. As shown in Figure 2-6, the OCM is composed of three levels (Controller, Reflective Operator and Cognitive Operator). Each level of the OCM is explained in the following paragraphs.

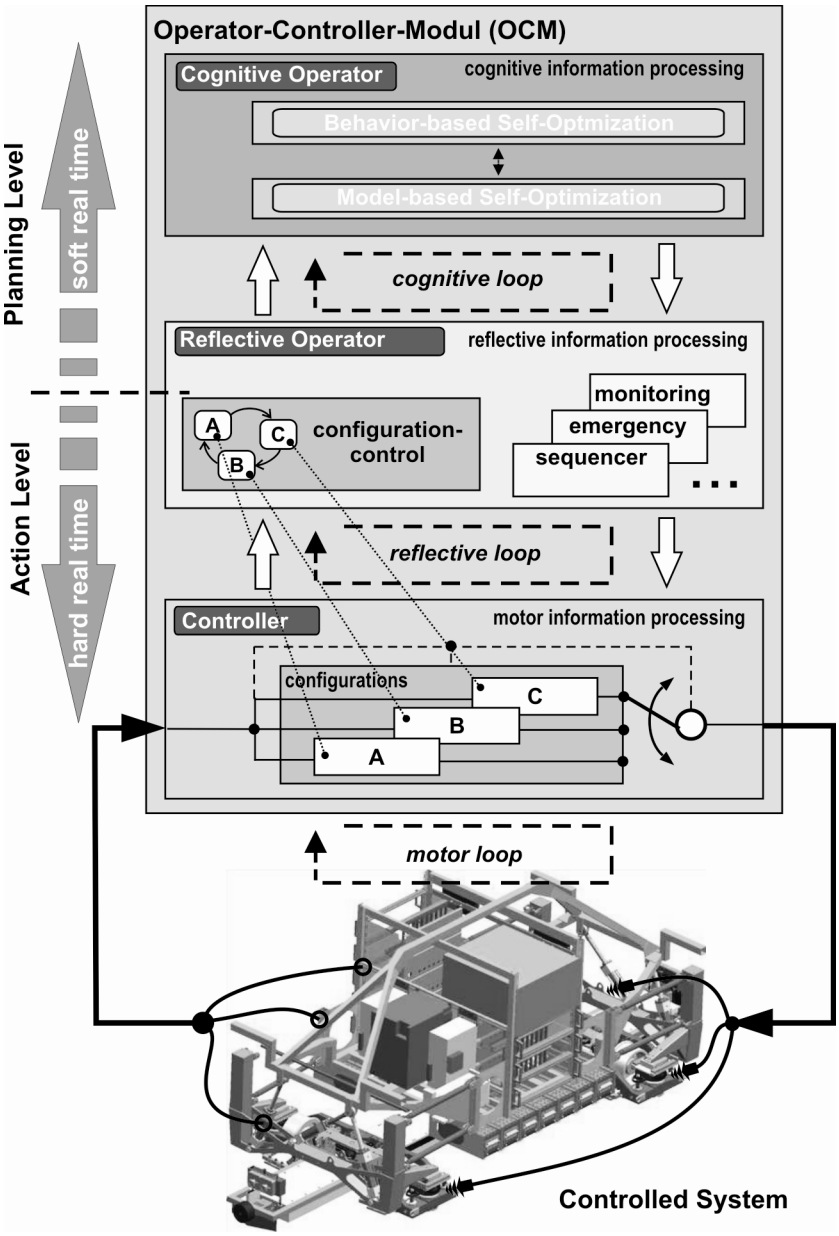


Figure 2-6: Architecture of the Operator Controller Module (OCM)

- **Controller:** The controller is positioned at the lowest level of the OCM which accesses through the controlled technical system. This control loop is an active chain that obtains measurement signals, determines adjustment signals and outputs them. For this reason it is called the "motor loop". The software at this level operates continuously under hard real-time conditions. The controller itself can be made up of a number of controller units with the possibility of switching control between them.

- **Reflective Operator:** The reflective operator monitors and directs the controller. It does not access the system's actuators directly, instead it modifies the controller by initiating changes to parameters or structures.

A structural change, such as a reconfiguration, not only replaces the control units, it also switches over the corresponding control flows and/or signal flows in the controller. Combinations of control units, switch elements and the associated control or signal flows are called "controller configurations". As shown in Figure 2-6, the blocks labeled A, B, and C represent different configurations of the controller. The configuration control – realized by means of a state machine – defines which configuration is valid in which system state, and how and under what circumstances it switches between them. The reflective operator is essentially event-oriented. Its close connection with the controller requires it to process events in hard real-time. As a connective element to the cognitive operator, the reflective operator serves as an interface between the controller and those elements which are not capable of real-time operation, or in another words, those element which work in soft real-time. In the context, it filters the incoming signals and feeds them to the lower levels. Apart from that, the reflective operator is also responsible for the real-time communication between a number of OCMs which together constitute a composed self-optimizing system.

- **Cognitive Operator:** At the highest level of the OCM, the system can employ a variety of methods (such as learning methods, model-based optimization, or knowledge-based systems) to use information about itself and its environment to improve its own behavior. Here the emphasis is on the cognitive ability to perform the self-optimization. The used method permits a pre-emptive optimization that does not interact in real-time with the actual system.

The underlying processes of the self-optimization (1. analysis of the current situation, 2. determination of the system objectives, and 3. adaptation of the system behavior) can be carried out in a multitude of ways within the OCM architecture. When the self-optimizing adaptation needs to fulfill real-time requirements, all three actions are carried out in the reflective operator. Systems that do not need to satisfy real-time conditions can use more complex procedures that are located in the Cognitive Operator. In this case the behavioral adaptation is carried out indirectly, relayed by the Reflective Operator. It has to synchronize the instructions to adapt the behavior with the real-time operation of the Controller. There are also hybrid forms that occur within a single OCM, when the two described forms of self-optimization take place simultaneously and asynchronously.

2.3 Introduction into the Application Examples

Two demonstrators of the Collaborative Research Center 614 “Self-Optimizing Concepts and Structures in Mechanical Engineering” are selected as the application examples in this work. They consist of a self-optimizing motor drive and an autonomous railway convoy.

2.3.1 Self-Optimizing Motor Drive

A motor drive is a device which converts electrical power into mechanical power in order to provide motion. In this application example, the early development phases of a self-optimizing motor drive controller are considered. The task is the control of the angular dynamics of the motor drive, which is elementary for a large number of applications, e.g. machine tools and general drives for automation purposes. Figure 2-7 shows the test bench for the self-optimizing motor drive developed in the laboratory of the Institute of Power Electronics and Electrical Drive, University of Paderborn. The test bench consists of a host computer with Field Programmable Gate Array (FPGA), a permanent magnet motor, a load machine, and power electronics.

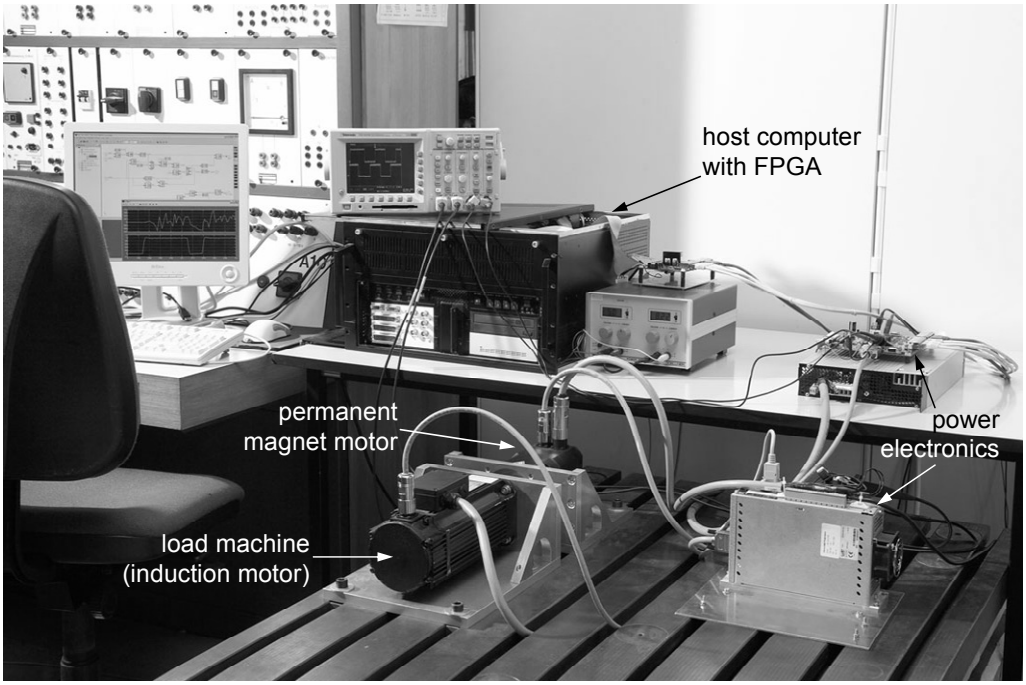


Figure 2-7: Test bench for the self-optimizing motor drive

In this application example, the controllers of the motor drive run alongside the other applications on the same computation platform. That means the controllers have to compete for the available resources with the other applications. In this context, available resources refer to the available memory, the available computation time, as well as the surface area of hardware-logic-cells available

on the FPGA [SPH+07]. An application can only be activated if there are sufficient resources available in the system. The resources are allocated during run time depending on whether the applications are compulsory to be executed, can be temporarily suspended or can be totally avoided. Depending on the changing operating and environmental condition, an appropriate controller structure has to be determined for the motor drive. In the test bench, the different system states are emulated by changing the torque or the speed of the load machine with respect to time. The other applications running alongside the motor drive are not real physical devices, but only simulations.

2.3.2 Autonomous Railway Convoy

"Neue Bahntechnik Paderborn/RailCab" is a research project at the forefront of innovative railway technology. The core of the system comprises autonomous railway vehicles, which are called "RailCabs" [Trä06]. In contrast to the traditional railway system, the RailCabs feature partial intelligence by means of self-optimization [TMV06]. Being autonomous, the RailCabs can transport passengers and goods based on individual demands rather than a fixed timetable. Besides being able to ensure a high level of comfort while travelling on changing terrain and traffic conditions, the RailCabs have the ability to form a convoy in order to reduce energy consumption. The RailCabs as well as the test track are built on a scale of 1:2.5 at the University of Paderborn. Figure 2-8 shows two RailCabs during the field test of convoy operation on the test track at the university.

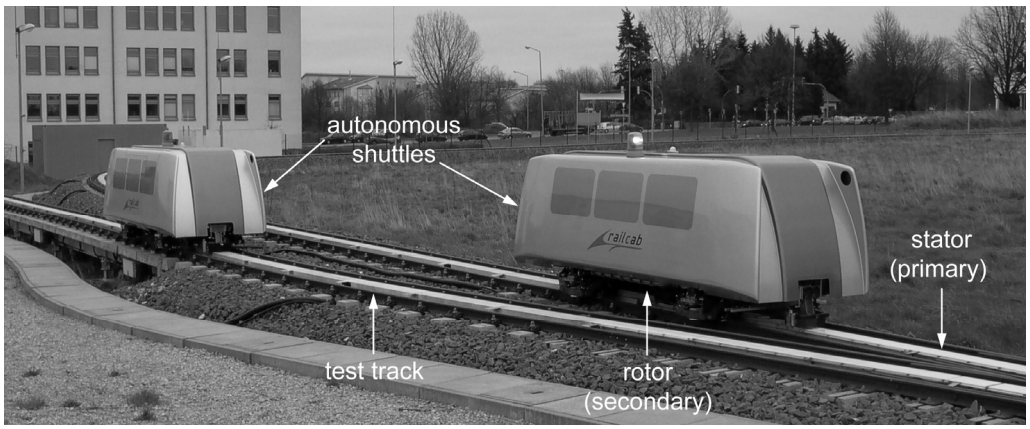


Figure 2-8: Field test of convoy operation on the test track (scale 1:2.5)

A RailCab consists of a number of modules such as the drive-and-brake module, the active guidance module, the spring-and-tilt module, the air gap adjustment module, the energy management module, as well as the communication module. This application example deals with the control of the longitudinal dynamics demanded by the autonomous convoy operation of the RailCabs

[HTS+08a] [HTS+08b]. In this context, only a convoy of two RailCabs, which is the simplest configuration for convoy operation, is considered here.

2.4 Development of Advanced Mechatronic Systems

Increasing functionality of mechatronic products has resulted in increased complexity in their development. The established design methodologies for technical systems, for instance [PBF+07] and [VDI2206], lay the foundation for the development of mechatronic systems. On a generic level, the development of mechatronic systems starts with domain-spanning conceptual design, followed by domain-specific concretization, and ends with system integration. This generic flow for the development of mechatronic systems is shown in Figure 2-9.

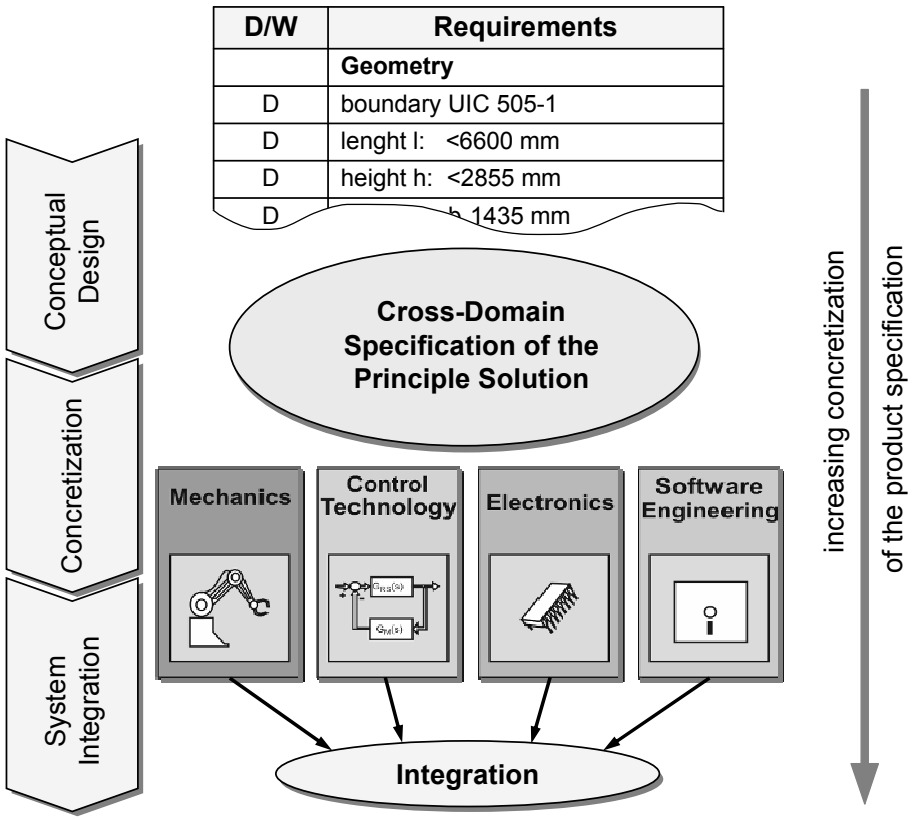


Figure 2-9: Generic development flow of mechatronic systems

A significant milestone during the development of advanced mechatronic systems lies at the beginning of the development project, i.e. the conceptual design phase. The conceptual design of advanced mechatronic systems involves planning and clarifying the task, conceptual design on system level, conceptual design on module level, and concept integration [GZD+08b, p. 1277]. During the conceptual design phase, the aim is to specify the principle solution of the system to be developed. Within the principle solution, fundamental decisions

concerning the physical structures and the logical modes of operation of the system are made. By omitting irrelevant details, it portrays a holistic overall system design using a vocabulary that spans across the boundaries of technical domains. Hence, a common understanding about the system to be developed is enabled.

The basis for the fundamental understanding of mechatronics during the conceptual design phase is a common vocabulary. While a developer conceives his/her design using construction drawings, the other thinks in circuitry diagrams, and the third within lines of code, a common technical language spanning all engineering domains is mandatory. Such a common language has to be used for describing the principle solution, which is the result of the conceptual design phase. It has to describe the components from various domains the system consists of, and thus the interfaces among them. Such is the domain-spanning principle solution for the system to be developed.

The principle solution forms the basis for the subsequent concretization [GFL+07a] [GGS+07]. On the basis of this jointly developed principle solution, further concretization takes place in parallel in the domains of mechanics, electric/electronics, control engineering, and software engineering. These domains make use of well established methods, for instance, [PBF+07] with respect to mechanics, [BGH+93] with respect to electric/electronics, [Föl08] with respect to control engineering, and [Som06] with respect to software engineering. It is indispensable to transfer all design concepts formulated in the principle solution for the deployment of these different domains without any information loss. At this point, clear design goals have to be understood by the specialists of different domains and sufficient system information must be available as prerequisites before concretization in the respective domains could be continued. During the concretization phase, for instance, mechanical engineers analyze the kinematics and dynamics of the system, electronic engineers design the printed circuit boards, software engineers develop the software components, and control engineers develop the different controllers. The outcomes of the concretization phase consist of the validated CAD drawings, schematic diagrams, block diagrams, UML diagrams, etc. A number of specialized tools are used during the concretization phase.

- **Mechanical engineering:** There are numerous CAD/CAM/CAE softwares for designing mechanical systems. For instance, CATIA can be used for design (CAD), manufacturing (CAM), and analysis (CAE). Other well known softwares are ADAMS, Unigraphics, etc.
- **Electrical/electronic engineering:** For instance, EAGLE is used to design an electronic schematic and lay out a printed circuit board (PCB).

The other tools for board-level design are CADSTAR, Cadence Allegro, etc.

- **Control engineering:** MATLAB/Simulink/Stateflow is the de facto tool for modelling, simulation and prototyping. The other comparable tools are, for instance, CAMeL-View.
- **Software engineering:** The Unified Modeling Language (UML) is the industry standard for modeling software-intensive systems. The UML tools are, for instance, Eclipse, Fujaba, Telelogic Rhapsody, etc.

In the course of the concretization phase, the engineers of different domains work in parallel. The principle solution continues to serve as the basis of communication and cooperation between the engineers of different areas of expertise. Finally, during the system integration phase, the outcomes from the individual domains are integrated to form an overall system.

2.5 Problem Definition

Facing the paradigm shift from mechatronics to self-optimization, there is a rising concern on whether the design methodology of mechanical engineering have to be fundamentally extended, particularly during the initial phases: “planning and clarifying the task” as well as “conceptual design” [GZD+08a] [GFD+08b]. In this context, it has emerged that the basic structure of the design methodology of mechanical engineering (formulating requirements, defining functions, searching for active principles to fulfil those functions, etc.) also applies to mechatronic and self-optimizing systems [PBF+07]. However, a deeper investigation reveals that the design methodology is insufficient in dealing with the new aspects of self-optimizing systems. For instance, the integrative use of solution patterns and the need to model the environment, application scenarios, and the complex system of objectives. As such, the classical design methodology has to be adequately supported by methods and tools for their effective implementation in face of this paradigm shift.

This work focuses on two interrelated aspects within the early development phases of advanced mechatronic systems. The first aspect deals with the specification of control concepts within the domain-spanning principle solution of advanced mechatronic systems. The second aspect deals with the management of information extraction from the domain-spanning principle solution for the domain-specific controller design of advanced mechatronic systems. Both aspects have to be addressed in transition from the conceptual design phase towards the controller design phase of advanced mechatronic systems.

2.5.1 Specifying the Basic Control Concepts within the Principle Solution of Advanced Mechatronic Systems

In current practices, despite having specification techniques for the conceptual design of advanced mechatronic systems, the basic concepts of the various domains that have to be taken into consideration when specifying the principle solution are yet to be defined. From the viewpoint of control engineering, the following problems are identified during the conceptual design of advanced mechatronic systems.

Undefined basic control concepts for the conceptual design of advanced mechatronic systems. It is still ambiguous how the ideas from the domain of control engineering will receive equal treatment as per their counterparts during the conceptual design of advanced mechatronic systems. This is due to the fact that the basic concepts of control engineering that have to be taken into consideration when specifying the principle solution are undefined. Only by allowing the articulation of the basic concepts from every domains among the engineers of diverse backgrounds can there be an equal treatment on these concepts when specifying the principle solution of advanced mechatronic systems.

Insufficient guidance for the specification of control concepts within the principle solution. At the moment, the specification of the basic control concepts of advanced mechatronic systems within the various partial models of the principle solution is still insufficiently guided. An approach to point out how this can be done is still lacking. As such, the role of the principle solution as the starting point for the controller design of advanced mechatronic systems is still to be honed.

2.5.2 Managing the Extraction of Information from the Principle Solution of Advanced Mechatronic Systems

Having specified the principle solution, the design concepts have to be transferred from the principle solution into the respective domains for further concretization. As the development progresses from the domain-spanning conceptual design phase towards the domain-specific controller design phase, discontinuity of development flow arises due to the different points of view, the different approaches, the different specification techniques and the different degrees of granularity involved in the two phases. Due to the factors stated above, the synergistic impacts of specifying the principle solution may not be sustained beyond the conceptual design phase.

Different points of view: During the conceptual design phase of advanced mechatronic systems, a holistic point of view is taken which spans across the domains of mechanics, electric/electronics, control technology and software

engineering. Such a holistic point of view emphasises that there is usually no single correct design for a development order. Instead, there are several alternatives that can be conceptualized, developed, and implemented. These solutions differ depending on the purposes a system is to serve, as well as the values of the stakeholders. In this context, stakeholders refer to the clients, developers, and users who have a stake in the solutions. On the contrary, the point of view taken during the design of the controllers is rather domain-specific. Within this phase, one of the alternative solutions is further concretized in the domain of control engineering. As such, the specific aspects of system behavior, their characteristics, and their controller design are concretized. These differences in the point of view have to be aligned as the development progresses from the conceptual design towards the concretization of controller design. It is not assured that the principle solution specified during the conceptual design will meet the expectations of control engineers during the concretization phase.

Different approaches: Conceptual design and controller design are carried out in contradictory progression flows [GFL+07b]. During the conceptual design phase, engineers formulate the domain-spanning principle solution following a top-down approach. This approach is useful during the conceptual design especially when decomposing the principle solution from the system level into the module level. On the contrary, during the controller design phase, control engineers adopt a bottom-up approach. This approach is useful for the control engineers as the design of the superimposing control loop is directly dependent on the characteristics of the underlying control loop. There is currently no approach to manage the interdependencies between these different approaches used by the engineers in the two separate phases. As a consequence, discontinuity may arise as the development of an advanced mechatronic system progresses from the conceptual design phase towards the concretization phase.

Different specification techniques: Due to its multidisciplinary nature, various types of specification techniques are involved along the development of advanced mechatronic systems. Within the conceptual design phase, a set of semi-formal specification techniques was developed by FRANK et al to describe the principle solution of self-optimizing mechatronic systems. This involves the integrative use of solution patterns. During the controller design phase, the block diagram is the standard specification technique used by the control engineers nowadays. In contrast to the semi-formal specification of the principle solution, the specification of the controller design is rigorous and strictly formal. The differences in syntax and semantics between the two specification techniques are yet to be addressed. Furthermore, an engineer who is familiar with the specification technique for describing the principle solution may not be familiar with the specification of block diagrams, and vice versa.

Different degrees of granularity: Granularity refers to the level of detail. As described earlier, the outcome of conceptual design is the principle solution while the outcome of controller design is the validated block diagram. Serving as the platform for interdisciplinary technical communication, the emphasis during the conceptual design phase is the intuitiveness of the principle solution. As such, the principle solution must be easily understandable by the engineers of diverse backgrounds. Any unnecessary details should be avoided in the principle solution. Such is the degree of granularity during the conceptual design phase. On the contrary, rigorous mathematical formulation of the control solution is a must rather than a preference. The block diagrams for controller design are drawn based on the differential equations governing the system to be controlled. It involves modelling and the subsequent numerical simulation, analysis and synthesis based on these block diagrams. Comparing the degree of granularity between the principle solution and the block diagram, the principle solution is coarse-grained while the block diagram is fine-grained. It is difficult to guarantee a one-to-one transformation between the principle solution and the block diagram. As such, the control concepts specified within the principle solution have to be first identified and then concretized.

2.6 Requirements

With reference to the problems defined above, the following requirements to be fulfilled when managing the transition from the principle solution towards the controller design of advanced mechatronic systems are identified.

R1 ― A Holistic Principle Solution as a Starting Point for Concretization

The method should point out how the domain-spanning specification of the principle solution can serve as a starting point for domain-specific concretization of controller design.

R2 ― Equal Treatment on the Basic Concepts from Different Domains

The method should point out the basic control concepts so that they can be treated equally along with their counterparts of the other domains of mechatronics during the conceptual design phase. These basic concepts from the domain of control engineering must be easily interpretable by the engineers of diverse backgrounds. Only by allowing the engineers to articulate the basic concepts from different domains, can there be an equivalent treatment on the different domains when specifying the principle solution of advanced mechatronic systems.

R3 – Systematic Extraction of Information from the Principle Solution

Systematic structuring of the activities involved during the transition from the domain-spanning conceptual design phase towards the concretization phase in the domain of control engineering is essential. Such a structuring should allow stepwise transition from the principle solution towards the controller design for the advanced mechatronic systems to be developed.

R4 – Integrating Top-Down and Bottom-Up Approaches

Along the development of advanced mechatronic systems, different approaches are deployed during the conceptual design phase and the concretization phase. The method should bridge the gap between the top-down approach deployed when specifying the principle solution and the bottom-up approach deployed for the controller design of advanced mechatronic systems.

R5 – Linking Semi-Formal and Formal Specifications

Different kinds of specification techniques are deployed during the conceptual design phase and the concretization phase of advanced mechatronic systems. The method should bridge the gap between the easily interpretable concepts represented by the semi-formal specification of the principle solution with the precise designs represented by the formal specification of the controller design.

3 State-of-the-Art

Despite the relatively young history of mechatronics, numerous design methodologies and specification techniques for mechatronic systems can be found. This chapter reviews the existing domain-spanning design methodologies and specification techniques for the development of advanced mechatronic systems as well as the domain-specific design methodologies and specification techniques for the controller design of such systems. These design methodologies and specification techniques are evaluated against the requirements defined in Section 2.6. Based on the evaluations, the call for action addressing the urgency for research is described at the end of this chapter.

3.1 Domain- Spanning Design Methodologies for Mechatronic Systems

In this section, the established design methodologies for mechatronic systems are summarized. The focus here is the development of the second category of mechatronic systems which deals with the controlled movements of multi-body systems. Only the design methodologies which span across the different domains of engineering are presented here.

3.1.1 Axiomatic Design

Axiomatic Design developed by SUH is a theory for the development of various kinds of systems such as mechanical systems, software systems, or mechatronic systems. The design methodology gets its name from its use of design principles or design axioms governing the analysis and decision making process, which enable him to derive modules and determine the ideal solution concept of high quality product or system designs. As shown in Figure 3-1, the four domains of the axiomatic design framework are: customer, function, physics and process [Suh01, p. 10]. SUH formally models the domains and the relationships between them.

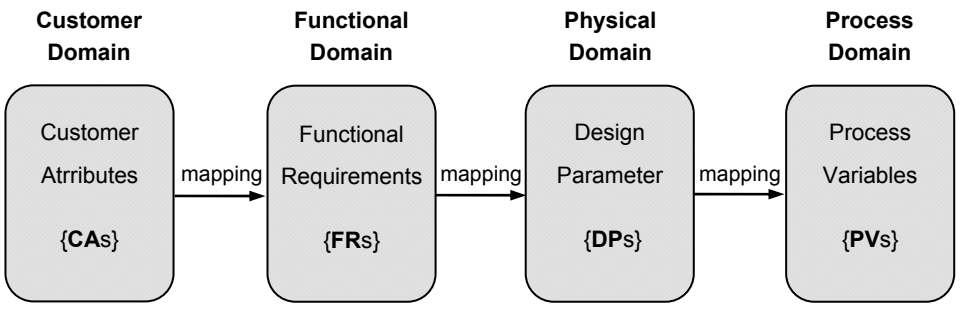


Figure 3-1: The domains of the axiomatic design framework [Suh01, p. 11]

The steps involved in Axiomatic design can be summarized as follows: systematically analyzes the transformation of customer needs or attributes (CAs) into functional requirements (FRs), design parameters (DPs), and process variables (PVs). The designers first break up customer needs into functional requirements (FRs), then break up these requirements into design parameters (DPs), and then finally figure out a process to produce those design parameters. In another word, axiomatic design is a decomposition process going from customer needs to functional requirements (FRs), to design parameters (DPs), and then to process variables (PVs), thereby crossing the four domains mentioned above: customer, function, physics, and process.

The domain customer specifies the requirements of the system. In the domain function, these requirements are concretized into functional requirements and constraints. The functional requirements correspond to the functions of the system. In the domain physics, these functional requirements are transformed into design parameters. Design parameters describe the physical characteristics of the system to be developed. The domain process describes the manufacturing process of the system by means of process parameters. The parameters involved in a domain are graphically represented by hierarchical trees and mathematically represented by vectors. The transition from one domain into another is specified by design matrices [Suh01, p. 18]. Design matrices determine how the variables of a domain are transformed into the variables of another domain.

Along the way, two basic axioms are taken into consideration: the independence axiom and the information axiom. The first axiom says that the functional requirements within a good design are independent of each other. The second axiom says that when two or more alternative designs satisfy the first axiom, the best design is the one with the least information. Application of axiomatic design as exemplified by mechanical systems, software systems and control systems can be found in [Suh95], [SD00] and [LSO01].

Evaluation

The advantage of axiomatic design is its general applicability to the diverse kinds of systems, including the mechatronic systems. The approach systematically transforms the customer needs at the one end towards the production concepts at the other end. With the help of design matrices, the product data can be represented not only graphically but also mathematically. However, the axiomatic design theory has to be applied with slight variations for the development of advanced mechatronic systems. Axiomatic design distinguishes between the customer domain, functional domain, physical domain and process domain. Such an approach is not customized for the development of mechatronic systems as the synergistic integration of mechanics, electric/electronics, control

engineering and software engineering. Furthermore, issues concerning the different points of view, different approaches and different specification techniques used during the domain-spanning conceptual design phase and the domain-specific concretization phase of advanced mechatronic systems are not explicitly addressed.

3.1.2 The SYSMOD Approach

The SYSMOD approach [Wei07] is a modeling approach for the development of systems. It is used in combination with the specification technique SysML. The SYSMOD procedure consists of a procedural model for analysis and another procedural model for design. The procedural models are shown in Figure 3-2 and Figure 3-3 respectively. The activities of the two procedural models are described in the following.

Determine requirements: The determination of requirements involves the description of system idea and objectives, the identification of stakeholder, and the collection of requirements. For this purpose, the system idea and the basic objectives to be achieved by the system are described. Besides that, all persons and institutions who or which have a stake on the requirements or an interest on the system are identified. Furthermore, the stakeholders are inquired about the requirements to be fulfilled by the system to be developed.

Model system context: The modeling of system context involves the identification of system actors, the modeling of system/actor information flow, and the identification of system interaction points. For this purpose, all persons and systems that directly interact with the system to be developed are identified. Besides that, information that the system shares with its surroundings is described. Furthermore, the points of the system where the information exchange with the environment takes place are described.

Model use cases: The modeling of use cases involves several activities. Identification of use cases refers to the identification of services which offered by the system to the actors. Description of use case essences refers to the description of the specific intention of the use cases in the form of essential steps, which technical details and specific processes are not taken into account. Description of system processes refers to the description of the timing dependencies between uses cases and summary of the related processes in system processes. Modeling of use cases without redundancies refers to the identification of the commonalities between the processes of use cases and modeling of the area by means of isolation in order to avoid redundancies. Modeling of use case flows refers to the description of the processes of the use cases with all exceptions and variations in a reasonable detail. Modeling of object flows refers to the

description of the incoming and outgoing data of each use cases and modeling of their dependences.

Model domain knowledge: This involves the modeling of the structure regarding the specific terms of the system.

Create glossary: Here the technical terms associated with the system are described.

Realize use cases: Realization of use cases involves the modeling of system/actor interaction, derivation of system interfaces, and modeling of system structures. During this phase, the interactions between the system and actors relating to the use cases are described. Besides that, the interfaces of the system with the actors in relation to the various interaction points are described. Furthermore, system components and their composition, which are necessary for the entire system in order to meet the requirements, are modeled.

Evaluation

The SYSMOD approach is generally applicable for the development of all kinds of systems. It consists of two approach models for the analysis and design of systems. Both approaches can be applied for the development of advanced mechatronic systems. Nevertheless, the development flow from conceptual design, towards concretization, and finally system integration is not explicitly addressed. It is not clear how the basic concepts from the different domains of mechatronics can be intuitively specified and equally treated during the conceptual design phase. Furthermore, the SYSMOD Procedure does not address its interdependencies with the established methodologies of the respective domains of mechatronic, for instance, the established methodology for controller design.

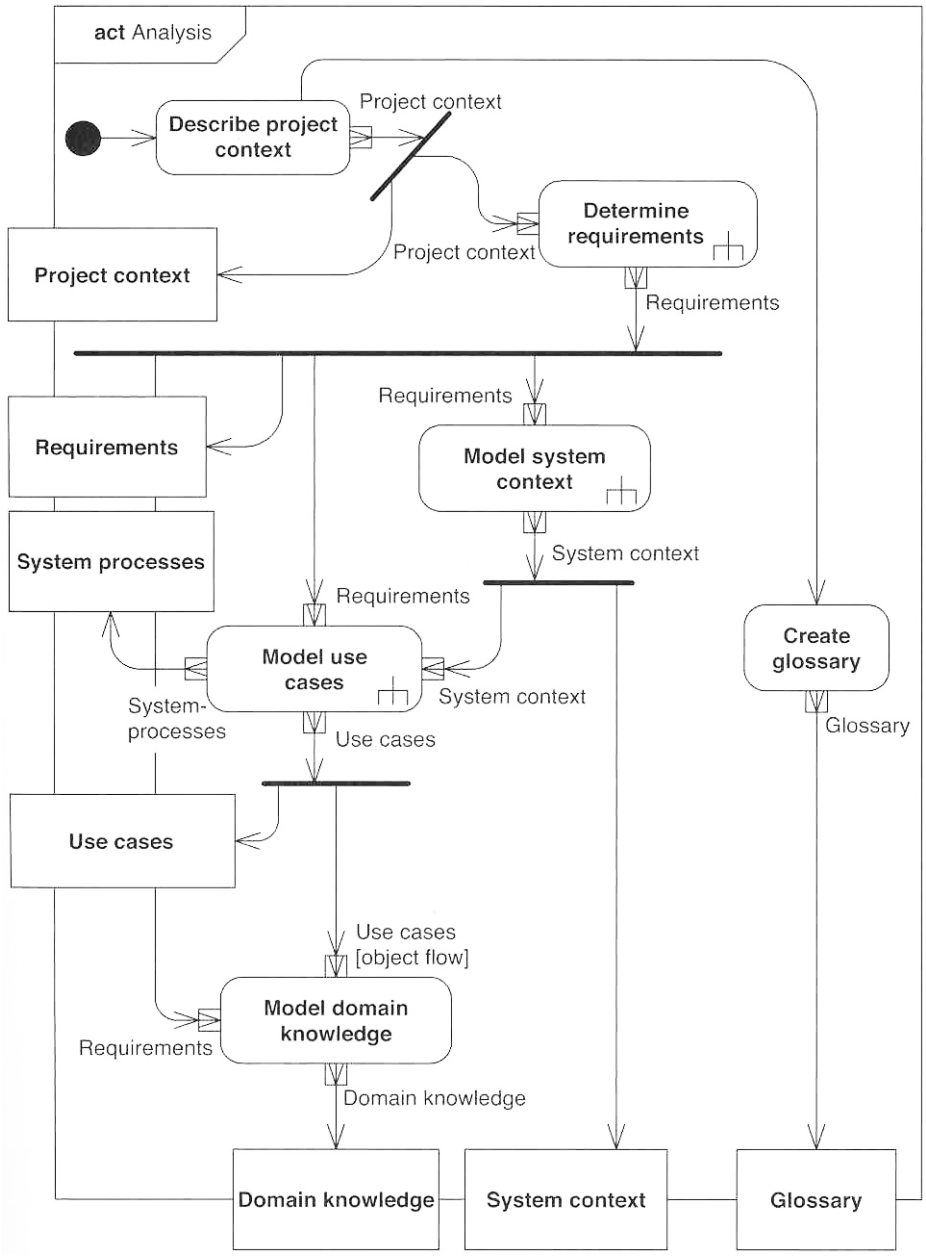


Figure 3-2: The approach model for analysis [Wei07, p. 25]

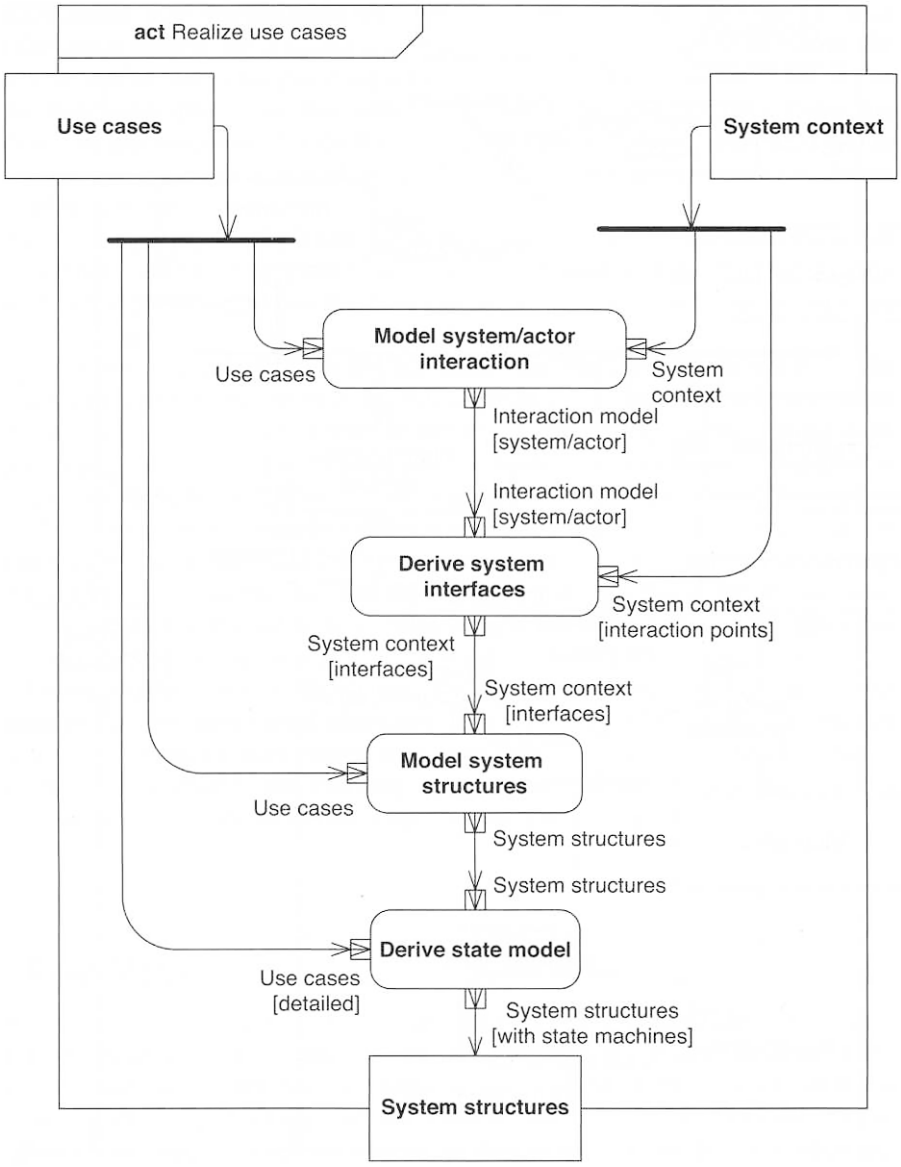


Figure 3-3: The approach model for designs [Wei07, p. 26]

3.1.3 VDI-Guideline 2206: Design Methodology for Mechatronic Systems

The VDI-Guideline 2206 "Development Methodology for Mechatronic Systems" is a universal cross-domain guideline intended to describe the methods of developing mechatronic systems. As stated in the guideline, both the experiences of industrial practice and the results of empirical design research from recent years make it clear that there is no “canonizable optimal form of the design process which the designer can follow in a fixed schedule” [Dör98]. In order to allow for this realization also in the development of mechatronic systems, a more flexible procedural model is proposed in VDI 2206, which is supported essentially on three elements [VDI2206, p. 26]:

- general problem-solving cycle on the micro-level
- V model on the macro-level
- predefined process module for the handling of recurrent working steps in the development of mechatronic systems.

1. Problem-solving cycle as a micro-cycle: The structuring of the procedure in the development process takes place in this case on the basis of a general problem-solving cycle, such as that known for example from systems engineering [DH02, p. 47]. By arranging procedural cycles in series and one within the other, process planning can flexibly adapted to the peculiarities of any development task. The presented micro-cycle is intended in particular to support the product developer engaged in the process to work on predictable, and consequently plannable, subtasks, but also to solve suddenly occurring, unforeseeable problems.

2. The V model as a macro-cycle: A guide for the basic procedure is offered by the V model adopted from software development and adapted to the requirements of mechatronics; it describes the logical sequence of important substeps in the development of mechatronic systems. When using this model in practice, it must be taken into account that the time sequence of the substeps may deviate from the logical sequence: for example, to minimize the development risk, it may be advisable to bring critical systems almost up to readiness for mass production before commencing development of the complex overall system dependent on it. The V model is shown in Figure 3-4.

Requirements: The starting point is formed by an actual development order. The defined object was specified more precisely and described in the form of requirements. These requirements at the same time form the measure against which the later product is to be assessed.

System Design: The aim is to establish a cross-domain solution concept which describes the main physical and logical operating characteristics of the future product. For this purpose, the overall function of a system is broken down into main subfunctions. These subfunctions are assigned suitable operating principles or solution elements and the performance of the function is tested in the context of the system.

Domain-Specific Design: On the basis of this jointly developed solution concept, further concretization usually takes place separately in the domains involved. More detailed interpretations and calculations are necessary to ensure the performance of the function, in particular in the case of critical functions.

System integration: The results from the individual domains are integrated to form an overall system, to allow the interaction to be investigated.

Assurance of properties: The progress made with the design must be continually checked on the basis of the specified solution concept and the requirements. It must be ensured that the actual system properties coincide with the desired system properties.

Modeling and model analysis: The phases described are flanked by the forming and investigating of the system properties with the aid of models and computer-aided tools for simulation.

Product: The result of a continuous macro-cycle is the product. In this case, a product is understood as meaning not exclusively the finished, actually existing product but the increasing concretization of the future product (product maturity). Degrees of maturity are, for example, the laboratory specimen, the functional specimen, the pilot-run product, etc.

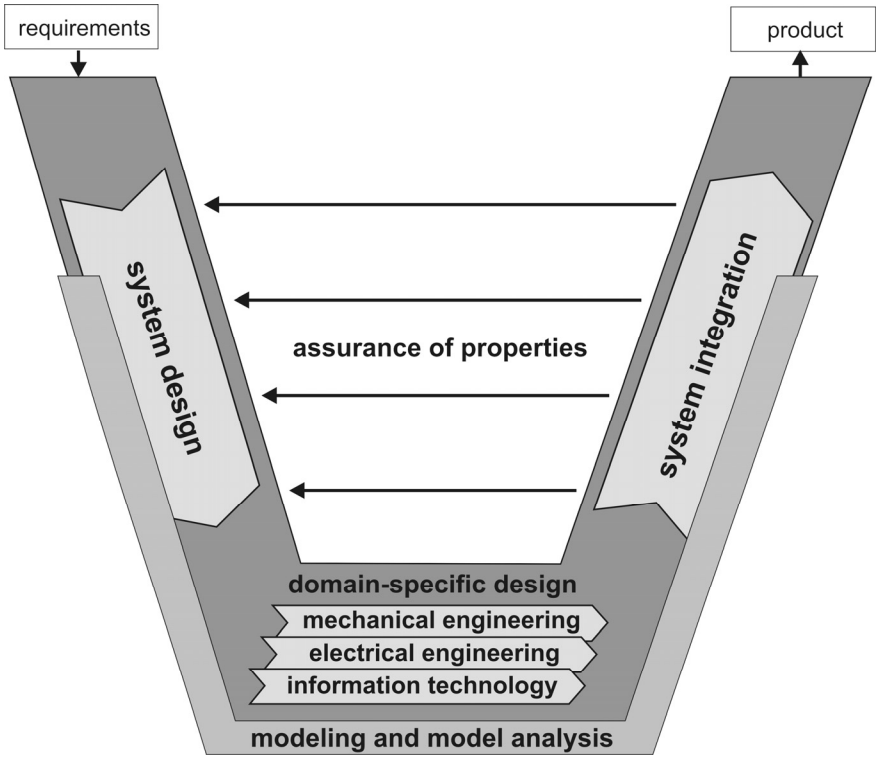


Figure 3-4: V model as a macro-cycle [VDI2206, p. 29]

3. Process modules for recurrent working steps: The handling of individual substeps of the process planning worked out on the basis of the V model is governed by the already mentioned problem-solving cycle. However, for some mechatronic systems that are in development for recurring defined tasks, handling can be described in more concrete terms in the form of partly predefined process modules. In this guideline, process modules for system design, modelling and model analysis, domain-specific design, system integration and assurance of properties are described.

Evaluation

The VDI-Guideline 2206 was established under the contribution of the Heinz Nixdorf Institute. It portrays the current consensus of the experts practising in the field of mechatronics and hence serves as a first step on the way to a comprehensive design methodology for mechatronic systems. Being a practical guideline, it focuses on the applicability for general mechatronic systems to allow product-specific and enterprise-specific adaptation. As far as the scope of this work is concerned, the guideline is yet to be comprehended by a specification technique customized for the conceptual design of mechatronic systems. Besides that, during the domain-specific design, the domain of control engineering which involves extensive modeling and model analysis is not explicitly pointed out in the V model. Furthermore, the handling of the transition from the system design towards domain-specific design is not explicitly addressed in the guideline. Last but not least, none of the application examples showcase the emerging capability of advanced mechatronic systems such as self-optimization.

3.1.4 3-Level Procedural Model according to BENDER

BENDER concretized the V model of the VDI-Guideline 2206. The concretized V model is called the 3-Level Procedural Model, as shown in Figure 3-5. The procedural model was designed especially for the development of embedded systems [Ben05, p. 44]. The procedural model classifies the development phases into the system level, subsystem level and component level.

On the **system level**, questions concerning the overall system are of interest. Starting from the requirements, the logical architecture of the system is first developed. It serves as the basis to decompose the overall system into subsystems. On the **subsystem level**, the requirements of each subsystem are analyzed once more. The interdependencies between the subsystems should be kept as marginal as possible. Besides that, the subsystems should belong to a single domain: mechanical, software or electronics (hardware). Thereby it is easier to independently concretize the subsystems. On this level, the subsystems of software and electronics (hardware) are considered to be part of the subsystem IT. The subsystems are now subdivided into components. On the **component level,** the product is effectively realized. On this level, the components are developed in detail based on the division of labour. After that, the components are integrated and tested step-by-step, as shown on the ascending bough of the 3-Level Procedural Model. Before the IT integration is carried out, the potential strong interdependencies between the hardware and the software components necessitate an early risk analysis. This is done by taken into account the integration and test of both the software and the hardware.

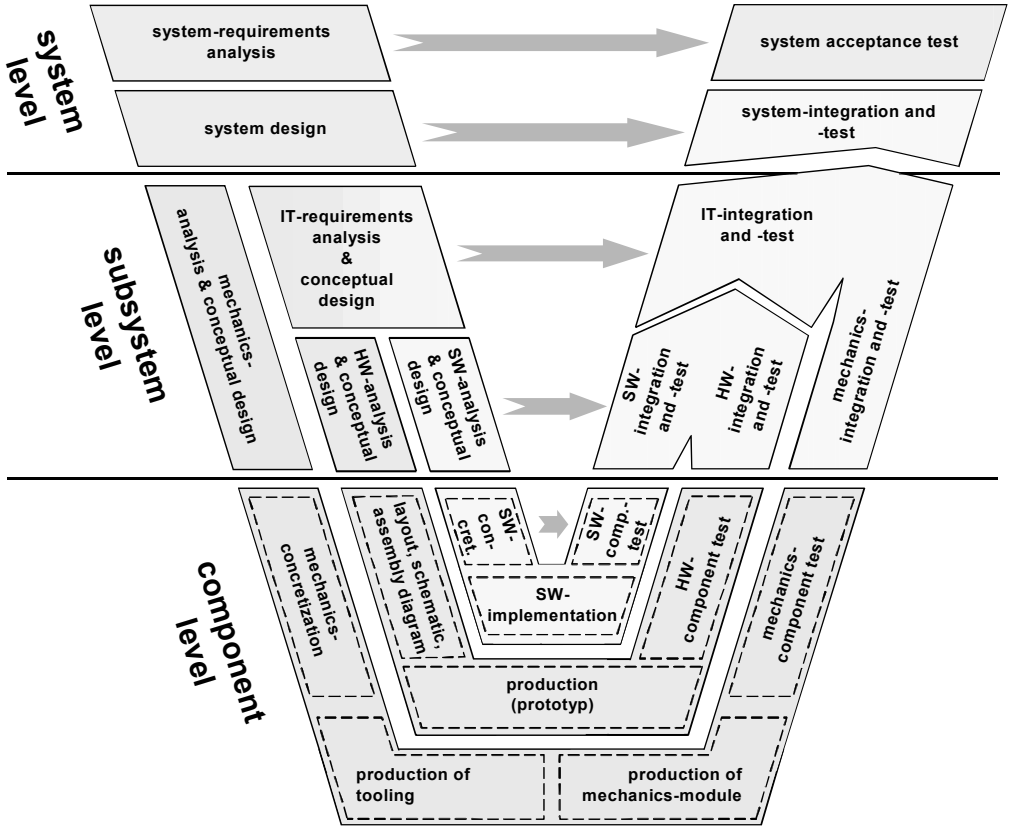


Figure 3-5: 3-Level Procedural Model by BENDER [Ben05, p. 45]

The domain-spanning phases on the system and subsystem levels provide the points of synchronization between the different domains. There are two different kinds of synchronization, i.e. **functional/technical synchronizations** and **organizational synchronizations**. On the right bough of the 3-Level Procedural Model, functional/technical synchronizations are required during the respective integration phases. At these points of synchronization, subsystems which realize specific functionalities are integrated. On the left bough of the 3-Level Procedural Model, organizational synchronizations are required during the respective conceptual phases. Besides that, there are dependences between the functional/technical synchronizations and the organizational synchronizations.

Evaluation

The 3-Level Procedural Model is more detail than the V model of the VDI-Guideline 2206. An edge of the 3-Level Procedural Model is the division of the V model into the levels of system, subsystem and component. Such a classification reduces the complexity of development. However, such structure does not include the networked mechatronic systems, which is one level higher than the system level. Similar as the V model of the VDI-Guideline 2206, besides the domains of software, electronic hardware and mechanics, the domain of control engineering is not pointed out in the 3-Level Procedural Model. As

such, it is not clear when and how the concepts of control engineering should be integrated into the procedural model. The potential inconsistencies due to the different points of view, different approaches, and different specification techniques used from one level into another level across the various domains are not explicitly addressed.

3.1.5 Methodology for Mechatronic Design according to LÜCKEL

The Institute of Control Engineering and Mechatronics (RtM) at the University of Paderborn has developed a methodology for mechatronic design [LKS00], as shown in Figure 3-6. The methodology is developed based on the classical design methodology for mechanical systems according to PAHL and BEITZ [PBF+07]. According to the methodology, product planning and clarification of task has to be done upon receiving the development task. After that, LÜCKEL transcends the classical design methodology by adding another design step ─ the mechatronic composition. The mechatronic composition consists of three steps: modeling, analysis, and synthesis.

Modeling: This step involves the computer-aided representation of the main physical features of the systems by means of a model. Generally, the entities which have to be modeled are the plant, the control algorithms, and the environment of the plant. The focus here is the motional functions of the system that can be analyzed by means of a model. For this purpose, software tools such as CAMeL-View (Computer Aided Mechatronics Laboratory ─ Visual Engineering Workbench) can be used.

Analysis: This step involves the deployment of computer-aided methods to analysis the behavior of motion of the model. As such, engineers can investigate whether the system fulfills the desired requirements or not.

Synthesis: With reference to the simulation results obtained from the previous step, the system will be improved and the model will be adapted accordingly. If necessary, measurements results on subcomponents can be taken into account. During this step, the controllers are designed in such a way to imprint a desired behavior on the model of the plant.

These steps of modeling, analysis, and synthesis are an iterative process. It has to be performed several times until the system is sufficiently optimized. A vital part of mechatronic composition is the computer-aided model-based design. The model-based design allows the minimization of iterations in cost-intensive process steps (e.g. field tests). After the mechatronic composition, computer-aided design tools can be used to assist the engineers. Now the system has to be manufactured. After passing the laboratory and field test, a finished product can be obtained.

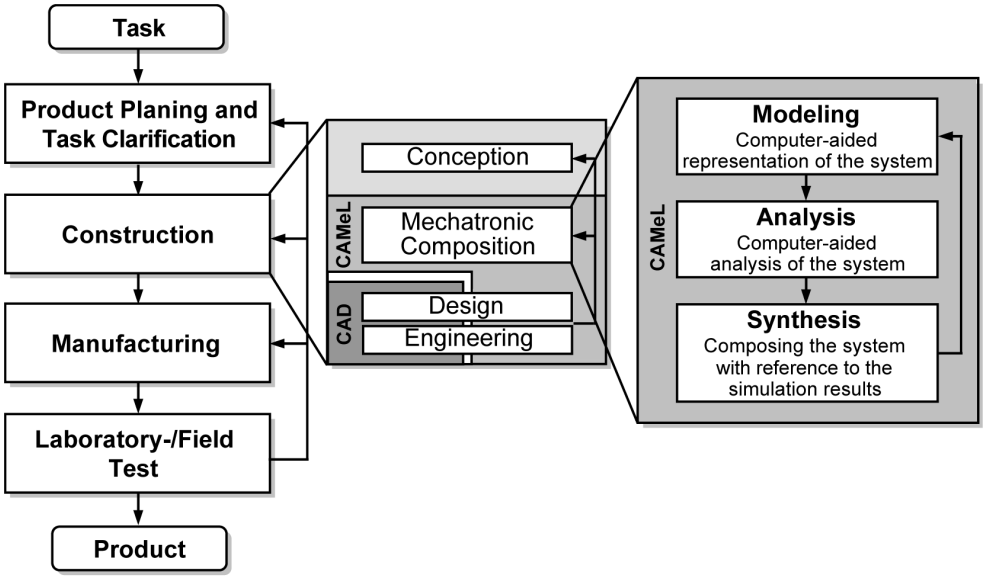


Figure 3-6: Methodology for mechatronic design according to LÜCKEL [LKS00, p. 16]

Evaluation

The mechatronic design method developed by LÜCKEL focuses on the controller design of the mechatronic system. In this context, extensive computer aided modeling and simulation methods are used for the purposes of analysis and synthesis. However, it does not sufficiently support the conceptual design of mechatronic systems at the beginning of the development. In this context, the approach does not describe how the principle solution of mechatronic systems should be specified. Besides that, the handling of the transition between the domain-spanning conceptual design and the domain-specific concretization is not clarified.

3.2 Domain-Spanning Specification Techniques for Mechatronic Systems

This section reviews the domain-spanning specification techniques for mechatronic systems. Some of the specification techniques are deployed in conjunction with a specific design methodology whereas the others are independent of any design methodology.

3.2.1 Specification Technique for Axiomatic Design

In axiomatic design, there are three different but equivalent ways of representing a system: hierarchies with corresponding design matrices, the module-junction diagram, and the flow diagram or flow chart [Suh01, p. 207] [Suh98] [Suh04]. Although all these different representations of the system architecture

are equivalent, they emphasis different aspects of the system. Figure 3-7 exemplifies the hierarchies of functional requirements and design parameters while Figure 3-8 exemplifies the module-structure diagram and the flow diagram.

As shown in Figure 3-7, the hierarchical diagram gives the entire decomposition steps and all functional requirements and design parameters. As shown at the left of Figure 3-8, the module-junction diagram is created to show the hierarchical structure of modules and their interrelationships. As shown at the right of Figure 3-8, the flow diagram illustrates the design relationships of all modules at the leaf level and the precedence of implementation based on design matrices of each level of design decomposition. The flow diagram is a concise and powerful tool that provides a comprehensive view of the system design and a road map for implementation of the system design.

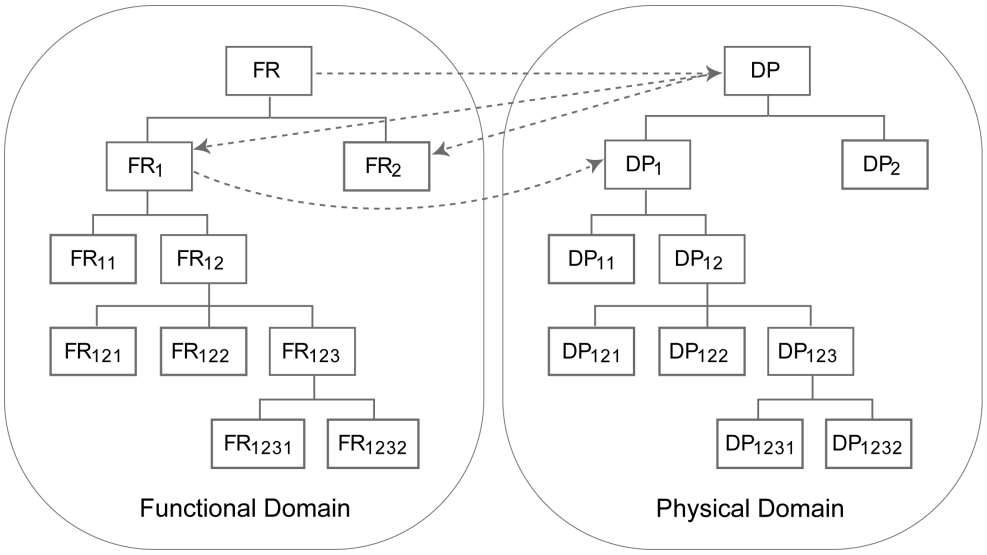


Figure 3-7: The hierarchy of functional requirements (FRs) (left) and the hierarchy of design parameters (DPs) (right) [Suh01, p. 30]

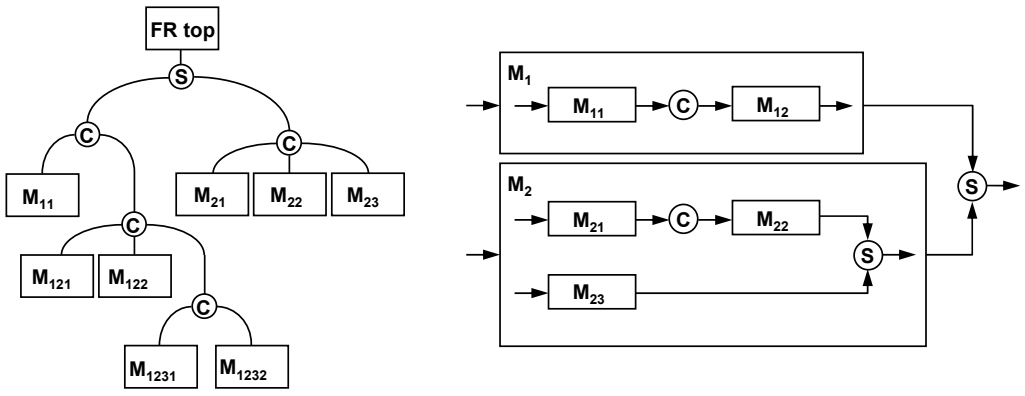


Figure 3-8: The module-junction diagram (left) and the flow chart (right) used in axiomatic design [Suh01, p. 211 and p. 212]

Evaluation

The specification technique for axiomatic design does not sufficiently support the development of advanced mechatronic systems, especially those with self-optimizing capability. It is not clear how the key aspects of self-optimizing systems such as the adaptation of the system objectives, the behavioral adaptation, and the required structural reconfiguration or parameter adjustment can be specified. As such, the interdependencies between the adaptation of system objectives and the behavioral adaptation of the system cannot be specified.

3.2.2 Systems Modeling Language (SysML)

The aim of the Systems Modeling Language (SysML) is to provide a language that supports the systems engineer [OMG03] [Sys04] [OMG07]. SysML is a new visual modeling language for the development of systems. The version V1.0 is available since 2007. SysML is based on UML 2.0. As such, it uses the UML constructs, with some modifications and some extensions. SysML customizes the UML for systems engineering applications. These systems may include hardware, software, information, processes, personnel, and facilities. SysML supports the specification, analysis, design, verification and validation of a broad range of systems and systems-of-systems. The SysML diagrams can be classified into requirement diagram, behavior diagrams, structure diagrams and parametric diagram [Hau01]. Figure 3-9 exemplifies these different diagrams with an antilock braking system.

The **requirement diagram** captures requirements hierarchies and the derivation, satisfaction, verification and refinement relationships. The relationships provide the capability to relate requirements to one another and to relate requirements to system design models and test cases. The requirement diagram provides a bridge between typical requirements management tools and the system models.

The **behavior diagrams** include the **use-case diagram**, **activity diagram**, **sequence diagram** and **state machine diagram**. A use-case diagram provides a high-level description of the system functionality. The activity diagram represents the flow of data and control between activities. A sequence diagram represents the interaction between collaborating parts of a system. The state machine diagram describes the state transitions and actions that a system or its parts performs in response to events.

The **system structure** is represented by **block definition diagrams** and **internal block diagrams**. A block definition diagram describes the system hierarchy and system/component classifications. The internal block diagram de-

scribes the internal structure of a system in terms of its parts, ports, and connectors. The **package diagram** is used to organize the model.

The **parametric diagram** represents constraints on system parameter values such as performance, reliability and mass properties to support engineering analysis. SysML includes an allocation relationship to represent various types of allocation including allocation of functions to components, logical to physical components and software to hardware.

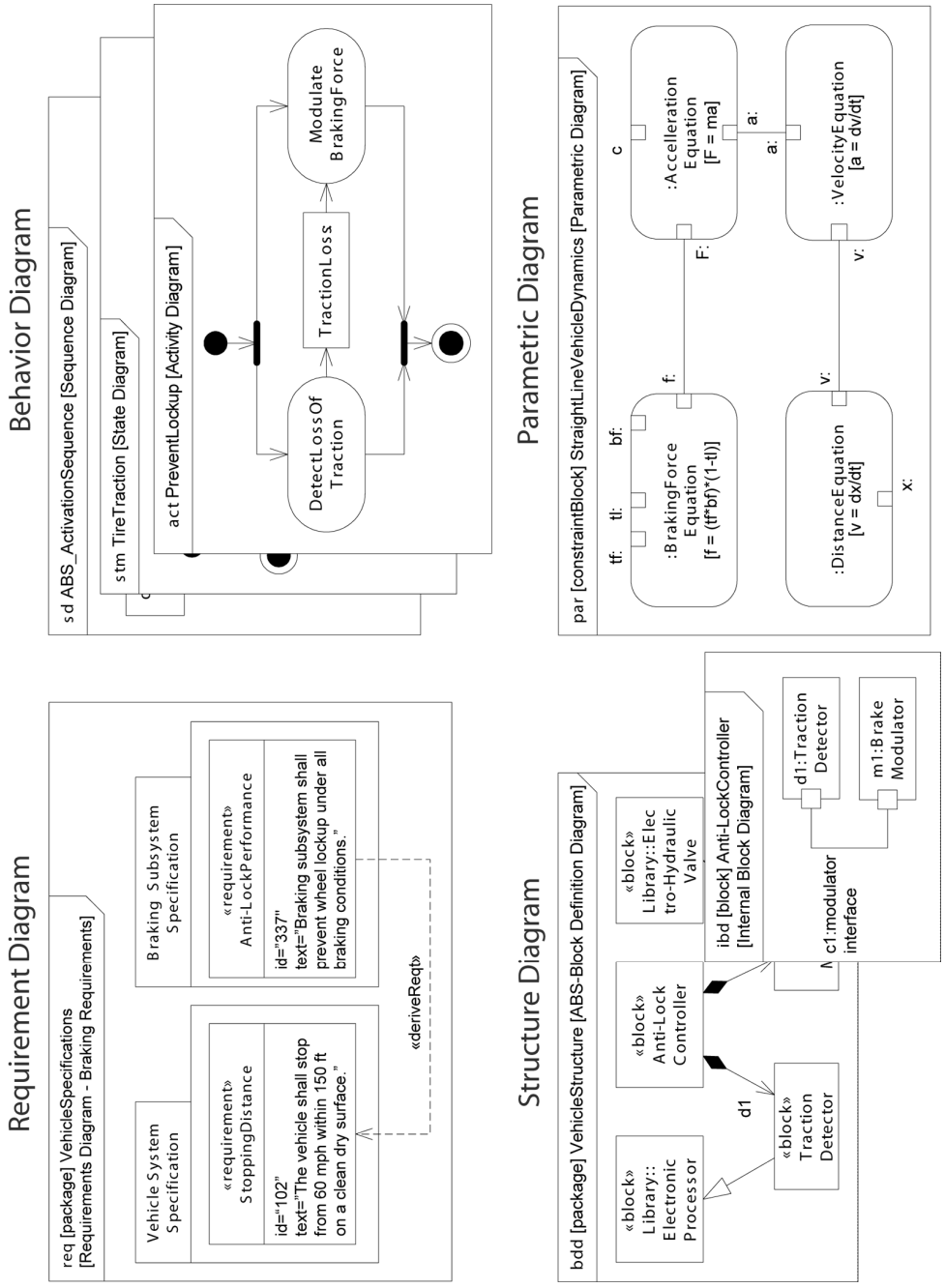


Figure 3-9: The diagrams of SysML for specifying the system requirements, behavior, structure and parametric relationships [Hau01]

Evaluation

SysML is a visual modeling language that provides semantics and their notations for systems engineering. Since the UML is widely accepted, an advantage for SysML is that the UML tools and trainings are easily available. Besides that, system engineers using SysML can work together with the software engineers using UML in an efficient way. SysML is not customized for advanced mechatronic systems. The basic control concepts for advanced mechatronic systems are not described. An approach to show how controller design can be started base on the specification of SysML is also lacking.

3.2.3 Function-Oriented Specification of Mechatronic Systems according to BUUR

In engineering design, a function is an intended input/output relationship of a system whose purpose is to perform a task [PBF+07, p. 31]. Technical products can be developed with respect to the functions of the products. Using function-oriented approaches for product development is the current trend in enterprises developing technical systems with a high proportion of electronics and software components. The function-oriented approaches are exemplified by BUUR's work.

Main emphasis of BUUR's work is on the modeling of mechatronic systems by functions, in dependence of the system's current state. According to him, an entire function specification consists of a description of the states and of the transition states of the system as well as of transformation functions and also purpose functions. Those transformation functions describe the converting and transferring of energy, material and information by the system. The so-called purpose functions make necessary effects available so that the system can carry out the required transformations. The active transformation and purpose functions are assigned to the current state [Buu89], [Buu90]. Figure 3-10 exemplifies the function-oriented specification of a telephone according to BUUR.

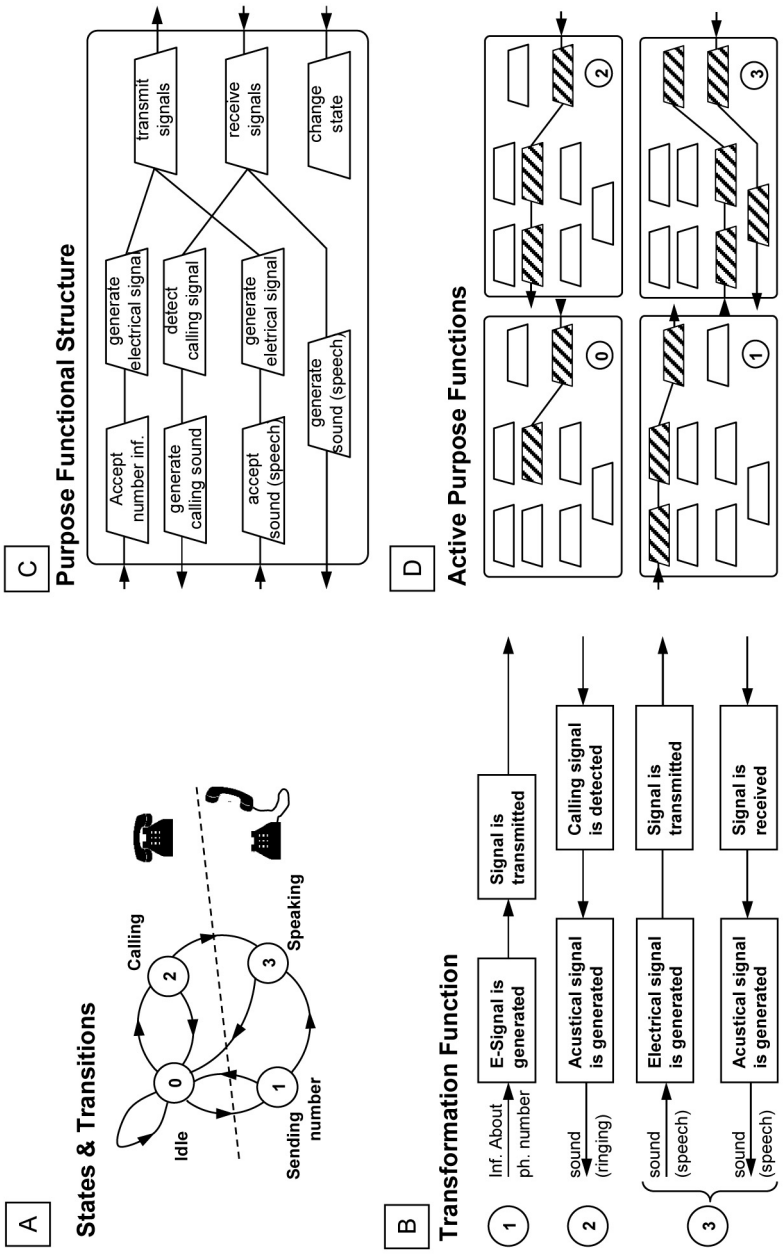


Figure 3-10: Specification of the functions of a telephone [Buu90]

Evaluation

The specification technique developed by BUUR does not sufficiently support the development of advanced mechatronic systems, especially those with self-optimizing capability. It is not clear how the changing objective of the system, its behavioral adaptation and the required structural reconfiguration or parameter adjustment can be specified. The potential inconsistencies due to the different points of view, different approaches, and different specification techniques used during the development of mechatronic systems are not explicitly addressed.

3.2.4 Specification Technique for the Principle Solution of Self-Optimizing Systems according to FRANK

Within the collaborative research centre CRC 614 “Self-Optimizing Concepts and Structures in Mechanical Engineering”, a set of specification techniques for the description of the principle solution of self-optimizing systems was developed on the work of FRANK, GAUSEMEIER, and KALLMEYER [Fra06], [GEK01] [Kal98]. For a complete description, several aspects of the advanced mechatronic system are needed. Each aspect is mapped by a computer onto a partial model. As shown in Figure 3-11, the principle solution is made up of the following aspects: requirements, environment, system of objectives, functions, active structure, shape, application scenarios and behavior. The aspect ‘behavior’ is considered as a group because there are various types of behavior (e.g. the dynamic behavior of a multibody system, the cooperative behavior of system components etc.). There are close interplay between the aspects, leading to a coherent system of partial models that represents the principle solution of advanced mechatronic systems. Such a principle solution provides the basis for the communication and cooperation between the engineers from different areas of expertise.

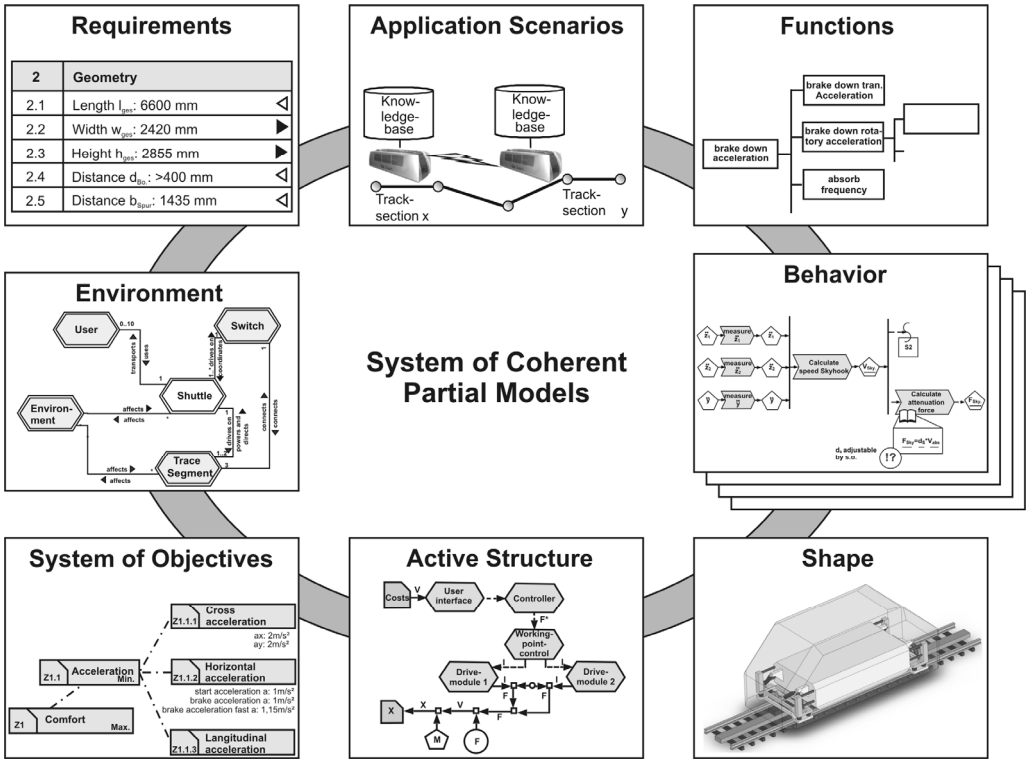


Figure 3-11: Interconnected system of partial models for the description of the principle solution of self-optimizing systems [Fra06, p. 80]

Environment: This model describes the environment of the system that has to be developed and its embedding into the environment. The relevant spheres of influence (such as weather, mechanical load, higher-level systems) and influences (such as thermal radiation, wind force, information) are identified in this model. Undesirable influences disturbing the operation of the system are marked as disturbance variables. Furthermore, the interplays between the influences will be examined. We also investigate the possibility of the concurrent occurrences of the influences. In this context, we consider a 'situation' to be a consistent set of collectively occurring influences, in which the system has to work properly. We mark influences that cause a state transition of the system as events. Catalogues, that imply the spheres of influences and the influences, can be used to support the creation of environment models.

Application scenario: Application scenarios form the first concretizations of the system. They concretize the system's behavior in a particular state and a particular situation, and even the kinds of events that initiate a certain state transitions. Application scenarios characterize a problem, which needs to be resolved in special cases, and then roughly describe the possible solution.

Requirements: This aspect considers the representation of the requirements in a computer. The list of requirements sets up its basis. It presents an organized collection of the requirements that need to be fulfilled during the product development (such as overall size, performance data) [AGK+06] [PBF+07]. Among the requirements, there is a distinction between demands and wishes. Every requirement is verbally described and, if possible, concretized by attributes and their characteristics. Checklists can be used to assist the setting up of requirements, see for example [PBF+07], [Rot00], [Ehr03].

System of objectives: This aspect includes the representation of external, inherent and internal objectives as well as their interrelations. The external and inherent objectives are represented in the form of a hierarchical tree. The hierarchical relations are specified by the logical relation "is part-objective of". The internal objectives are derived from the external and inherent objectives. An influence matrix can be used to show if the objectives can work in mutual support, or if they influence each other negatively, or if they are in a neutral relationship. In the case of the mutually supporting relation and the neutral relation, the system is able to follow simultaneously without any problems. But if the objectives influence each other in a negative way, this is an indication for the need of an optimization. Instead of an influence matrix, graphs that model objectives and their interplays can be used.

Functions: This aspect concerns the hierarchical decomposition of the system's functionality. A function is the general and required coherence between input and output parameters, aiming at fulfilling a task. For the setting up of

function hierarchies, there is a catalogue of functions which is based on BIRKHOFER [Bir80] and LANGLOTZ [Lan00]. This catalogue has been extended by the functions used for self-optimization. Functions are realized by solution patterns and their concretizations. As such, the decomposition into sub functions takes place until the useful solution patterns are found for the functions.

Active structure: The active structure describes the system elements, their attributes as well as the relations between the system elements. The aim is to define a basic structure which includes all system configurations that can be thought ahead. The system elements can be structured into logical groups in order to improve the clarity of representation. The system elements, which deal with the self-optimization process, are marked by a slanting arrow.

Shape: This aspect has to be modeled because the first definitions of the system's shape have to be carried out already during the conceptual design phase. In particular, this model concerns the working surfaces, working places, surfaces and frames. The computer-aided modeling takes place by using three dimensional CAD systems.

Behavior: This group of partial models comprises several kinds of behavior. Basically, what needed to be modeled are the system's states with their operation activities and the state transitions with their adaptation activities. The adaptation activities lead to the realization of the self-optimizing process. If there are several systems involved, the interplay of these systems needs to be described. Depending on the development task, more kinds of behavior, such as kinematics, dynamics or electro-magnetic compatibility of the system's components need to be specified.

- The partial model **Behavior – States** defines the states and state transitions of a system. All the system's states and state transitions which can be thought ahead have to be considered. This includes the descriptions of the events that trigger a state transition. Events can be characteristic influences on the system or the already finished activities.
- The partial model **Behavior – Activities** describes the aforementioned operation activities which take place in a system's state and the adaptation activities which have the typical features of self-optimization. An activity can be, for instance, determination of the fulfillment of current objectives, selection of the adequate parameters and configurations, etc.

In classical design methodology of mechanical engineering, the *active structure* is the important partial model. However, for the development of advanced mechatronic systems, the system states and the state transitions play an important role [GFS06].

Evaluation

This specification technique was developed for describing the principle solution of self-optimizing systems in a domain-spanning way. The specification technique provides a basis for effective technical communication and cooperation among the engineers of diverse backgrounds. With such a specification technique, equal treatment on the different domains during the conceptual design phase is possible. Besides that, such a specification technique ensures that the different aspects of the system are sufficiently considered, and therefore be able to serve as the starting point for the respective domain-specific concretization. Nevertheless, an approach regarding how to specify the control concepts within the principle solution is still lacking. Besides that, a systematic approach is yet to be developed to point how the right information can be identified and then extracted from the principle solution for the concretization of controller design.

3.3 Domain-Specific Design Methodologies in Control Engineering

Design methodologies in control engineering are well established. In this section, the well known DIN 19226 German Standard for Control Technology and the methodology for controller design according to FÖLLINGER are reviewed.

3.3.1 DIN 19226: German Standard for Control Technology

The DIN 19226 is the German Standard for control technology [DIN19226]. This standard contains the general principles of control technology and the terms and definitions used in control engineering. It consists of the following six parts.

Part 1 ─ General terms and definitions: First and foremost, the standard explains the field of application and the aim of the standard. After that, it describes the general terms and definitions in control engineering, which include: system, variable, vector, action, process, model, algorithm, action diagram, as well as open-loop and closed-loop control.

Part 2 ─ Terms and definitions of dynamic systems behavior: The second part of the standard focuses on the behavior of dynamic systems. The terms and definitions here cover the transfer behavior, the classification and the state description of transfer elements, stability, characteristic curve, responses on specific input variables, characteristic functions of linear time-invariant transfer elements, and last but not least, nonlinear time-invariant transfer elements.

Part 3 ─ Terms and definitions of switching systems behavior: The third part of the standard focuses on the behavior of switching systems. It first defines the terms pertaining to a switching system such as switching variable, switching function, and switching element. Subsequently, it elaborates the Boolean switching functions, storage switching functions, binary timing elements, sequential circuits and switching system for sequential control.

Part 4 ─ Terms and definitions of control systems: The fourth part of the standard covers the variables, functional units and structures of control systems. Besides that, it describes the types of the influences of the plant, the requirements on control systems, the structures of control and the control of multivariable systems.

Part 5 ─ Functional terms: The fifth part of the standard covers the operating modes, errors, variables and parameters pertaining to the open-loop and closed-loop control. Besides that, it describes the characteristics and parameters of the final controlling equipments as well as the plants. On one hand, in the context of open-loop control, the types of digital and binary control, the types of control signal and the elements of a control program are described. On the other hand, in the context of closed-loop control, the types, variables, characteristics, and parameters of the control loop are described.

Part 6 ─ Terms and definitions of functional and physical units: The last part of the standard describes the functional and physical units in control engineering. It covers the generation of variables and signals from the plants and their environment (e.g. measuring equipment) as well as the adjustment-and-transduction (e.g. transducer), input (e.g. input device), transfer (e.g. signal line), processing (e.g. arithmetic logic), control (e.g. reference variable adjuster), manipulation (e.g. actuator) and output (e.g. indicator) of variables and signals.

Evaluation

The DIN 19226 focuses on the definition of terms and the classification of components in control engineering. As such, the meaning of the technical terms used in control engineering is precisely defined and the components used are rightly classified. Nevertheless, the specific terms defined in DIN 19226 can not contribute towards the creation of a common understanding among the engineers of diverse backgrounds. On the contrary, confusion may arise if the same term is used in the other domain but carries a different meaning. Besides that, the standard did not cover the delineation of any design procedures. The conceptual design of mechatronic design is not addressed by the standard. The standard can fulfill none of the requirements.

3.3.2 Methodology for Controller Design according to FÖLLINGER

The design of classical control as per [Föl08, p. 12] is summarized in the form of a procedural model as shown in Figure 3-12. The phases, activities and results of the procedural model are explained below.

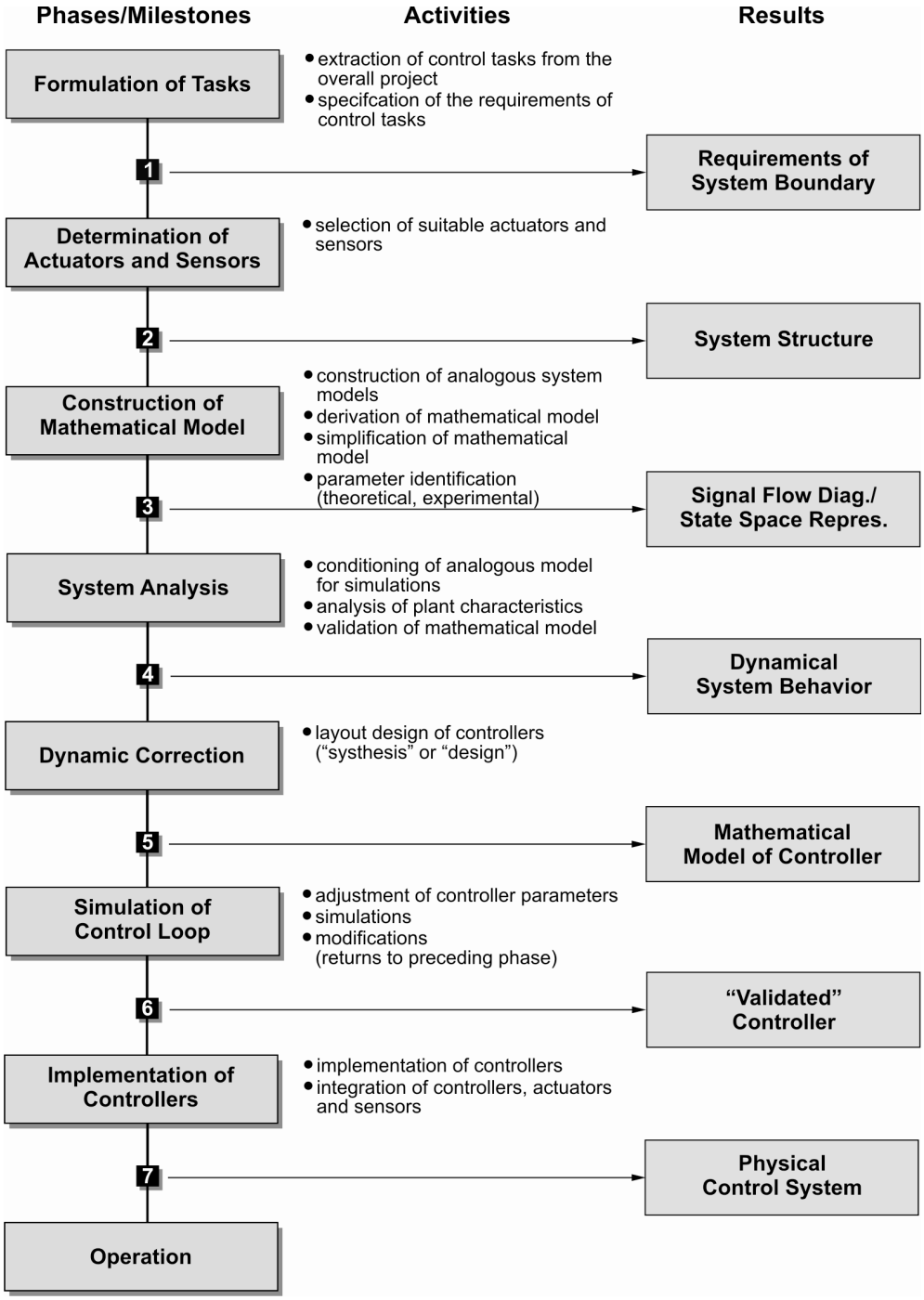


Figure 3-12: Procedural Model for the Design of Controller [Föl08, p. 12]

Formulation of tasks: At this stage, the control task is specified as precise as possible. The control task is only a sub-task of an overall system. Therefore,

the system boundaries for the control tasks have to be determined. Thereby, the system boundaries are selected in such a way that the interaction with the other system environment is neglected, as the case may be, the relevant disturbance variables are sufficiently taken into account. The outcome of the setting of the system boundaries is the basic system to be controlled. The specification of the control task includes the requirements for the dynamic behavior of the control loop. These can be subdivided into basic requirements, qualitative requirements and quantitative requirements.

Determination of actuators and sensors: For state space control, all relevant state variables of the system are considered. From the cost criteria, not all relevant system variables can be measured due to the big amount of the relevant system variables. Therefore it is necessary to clarify which of the state variables to be measured and which of the state variables to be approximated by a model. Since the sensor to measure the system variables as well as the actuators to influence the output variables carry an impact on the dynamic behavior of the overall system, the selected sensors and actuators are usually taken in consideration while modelling the system.

Construction of mathematical model: The objective of this phase is to describe the system behavior with sufficient accuracy using the analogous mathematical models of the system (basic system, sensors and actuators). Figure 3-13 shows the steps to create a dynamic analogous model in state space with the example of a simple gear system. The starting point is the principle sketch of the gears. For modelling the dynamics of the gears, the analogous model takes into account the mass moments of inertia, friction and stiffness. The decision regarding which influences and physical effects to be modelled in detail is depending on the respective task. To be able to be processed by the digital computer, a mathematical analogous model is derived, which has to be done by applying the law of physics. This results in a system of differential equations. The system is then represented in state space for the analysis of the dynamic system behavior and the design of controller. In practice, this is often done only after the simplification of the system of differential equations. The state space representation requires a set of first order differential equations in matrix form. The value of the system variables (e.g. torque, mass are displacement) are determined by measurement on the physical system. This is also known as parameter identification.

Systems Analysis: In this phase, the dynamic characteristics of the basic system are analyzed in conjunction with the selected actuators and sensors. This is the phase which the model constructed previously is converted into codes to be simulated on a digital computer. The simulation is usually carried out using the excitation of the system through excitation functions. Typical excitation functions are the step function or the superposition of harmonic oscillations. Based

on the output variable of the system, comparison with the behavior of the simulated system can be made, i.e. the analogous models are validated. If the behavior specified by the models does not correspond with sufficient accuracy with the underlying system, changes have to be made in the previous phases. For example, the simulation model may show that a simplification (e.g. the negligence of friction) was not allowed, and therefore the analogous model must be modified.

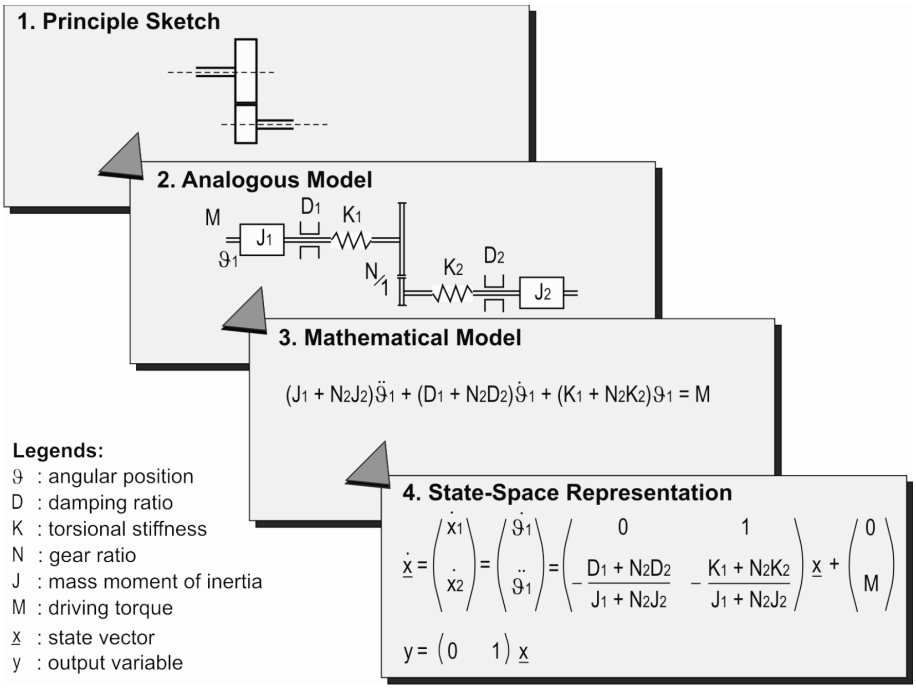


Figure 3-13: Steps of representing the system in state-space as exemplified by the dynamics of a gear system [GEK01, p. 300]

Dynamics Correction: In the phase of dynamic correction, the structure of the controller will be laid out. A controller structure has to be selected, which can achieve the desired dynamic behavior within the closed loop. The controller is described by a mathematical model. The controller contains free, i.e. yet unspecified parameters (the so-called controller gains), which have to be determined based on the requirements set during the phase "formulation of tasks".

Simulation of the control loop: The purpose of the simulation is to examine the suitability of the selected controller as well as to determine the controller gains. Therefore the models of the plant, sensors, controllers and actuators are simulated on the digital computer in terms of their dynamic behaviour. Modifications in the earlier steps are needed if there is inadequacy of the controller or the models. The result of this phase is the validated mathematical description of the controller.

Implementation of the controller: In this phase, the controller is implemented as a physical subsystem ─ usually as algorithm on a microprocessor ─ and in-

tegrated with the actuators, sensors and the basic system to form a physical control system.

Operation: During the initial phase of operation, further adjustment of the controller gains is often needed. This is due to the fact that, on one hand, the model of the plant often reflects its behavior insufficiently, and on the other hand, the controller gains depend directly on the parameters of the plant.

Evaluation

The methodology for controller design developed by FÖLLINGER is widely accepted in both academia and industry. Though the methodology covers the formulation of tasks, the steps involved in formulating the tasks are not sufficiently structured. Thus the methodology can neither support the holistic conceptual design phase nor ensuring an equal treatment on the different domains during that particular phase. Furthermore, no references to the principle solution are made in the methodology. Therefore, it is not sure how the principle solution can serve as the starting point of controller design for advanced mechatronic systems. Without a systematic approach as the continuation from the conceptual design phase, the synergistic impacts enable of the domain-spanning specification of the principle solution may not be sustained.

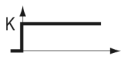
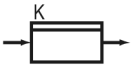
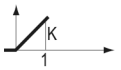
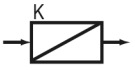
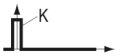
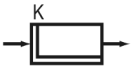
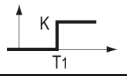
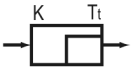
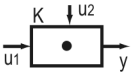
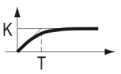
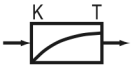
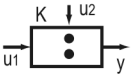
3.4 Domain-Specific Specification Techniques for Controller Design

This section reviews the domain-specific specification techniques for the controller design of mechatronic systems. For the purpose of design and analysis, it is necessary to have a mathematical model of the controlled system. A "model" is the encoded form of the knowledge available about the system under study [Lev96, p. 416]. Such a model can be obtained by applying physical laws governing the system. The models of the controlled system are generally highly coupled nonlinear differential equations. Fortunately, many physical systems behave linearly around an operating point within some range of the variables and it is possible to develop linear approximations to the physical systems. The linear approximation to the physical system is described by linear, constant coefficient ordinary differential equations [Bis93, p. 25]. Such equations provide a complete description about the dynamics of the system. For any given stimulus, the output response is obtained by solving these equations. However, this method can be rather cumbersome and difficult for the designer to handle [Buc78, p. 1]. For these reasons, the transfer function concept and the state-space approach are developed. Having obtained the mathematical model, a controlled system can be represented graphically by means of block diagrams or signal flow graphs. Each of the specification techniques for system representation is briefly explained in the following subsections.

3.4.1 Block Diagram

A block diagram consists of blocks connected by lines. A block describes an underlying input/output relationship where the input variable is transformed to the output variable of the system. A block represents a special dynamic behavior and can be designated as a transfer element. Complex dynamic systems can be represented by the coupling of simpler transfer elements. Table 3-1 shows some basic transfer elements stated in [Föl08, p. 46]. As shown in the second column of Table 3-1, the functional relation between input and output characterizes the output variable *y* resulting from an input variable *u* for a given system in the time domain. Applying Laplace transformation on the functional relation, the result is the so called transfer function of a transfer element. The transfer elements can be represented by means of block diagrams, shown in the last column of Table 3-1. A method of analysing a system is to represent the system by an equivalent block diagram and then apply several simplifications to the block diagram circuitry. A complex block diagram can be simplified using easily derivable transformations [Buc78, p. 14].

Table 3-1: Basic transfer elements and their block oriented representation [Föl08, p. 46]

Description	Functional Relation	Transfer Function	Step Response (zero für t<0)	Behavior of Step Response	Symbol
proportional element	$y = K\,u$	K	K		
integral element	$y = K\int_0^t u(\tau)\,d\tau$	$\frac{K}{s}$	$K\,t$		
derivative element	$y = K\,\dot{u}$	$K\,s$	$K\,\delta(t)$		
dead time element	$y(t) = K\,u\,(t - T_t)$	$Ke^{-T_t s}$	$K\,\sigma(t - T_t)$		
summing element	$y = u_1 \pm u_2 \ldots \pm u$				
characteristic curve	$y = F(u)$				
multiplying element	$y = K\,u_1\,u_2$				
first-order time-delay element	$T\dot{y} + y = K\,u$	$\frac{K}{1 + T\,s}$	$K\,(1 - e^{-t/T})$		
dividing element	$y = K\,\frac{u_1}{u_2}$				

The state-space approach is especially suitable for complex systems with multiple inputs and outputs. Figure 3-14 exemplifies the block diagrams representing the general structure of state space control for a linear system. The con-

trolled system in Figure 3-14 is described by the following system of equations:

$$\dot{\underline{x}} = \underline{A} \cdot \underline{x} + \underline{B} \cdot \underline{w} \text{ with } \underline{w} = \underline{u}$$
$$\underline{y} = \underline{C} \cdot \underline{x}$$

Taken into account the pre-filter and the state controller:

$$\dot{\underline{x}} = \left(\underline{A} - \underline{B} \cdot \underline{R}\right) \cdot \underline{x} + \underline{B} \cdot \underline{M} \cdot \underline{w}$$

$$\underline{y} = \underline{C} \cdot \underline{x}$$

By determining the appropriate values of the matrix $\underline{R}$, the state variables $\underline{x}$ of the system ─ and thus also the output variables $\underline{y}$ of the system ─ can be controlled in the desired way. What necessary here is the so-called "controllability" of the system [Föl08, p. 442]. If this criterion is not met, then not all system variables are controlled as per the desired way of the input vector $\underline{w}$.

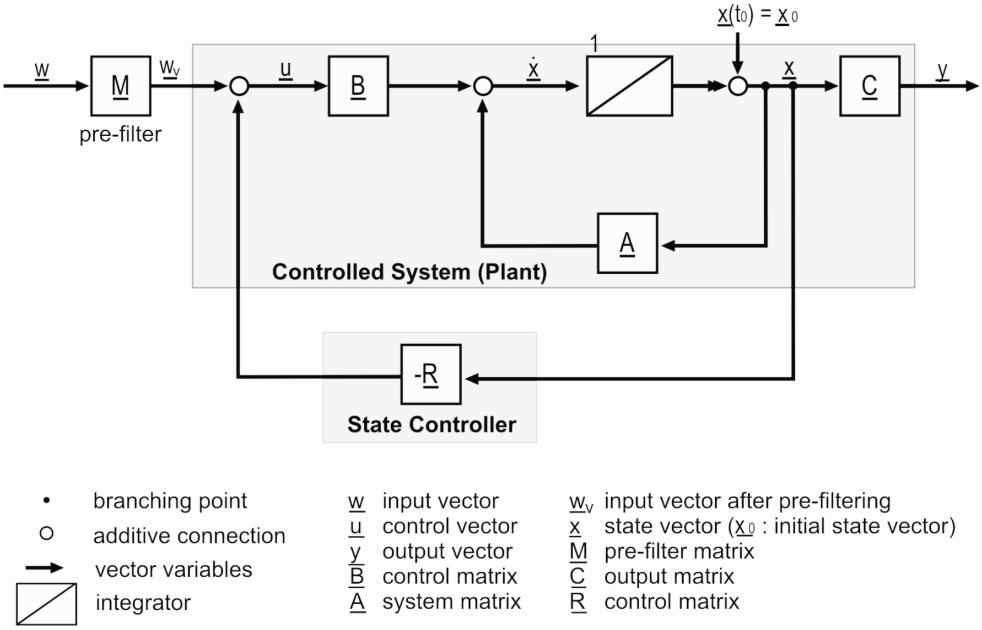


Figure 3-14: The general structure of state space control for a linear system [GEK01, p. 297]

Evaluation

The block diagram is the most prevalent specification technique used by the control engineers nowadays. It is possible to use block diagrams to describe any type of system. In comparison with the mathematical equations, the block diagram is more intuitive. By means of abstraction, the block diagrams can provide a higher level and less detailed description aimed at the understanding of the overall concepts. However, a block diagram has the disadvantage of losing the topological significance of the system specified. For instance, signals that physically belong together and are inseparable from each other get sepa-

rated in the block diagram into two totally independent signals. As pointed out in [Lev96, p. 416], block diagram is certainly not the right tool to describe, for instance, electrical circuits or multibody systems. Besides that, behavioral adaptation of advanced mechatronic system cannot be represented by the block diagrams. Therefore, the block diagram is not the ideal tool for conceptual design.

3.4.2 Signal Flow Graph

Signal flow graphs, as shown in Figure 3-15, consist of a network in which nodes representing each of the system variables are connected by directed branches. A branch acts as a one-way signal multiplier, the ratio of the output to the input is defined as the transmittance, with the arrowhead used to indicate the flow of information as in a block diagram. [Buc78, p. 26]

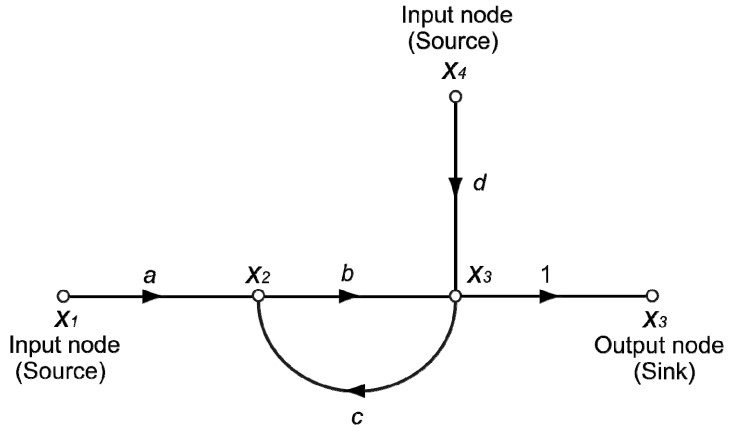


Figure 3-15: An example of signal flow graphs [Oga02, p. 105]

A signal flow graph contains essentially the same information as a block diagram. The important properties of signal flow graphs are the nodes and the branches. A branch indicates the functional dependence of one signal on another. A signal passes through only in the direction specified by the arrow of the branch. A node adds the signals of all incoming branches and transmits this sum to all outgoing branches. It is important to note that for a given system, a signal flow graph is not unique. Many different signal flow graphs can be drawn for a given system by writing the system equations differently. [Oga02, p. 105]

The usual application of signal flow graphs is in system diagramming. The set of equations describing a linear system is represented by a signal flow graph by establishing nodes that represent the system variables and by interconnecting the nodes with weighted, directed, transmittances, which represent the relationships among the variables. Mason's gain formula [Oga02, p. 111] may be used to establish the relationship between an input and output. Mason's gain formula

is especially useful in reducing large and complex system diagrams in one step, without requiring the step-by-step reductions.

Evaluation

The signal flow graphs and the block diagrams are about equally found in the control engineering texts. In comparison to the block diagrams, there are correlations between the elementary constructs of the block diagram and the equivalent elementary constructs of the signal flow graph. However, the signal flow graph is a little less powerful than the block diagrams in some specific aspects, for instance, there is no signal flow graph equivalent exists for a multiport block used in the block diagram. Similarly with block diagrams, signal flow graphs can capture the computational structure, whereas they do not preserve the topological structure of the system they represent [Cel91, p. 257]. Furthermore, the signal flow graph is not intuitive, in the sense that it is not easily interpretable for a beginner. Furthermore, it can be difficult to draw the signal flow graphs systematically as the complexity of the system increases. Signal flow graph is not the ideal specification technique for the conceptual design of advanced mechatronic systems.

3.5 Call for Action

The literature review shows that approaches to manage the transition from domain-spanning principle solution towards domain-specific controller design are nearly nonexistent, except for a few scattered basic thoughts that were written under the Collaborative Research Center 614. Nevertheless, the existing domain-spanning design methodologies and specification techniques for mechatronic systems as well as the domain-specific design methodologies and specification techniques for controller design are reviewed. The evaluation of the state-of-the-art shows an acute urgency for research and calls for appropriate actions to be taken to fulfill the requirements listed in Section 2.5. Table 3-2 shows the evaluation of the state-of-the-art at a glance.

R1 ― A Holistic Principle Solution as a Starting Point for Concretization

Each of the domain-spanning design methodologies that were reviewed partially fulfills the requirement for a holistic principle solution as a starting point for domain-specific concretization. Out of the domain-spanning specification techniques reviewed, only the specification technique developed by FRANK et al for describing the principle solution of advanced mechatronic systems can completely fulfil this requirement. This specification technique portrays a holistic principle solution, which covers the different aspects of the advanced mechatronic systems to be developed. Neither the domain-specific design methodologies nor the domain-specific specification techniques reviewed de-

ploy the principle solution as a starting point for domain-specific concretization. They simply start at a high level abstraction within their respective domains.

R2 – Equal Treatment on the Basic Concepts from Different Domains

The reviewed domain-spanning specification techniques are domain-independent and thus allow for equal treatment of the basic concepts from different domains of mechatronics during the conceptual design phase. However, none of the domain-spanning design methodologies address what the basic concepts are so that the principle solution can be specified not only spanning the different domains but also treating each of the domains equally. Subsequently, none of them explicitly address the basic concepts in control engineering so that they can be treated as per their counterparts from the other domains during the conceptual design phase. None of the domain-specific design methodologies and specification techniques reviewed can effectively fulfill this requirement as they are customized for the domain of control engineering.

R3 – Systematic Extraction of Information from the Principle Solution

All design methodologies and specification techniques reviewed progress from a coarse-grained design towards a fine-grained design. However, they are limited within their respective development phases. None of them address the transition from the conceptual design phase towards the concretization phase. This is due to the fact that the specification of the principle solution for advanced mechatronic systems is itself an emerging trend for engineering design. As such, all the design methodologies and the specification techniques fail to address how information can be extracted from the principle solution for further concretization in each of the domains of mechatronics.

R4 – Integrating Top-Down and Bottom-Up Approaches

On one hand, top-down approaches are used during the early phases of development, for instance, during the decomposition of functional requirements in axiomatic design and along the left bough of both the V models of the VDI-Guideline 2206 as well as the 3-Level Procedural Model. On the other hand, bottom-up approaches are used during the concretization phase for the dynamics correction of the controlled systems, where a superimposing control loop is designed based on the characteristics of the underlying control loop. None of the design methodologies reviewed address the differences between the approaches deployed during the conceptual design phase and the approaches deployed during the concretization phase, let alone the integration of both approaches. As such, they can only partially fulfill the requirement for integrating domain-spanning and domain-specific approaches.

R5 – Linking Semi-Formal and Formal Specifications

During the conceptual design phase for advanced mechatronic systems, both formal and semi-formal specification techniques are deployed in different design methodologies. For instance, axiomatic design utilizes a formal specification technique while FRANK et al utilize a semi-formal specification technique for conceptual design. On the contrary, only formal specifications are deployed during the concretization phase for controller design. None of the reviewed methodologies offer any approach which links the easily interpretable semi-formal specifications used for conceptual design with the precise formal specifications used for the controller design of advanced mechatronic systems.

Table 3-2: Evaluation of the state-of-the-art against the requirements

Legend

Requirement is

● fully fulfilled

◐ partially fulfilled

○ not fulfilled

Design Methodologies/ Specification Techniques	A holistic principle solution as a starting point for concretization	Equal treatment on the basic concepts from different domains	Systematic extraction of information from the principle solution	Integrating top-down and bottom-up approaches	Linking semi-formal and formal specifications
Domain-Spanning Design Methodologies for Mechatronic Systems					
Axiomatic Design	◐	●	○	◐	○
The SYSMOD Approach	◐	●	○	◐	○
VDI-Guideline 2206: Design Methodology for Mechatronic Systems	◐	●	○	◐	○
3-Level Procedural Model according to BENDER	◐	●	○	◐	○
Methodology for Mechatronic Design according to LÜCKEL	◐	◐	○	◐	○
Domain-Spanning Specification Techniques for Mechatronic Systems					
Specification Technique for Axiomatic Design	◐	●	○	○	○
Systems Modeling Language (SysML)	◐	●	○	○	○
Function-Oriented Specification according to BUUR	◐	●	○	○	○
Specification Technique for the Principle Solution of Self-Optimizing Systems according to FRANK	●	●	○	○	○
Domain-Specific Design Methodologies in Control Engineering					
DIN 19226: German Standard for Control Technology	◐	◐	○	○	○
Methodology for Controller Design according to FÖLLINGER	◐	◐	○	○	○
Domain-Specific Specification Techniques for Controller Design					
Block Diagram	◐	○	○	○	○
Signal Flow Graph	◐	○	○	○	○

4 A Method to Manage the Transition from the Principle Solution towards the Controller Design of Advanced Mechatronic Systems

The review of the state-of-the-art reveals acute urgency for research within the early development phases of advanced mechatronic systems. There is neither any approach to specify the basic control concepts within the principle solution of advanced mechatronic systems nor any approach to extract this information from the principle solution for the controller design of such systems. These two interrelated approaches are addressed in this chapter. They constitute a method to manage the transition from the principle solution towards the controller design of advanced mechatronic systems. At the end of the chapter, the method is evaluated against the requirements stated in Section 2.6.

4.1 Specifying the Basic Control Concepts within the Principle Solution of Advanced Mechatronic Systems

In order to resolve the problems defined in section 2.5.1 this section describes an approach to specify the basic control concepts within the principle solution of advanced mechatronic systems. For this purpose, the specification technique developed by FRANK et al as described in Section 3.2.4 is used. This section is divided into two parts. The first part describes the specification of the basic control concepts in each of the partial models of the principle solution of advanced mechatronic systems. The second part describes the essential cross-references to be made between these partial models in order to produce a complete specification of the control concepts during the conceptual design phase.

4.1.1 Basic Control Concepts within Individual Partial Models of the Principle Solution

During the conceptual design phase of advanced mechatronic systems, the specification of the basic control concepts is of such significance as the specification of the basic concepts of mechanics, electric/electronics, and software engineering. During this phase, engineers must first formulate the fundamental concepts of the control tasks. They must be clear of where the essential control problems lie and what is to be achieved before they worry about how to solve the problem. The emphasis is the fundamental concepts and the alternative design strategies rather than the detailed derivations of mathematical rigor. Any unnecessary details of the domain-specific control techniques must be avoided in order to focus on the main features. The aim is to specify the control con-

cepts of advanced mechatronic systems within their principle solution in a holistic way spanning the different domains of mechatronics.

Figure 4-1 delineates the basic control concepts to be specified within the principle solution of advanced mechatronic systems. The following partial models of the principle solution are involved: environment, application scenarios, requirements, system of objectives, functions, active structure, and behavior. The partial model behavior includes the partial models behavior ─ states and bahavior ─ activities. The partial model shape is omitted here as the shape of the system to be developed is of little relevance when it comes to its controlled movements. The following subsections describe the specification of the basic control concepts within each of the partial models of the principle solution of advanced mechatronic systems.

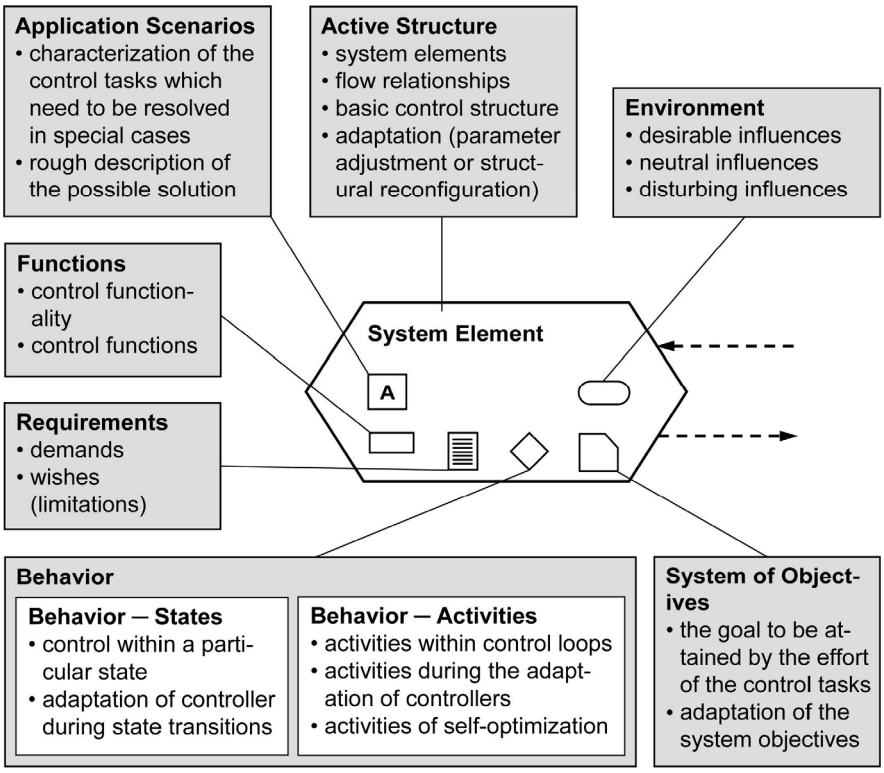
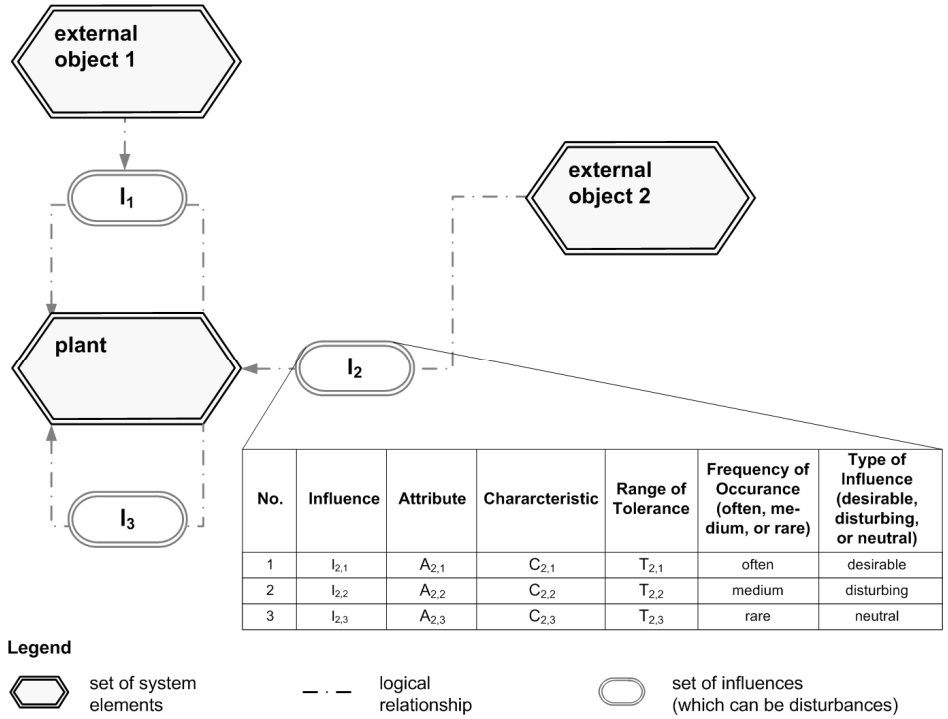


Figure 4-1: Basic control concepts within the principle solution of advanced mechatronic systems

4.1.1.1 Environment

Advanced mechatronic systems are often subjected to various types of influences. Within the conceptual design phase, these influences have to be anticipated and specified in the principle solution. This can be done in the partial model environment as illustrated in Figure 4.2. Crucial for the control of an advanced mechatronic system are the influences. Influences can be internal influences generated within the plant $\mathbf{l_3}$, external influences generated outside

the plant **I_2**, or influences generated within and outside the plant **I_1**. In this partial model, an influence table [Fra06, p. 106] [GFS05] can be added to the set of influences. The influence table outlines the influences, their attributes, characteristics, range of tolerance, frequency of occurrence, and types.



No.	Influence	Attribute	Chararcteristic	Range of Tolerance	Frequency of Occurance (often, medium, or rare)	Type of Influence (desirable, disturbing, or neutral)
1	$I_{2,1}$	$A_{2,1}$	$C_{2,1}$	$T_{2,1}$	often	desirable
2	$I_{2,2}$	$A_{2,2}$	$C_{2,2}$	$T_{2,2}$	medium	disturbing
3	$I_{2,3}$	$A_{2,3}$	$C_{2,3}$	$T_{2,3}$	rare	neutral

Figure 4-2: Specification of influences in the partial model environment

These influences constitute the essential points of consideration during the formulation of a control concept for the system to be developed. An influence can be desirable, neutral, or disturbing to the system's behavior. Disturbing influences can adversely affect the response of the system and result in inaccurate output of their controlled variables. In order to avoid these negative impacts, the unwanted disturbing influences have to be compensated to ensure the performance and safety of the system. As such, the foreseeable disturbing influences ensure that potential means of disturbance compensation are dealt with as early as during the conceptual design phase of advanced mechatronic systems.

4.1.1.2 Application Scenarios

During the conceptual design phase, the application scenarios of the advanced mechatronic systems have to be conceptualized. An application scenario focuses on a partial development task, characterizes the problem that has to be resolved in that special case, and roughly describes the possible solution [GFD+08c, p. 92]. The specification of application scenarios in the principle solution involves three parts: a description of the partial development task, a

sketch, and cross-references to the application-specific partial models. The basic control concepts to be specified in these three parts are described as follows.

Description of the partial development task: In this section, it is essential to point out how the system should be controlled in this particular application scenario. As such, the control tasks which have to be carried out in this application scenario have to be characterized. Such a characterization involves the description of the operational and environmental conditions, the circumstances under which the requirements of the control tasks have to be met, and the desired response of the controlled behavior in this application scenario. By comparing between the descriptions of the partial development task of different application scenarios, a rough control concept which is feasible for the different application scenarios can be conceptualized.

Sketch: A sketch is used to support the description of the partial development task above. While the level of information is still relatively low, sketching helps in conceptualizing a rough solution concept for the control of the system. In a particular application scenario, simple and intuitive sketching visualizes when and where which control strategy has to be used. By comparing the sketches of different application scenarios, the key differences between the application scenarios can be spotted at a glance.

Cross-references to the application-specific partial models: Cross-references can be drawn between an application scenario and its application-specific partial models. These application-specific partial models point out the different aspects of the system which are involved in the application scenario. By means of cross-references, the control concepts required in a particular application scenario but specified in different partial models can be related together.

Different kinds of information can be revealed by such cross-references. For instance, cross-references to the **environment** reveal the influences within and outside the system in a particular application scenario; cross-references to the **requirements list** reveal the involved demands and wishes; cross-references to the **function hierarchy** reveal the involved control functions; cross-references to the **active structure** reveal the involved controllers and the information scheme; cross-references to the **behavior ─ states** reveal the involved states and state transitions; cross-references to the **behavior ─ activities** reveal the activities carried out to control the behavior of the system; cross-references to the **system of objective** reveal the system objective which has to be prioritized in an application scenario.

4.1.1.3 Requirements

The requirements list is the core of the partial model ‘requirements’. Requirements refer to the demands and wishes derived from the various domains of advanced mechatronic systems. Demands are requirements that must be met under all circumstances while wishes are requirements that should be taken into account whenever possible [PBF+07, p. 147]. The focus here is the specification of demands and wishes pertaining to the control of advanced mechatronic systems. These requirements can be specified in qualitative or quantitative terms. In order to avoid communication barriers, well clarified terminologies should be used in the requirements list during the conceptual design phase of advanced mechatronic systems.

Numerous kinds of requirements have to be fulfilled by an advanced mechatronic systems, for instance, requirements with respect to the geometry of the system, the construction material to be used, ergonomics, etc. Three main headings of requirements were identified as of direct relevance for the control of advanced mechatronic systems. They pertain to the kinematics, forces, and safety of the system.

Crucial for the control of advanced mechatronic systems are the limitations. These limitations constrain the system dynamics so that the system behaves in the desired way. These limitations are specified as demands as wishes in the requirements list. The circumstances under which these demands and wishes have to be met must be specified. Limitations exist due to the fact that the actual control is realized through actuators with a limited range of actions. For instance, the time response of a RailCab cannot be made very fast but at the cost of prohibitively large control input to the linear motor producing the thrust.

Requirements on kinematics refer to the demands and wishes pertaining to the controlled motions of multi-body systems. The term kinematics is used for the study of motion without regard to forces [Bol03, p. 142]. They can involve complex motions which consist of combinations of translational and rotational motions in a three dimensional space up to six degress of freedom. It involves the description of the desired direction of motion, its desired magnitude or range of motion, as well as the type of motion.

Requirements on forces describe the required action of the forces that resulted in the controlled motion of the system. Similarly, it involves the description of the acting direction of the forces, their ideal magnitudes or ranges, as well as the type of forces. The interaction of forces which results in the controlled motion of multi-body systems has to be taken into account. It can involve moments of inertia, torque, load, and mass of the system.

The **safety requirements** must be strictly adhered by safety critical mechatronic systems in order to avoid injury, fatality, and destruction. Safety can be enhanced through precise control of system behavior so that malfunctions or abnormal system behavior can be prevented. In the context of networked mechatronic systems, the synchronization of their constituent systems and modules is essential to ensure the safety of the overall system.

4.1.1.4 System of Objectives

The behavioral adaptation of classical mechatronic systems does not involve the adaptation of the objectives of the systems. On the contrary, self-optimizing mechatronic systems adapt their objectives in order to adapt their behavior. The principles of self-optimization are described in Section 2.2.2. During the conceptual design phase, the objectives of the system to be developed are conceptualized and specified in the partial model 'system of objectives' as exemplified in Figure 4-3. There are three kinds of objectives, i.e. the external, inherent, and internal objectives [Fra06, p. 15]. The specification of the system of objectives distinguishes the conceptual design of self-optimizing mechatronic systems from classical mechatronic systems.

The control of a self-optimizing mechatronic system has to correspond to the objective currently pursued by the system. The objective pursued by the system determines the goal to be attained by the effort of the control tasks performed by the system. For instance, the effort of a control task can be directed towards the control quality, resource consumption, or safety of the system in different application scenarios. In this context, the adaptation of the system of objectives determines if the quality of control, resource consumption, or safety of the system should be given priority when carry out a control task. During the conceptual design phase, these objectives to be attained by the control tasks are specified using verbal descriptions. A higher level objective can be broken down into lower level objectives. Nevertheless, mathematical derivation of the objective functions is not desirable during the conceptual design phase.

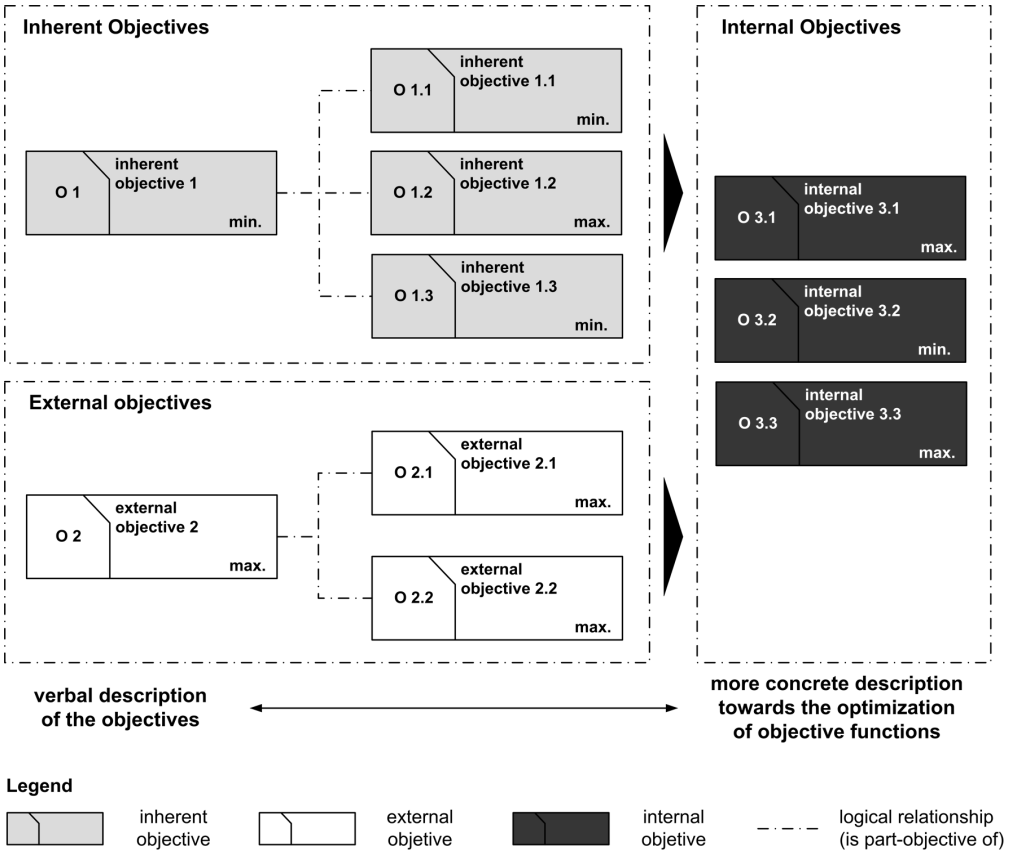


Figure 4-3: Conception of the system of objectives for self-optimizing mechatronic systems

4.1.1.5 Functions

The function hierarchy is the core of the partial model ‘functions’. The term ‘function’ refers to the intended input/output relationship of a system whose purpose is to perform a task. A function thus becomes an abstract formulation of the task, independent of any particular solution [PBF+07, p. 31]. This principle also applies for the conceptual design of advanced mechatronic systems. During the conceptual design phase, the task to be performed by an advanced mechatronic system is clarified and hence its functions are specified.

In this context, it is often that an advanced mechatronic system has to perform tasks such as controlling the velocity, pressure, torque, temperature, and so on. This category of tasks is called the control tasks to be performed by technical systems. Similarly, the function whose purpose is to perform a control task is called a ‘control function’. At a generic level, the control of advanced mechatronic systems involves the control of their longitudinal, lateral, vertical or angular dynamics. Such an overall control function is called a ‘control functionality’. A control functionality realizes a main control task of the system. It can be divided into simpler control functions which correspond to the sub control tasks. Together with the other functions of advanced mechatronic systems,

these control functions are specified in a function hierarchy. Figure 4-4 exemplifies a function hierarchy which includes a control functionality and two control functions.

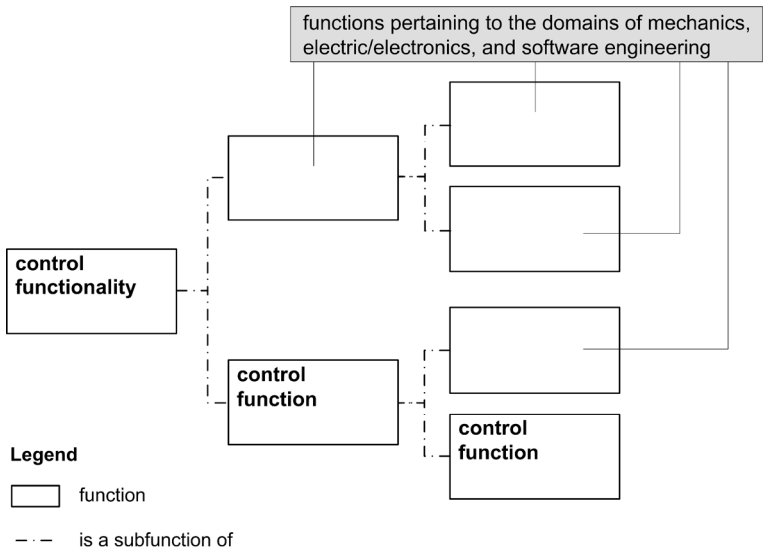


Figure 4-4: A function hierarchy exemplifying a control functionality and two control functions

4.1.1.6 Active Structure

During the conceptual design phase, the active structure plays a dominant role for specifying the control concepts within the principle solution of advanced mechatronic systems. Figure 4-5 illustrates the basic control concepts to be specified in the active structure. In the active structure, the system elements as well as the relationships between them are defined. Together they form a basic control structure for the system to be developed. The means of adaptation have to be taken into consideration within the basic structure, if necessary.

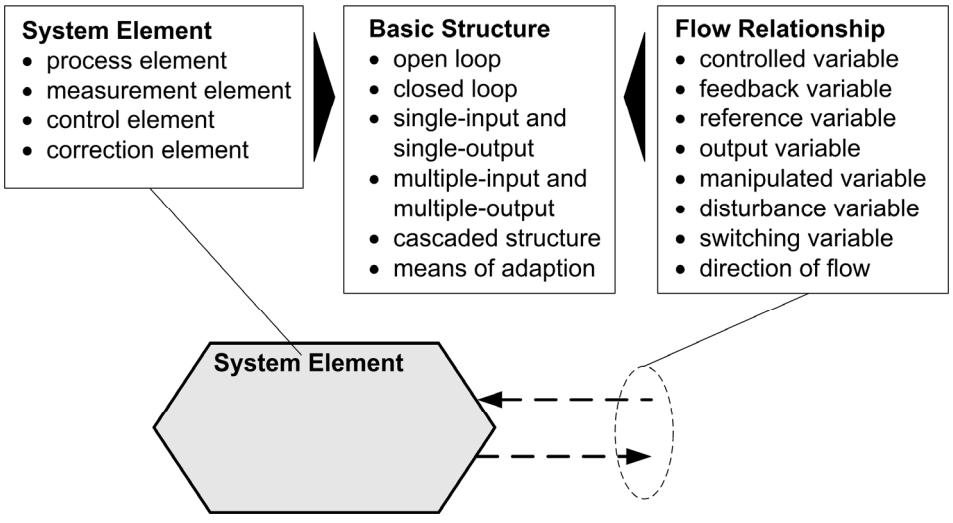
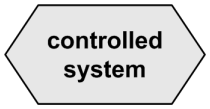


Figure 4-5: The basic control concepts to be specified in the active structure

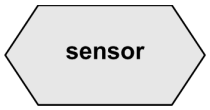
The basis of a control task is the physical system to be controlled. Thereby it does not matter whether the system exists physically or is just a concept with all its system elements defined. Within the active structure, the system elements that are involved in the execution of the control task are defined. These system elements can be categorized into process elements, measurement/estimation elements, control elements, and correction elements.

Process Element



The process elements represent the process dynamics of the physical system to be controlled. Most of the time, it consists of the basic mechanical structure. During the conceptual design phase, the process elements can be simplified as a plant or a controlled system.

Measurement/ Estimation Element

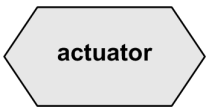


The measurement elements refer to the sensors and their associated components which determine the value of certain system variables. A sensor is a device that measures a physical quantity and converts it into a readable signal. The analogue values measured are converted into digital values and possibly some pre-processing of the acquired data. During the conceptual design phase, the measurement elements can be simplified as a sensor. In cases where the system variable is not measurable or not economical to do so, an observer can be used to estimate the value of the system variables.

Control Element

The control elements refer to the information processing pertaining to a controller. It compares the actual value of the controlled variable with its reference value and subsequently generates the necessary control signals. As such, it decides the corrective action to be taken upon receiving the error signal. During the conceptual design phase, the control elements can be simplified as a controller.

Correction Element



The correction elements refer to the actuators and their associated components which receive the control signal from the controller and transform it into an action which corrects the behavior of the plant. Upon receiving the control signal from the controller, digital to analog conversion as well as energy-based amplification can be required. During the conceptual design phase, the correction elements can be simplified as an actuator. In some cases, even the actuators are abstracted into the plant.

Besides the conception of system elements, the flows between the system elements play an equally important role during the conceptual design of advanced mechatronic systems. Within the active structure, the inputs and the outputs of the system elements are connected by the flows of information, energy, or material. Innovations are often obtained by a critical conception regarding the flows between system elements. For instance, the fly-by-wire or drive-by-wire technology is fundamentally a result of replacing mechanical connections with electric wires carrying signals.

For the purpose of specifying the control concepts, it is essential to maintain the clarity of the information scheme of the system. The information scheme depicts the control flow and the system variables involved in performing the control task. The pertinent system variables within the flows have to be properly labeled, for instance, the output variables which must be held at certain desired values or within a limited deviation from the desired values. The other system variables that have to be identified are stated in Figure 4-5.

The conception of system elements and the relationships between them leads to a basic control structure. Different constellations of system elements are possible in order to realize the envisaged control functions. A proper control structure enables a proper understanding of the underlying input-output relations which result in the desired system response. The conception of a poor or inadequate control structure will eventually result in complicated errors, to the extent that the resulting system response may prohibit proper system operation. As such, the conception of a basic control structure has to be paid considerable attention during the conceptual design phase of advanced mechatronic systems.

A basic control structure can be open, closed, or where appropriate, a combination of both. A proper combination of open-loop and closed-loop controls is usually less expensive and will give satisfactory overall system performance. Figure 4-6 exemplifies the specification of a single control loop in the active structure. As shown in the figure, a single control loop can be simplified by just two system elements.

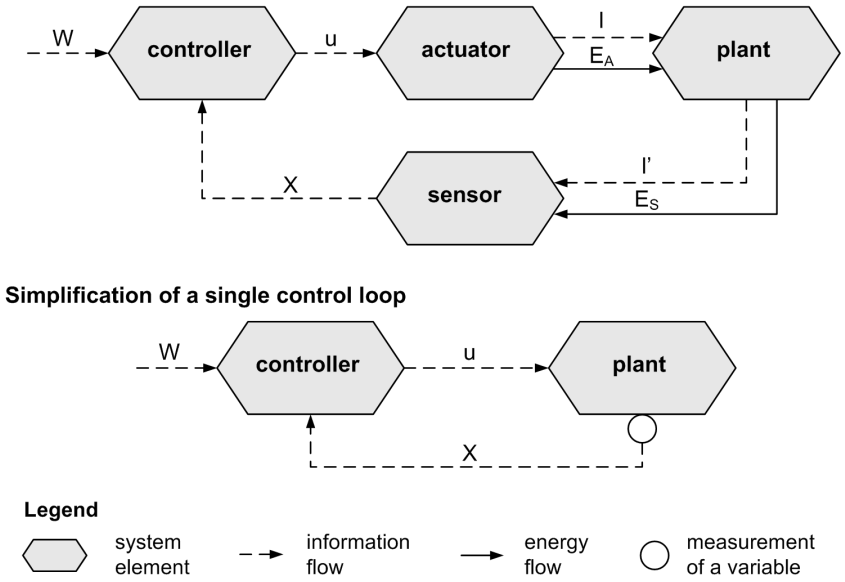


Figure 4-6: Specification of a single control loop in the active structure

The single control loop alone is obviously not sufficient to solve all control problems. Figure 4-7, Figure 4-8, and Figure 4-9 exemplify the specification of the other conventional control structures in the active structure. They consist of cascaded control, multivariable control, and feedforward control. Lengthy explanations of these control structures are omitted here. Further details can be found in the literature cited in the figure. The combinations of these basic structures can be necessary as the complexity of the control task increases.

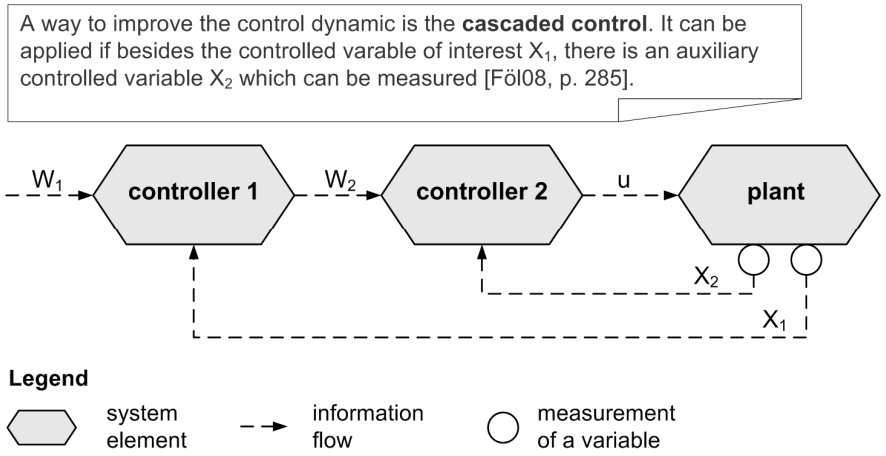


Figure 4-7: Specification of cascaded control within the active structure

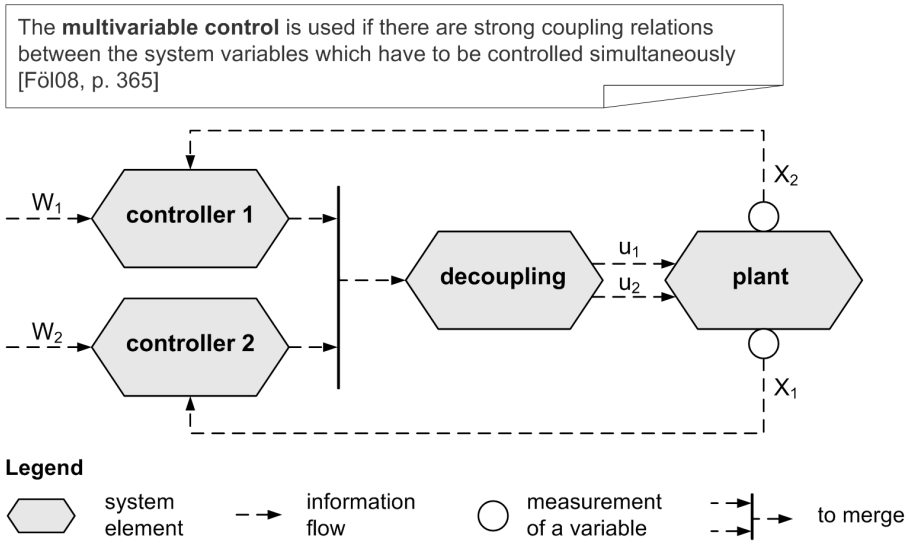


Figure 4-8: Specification of multivariable control within the active structure

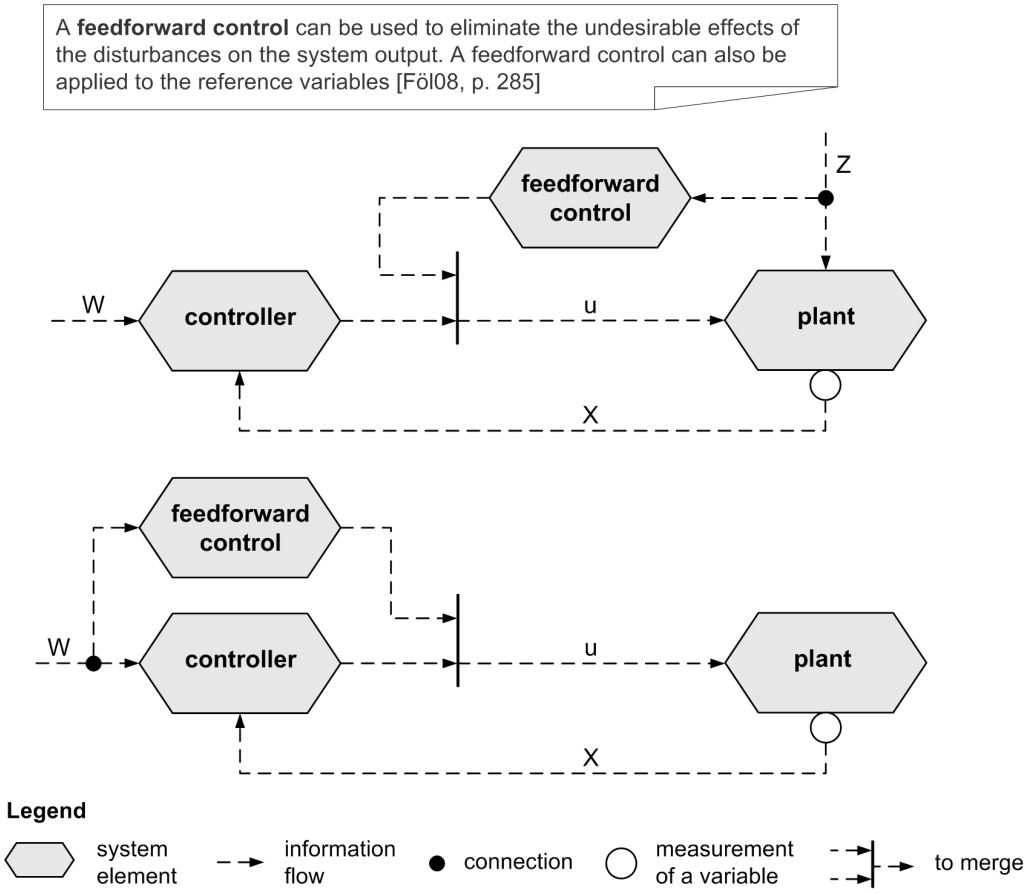


Figure 4-9: Specification of feedforward control within the active structure

The operational and environmental condition of advanced mechatronic systems can vary significantly. Hence, adaptation of system behavior is required in response to the changing operational and environmental conditions. In this context, classical controllers with constant controller parameters and a fixed structure do not always provide satisfactory control performance. This urgency calls for means of adapting the parameters and where necessary the structure of the controllers. During the conceptual design phase, early decisions pertaining to the need to adjust the controller parameters or to reconfigure the controller structures are addressed within the active structure.

In cases where the adjustment of controller parameter is deemed necessary, a system element that performs the required adaptation is added to the active structure. As shown in Figure 4-10, there are two main approaches for the adjustment of controller parameters, i.e. feedforward adaptive control or feedback adaptive control. Under different circumstances, the controller parameters can be adjusted to compensate for changes within the process of the plant or in the environment.

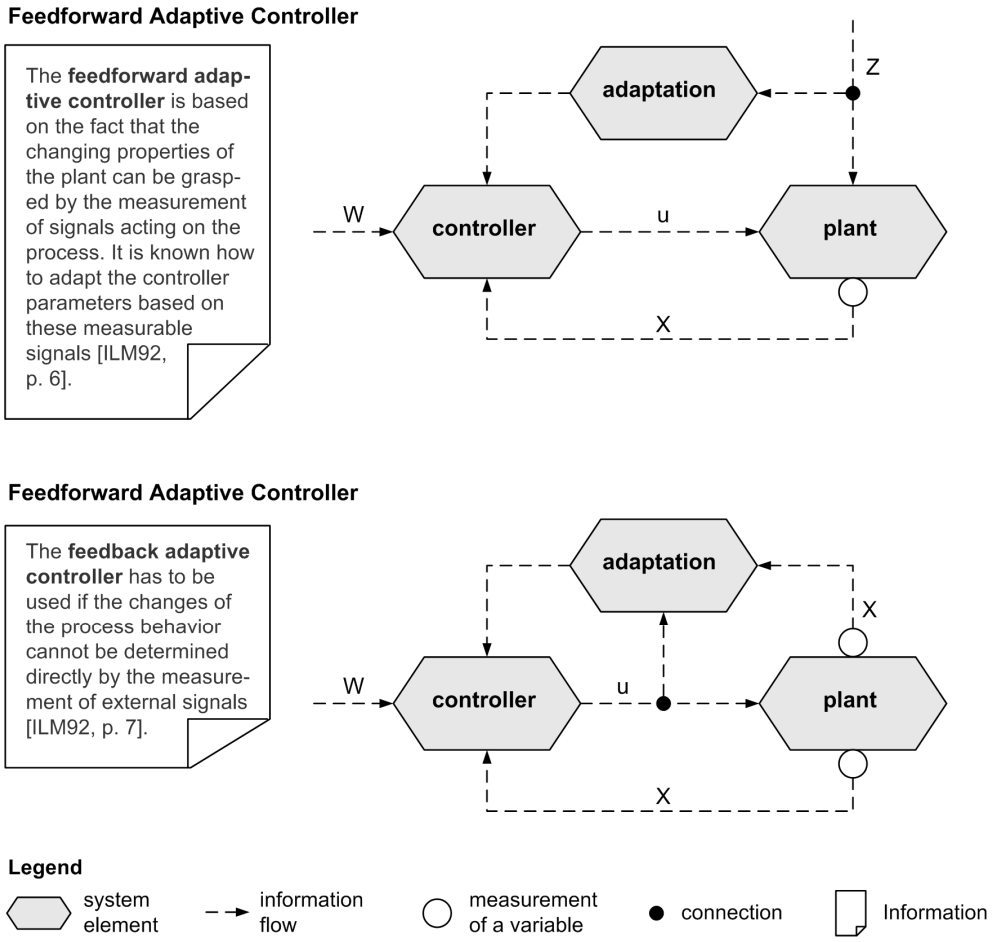


Figure 4-10: Specification of adaptation by means of parameter adjustment in the active structure

The alteration of the controller parameters will not lead very far because many characteristics can only be altered in the internal structures of the controller. As such, behavioral adaptation through the reconfiguration of the controller structures is required. In this context, the alternative controller structure should be conceptualized, for instance, the alternative A, B and C as illustrated in Figure 4-11. In this case, different control algorithms and generation of switching commands are possible. A switching can be made between the controllers controlling a same system variable, or, between the controllers controlling different system variables. For instance, a motor drive has to switch between its current controllers in different modes of operation, while a follower RailCab has to switch from distance control when driving in a convoy to velocity control when driving alone.

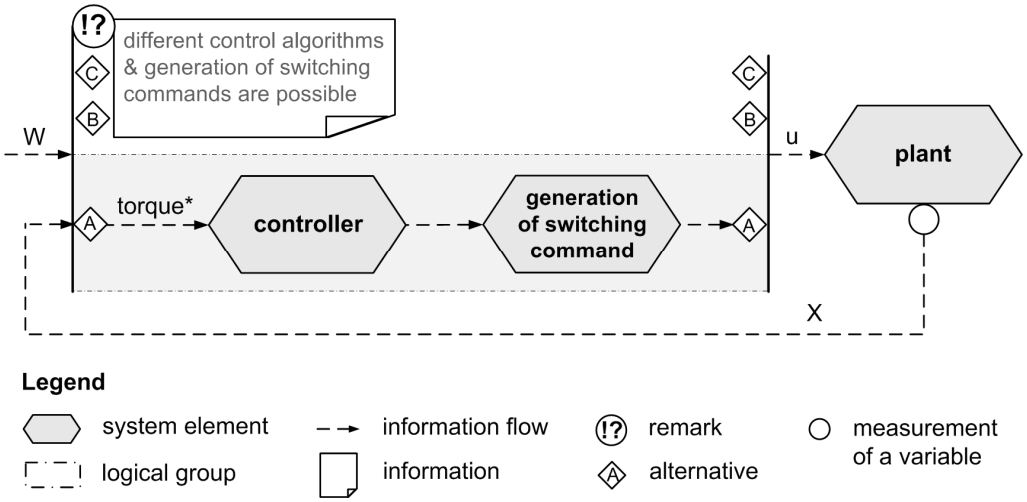


Figure 4-11: Specification of adaptation by means of structural reconfiguration in the active structure

So far, the conception of controllers in the active structure has been described. In order to clarify the information processing above the control loops, the Operator Controller Module (OCM) has to be included in the active structure. Figure 4-12 exemplifies the conception of the OCM in the active structure. The reflective operator involves system elements such as configuration control, reference generator, monitoring, emergency handling, sequencer, and risk management. The cognitive operator involves system elements such as behavior-based optimization and model-based optimization. Different combinations of these system elements of the reflective operator and the cognitive operator are required depending on the system to be developed. The cognitive operator and the reflective operator decide and instruct the adaptation by means of parameter adjustment and/or structural reconfiguration of the controllers. The fundamentals of the OCM are described in Section 2.2.2. System elements which carry out the self-optimization process are indicated by an arrow [Fra06, p. 99].

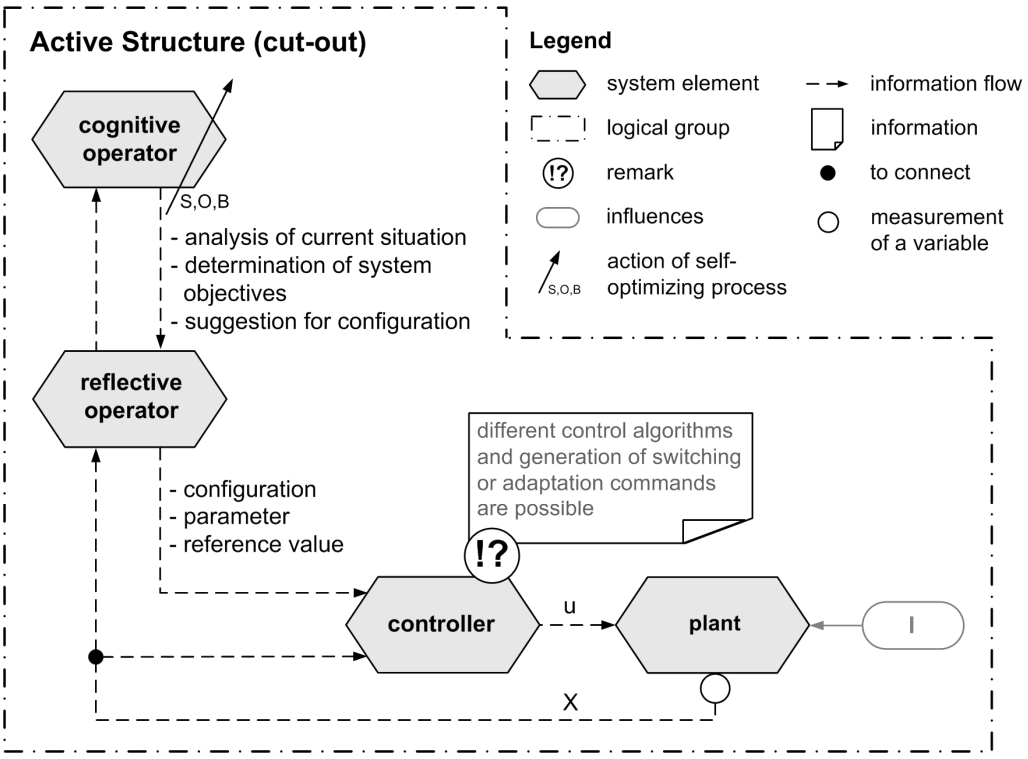


Figure 4-12: Conception of an Operator Controller Module in the active structure

4.1.1.7 Behavior – States

Advanced mechatronic systems are operational in several modes of operation. Each of these modes of operation can be represented as a particular state of the system. The states of the system and the state transitions as well as the events that trigger a state transition can be described in the partial model behavior – states. As such the discrete behavior of the system to be developed is conceptualized.

A system may have to carry out a particular control task in a particular state. From one state to another, a different control task may have to be carried out by the system. Variations can be due to the different control functions to be realized and the different requirements to be met in those states. As such, a particular state can require a particular controller which suits the current mode of operation in the best way. In order to distinguish between the control of the system in different states, the quality of control and the amount of resources demanded to carry out the control task in a particular state have to be specified. If the type of controller required in that particular state can be anticipated, the controller type has to be specified as well. Furthermore, the initialization of the controllers can be specified in order to clarify the flow of control between the main states of the system.

Figure 4-13 exemplifies the specification of control concepts within the partial model behavior-states of an advanced mechatronic system. The system has to be operational in two states, i.e. state A and state B. Upon receiving a triggering event, state A transits into state B and vice versa. Two different controllers are required in the two states of the system. As such, there is a need to switch to another controller during the transition between state A and state B. In order to distinguish between these states, the control quality, the resources demand and the types of controller deployed in both the states are stated. Besides that, the initialization of the controller during the transition between state A and state B is pointed out. Such a description depicts the flow of control and the transition which connects the two states.

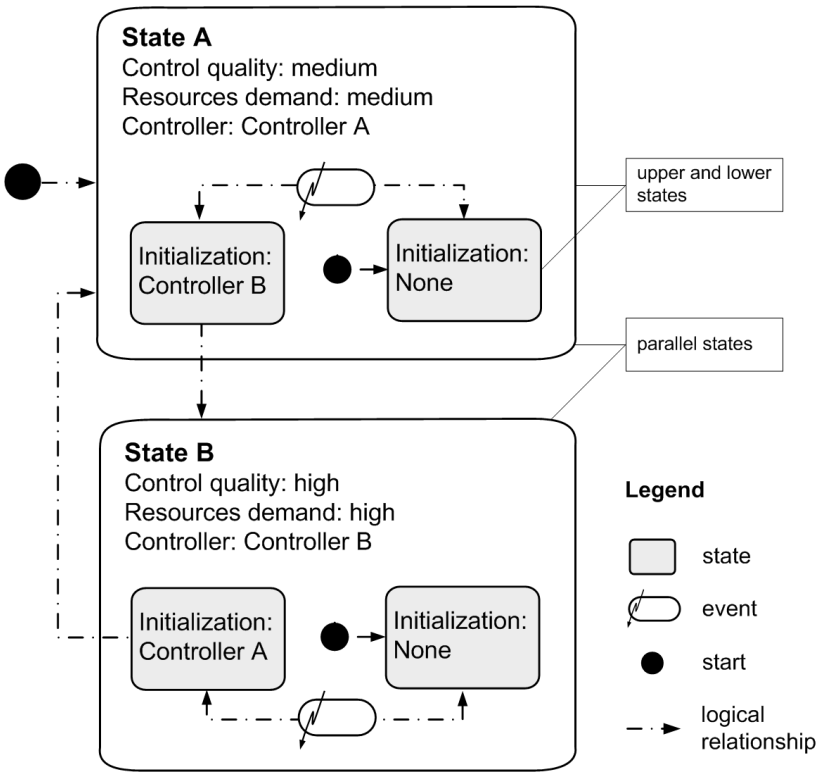


Figure 4-13: Specification of control concepts within the system states

4.1.1.8 Behavior – Activities

During the conceptual design phase, a control task to be carried out by an advanced mechatronic system can be broken down into a number of activities. The activities of a control task describe how a controller works in principle and how the adaptation of the controlled behavior can be carried out. These activities can be very different depending on the control task and the physical system to be controlled. During the conceptual design phase, these activities are conceptualized together with the other system activities in the partial model behav-

ior-activities. Activities can be carried out in parallel, in series, or a combination of both.

Figure 4-14 illustrates the basic activities of a single control loop as shown in Figure 4-6. In a single control loop, the actual value of the controlled variable is measured and compared with the desired value. Subsequently the required control signal is calculated and generated based on the deviation between the measured value and the desired value of the controlled variable. Finally, the control signal is transformed into action that corrects the dynamics of the system which eventually brings the controlled variable of the system to the desired value or within a limited deviation from the desired value. These activities are carried out continuously in performing a standard control task.

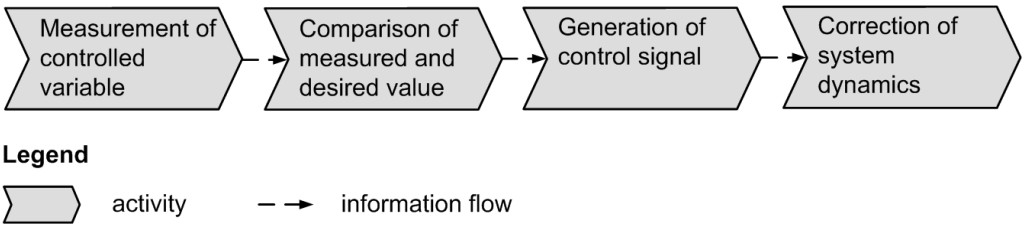


Figure 4-14: Basic activities of a single control loop

Though it is essential to conceptualize the basic activities of a single control loop, these activities are not sufficient for the conceptual design of self-optimizing mechatronic systems. For such systems, the self-optimization process has to be taken into consideration. During the conceptual design phase, the activities of the self-optimization process have to be anticipated and structured. Figure 4-15 exemplifies the structuring of these activities according to the self-optimization process, i.e. analysis of current situation, determination of system objectives, and adaptation of system behavior.

It is possible to outline the common activities for each step of the self-optimization process. **Analysis of current situation** involves activities such as measurement, identification, communication, evaluation, inquiry, prediction, acquisition, and extraction of particular influences or system variables. **Determination of system objectives** involves activities such as weighting of objective functions, Pareto optimization, selection of a Pareto point, and evaluation of a fuzzy-rule base. **Adaptation of system behavior** involves activities such as loading of a controller, initialization of a controller, activation/deactivation of a controller, adjustment of controller gains, and generation of reference profiles for a controller.

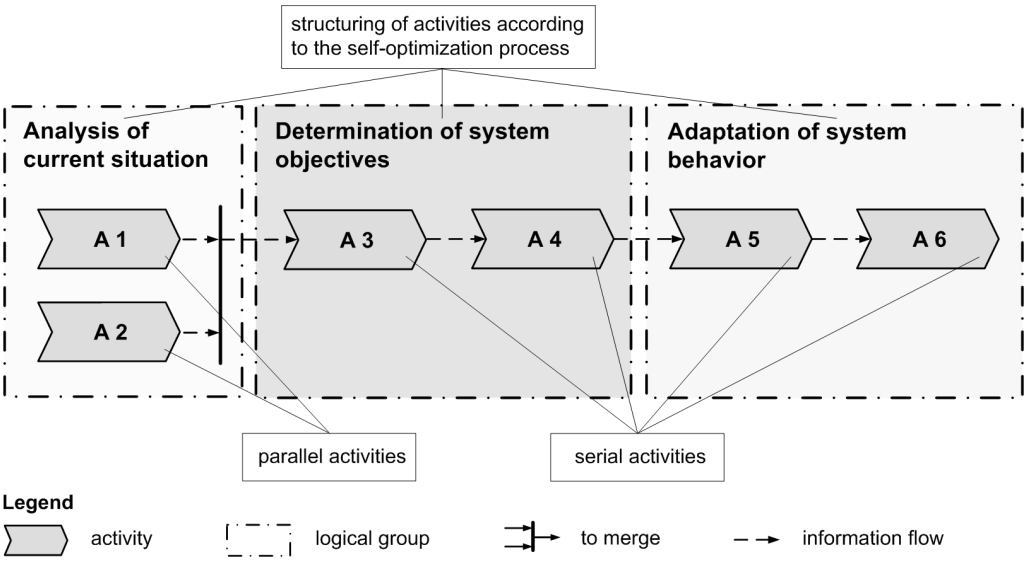


Figure 4-15: Structuring of the activities of a self-optimizing mechatronic systems

4.1.2 Basic Control Concepts within the Cross-References between the Partial Models of the Principle Solution

During the conceptual design phase of advanced mechatronic systems, numerous types of cross-references exist between the partial models of the principle solution. Such a cross-reference is understood as a conceptual link between a construct (or description) of a partial model with another construct (or description) of another partial model. As each of the partial models describes a particular aspect of the system, the constructs specified in the individual partial models have to be in conformity and mutually supplement each other. The same applies to the control concepts specified within the different partial models of the principle solution. Nine cross-references which are essential for the specification of control concepts within the principle solution are listed in Table 4-1. These cross-references ensure the completeness of the control concepts of advanced mechatronic systems during the conceptual design phase. Each of the cross-references is described in the following subsections.

Table 4-1: Interrelations between the partial models (cut-out)

Construct	Partial Model	Type of Inter-relation	Construct	Partial Model
pictogram	application scenarios	characterizes	objective	system of objectives
pictogram	application scenarios	characterizes	function	functions
function	functions	is determined by	(text)	requirements
function	functions	is realized by	system element	active structure
system element	active structure	is influenced by	influence	environment
system element	active structure	fulfills	(text)	requirements
system element	active structure	operates in	state	behavior – states
system element	active structure	carries out	activity	behavior – activities
activity	behavior – activities	occurs in	state	behavior – states

4.1.2.1 Cross-References between Application Scenarios and System of Objectives

Figure 4-16 exemplifies the cross-references between the application scenarios and the system of objectives within the principle solution. Such cross-references clarify which objective of the system is to be prioritized in which application scenario. (A system with two application scenarios is exemplified here.) In each of the application scenarios, a different objective has to be pursued by the system. The inherent objective O1 has to be pursued by the system in application scenario AS1 while the external objective O2 has to be pursued by the system in application scenario AS2. As such, the inherent objective O1 has a higher priority than the external objective O2 when the system operates in application scenario AS1 and vice versa. The adaptation of the system of objectives in face of changing application scenarios is enabled by the self-optimization process. As a result, the control of the system has to be adapted to correspond with the objective currently pursued by the system.

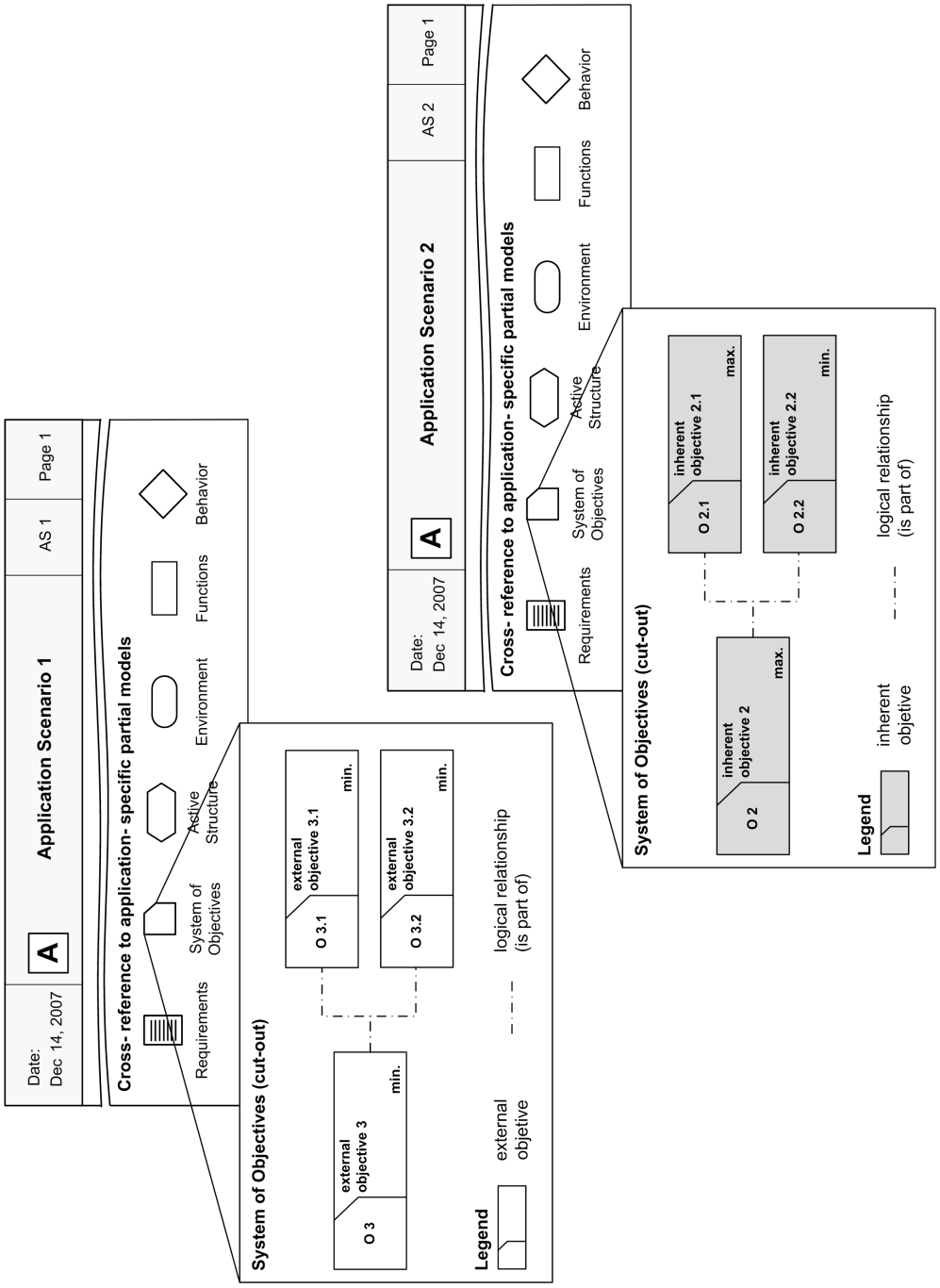


Figure 4-16: Cross-references between application scenarios and system of objectives (cut-out)

4.1.2.2 Cross-References between Application Scenarios and Functions

Figure 4-17 exemplifies the cross-references between the application scenarios and the functions within the principle solution. A system with a control functionality that consists of control functions CF1 and CF2 is exemplified here. The system is expected to behave as desired in the two application scenarios

AS1 and AS2. In this case though the overall control functionality of the system remains the same, different control functions are involved in different application scenarios. In application scenario AS1, only the control function CF1 is required. However, both the control functions CF1 and CF2 are indispensable in application scenario AS2. By pointing out which control functions are required in which application scenario, such cross-references reveal if the control functions to be realized by the system vary from one application scenario to another.

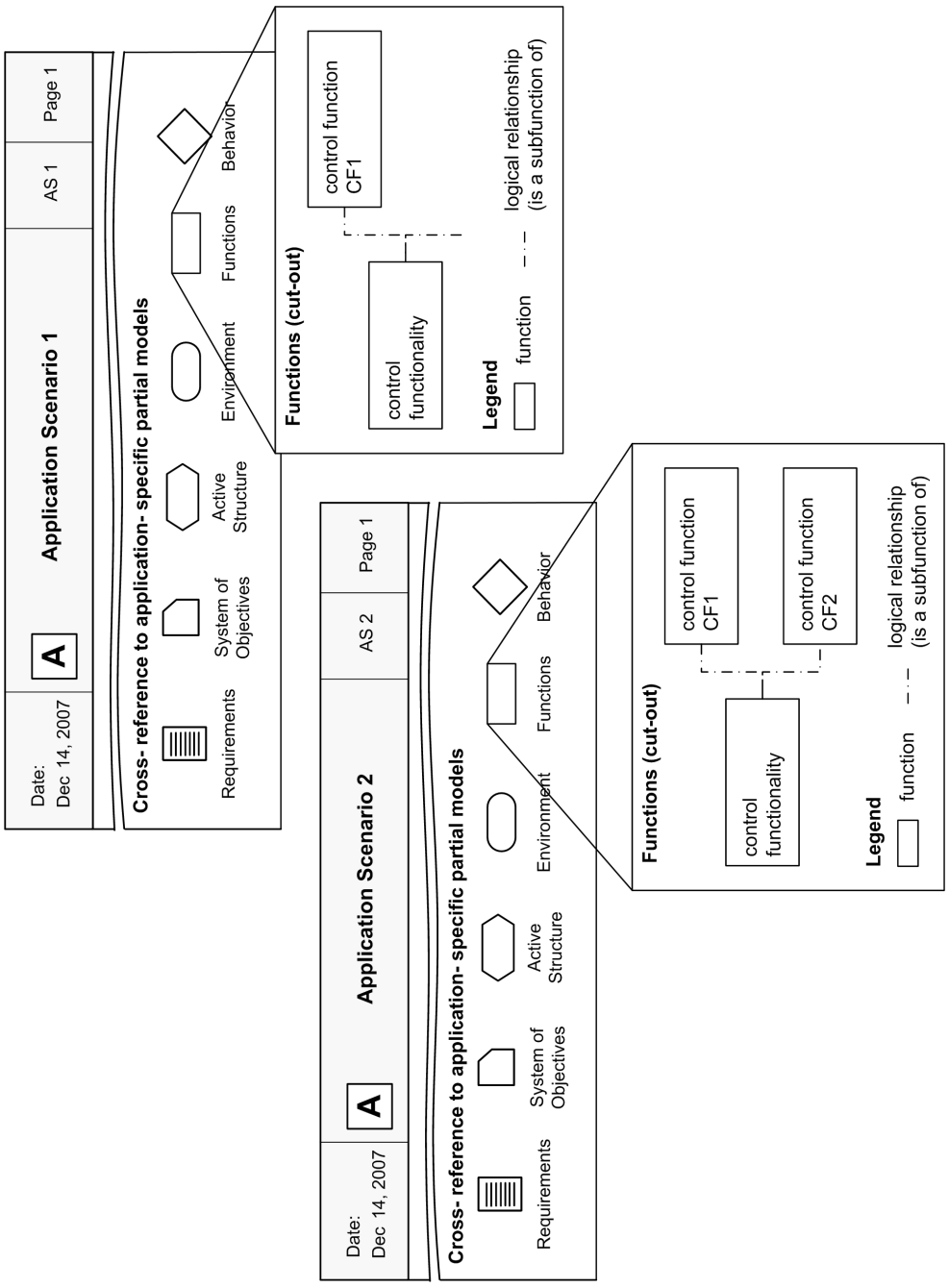


Figure 4-17: Cross-references between application scenarios and functions (cut-out)

4.1.2.3 Cross-references between Functions and Requirements

Figure 4-18 exemplifies the cross-references between the functions and the requirements within the principle solution. A system with two control functions is exemplified here. By cross-referencing between functions and requirements, control functions are linked to the demands and wishes concerning the kinematics, forces, and safety of the system. These demands and wishes determine the control functions to be realized by the system. As shown in the figure, the control function CF1 is determined by the demands regarding the kinematics and the safety of the system, while the control function CF2 is determined by the demands regarding the forces acting on the system. From the requirements list, information about the limitations pertaining to the individual control functions can be traced.

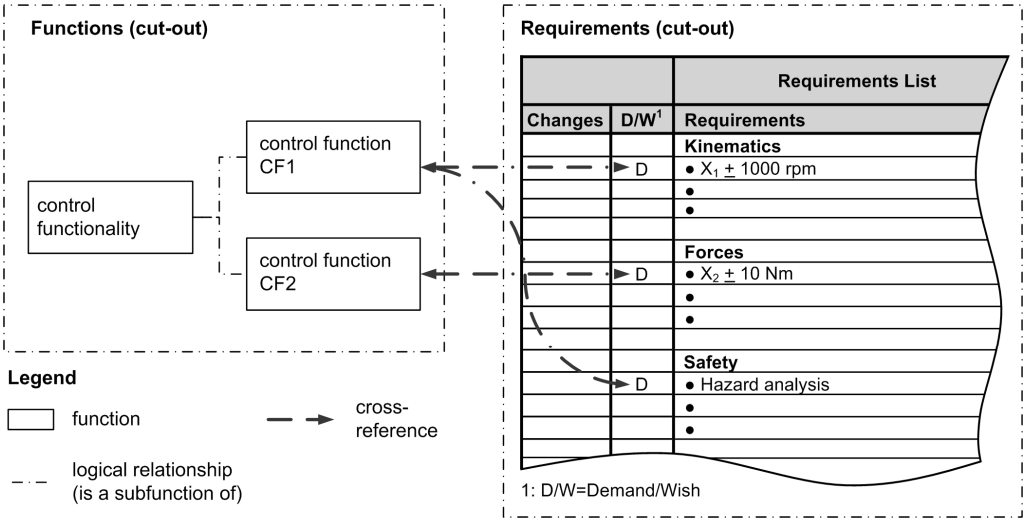


Figure 4-18: Cross-references between functions and requirements (cut-out)

4.1.2.4 Cross-references between Functions and Active Structure

A function is realized by a system element. Cross-references between the functions and the active structure link the control functions in the function hierarchy with the system elements assigned for the realization of the control function. Such cross-references are illustrated in Figure 4-19 and Figure 4-20. There are two fundamental concerns within such cross-references. The first concern pertains to the system variable to be controlled when the system carries out its tasks. The second concern pertains to the interdependencies between the control functions.

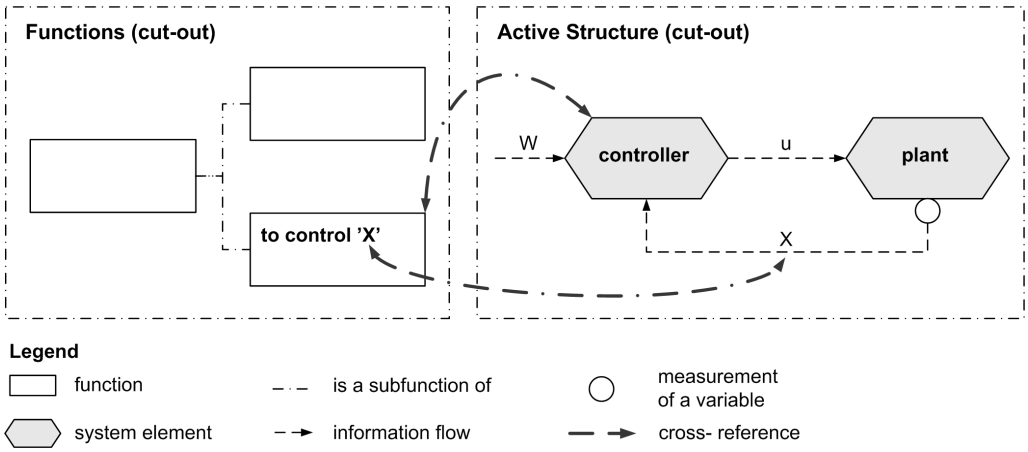


Figure 4-19: Cross-references between functions and active structure — identification of controlled variables (cut-out)

As illustrated in Figure 4-19, the system variable to be controlled when a system carries out its tasks is represented by X. Such variables are referred as the controlled variables which are included in the specification of both the function hierarchy and the active structure. Cross-references can be drawn between the controlled variables specified in the function hierarchy and their counterparts specified in the active structure to make sure that one is suitably matched with another. As such, the conformity between the controlled variables specified in the two partial models can be ensured. Besides preventing flaws in the concept, such cross-references point out the controlled variables which may be overlooked in one of the partial models due to the complexity of the development tasks.

In the function hierarchy, the structuring of the functions is based on the decomposition of an overall function into its subfunctions. In the active structure, the structuring of the system elements is based on the hierarchical structure of advanced mechatronic systems, namely from the level of networked mechatronic system (NMS) to the level of autonomous mechatronic system (AMS) and further on to the level of mechatronic function modul (MFM). Due to the different structuring approaches used in the two partial models, it happens quite often that the system elements at the same hierarchical level in the active structure actually have their corresponding functions specified at different hierarchical levels in the function hierarchy, and vice versa. Cross-references between the function hierarchy and the active structure relate the two different aspects of the system in order to portray an overall system design.

Figure 4-20 exemplifies the cross-references between the function hierarchy and the active structure, where the hierarchical structuring of the control functions in the function hierarchy corresponds to the hierarchical structuring of the controllers in the active structure. The interdependencies between the control functions can be identified from such cross-references. As such, the way the

controllers and the control functions rely on each other in order to perform the control task is clarified. Crucial are the logical relationship between the control functions and the information flow between the inputs and outputs of the system elements.

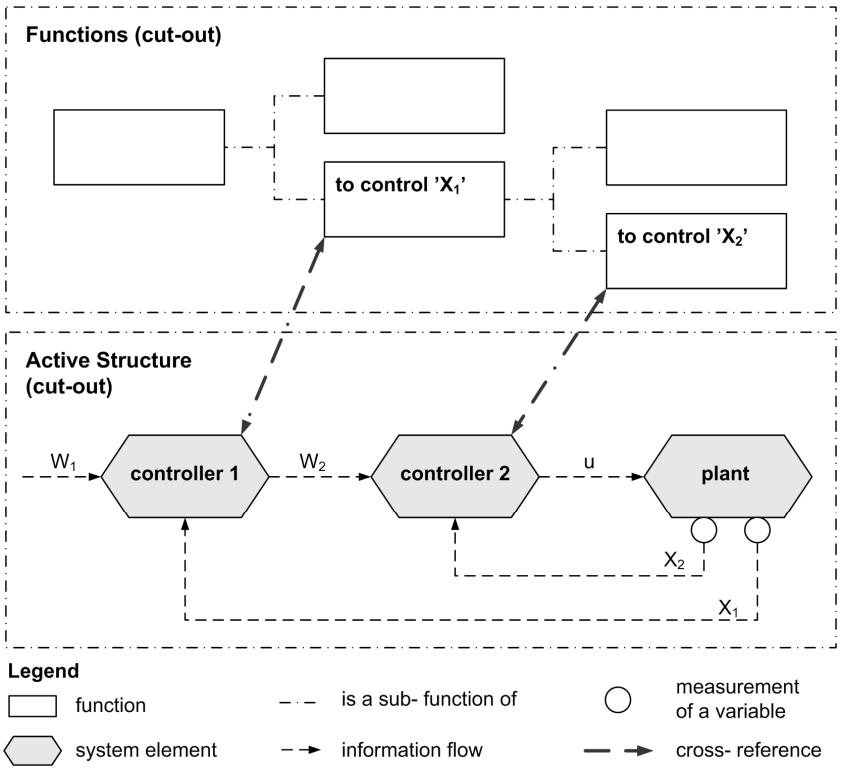


Figure 4-20: Cross-references between functions and active structure — analysis of interdependencies among control functions (cut-out)

4.1.2.5 Cross-References between Active Structure and Environment

Nevertheless, the basic constructs used in the active structure and the environment are similar. The focuses in the active structure are the constellations of system elements and the flows between them. In the partial model environment, the focuses are the influences from the environment as well as influences generated within the system itself. Figure 4-21 exemplifies the cross-references between the active structure and the environment model. As shown in the figure, the influences $I_{1.1}$, $I_{1.2}$, and $I_{1.3}$ specified in the active structure are related to an influence table specified in the partial model environment. Such cross-references reveal further information about the influences acting on the controlled system as well as the source of influences. As such, the impact caused by these influences on the controlled system can be anticipated. Decision has to be made if the undesirable or disturbing impacts caused by the influences have to be compensated. Feedforward control, as illustrated in Figure 4.9 can be added to the active structure in order to compensate for the impacts of disturbing influences on the behavior of the controlled system.

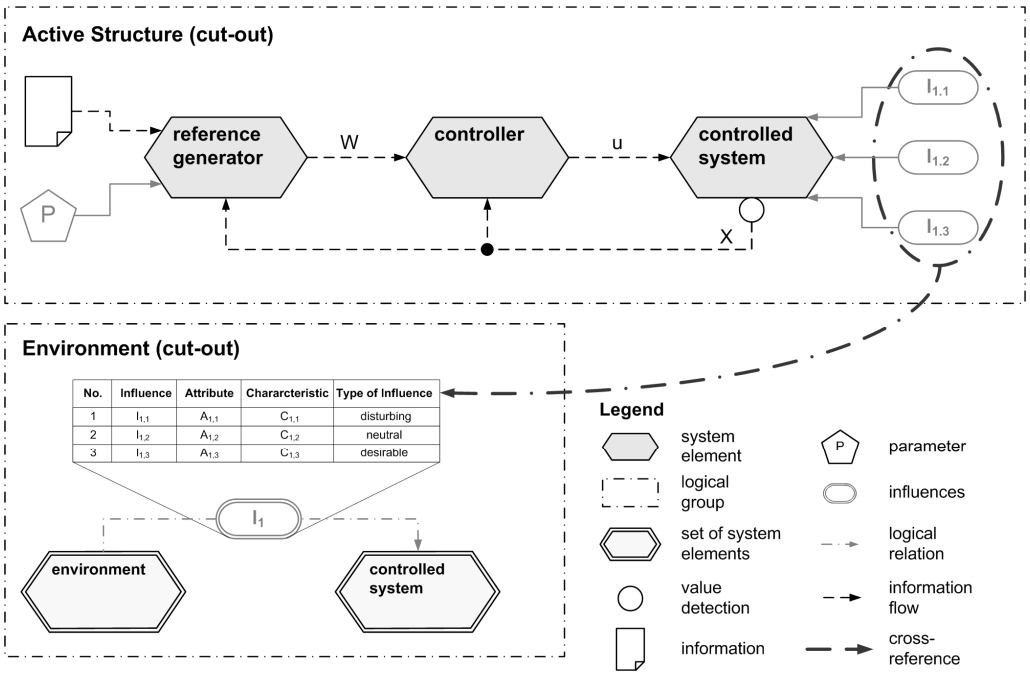


Figure 4-21: Cross-references between active structure and environment (cut-out)

4.1.2.6 Cross-References between Active Structure and Requirements

Cross-references between the active structure and the requirements list reveal the requirements to be fulfilled by the constructs of active structure. Figure 4-22 exemplifies the cross-references between the active structure and the requirements list. As shown in the figure, the constructs of active structure include the system elements, information, parameter, and value detection. Each of these constructs is related to the requirements to be fulfilled by the system. The system element representing the controller has to fulfill a demand which limits a particular aspect regarding the kinematics of the system. The reference generator has to take into account the risk management and the limitation on the variable P when generating the reference values for the controller. Besides that, redundant measurements on the controlled variable X are required. Such cross-references ensure that no requirement is left unidentified and unfulfilled from the beginning of a mechatronic development project.

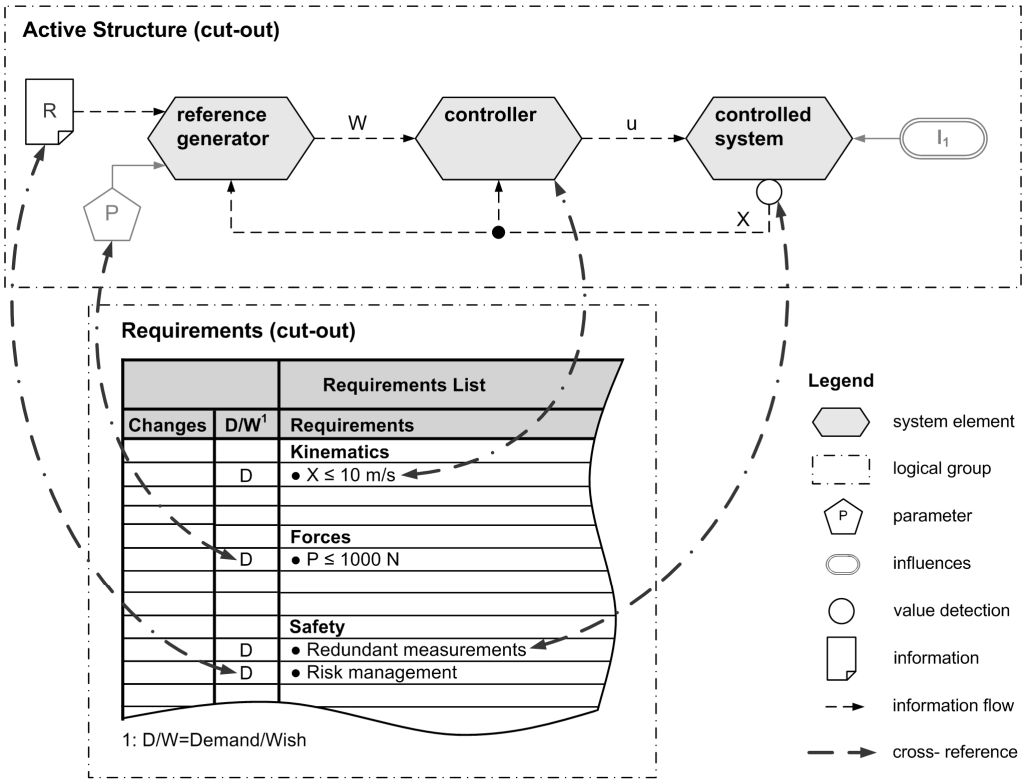


Figure 4-22: Cross-references between active structure and requirements (cut-out)

4.1.2.7 Cross-References between Active Structure and Behavior-States

A state refers to a particular mode of operation of the system. In a particular state, some system elements are activated while the others are deactivated. Cross-references between the active structure and the behaviour-states reveal which system elements have to be activated or deactivated in a particular state of the system. Figure 4-23 exemplifies a system with two discrete states, i.e. state A and state B. The controller A has to be deployed in state A while the controller B has to be deployed in state B. Upon the triggering of a state transition, a switching command is generated and the controller is switched. Such cross-references link the different states of the system with their corresponding controllers in the active structure. The active structure corresponds to a continuous-time representation while the system states correspond to a discrete-time representation. As such, such cross-references can be understood as the conceptual integration of the discrete representation and the continuous representation within the principle solution of advanced mechatronic systems.

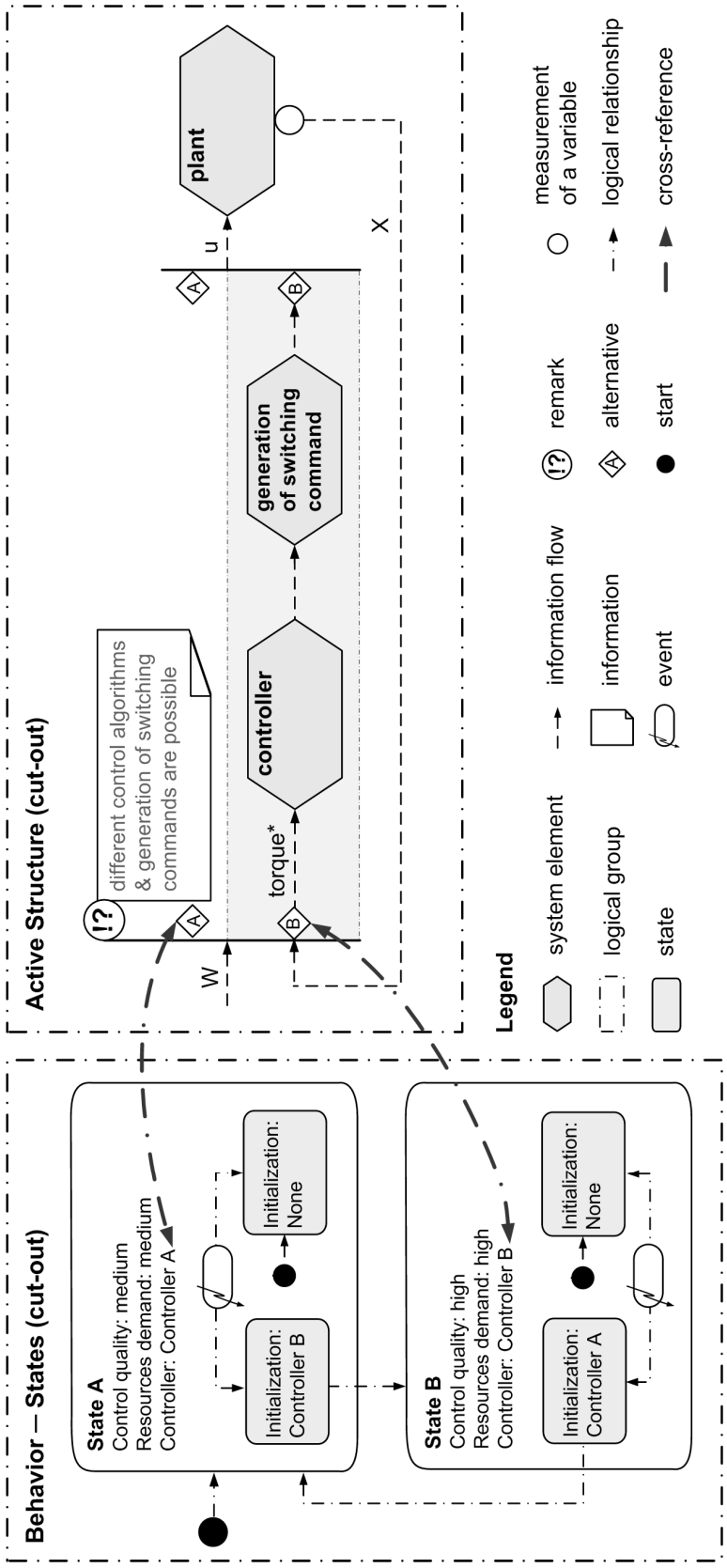


Figure 4-23: Cross-references between active structure and behavior-States (cut-out)

4.1.2.8 Cross-References between Active Structure and Behavior-Activities

Figure 4-24 illustrates the cross-references between the active structure and the behaviour-activities within the principle solution. As shown in the figure, activities are linked to the system elements which carry out them. The activities of a single control loop are exemplified here. The sensor measures the value of the controlled variable X. The controller compares the actual value of X with its desired value W, and subsequently calculates the required control signal u. Finally, the actuator corrects the system dynamics by transforming the control signal into action which eventually brings the actual value of the controlled variable X to its desired value W, or within a limited deviation from the desired value W. These activities are carried out continuously in a loop. Such cross-references between the activities and their corresponding system elements clarify how the controllers work and how they adapt their parameters or structures. Flaws in the control concept can be revealed if the activities do not match with the system elements and vice versa.

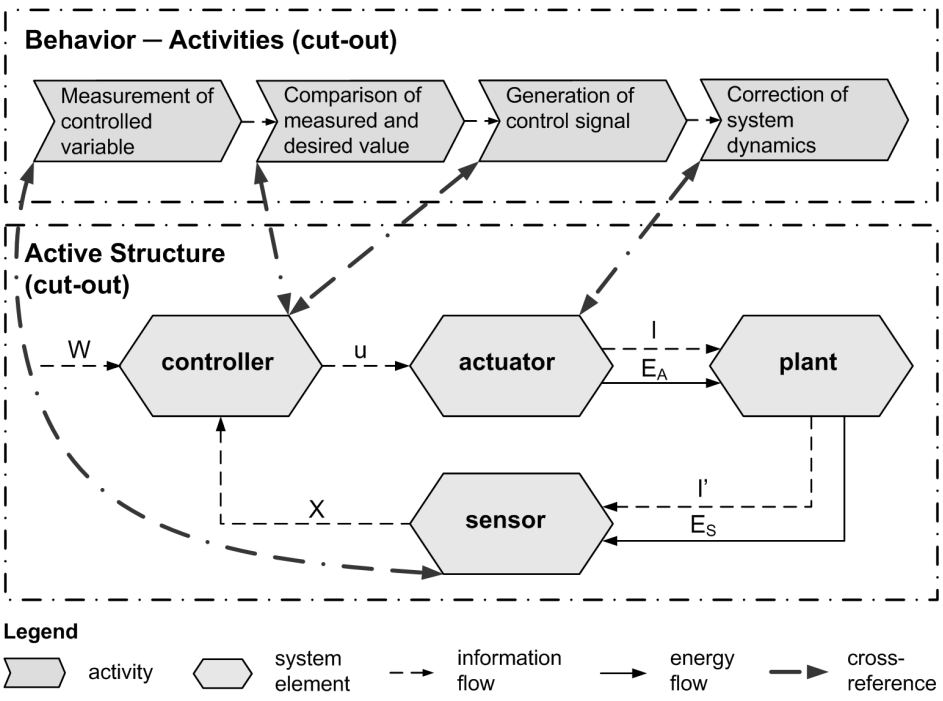


Figure 4-24: Cross-references between active structure and behavior-activities(cut-out)

4.1.2.9 Cross-References within Behavioral Models

The partial model behavior consists of a group of behavioral models. In this section, the **cross-references between partial model behaviour-states and the partial model behaviour-activities** are described. In this context the states or events of the system can be linked to the activities of the system and vice

versa. Such cross-references point out the behavioral adaptation of the system to be developed. Figure 4-25 illustrates such cross-references within the principle solution. A system of which the self-optimization process can be activated or deactivated is exemplified. As shown in the figure, all activities pertaining to the self-optimization process are essential when the system operates in a state which requires self-optimization. Nevertheless, the activities pertaining to the determination of the system objectives are deactivated when self-optimization is not required.

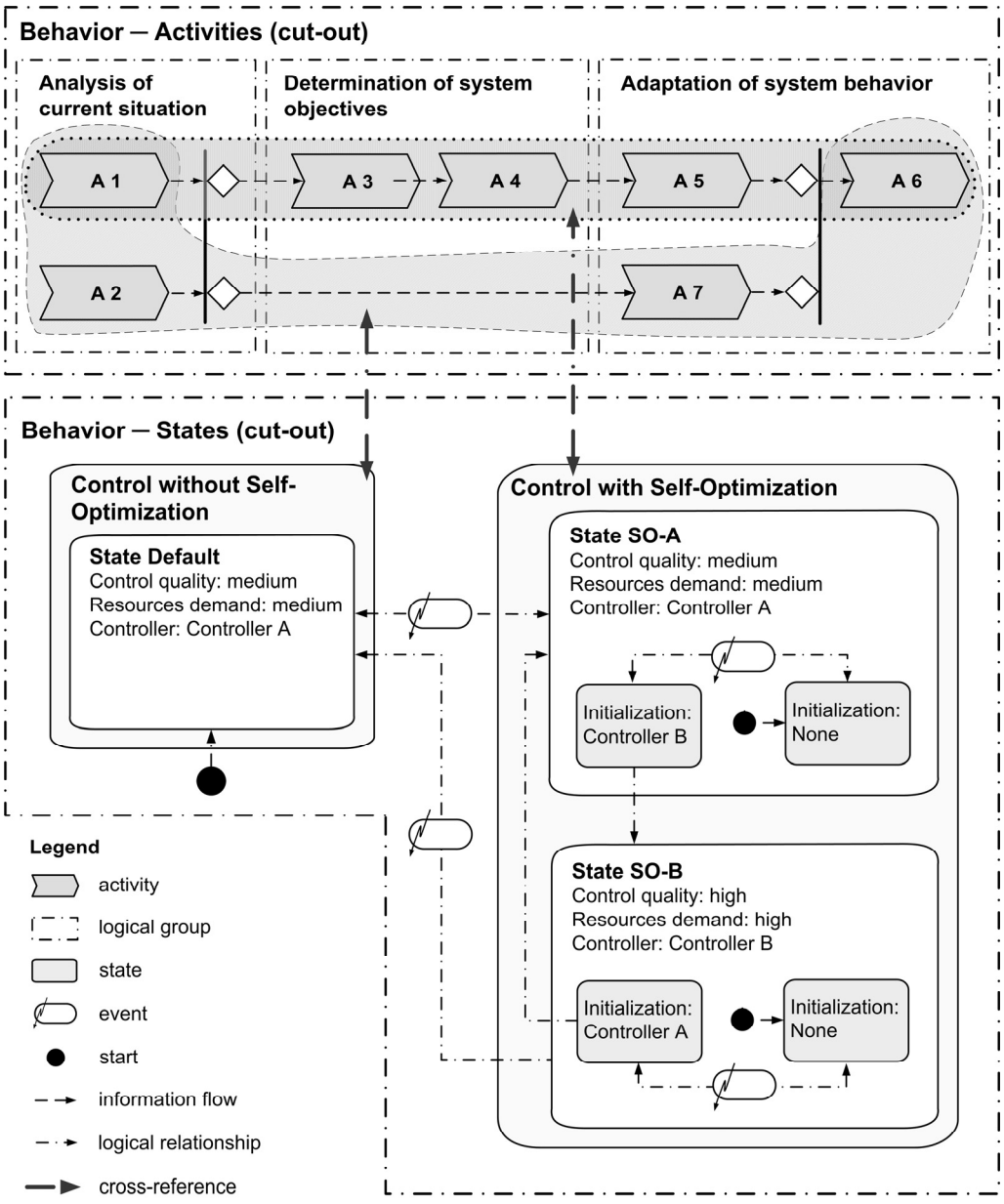


Figure 4-25: Cross-references between behavior-States and behavior-activities(cut-out)

4.2 Managing the Information Extraction from the Principle Solution for the Controller Design of Advanced Mechatronic Systems

Having specified the principle solution, the conceptual design of advanced mechatronic systems is completed. Further concretization takes place with this principle solution as a basis. During the transition from the conceptual design phase towards the concretization phase, an approach to manage the information extraction from the principle solution for the controller design of advanced mechatronic systems is lacking. In order to resolve the the problems defined in Section 2.5.2.2, an approach as illustrated in Figure 4-16 is developed.

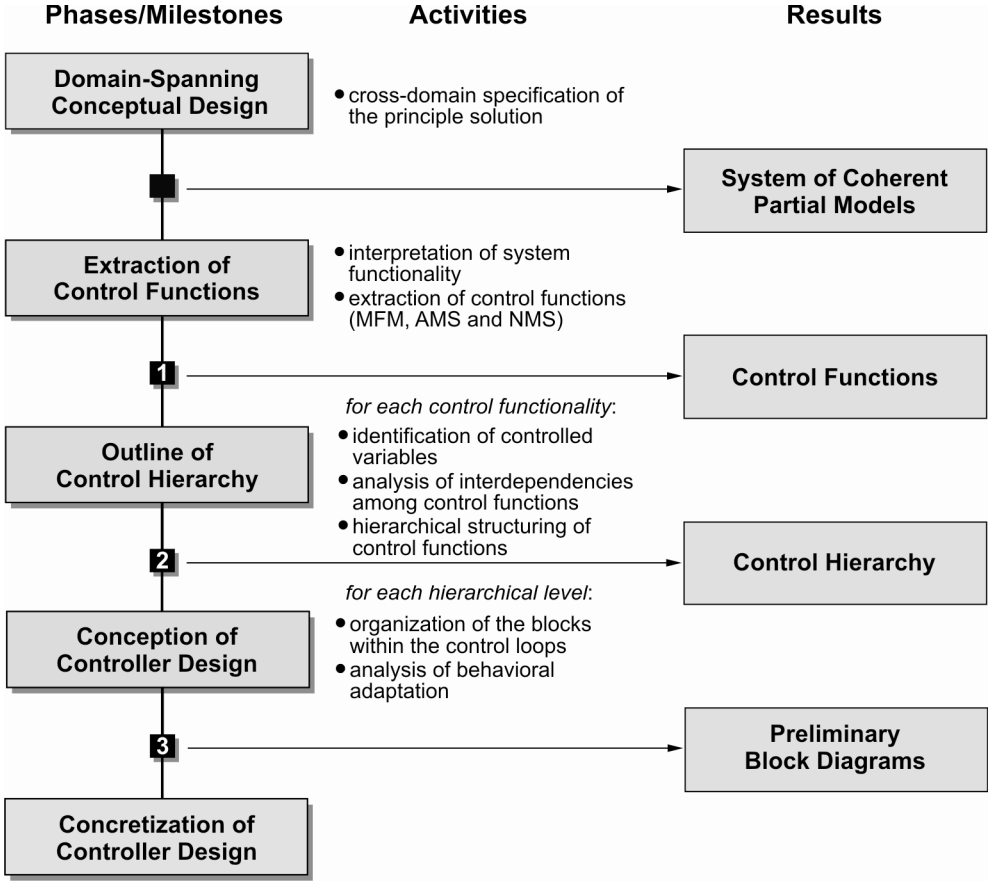


Figure 4-26: An approach to manage the information extraction from the principle solution for the controller design of advanced mechatronic systems

The approach is represented as a procedural model. The procedural model points out the way to identify the control concepts within the principle solution, extract them out of the principle solution, and subsequently transform them into the preliminary block diagrams. Such a procedural model allows the stepwise transition from the principle solution towards the controller design of advanced mechatronic systems. Three transitional phases are involved: extraction

of control functions, outline of control hierarchy, and conception of controller design. Along the transition, the control concepts required in each of the transitional phases are pointed out. These control concepts are extracted from the individual partial models as well as from the cross-references between the partial models. The approach is elaborated in the following subsections.

4.2.1 Extraction of Control Functions

The first transitional phase deals with the extraction of control functions to be realized by the system from the principle solution. As shown in the procedural model in Figure 4-26, the control functions are the outcome of two successive activities. They involve the interpretation of system functionality which is then used to guide the extraction of control functions. These activities are described as follows.

4.2.1.1 Interpretation of System Functionality

With the well formulated principle solution at hand, clear design goals have to be understood by control engineers before starting the concretization of controller design. For this purpose, control engineers have to interpret the functionality of the overall system in the first place. Control engineers are particularly interested in system functionality that is directly related to the correction of the dynamical behavior of the system, as this is the point where the significance of controller design comes in. Besides that, the interpretation of system functionality makes it clear whether the controller design to be carried out is aimed for classical mechatronic systems or self-optimizing mechatronic systems.

Information about system functionality can be extracted from the application scenarios as described in 4.1.1.2, the objectives of the system as described in 4.1.1.4, and the higher-level functions in the function hierarchy as described in 4.1.1.5. Besides that, two cross-references between the partial models have to be referred. They include the cross-references between the application scenarios and the system of objectives as illustrated in Figure 4-16, as well as the cross-references between the application scenarios and the function hierarchy as illustrated in Figure 4-17.

4.2.1.2 Extraction of Control Functions

The understanding of the system functionality is used to guide the extraction of the control functions to be realized at each hierarchical level of the system. As advanced mechatronic systems require multiple control algorithms, the control functions of the system have to be grasped by the control engineers early on.

The main reference for the extraction of control functions is the function hierarchy as described in 4.1.1.5. In some cases, the control functions are explicitly specified in the function hierarchy. In most of the cases, this may not be so direct. Table 4-2 lists down the functions in the function hierarchy, which can be control functions, or may involve a control function.

Table 4-2: List of functions that can be or may involve a control function

- to control *X*
- to regulate *Y*
- to adjust *Z*
- to maintain *M*
- to maximize *N*
- to minimize *O*
- to coordinate *P*

  etc

In cases where the control functions are not explicitly specified, cross-references as illustrated in Figure 4-18 between functions and their counterparts in the requirements list should be traced. If none of their counterparts in the active structure and the requirements list indicates that these functions serve for the purpose of control, they are excluded from the list of control functions. The precise insight into their control functions will help enable the decomposition and the integration of control solutions in the later stages.

4.2.2 Outline of Control Hierarchy

The second transitional phase deals with the outline of a control hierarchy. As shown in the procedural model in Figure 4-26, a control hierarchy is the outcome of three successive activities. These activities include the identification of controlled variables, the analysis of the interdependencies among the control functions, and the hierarchical structuring of the control functions. These activities are described as follows.

4.2.2.1 Identification of Controlled Variables

Various system variables are involved in order to realize the control functions of advanced mechatronic systems. Though with the same overall control functionality, the controlled variables involved can be different from one system to another depending on the physical characteristics of the plant and the selected sensors and actuators. All controlled variables of the system have to be extracted from the principle solution. These controlled variables are stated in the control functions in the function hierarchy as described in 4.1.1.5, and the system elements as well as the information flows between the system elements in

the active structure as described in 4.1.1.6. Besides that, cross-references between the function hierarchy and the active structure have to be referred. Such cross-references are illustrated in Figure 4-19.

4.2.2.2 Analysis of Interdependencies among Control Functions

The various control functions of advanced mechatronic systems rely on each other in order to perform a control task. Obviously, there are interdependencies among these control functions. Having extracted the control functions and identified the controlled variables, the interdependencies among the control functions have to be analyzed. The main interdependencies can be identified from the cross-references between the extracted control functions and their associated system elements in the active structure. Such cross-references are illustrated in Figure 4-20.

The interdependencies among the control functions characterize the coordination among the control algorithms, which are responsible for implementing the control functions. Such interdependencies partly determine how the control algorithm should be derived. The analysis of these interdependencies ensures the effective coordination among the control algorithms. A dependency can be, for instance, a strong mechanical coupling between two control functions. Without knowing this dependency, the two controllers of this strongly coupled mechanical structure may have a coordination problem. In the worst case, these two controllers may act against each other. The outcome can be that the strongly coupled mechanical structure is strained unnecessarily and/or too much actuator energy is consumed. This is against the aim of mechatronics to synergistically improve the behavior of technical systems. The type of interdependency between the control functions differs from case to case depending on the specification of the principle solution for the system to be developed.

4.2.2.3 Hierarchical Structuring of Control Functions

The hierarchical structuring of control functions puts the control functions into the form of a control hierarchy as an effort to facilitate further concretization of the control concepts specified in the principle solution. In the control hierarchy, the control functions in the function hierarchy are integrated with their corresponding system elements in the active structure. In this context, the interdependencies among the control functions previously analyzed are used as a guideline. Further decomposition or aggregation of the control functions can be involved, if necessary. The aim is a control hierarchy that can be implemented accordingly. As such, a basic understanding of the physical laws governing the plant will be an added advantage. This is the point where the first step of domain-specific concretization comes in. References to the proven controller de-

sign of standard applications can be made, if necessary. For instance, the controller with the fastest dynamics must be put at the lowest level of the control hierarchy.

During the hierarchical structuring of control functions, all the control functions, the controlled variables as well as their interdependencies must be taken into consideration. The outcome of this phase is a well structured hierarchy of control functions, as illustrated in Figure 4-27. Except the control functions at the uppermost and the lowermost levels, all the other control functions in a control hierarchy are two-faced entities. This means that they are a lower-level entity of the superimposing control function and a higher-level entity of the underlying control function at the same time. Going from the top to the bottom of the control hierarchy, the overall control task is decomposed into partial control tasks. On the contrary, going from the bottom to the top of the control hierarchy, individual solutions are combined to form a coherent overall solution. With such a control hierarchy, controllers can be designed and implemented to realize each of the control functions with their functional interdependencies across the hierarchical levels ensured.

4.2.3 Conception of Controller Design

The third transitional phase deals with the conception of controller design in the form of preliminary block diagrams. As shown in the procedural model in Figure 4-26, the preliminary block diagrams are the outcome of two successive activities. They involve the organization of the blocks within the control loops as well as the analysis of the behavioral adaptation of the system. These two activities are described as follows.

4.2.3.1 Organization of the Blocks within the Control Loops

Block diagram representation is the widely accepted specification technique deployed for controller design. In order to effectively bridge the gap between the principle solution and the controller design, the control concepts specified within the principle solution has to be transformed into the preliminary block diagrams. As such, the preliminary block diagrams conform to the specification in the principle solution. This preliminary block diagram serves as the initial controller layout for the system to be developed.

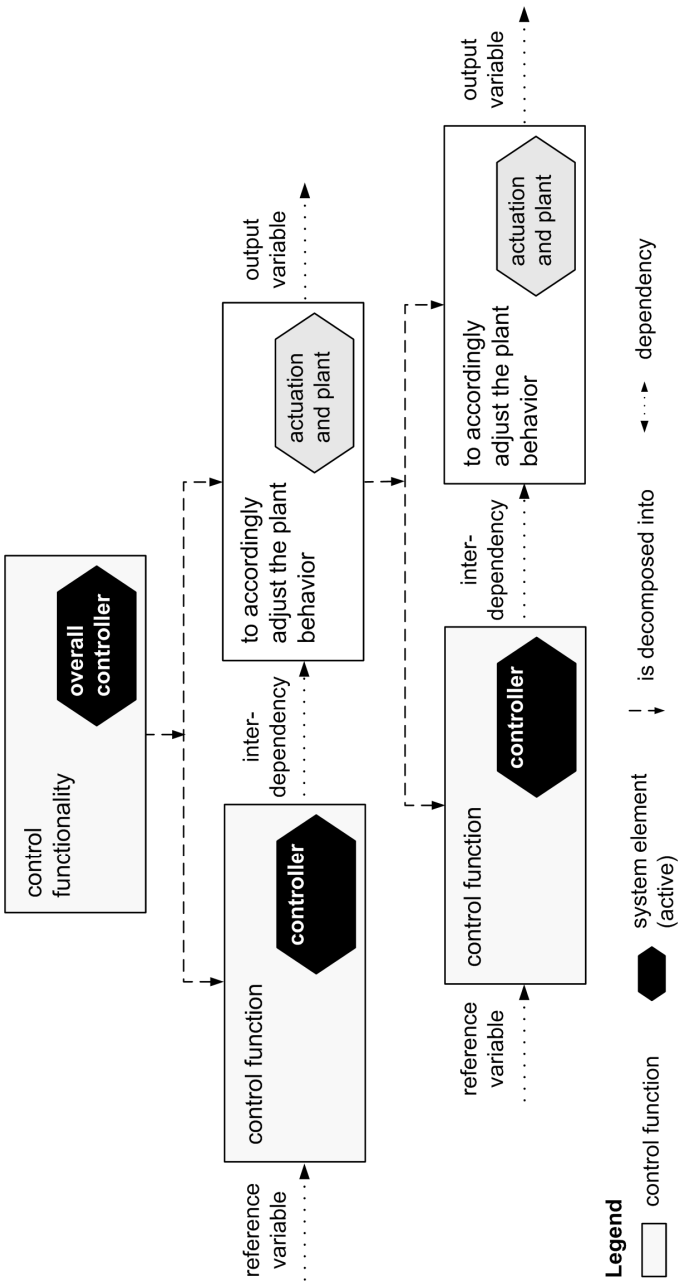


Figure 4-27: The representation of a control hierarchy

The basic control structures in the active structure as described in 4.1.1.6 serves as the main reference for the organization of the blocks within the control loops in the preliminary block diagram. In this context, the control elements, correction elements, process elements and measurement or estimation elements are transformed into the blocks in the block diagram. These blocks are connected together by means of information flows which include the feedback loops, as per the way they are specified in the active structure.

Nevertheless, the one-to-one mapping between the system elements in the active structure and the blocks in the block diagram is only possible in idealized cases. In practice, such a one-to-one mapping is rare and the control hierarchy

as illustrated in Figure 4-27 has to be referred. The control hierarchy supplements the essential information regarding the layout of the preliminary block diagrams. In the preliminary block diagram, the blocks and the information flows must be completely labelled. This includes the indication of the summation or the contraction of information flows.

Besides the active structure, cross-references from the active structure to the environment model and the requirement lists are required. Figure 4-21 and Figure 4-22 illustrate the cross-references between these partial models for the organization of the blocks within the control loops.

4.2.3.2 Analysis of Behavioral Adaptations

Having extracted a basic control structure in the form of preliminary block diagram, the task here is to enhance this basic structure so that it becomes conformal to the bahavioral adaptation required by the system.

It involves the addition of blocks such as reference generator, controller switch, or the block for the adaptation of controller parameter. A block representing a reference generator is used to generate the reference values for the controlled variables of advanced mechatronic systems, which can be a fixed set-point or a changing reference profile. A block representing a controller switch is used for the reconfiguration of the blocks representing the controller structures. A block for parameter adaptation is used to adjust the parameter of the blocks representing the controllers. Such means for behavioral adaptations can be required at different hierarchical levels of the system and should be specified in the preliminary block diagram.

Besides the active structure, cross-references between the active structure and the states of the system, between the active structure and the activities of the system, as well as between the states and the activities of the system have to be referred. These cross-references are illustrated in Figure 4-23, Figure 4-24, and Figure 4-25 respectively. Hence, the preliminary block diagrams are made conformal to the bahavioral adaptations required by the system. In this context, the blocks to be activated or deactivated in a particular state as well as the activities to be carried out by the blocks which result in the behavioral adaptations of the system are clarified.

4.3 Concretization of Controller Design

Concretization of controller design involves modeling, analysis, and synthesis of the controllers. By applying formal design techniques, the control concepts specified within the principle solution are realized.

Control technique: As pointed out in [BSK+06], control techniques range from the linear single-input single-output control, state-space methods, adaptive control, fuzzy control, neural networks, nonlinear control, H_2 and H_∞ designs, optimization algorithms, up to model predictive control. Among the established controller design techniques, there are heuristic and analytical approaches. On one hand, heuristic approaches refer to systematic approaches for parameter tuning such as the well known Ziegler-Nichols method. On the other hand, analytical approaches refer to the formula-based algorithms (e.g. coefficient comparison for time constants and cross-ratio) and the graphic-based algorithms (e.g. pole-placement) [Sch01].

Modelling & Simulation: In order to ensure a structured controller design process and to prevent design errors, the controlled system is modelled and simulated together with the controllers. This is required to predict the behavior of the non-linear part of the system and the uncertainties, which can be examined by simulation. The concretization of controller design involves activities such as mathematical modeling, specification of operating points, linearization, parameterization, or designing the details of the feedback controllers, prefilters, feedforward controllers, controller switch, etc. These activities are usually assisted by software tools which deploy extensive numerical simulations.

Software tool: The current industry standard for controller design is the MATLAB/Simulink and Stateflow. On one hand, Simulink is a graphical block diagramming tool for modeling, simulating and analyzing dynamical systems. On the other hand, Stateflow enables the graphical representation of hierarchical and parallel states and the event-driven transitions between them. Modeling reconfiguration of controller structures is achieved by adding discrete blocks, whose behavior is specified by statecharts, to the block diagrams. In order to exchange between controller structures, two types of switching are possible. A switching which can take place between two computational steps (atomic switching) [SPH+07], or a switching which can be specified by a fading function and an additional parameter which determines the duration of the cross fading [Vöc03].

Implementation: Controllers are realized in either continuous or discrete time. As such, an analog or a digital target-platform may be used. A time-continuous realization is usually implemented in hardware, for example, by combinations of operational amplifier and other electrical devices like resistors and capacitors. Multiple possibilities of time-discrete realization exist, for instance, using a Field Programmable Gate Array (FPGA) or software implementation [Bur06, p. 45]. The initial information about the implementation of the controllers can be traced from the logical relationship labelled by “running on” between the system elements in the active structure.

Test: The concretization of controller design is done in parallel with the concretization in the domains of mechanics, electric/electronics, and software engineering. A laboratory prototype of the advanced mechatronic system can be built for test purposes. For instance, the controllers and their switching concepts can be validated by the Hardware-in-the-Loop-Tests (HiL-Test). Test in a real plant application is the final step of the design concretization of the controller.

5 Application Examples

Chapter 5 exemplifies the method presented in the preceding chapter using two application examples. As stated in Section 2.3, the demonstrators consist of a self-optimizing motor drive and an autonomous railway convoy. The self-optimizing motor drive is an application example at the level of mechatronic function module while the autonomous railway convoy is an application example at the level of networked mechatronic system. Both application examples feature recent advances in their respective fields of drive technology and railway technology. They are the current demonstrators of the Collaborative Research Center 614 “Self-Optimizing Concepts and Structures in Mechanical Engineering”.

Both application examples demand complex information processing for adapting the parameter and where necessary the structure of the system. During the conceptual design phase of such systems, a valid concept for the control of the system is of paramount significance and has to be systematically structured. Following the method presented in the preceding chapter, this chapter exemplifies how the control concepts for both the application examples have to be specified within their principle solutions during the conceptual design phase. Subsequently, the management of information extraction from the principle solution for the controller design is exemplified. In such a way, the feasibility of the aforementioned method in practice is validated.

The principle solution of both the application examples is the outcome of continuous collaboration with the Institute of Power Electronics and Electrical Drive, University of Paderborn.

5.1 Self-Optimizing Motor Drive

An introduction for the self-optimizing motor drive is given in Section 2.3.1. As the continuation from Section 2.3.1, this section further describes the application example in three subsections. At the beginning, the basic control concepts to be specified within the principle solution of the self-optimizing motor drive are described. Subsequently, selected cross-references between the partial models are exemplified. At the end, the management of the information extraction from the principle solution for the controller design of the self-optimizing motor drive is described.

5.1.1 Specifying the Basic Control Concepts within the Principle Solution of a Self-Optimizing Motor Drive

As the outcome of the conceptual design phase, the domain-spanning principle solution for the self-optimizing motor drive is specified. This principle solution serves as a generalized solution concept which can be adapted for different applications, such as hydraulic pumps or as electrical part in the drive train of a hybrid car. Except the partial model shape, each of the partial models is described in the following.

5.1.1.1 Environment

Figure 5-1 exemplifies the partial model environment which describes the relevant areas of influences within the surrounding of the self-optimizing motor drive. The motor drive interacts with the load machine, the power supply system, the ambient temperature, and the other applications running alongside the motor drive. In such an environment, the influences acting on the motor drive are the load torque, the supply voltage, the ambient temperature, the resource storage, as well as the wear and tear of the motor drive itself. These influences have to be taken into consideration during the conceptual design phase as they can be undesirable or disturbing for the control of the angular dynamics of the motor drive. Undesirable influences for the motor drive are the fast changing load torque, insufficiently low voltage, extremely high ambient temperature, irregular resource storage, and severe wear and tear.

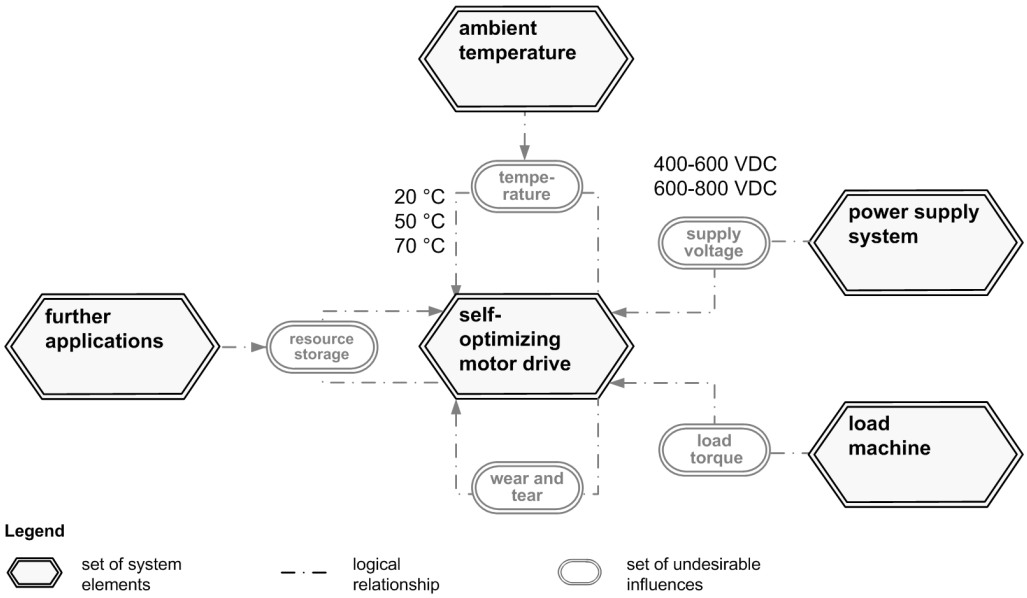


Figure 5-1: Environment of a self-optimizing motor drive (cut-out)

5.1.1.2 Application Scenarios

Figure 5-2 and Figure 5-3 exemplify the main application scenarios of the self-optimizing motor drive. In the first application scenario, the resource consumption has to be minimized while the motor drive drives a constant load at a constant speed. In the second application scenario, the control quality has to be maximized while the motor drive accelerates or drive a dynamic load at a constant speed. The description of the partial development task for the application scenarios is stated at the upper part of Figure 5-2 and Figure 5-3 respectively.

Besides the description in prose, a sketch about the application scenario is presented at the middle part of each figure. As shown in the sketch, the different controller structures for the motor drive are stored in a controller library. Each of the controller structures represents a different controller. An Operator Controller Module (OCM) as described in Section 2.2.2 is deployed. Depending on the actual operational and the environmental condition, an appropriate controller has to be determined for the motor drive. The selected motor drive controller runs alongside the other applications on the same computation platform, i.e. the Central Processing Unit (CPU) or the Field-Programmable Gate Array (FPGA). That means the controllers have to compete for the available resources with the other applications.

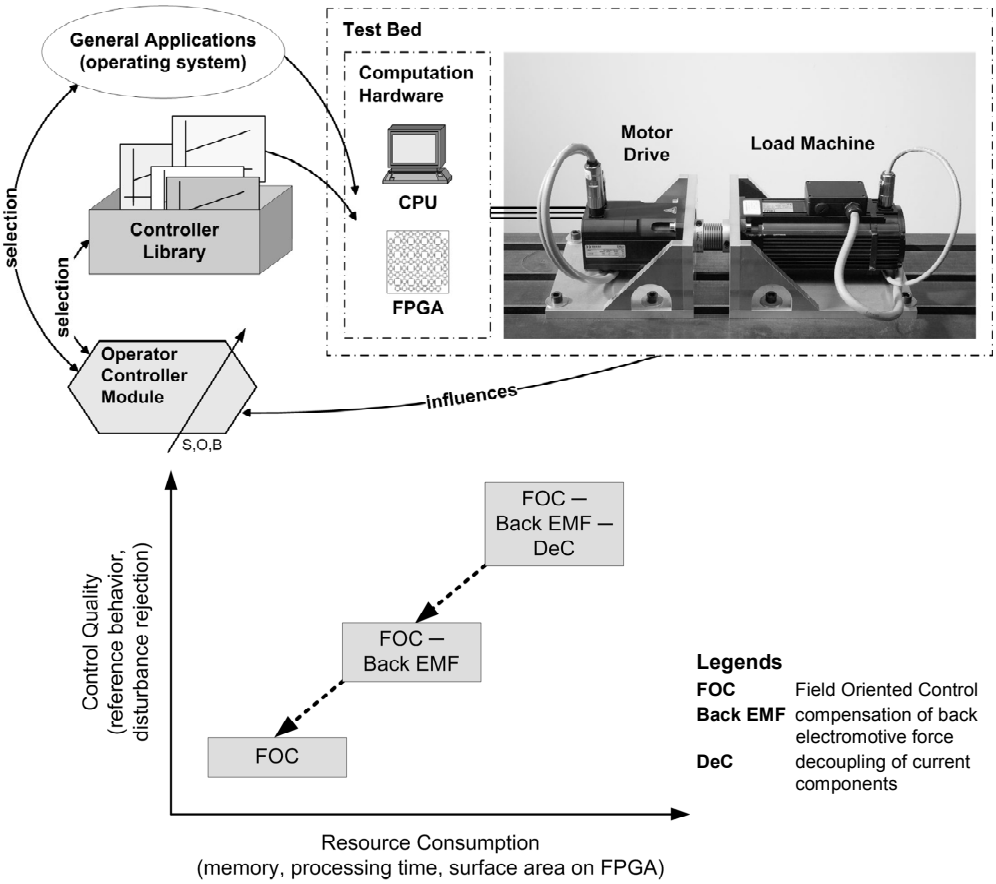
As shown in the graph, different controllers demand different amount of resources and deliver different degrees of control quality. Control quality refers to how good the motor drive follows the reference behavior and how good is the disturbance rejection. Resources refer to the available memory, the available CPU time, as well as the surface area of hardware-logic-cells available on the FPGA. An application or a controller can only be activated if there are sufficient resources available in the system. The resources are allocated during run time depending on whether the applications are compulsory to be executed, can be temporarily suspended or can be totally avoided.

Date: July 19, 2007	A	**Application Scenario** Minimization of Resource Consumption (constant load and constant speed)	**AS 1**	Page 1

Description of partial development task

In this application scenario, the motor drive drives a constant load at a constant speed. Obviously, it is not resource efficient to always use a high performance controller since it demands higher resource consumption. The other controllers which demand lower resource consumption should be used if the actual operating condition does not require such a high performance. For this purpose, the maximum resources available in the system have to be determined. If there are fewer resources available in the system than that required by a resource intensive controller, that controller cannot be loaded. As such, the choices of possible controllers are reduced. From the set of possible controllers, the controller which exhibits both the highest suitability in the current operating point and the lowest resource consumption are selected. In the case where several controllers exhibit a comparable resources demand, the selection of controller should be made upon the consideration of the best control quality.

Sketch



Cross-references to application-specific partial models

Requirements | System of Objectives | Active Structure | Environment | Functions | Behavior

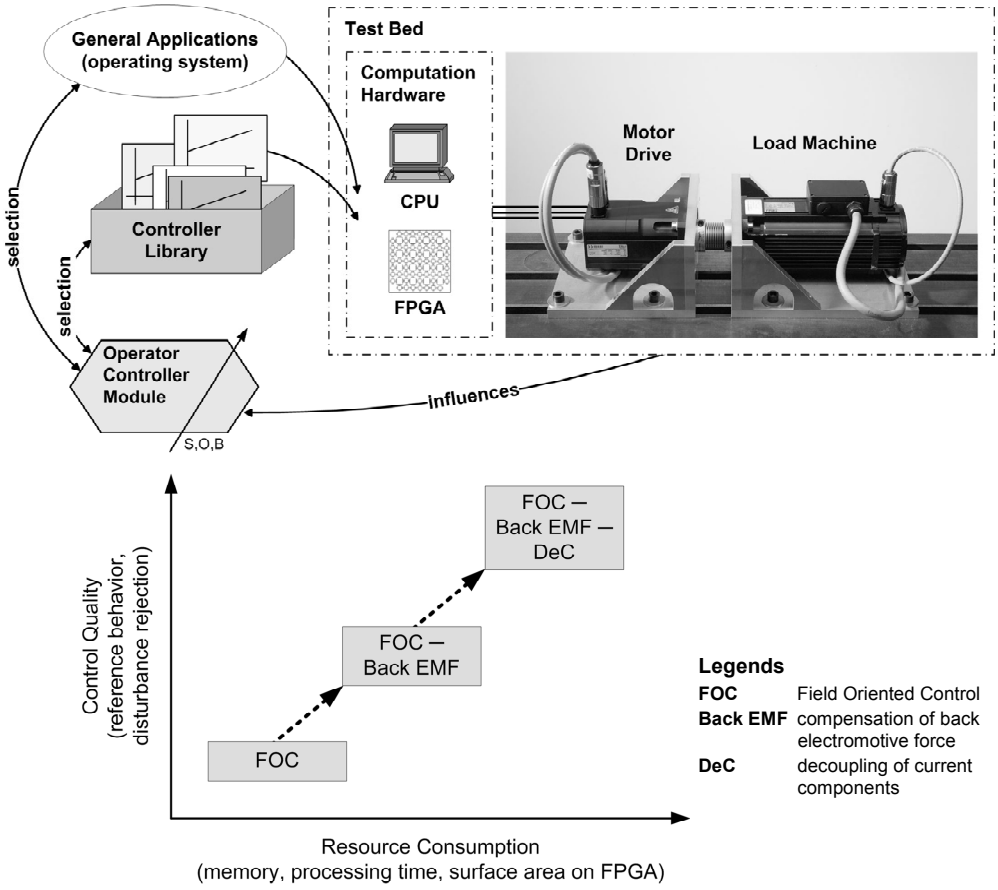
Figure 5-2: Application scenario for the minimization of resource consumption of a self-optimizing motor drive (cut-out)

Date: July 19, 2007	A	**Application Scenario** Maximization of Control Quality (changing load or accelerating)	**AS 2**	Page 1

Description of partial development task

In this application scenario, the motor drive either accelerates or be controlled at a constant speed under a changing load condition. In such circumstances, a simple controller cannot effectively perform the control task. The control quality of the motor drive has to be maximized. For this purpose, the maximum resources available in the system have to be determined and used as a criterion for the selection of a suitable controller structure. Instead of using the controller with the lowest resource consumption, the controller which suits the respective operating speed range and the driving torque the most will be selected. Subsequently, unused resources shall be freed and used for other applications such as the visualization tasks. In the case where several controllers exhibit comparable control quality, the controller with the lowest resource demand should be selected.

Sketch



Cross-reference to application-specific partial models

Requirements | System of Objectives | Active Structure | Environment | Functions | Behavior

Figure 5-3: Application scenario for the maximization of the control quality of a self-optimizing motor drive (cut-out)

5.1.1.3 Requirements

Figure 5-4 exemplifies a cut-out from the requirements list of the self-optimizing motor drive. The drive shaft of the permanent magnet synchronous motor should be able to rotate in both the clockwise and the counterclockwise directions. The nominal velocity of the motor drive is 3000 revolutions per minute. The maximal velocity of the motor drive is 6000 revolutions per minute. The nominal driving torque produce by the motor drive is 3.5 Nm while the overload torque produced is 5 Nm.

Date: July 20, 2007			**Requirements List** Self-Optimizing Motor Drive (overall system)	Page: 1
Changes	D/W[1]		Requirements	Responsible
		2	**Kinematics**	LEA
	D	2.1	Direction of motion: clockwise/counterclockwise	
		2.2	Range of angular velocity	
	D	2.2.1	Nominal velocity +/- 3000 revolutions per minute	
	W	2.2.2	Maximal velocity +/- 6000 revolutions per minute	
		3	**Forces**	LEA
		3.1	Range of torque	
	D	3.1.1	Nominal torque +/- 3,5 Nm	
	W	3.1.2	Overload torque +/- 5 Nm	

1: D/W = Demand/Wish

Figure 5-4: Requirements list of a self-optimizing motor drive (cut-out)

5.1.1.4 System of Objectives

Figure 5-5 exemplifies the system of objectives for the self-optimizing motor drive. The adaptation of the system of objectives distinguishes the self-optimizing motor drive from the classical industrial drives.

The self-optimizing motor drive has two inherent objectives, i.e. the reliable realization of the reference value of the driving torque as well as the reliable operation of the self-optimization process. On one hand, the reliable realization of the reference torque can be achieved by maintaining a high control performance, by keeping a high energy magnitude reserve for the manipulated variable of the controller, and by keeping the dependency on the system parameters low. On the other hand, the reliable operation of the self-optimization process can be assured by an efficient self-optimization process as well as by avoiding frequent switching and/or toggling on and off between the controller structures.

The external objective of the self-optimizing motor drive is to minimize the consumption of system resources. This external objective can be achieved by keep the demand for memory, computing power of the CPU, and surface area on the FPGA at a low level.

The internal objectives have to be derived from the inherent and external objectives of the motor drive. These internal objectives of the self-optimizing motor drive are omitted here as they involve mathematical derivations. Eventually they are the objective functions used for optimization. They have to be weighted by means of a fuzzy-rule base to determine the actual objective function to be pursued by the motor drive.

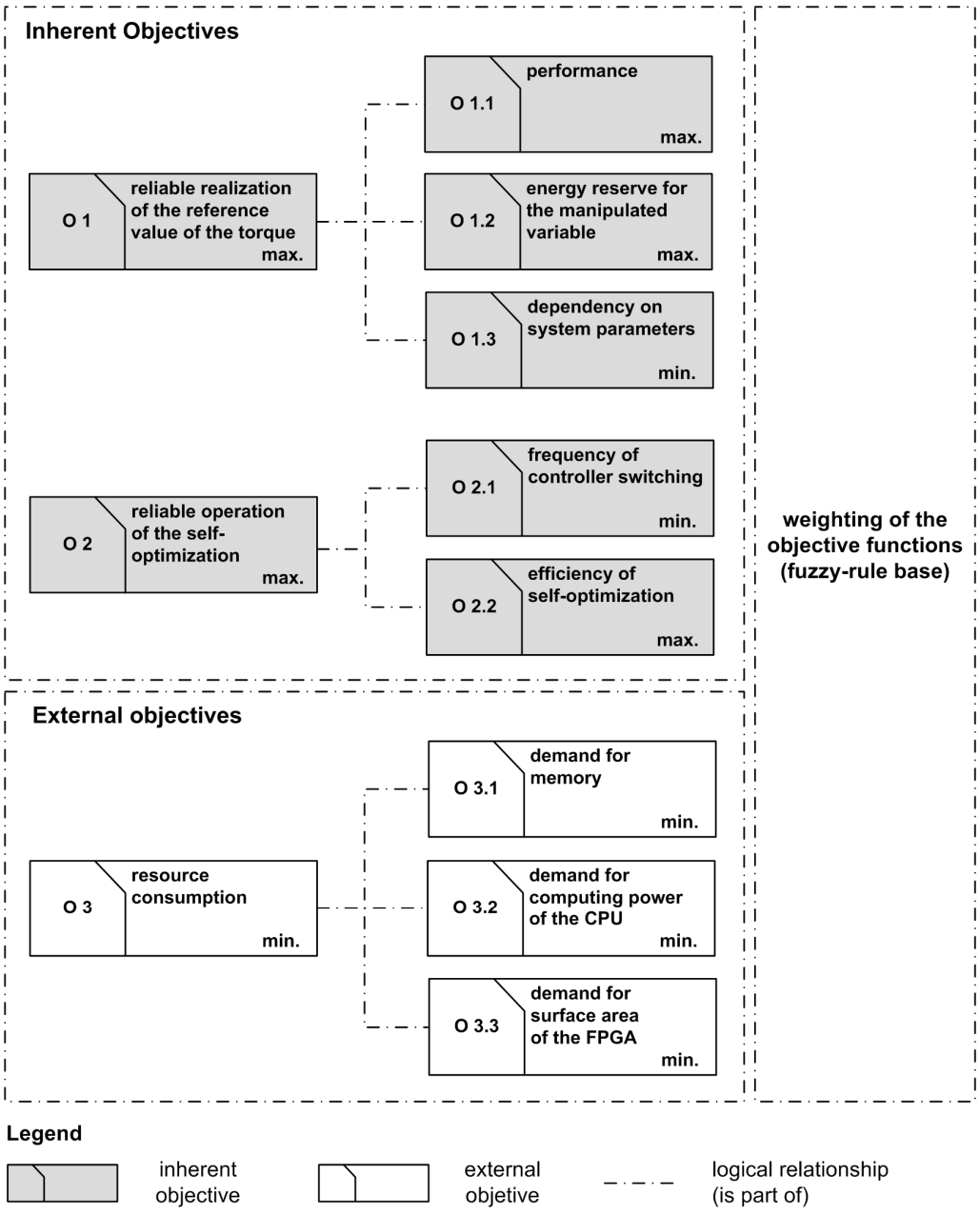


Figure 5-5: System of objectives of a self-optimizing motor drive (cut-out)

5.1.1.5 Functions

Figure 5-6 exemplifies the function hierarchy of a self-optimizing motor drive. As per the explanation in Section 4.1.2, the overall function of the motor drive is broken down into subfunctions.

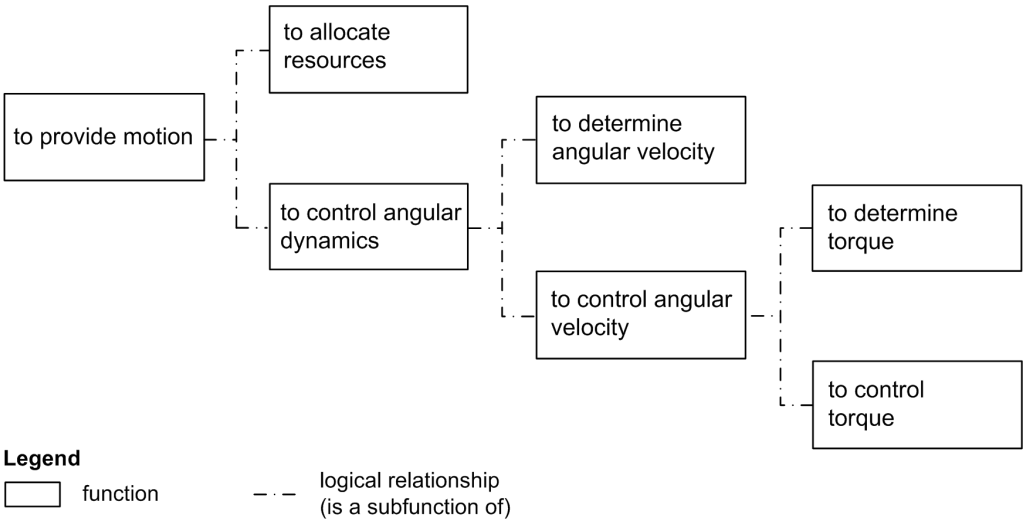


Figure 5-6: Function hierarchy of a self-optimizing motor drive (cut-out)

A motor drive provides regulated motion in technical systems by converting electrical power into mechanical power. In this application example, the self-optimizing motor drive is able to provide angular motion for a large number of applications. For this purpose, the central functionality required is the control of the angular dynamics of the motor drive. During the operation of the motor drive, the drive shaft has to rotate at a specific velocity or within a certain range of velocity. Depending on the changing operational and environmental conditions, the drive shaft may have to rotate faster or slower. The change of the angular velocity is resulted from the change in the driving torque. As such, both the angular velocity and the driving torque of the motor drive have to be controlled.

5.1.1.6 Active Structure

Figure 5-7 exemplifies the active structure of the self-optimizing motor drive. The motor drive is a mechatronic function module (MFM) which consists of an Operator Controller Module, power electronics, and a permanent magnet synchronous motor. Besides that, the motor drive is equipped with a resource management module, a CPU, a FPGA module, and a load machine. The system elements are linked by means of the flows of information and energy.

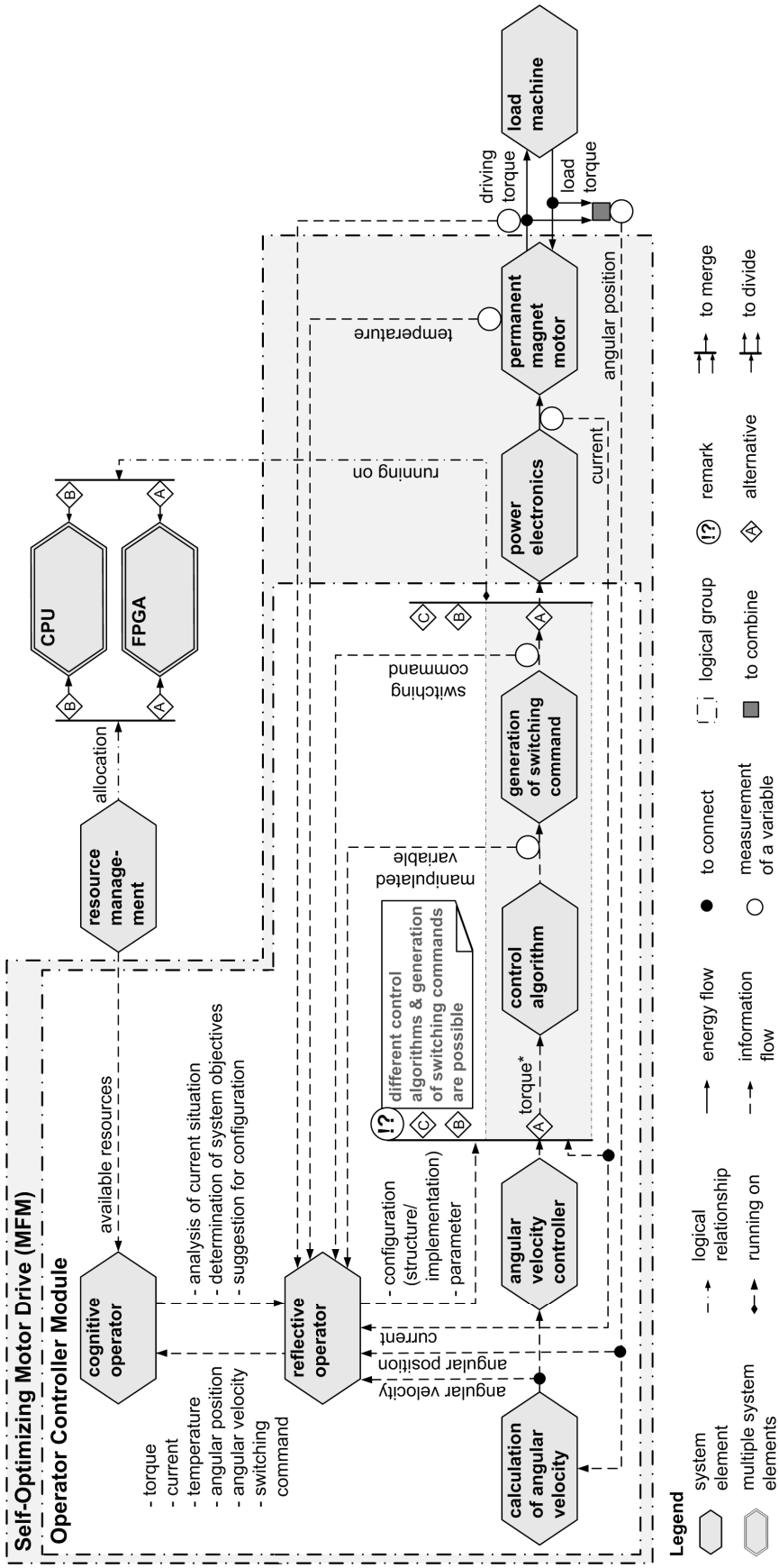


Figure 5-7: Active structure of a self-optimizing motor drive (cut-out)

The driving torque generated by the permanent magnet motor acts against the load torque and the frictional torque. Hence, angular motion is produced. The different modes of operation are emulated by varying the torque or the angular velocity of the load machine with respect to time. The other applications running alongside the motor drive are not real physical devices, but only simulations on the CPU. The simulations include the different test patterns of the memory demands, the required computing time, the surface area on the FPGA, as well as the performance of the applications.

The Operator Controller Module consists of the cognitive operator, the reflective operator, and the systems elements of the controller. The controller controls the angular dynamics of the motor drive and has to fulfill hard real-time requirements. The controller structures can be reconfigured in response to the. state transitions and the underlying adaptive processes. The switching between the controllers is represented by the alternative controller structures labeled with A, B, and C in the active structure. On top of the controller, the reflective operator activates the switching of the controller structures. It is event oriented and operates in hard real-time. At the highest level, the cognitive operator uses a preemptive optimization to improve the behavior of the motor drive in terms of control quality and resource consumption. It decides the most suitable controller structure to be deployed. It does not interact in real-time with the permanent magnet motor.

In this application example, the motor drive controller can be implemented in two ways. Both the CPU-based implementation and the FPGA-based implementation are possible. A CPU-based implementation implies a software controller while a FPGA-based implementation implies a hardware controller. The functions of the controller can be totally (or partly) implemented on the CPU or the FPGA. The run time switching between the different implementations of the controller is done by means of partial run-time reconfiguration [SPH+07].

5.1.1.7 Behavior – States

Figure 5-8 exemplifies the states and the state transitions within a self-optimizing motor drive. As the self-optimization process can be activated and deactivated, the motor drive can either be in a state featuring control with self-optimization or without self-optimization. Without self-optimization, the control quality and the resources demand are fixed depending on the standard design process. In this case, the control quality and resources demand are fixed in the medium range in the state without self-optimization. In the state of control with self-optimization, the motor drive has the ability to switch between three different modes of operation. These modes are the constant load operation mode, the acceleration operation mode, and the dynamic load operation mode.

Each of these operation modes can be denoted an inner state. The inner states include the State SO-A, State SO-B, and State SO-C. The transitions between the inner states are determined by the self-optimization process.

The control quality and the resources demand of the controller vary from one state of the motor drive to another. State SO-A is the default state where a medium control quality is required while a medium percentage of system resources are consumed. From State SO-A, a transition into State SO-B with a higher control quality and resources demand or into State SO-C with a lower control quality and resources demand is possible. The same principle applies to the transitions from State SO-B and State SO-C. As the control quality is proportional to the resources demand, a specific controller structure can be assigned to each of the states SO-A, SO-B, and SO-C. As shown in the figure, each of the states is assigned a controller structure, each of them with low, medium, or high control quality and resources demand respectively. A proper labelling of the controller structures eases the distinction between the different controller structures involved in each of the states.

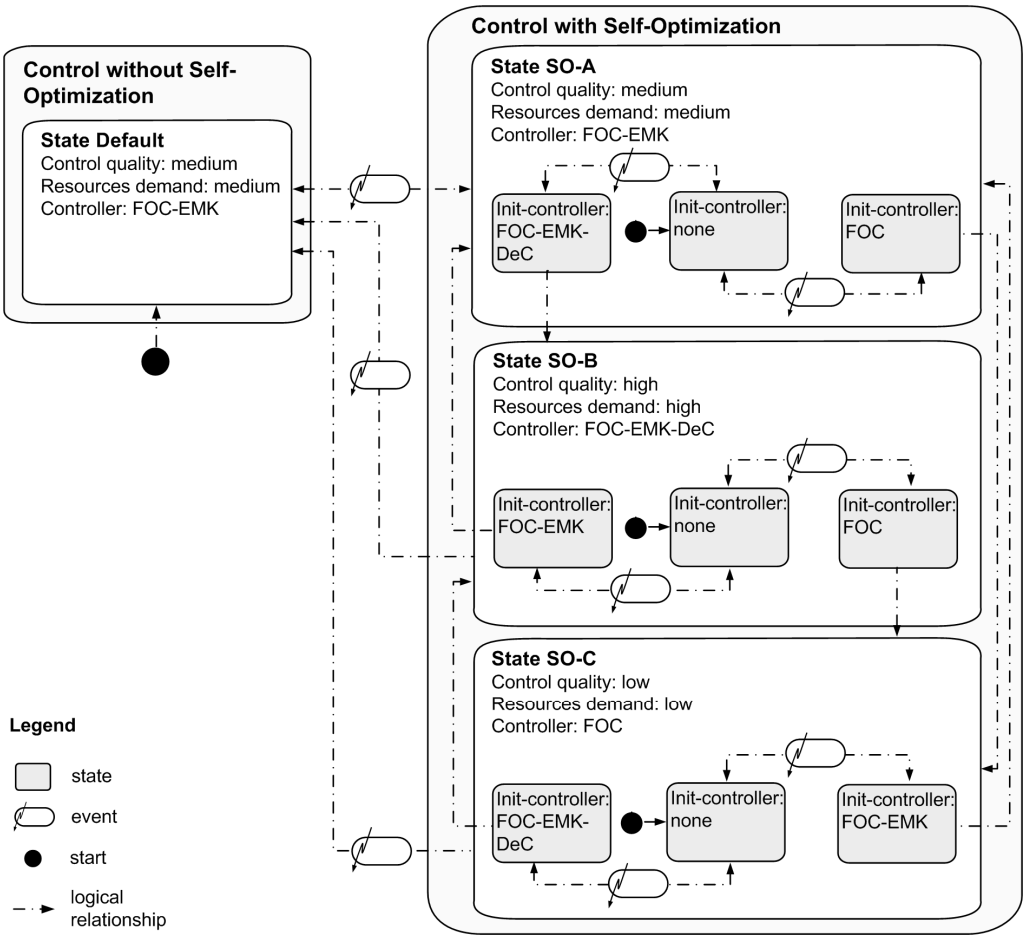


Figure 5-8: Different states of a self-optimizing motor drive (cut-out)

5.1.1.8 Behavior – Activities

Figure 5-9 exemplifies the behaviour activities of a self-optimizing motor drive. The activities of the motor drive are organized according to the self-optimization process, i.e. analysis of the current situation, determination of the system objectives, and adaptation of the system behavior.

The analysis of the current situation involves the analysis of the current state of the motor drive as well as the observations of its environment. The analysis starts with three parallel activities: inquiries of the resources available in the system, evaluation of the currently active controller, and prediction of the probable system behavior. This is followed by the acquisition of the environmental influences acting on the system as well as the influences within the system itself. Subsequently, the objective functions of the motor drive have to be weighted for the determination of the system objectives. The weighting of the objective functions is carried out by means of a fuzzy-rule base. As the operating condition and the environment change, the rules to determine an objective function can also be changed. The adaptation of system objectives leads to the adaptation of system behavior. If necessary, a new controller will be loaded, initialized, and subsequently activated. Finally, the unused resources will be released.

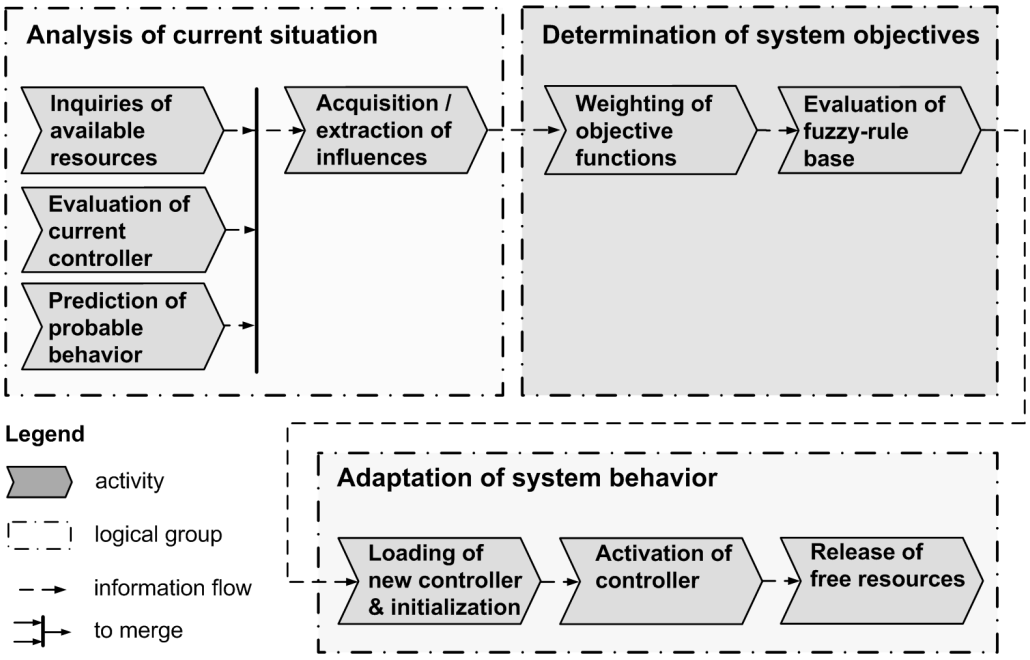


Figure 5-9: Activities of a self-optimizing motor drive (cut-out)

5.1.1.9 Cross-references between the Partial Models of the Principle Solution of a Self-Optimizing Motor Drive

Five cross-references between the partial models of the principle solution of the self-optimizing motor drive are exemplified as the following.

Cross-references between application scenarios and system of objectives as well as functions: Figure 5-10 and Figure 5-11 exemplify the cross-references from the application scenarios of a self-optimizing motor drive to its system of objectives as well as its functions. Though the functions of the motor drive remain unchanged in the application scenarios, the objective pursued by the motor drive has to be adapted from one application scenario to another. The external objective O3 is active in the application scenario AS1 while the inherent objectives O1 and O2 are active in the application scenario AS2. As the application scenario changes, the self-optimizing motor drive has to provide angular motion that suits the objective currently pursued by the system. In application scenario AS1, the resource consumption of the motor drive is to be minimized while controlling the angular dynamics of its drive shaft. In application scenario AS2, the control quality of the motor drive is to be maximized while controlling the angular dynamics of its drive shaft.

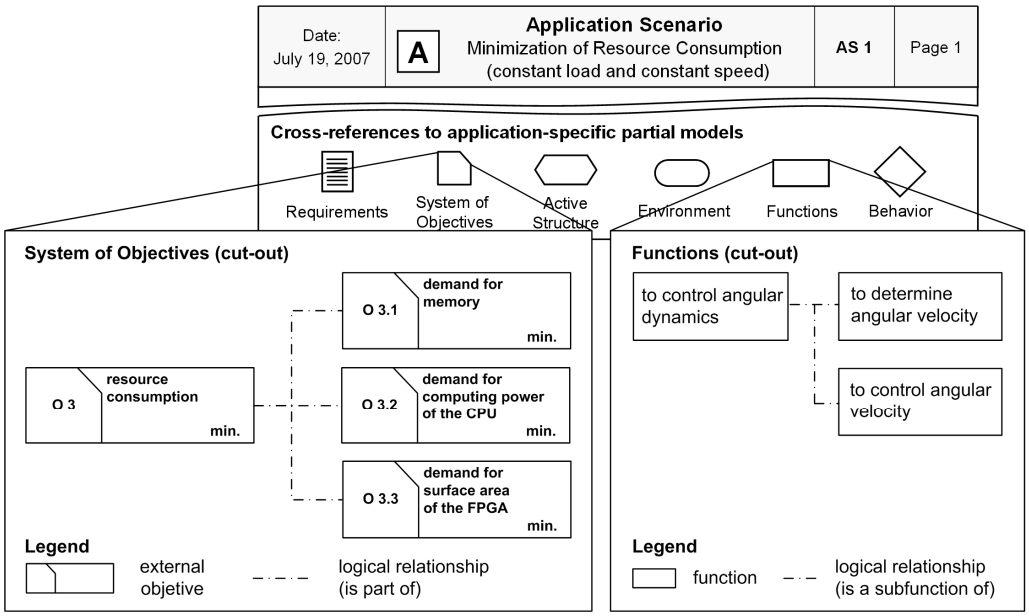


Figure 5-10: Cross-references from application scenario 1 to system of objectives and function hierarchy (cut-out)

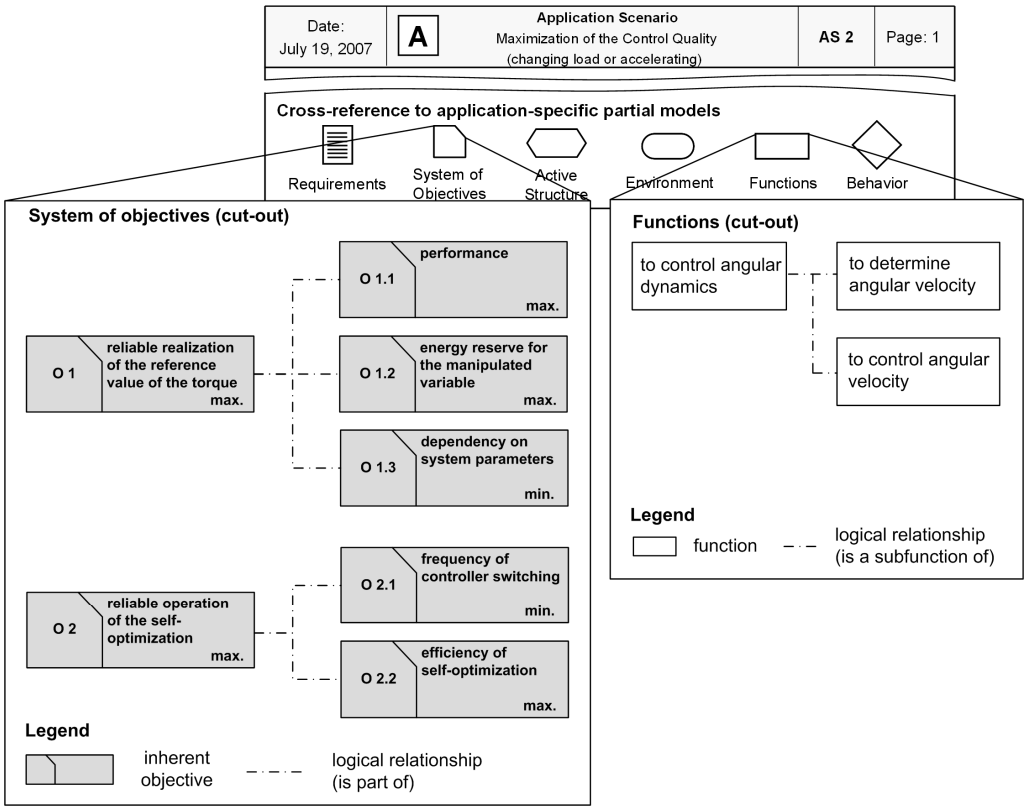


Figure 5-11: Cross-references from application scenario 2 to system of objectives and function hierarchy (cut-out)

Cross-references between functions and active structure: Figure 5-12 exemplifies the cross-references between the functions and the active structure of a self-optimizing motor drive. As illustrated in the figure, such cross-references link the control functions of the motor drive with its system elements assigned for the realization of the control functions. At the lowest level of the function hierarchy, three alternative control structures are assigned for the control of the driving torque of the motor drive. At the middle level of the function hierarchy, an angular velocity controller is assigned for the control of the angular velocity of the motor drive. As a whole, an Operator Controller Module is assigned for the control of the angular dynamics of the motor drive under changing operational and environmental conditions. The controlled variables specified in the function hierarchy correspond to their counterparts specified in the active structure. The interdependencies among the control functions are also clarified by such cross-references.

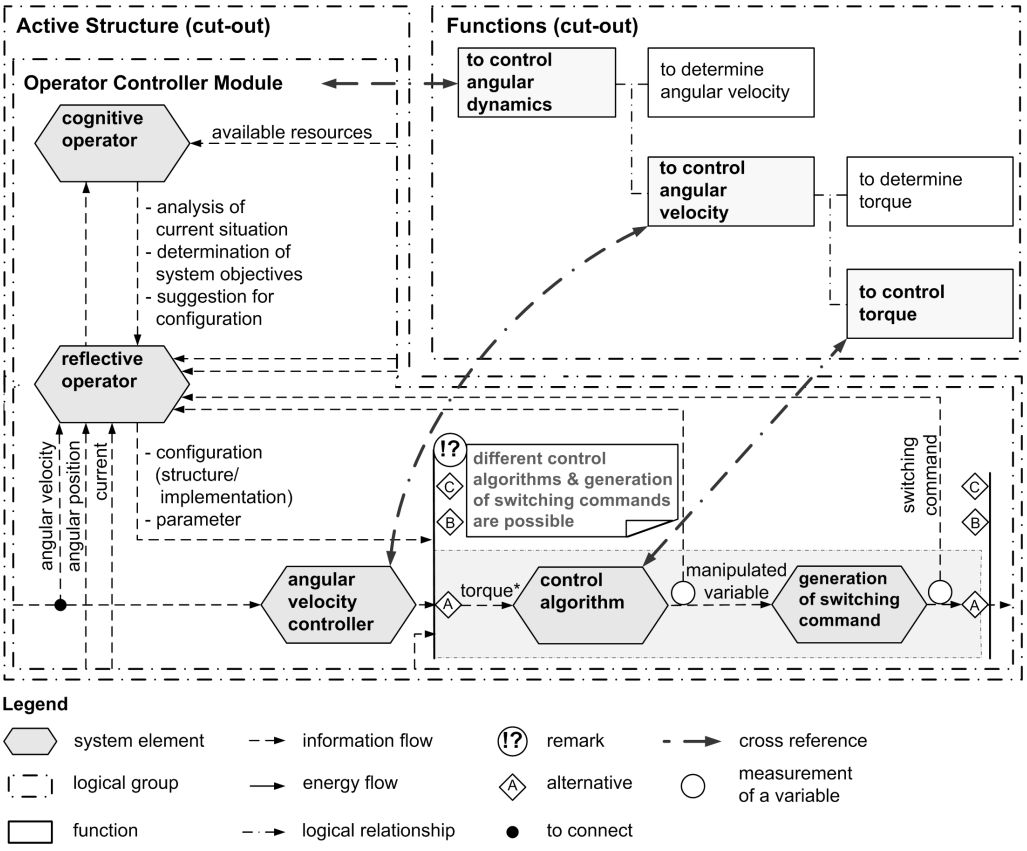


Figure 5-12: Cross-references between function hierarchy and active structure (cut-out)

Cross-references between active structure and requirements: The demands and wishes to be fulfilled when realizing the control functions of the self-optimizing motor drive can be revealed by the cross-references between the active structure and the requirements list of the motor drive. Figure 5-13 exemplifies such cross-references for the self-optimizing motor drive. As for the angular velocity controller, it is a demand that the angular velocity of the motor drive to be controlled at a nominal value of 3000 revolutions per minute in both the clockwise and counterclockwise directions. It is a wish that the motor drive could reach a maximal velocity of 6000 revolutions per minute. As for the control algorithm assigned for the control of the driving torque of the motor drive, it is a demand that driving torque to be controlled around a nominal value of 3.5 Nm while it is a wish that the driving torque could still be controlled around 5 Nm under an overload condition.

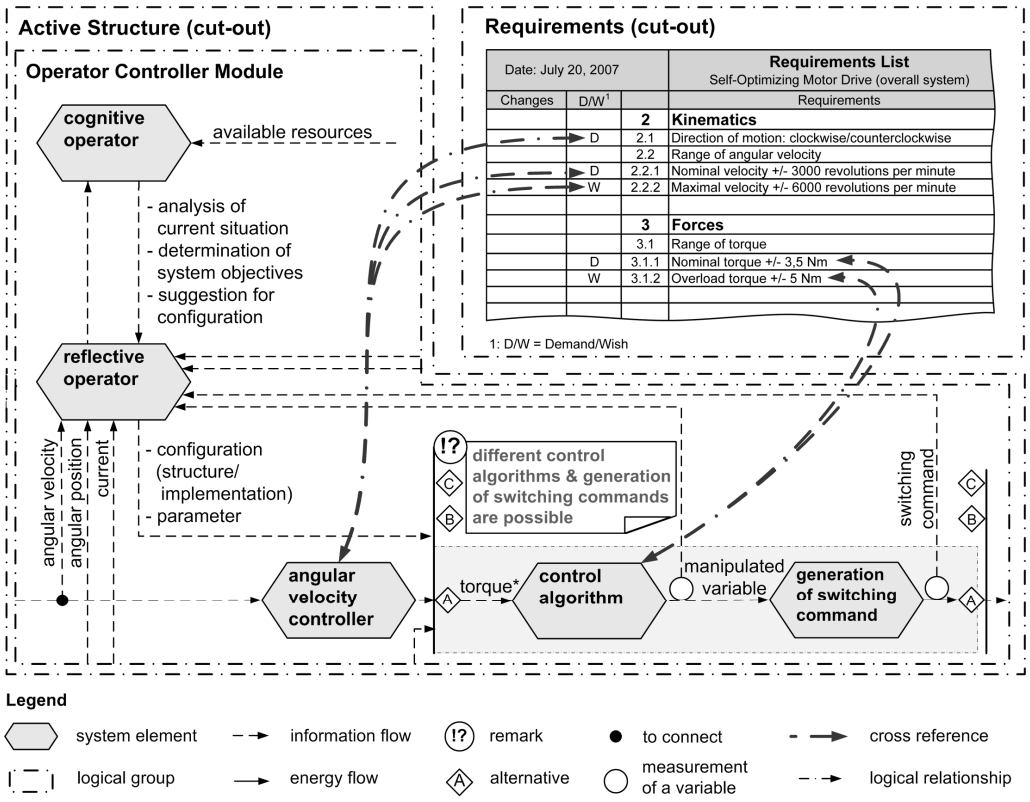


Figure 5-13: Cross-references between active structure and requirements (cut-out)

Cross-references between active structure and behaviour-states: Figure 5-14 exemplifies the cross-references between the active structure and the states as well as the events of a self-optimizing motor drive. As illustrated in the figure, the controller structure A which should be activated in the state SO-A are linked together. Besides that, the cognitive operator is linked to the events specified between the inner states. These events correspond to the optimization process that makes the decisions for the controller switching. The most viable controller structure is selected by evaluating both the desired control quality and the available system resources. In this context, the environmental influences acting on the system have to be taken into account. On the encounter of specific triggering events, both the initialization of the new controller and their switching take place as the system transits from an initial state into a new state. However, frequent switching and/or toggling on and off between the controllers have to be avoided. In the test bench, the different modes of operation are emulated using a load machine. It can be done by setting the load torque or the driving velocity with temporal changes.

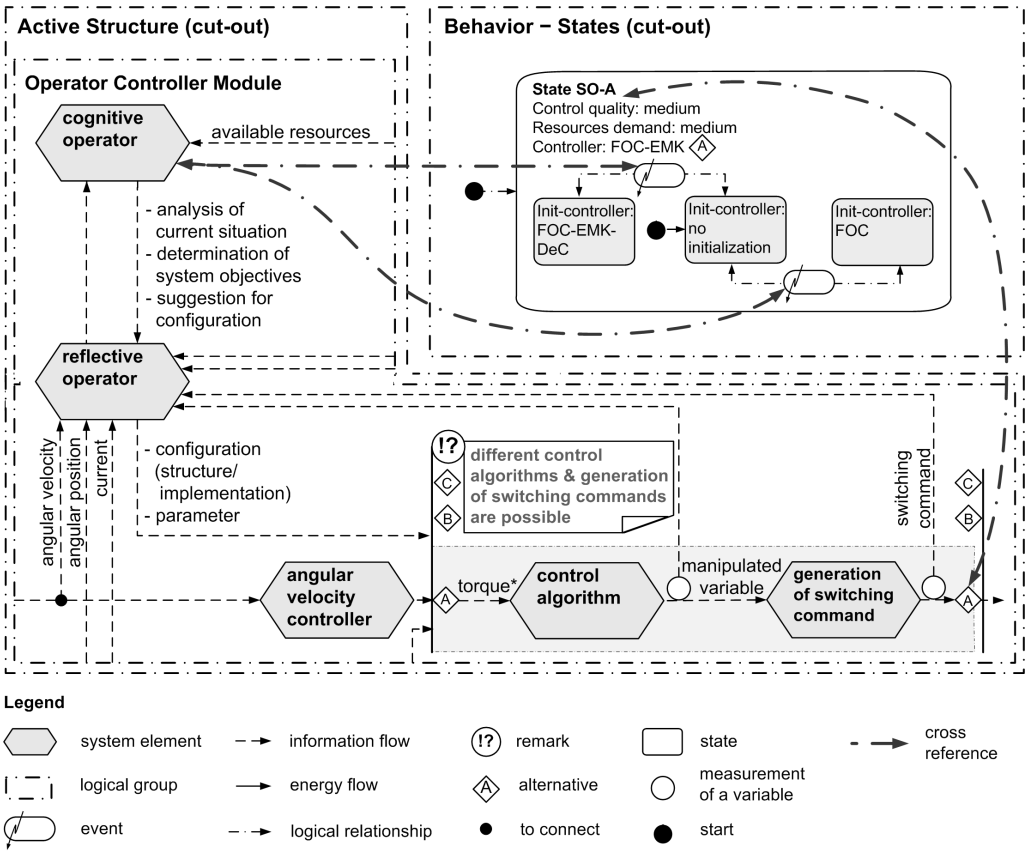


Figure 5-14: Cross-references between active structure and behavior – states (cut-out)

Cross-references between active structure and behaviour-activities: Figure 5-15 exemplifies the cross-references between the active structure and the activities of a self-optimizing motor drive. As such, activities are linked to the system elements which carry out them. The cognitive operator analyzes the current situation and subsequently determines the currently active objective function. The activities of loading, initialization and activation of controllers until the release of unused resources are executed by the reflective operator. The controllers control the angular dynamics of the motor drive.

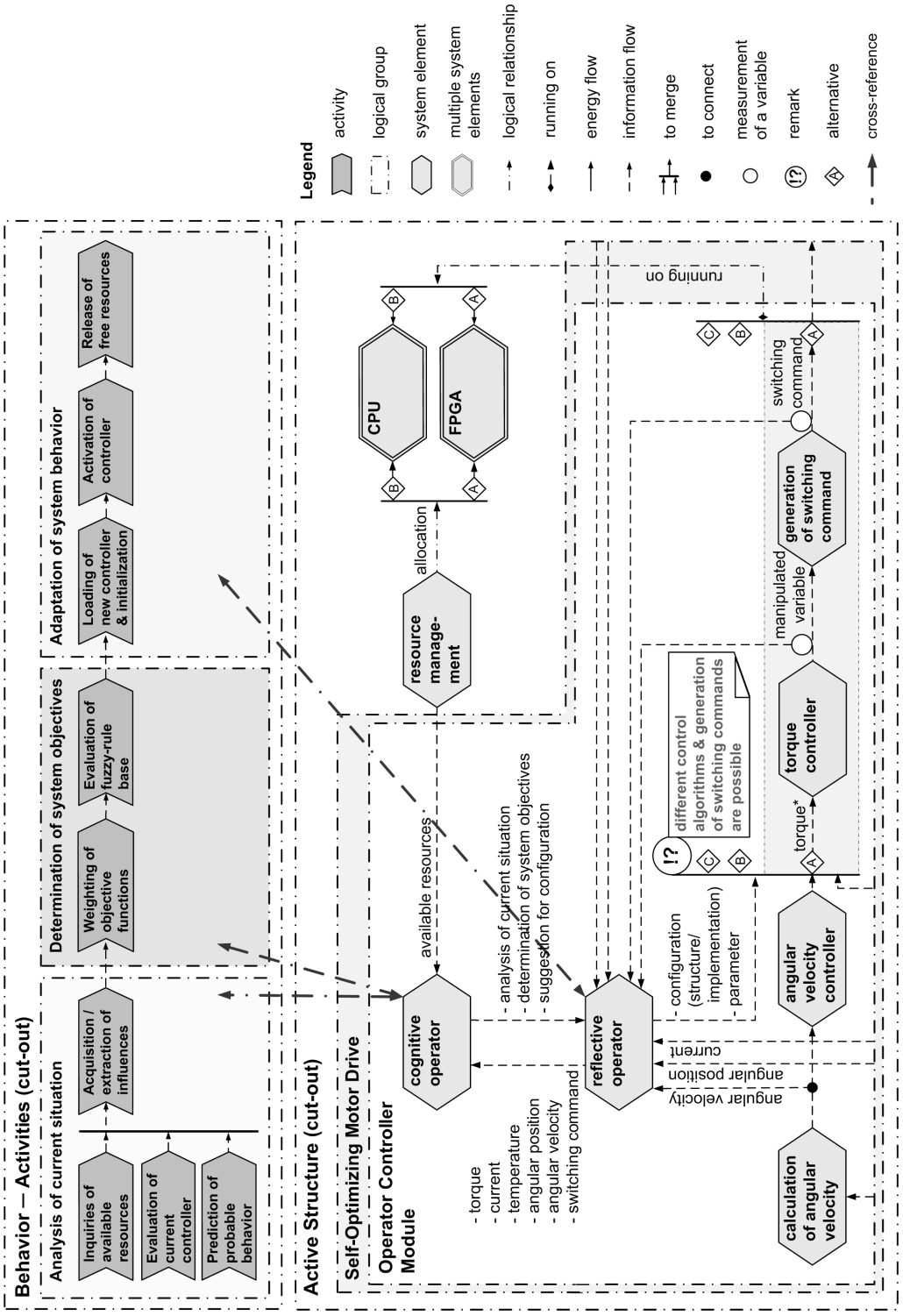


Figure 5-15: Cross-references between active structure and behaviour activities (cut-out)

5.1.2 Managing the Information Extraction from the Principle Solution for the Controller Design of a Self-Optimizing Motor Drive

Now the conceptual design phase has come to an end and the principle solution of the self-optimizing motor drive is well specified. At the interface between the conceptual design phase and the concretization phase in the domain of control engineering, information has to be extracted from the principle solution for the controller design of the self-optimizing motor drive. The procedural model to manage the information extraction during the transition from the principle solution towards the concretization of controller design as presented in Section 4.2 is exemplified here.

5.1.2.1 Extraction of Control Functions

Having formulated the principle solution, the functions to be implemented on the controllers have to be extracted. For this purpose, the system functionality specified within the partial models has to be well interpreted. The understanding of the system functionality is used to guide the extraction of control functions of the self-optimizing motor drive.

System functionality: With reference to the application scenarios (Figure 5-2 and Figure 5-3), the system of objectives (Figure 5-5), and the higher level functions of the system (Figure 5-6), it is clear that the system functionality of the motor drive is the ability of maximizing the control quality and yet minimizing the resources demand while the system is operating under changing operational and environmental conditions. As pointed out by the cross-references as illustrated in Figure 5-10 and Figure 5-11, the objective pursued by the motor drive has to be adapted according to its current application scenario. Hence, the controller design to be carried out is aimed for self-optimizing mechatronic systems.

Control functions: With reference to the function hierarchy (Figure 5-6) as well as the requirements (Figure 5-4) of the motor drive, the control functions to be realized by the motor drive can be extracted. The control functionality of the motor drive is the control of its angular dynamics, which is achieved through the control of the angular velocity by adjusting the torque of the motor drive.

5.1.2.2 Outline of a Control Hierarchy

Having extracted the control functions, the control hierarchy of the motor drive has to be outlined with reference to the principle solution. In order to be able to outline a control hierarchy, the controlled variables have to be identified and

the dependencies among the control functions have to be analyzed. References to the function hierarchy (Figure 5-6), active structure (Figure 5-7), and the cross-references between them (Figure 5-12) have to be made.

Controlled variables: In this application example, the control of the angular dynamics of the self-optimizing motor drive involves the adjustment of its angular velocity, torque, and current.

Interdependencies among control functions: Referring back to the active structure in Figure 5-7, three alternative controller structures labelled with A, B, and C are shown. It is to be noted that the inputs of the control algorithm of the three alternative controller structures consist of two information flows 'current' and 'torque'. Since the main dependency between the 'angular velocity controller' and the 'control algorithm' is the adjustment of the driving torque, the technique used to control the torque has to be made clear. It is to be understood that the adjustment of torque, is a result of current control, which directly influences the angular velocity of the motor drive. Therefore, in order to control the torque of the motor drive, the current of its permanent magnet synchronous motor has to be controlled.

Control hierarchy: A control hierarchy as shown in Figure 5-16 is outlined for the control of the angular dynamics of the self-optimizing motor drive. The function 'to control the angular dynamics' is first decomposed into the function 'to control the angular velocity', then decomposed into the function 'to control the torque', and further decomposed into the function 'to control the current' of the motor drive. The dependencies between the control functions are marked: 'adjustment of angular velocity', 'adjustment of driving torque', and 'adjustment of current components'. The controlled variables, which serve as the reference input and actual output at each hierarchical level, are indicated in the control hierarchy. At the bottom of the control hierarchy, the current controllers are represented as multiple function blocks. Implementation wise, this implies the possible switching between the controller structures which realize the functionality of current control. This information can be extracted from the alternative control algorithms A, B, and C, as indicated in the active structure. The aim is to realize the control of angular dynamics with different degrees of quality and resources demand.

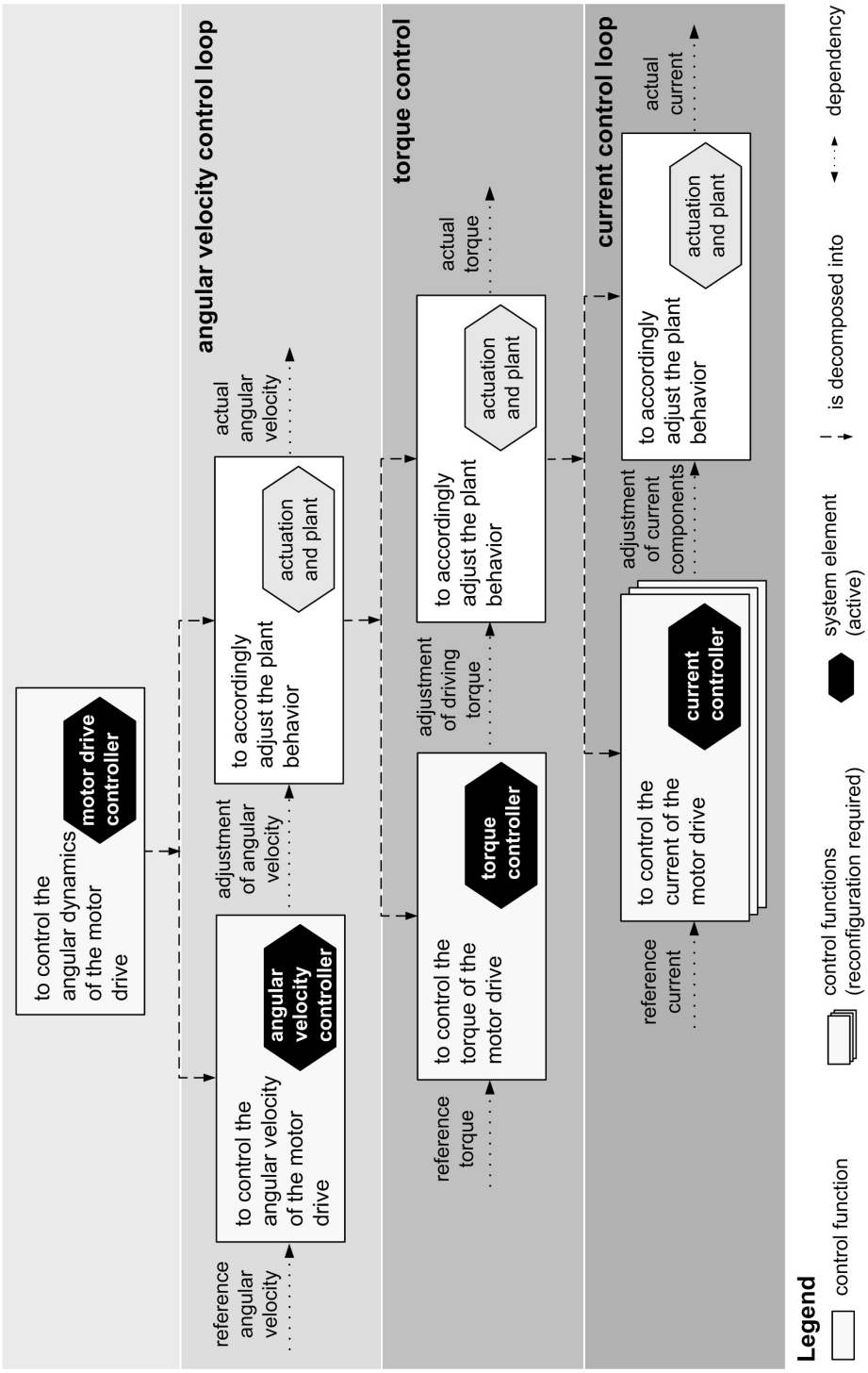


Figure 5-16: Control hierarchy for a self-optimizing motor drive

The ‘angular velocity controller’ and the ‘control algorithm’ used for torque control are placed side by side in the active structure. However, they are superimposing one another in the control hierarchy. This representation is clearer to the control engineers. Each of the control functionality in the hierarchy refers to a particular task to be carried out by a control algorithm in a cascaded con-

trol structure. Therefore, the hierarchical levels in the control hierarchy resemble the cascaded controller structure to be developed.

5.1.2.3 Conception of Controller Design

Having extracted the control functions and outlined a control hierarchy, the preliminary block diagram for the control of the angular dynamics of the motor drive has to be outlined. It involves the organization of the blocks within the control loops and the analysis of the behavioral adaptation of the motor drive.

Organization of the blocks within the control loops: For the organization of the blocks, the active structure (Figure 5-7) and the cross-references from the active structure to the requirements list (Figure 5-13) are referred. Figure 5-17 exemplifies the preliminary block diagram for the control of angular dynamics of the self-optimizing motor drive.

The block representing the electrical plant will contain power electronics and equations describing the electrical part of the motor that covers the voltage-current behavior. The blocks representing the mechanical plant includes the dynamics of the motor and the generation of driving thrust. The blocks representing the controllers will be added with control algorithms. In this example, while the torque is controlled by an open-loop controller, the PI-controller is used for both the angular velocity control and the current control. The current control involves the control of a vector of current components: the d-current and the q-current. On one hand, the d-current has to be controlled to zero in order to avoid energy losses in the motor. On the other hand, the q-current has to be controlled for the adjustment of the torque driving the motor. The focus here is the switching between the current controllers when the motor drive transits from a particular state into another. As shown at the innermost loops of the block diagram, the concept of using different control structures and the generation of switching command are clearly indicated. The reconfiguration of the control structures is a result of the self-optimization process.

All controller structures used to control the current are based on a Field Oriented Control (FOC) scheme. The properties of these controllers are well known by control engineers and have been presented by several authors, for instance [Bla72] [BWW72]. The difference between the controller structures A, B and C lie in the internal structure of the current control block.

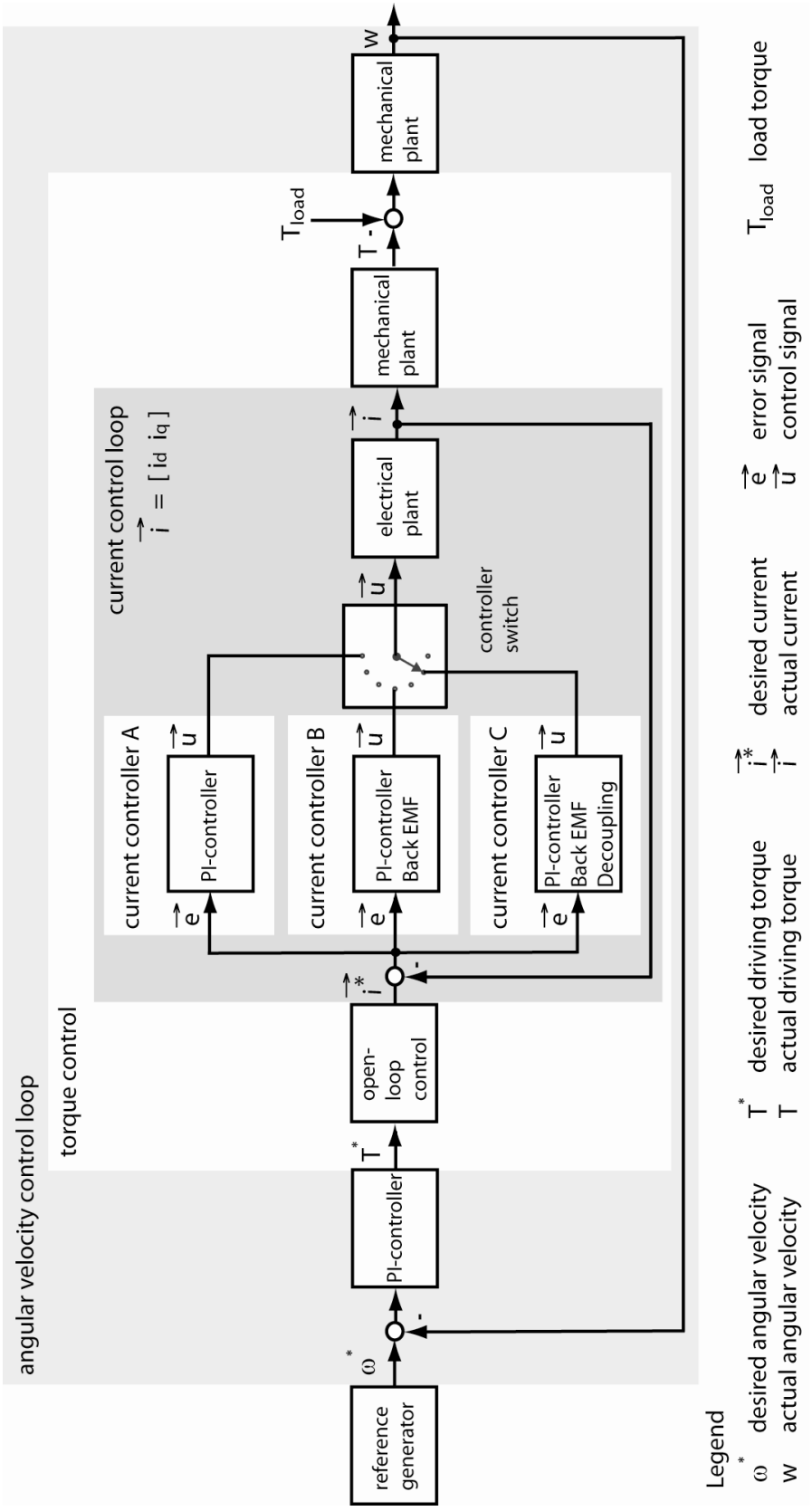


Figure 5-17: Preliminary block diagram for a self-optimizing motor drive [GKL+08a] [GKL+08b]

In control structure A, the current control block contains a FOC structure where the output of the PI-controller is directly the output of the block. Therefore, the dynamics of the control loop is determined by the PI-controller. Besides the PI-controller, the current control block in controller structure B contains a feed-forward for Back-EMF compensation. As such, the dynamics of the control loop is improved as compared with using the FOC alone. In controller structure C, the compensation for the Back-EMF and a decoupling of the current are incorporated in the feed-forward for the FOC.

Analysis of behavioral adaptation: Having organized the blocks within the control loops, the behavioral adaptation of the self-optimizing motor drive has to be analyzed. Eventually, the preliminary block diagram will have to be conformal to the behavioral adaptation required by the motor drive. In this context, a reference generator and a controller switch is involved. Switching between the different structures within the current control loop leads to the behavioral adaptation of the motor drive in delivering the self-optimizing capability. Cross-references between the active structure and the behavior ─ states (Figure 5-14) as well as between the active structure and the behavior ─ activities (Figure 5-15) are referred.

Concretization of Controller Design

The laboratory prototype of the self-optimizing motor drive has been developed in the Institute of Power Electronics and Electrical Drives at the University of Paderborn [Pet07] [Pet08] [Pik08]. The target platform is a rapid prototyping system with a FPGA-based and a CPU-based controller prototyping developed in the System and Circuit Technology group of the Heinz Nixdorf Institute, University of Paderborn. The current controllers are implemented using MATLAB Simulink and Xilinx System Generator, which is directly the source for the FPGA-platform.

The parameters of the controllers are tuned using the cross-ratio-approach. The controllers and their switching concepts are validated by the Hardware-in-the-Loop-Tests (HiL-Test) [SPH+07]. The hardware includes the power electronics, the permanent magnet motor drive and an induction motor as the load machine. The load machine emulates the impacts of changing load on the motor drive. The current controller A, B, and C as well as the switching between the controllers are implemented and tested under different conditions: constant load operation (constant motor speed), acceleration operation (speed up) and dynamic load operation (speed is varied by means of a load machine). Test in a real plant application, like hydraulic pump or machine tool, is the final step of the design concretization of the controllers.

5.2 Autonomous Railway Convoy

In this section, the method described in Chapter 4 is applied within the early development phases of an autonomous railway convoy. An introduction for the autonomous railway convoy is given in Section 2.3.2. As the continuation from Section 2.3.2, this section describes the principle solution of an autonomous railway convoy. Within each of the partial models of the principle solution, the basic control concepts required for the autonomous railway convoy are described. Subsequently, cross-references between the partial models of the principle solution are exemplified. At the end, the management of information extraction from the principle solution for the controller design required by the autonomous railway convoy is described.

5.2.1 Specifying the Basic Control Concepts within the Principle Solution of an Autonomous Railway Convoy

This section describes the conceptual design of an autonomous railway convoy. During the conceptual design phase, the principle solution for an autonomous railway convoy is specified in a domain-spanning way. Except the partial model shape, each of the partial models is described in the following.

5.2.1.1 Environment

Figure 5-18 exemplifies the specification of the environment of a RailCab. Passengers, cargo, track section, railway switch, and the outer environment all have certain influences on a RailCab. The mass of the passengers and the mass of the cargo contribute to the total moving mass of the RailCab which directly affects its driving behavior. The track set error is an influence exerted by the track section and the track switch on the RailCab driving over them. Besides that, abrasion is an influence generated within the RailCab itself.

The set of influences I_1 exerted from the outer environment onto the RailCab is expanded into an influence table. The set of influences consists of the downhill slope force, the air resistance, and the roll resistance. The downhill slope force due to the inclination of the terrain has a significant impact on the driving behavior of the RailCab. It constitutes 570 N at the biggest incline and therefore cannot be neglected [HTS+08b]. The air resistance is dependent on the travelling velocity of the RailCab. It is approximated to be less than 50 N because of the slow motion of the RailCab. When the head wind and the down wind are also considered, the air resistance is nearly independent of the velocity of the RailCab. The roll resistance of the RailCabs is small and is approximated to be less than 30 N. These influences act against the driving force produced by the

linear drive. The interaction of these forces results in the actual velocity of a travelling RailCab.

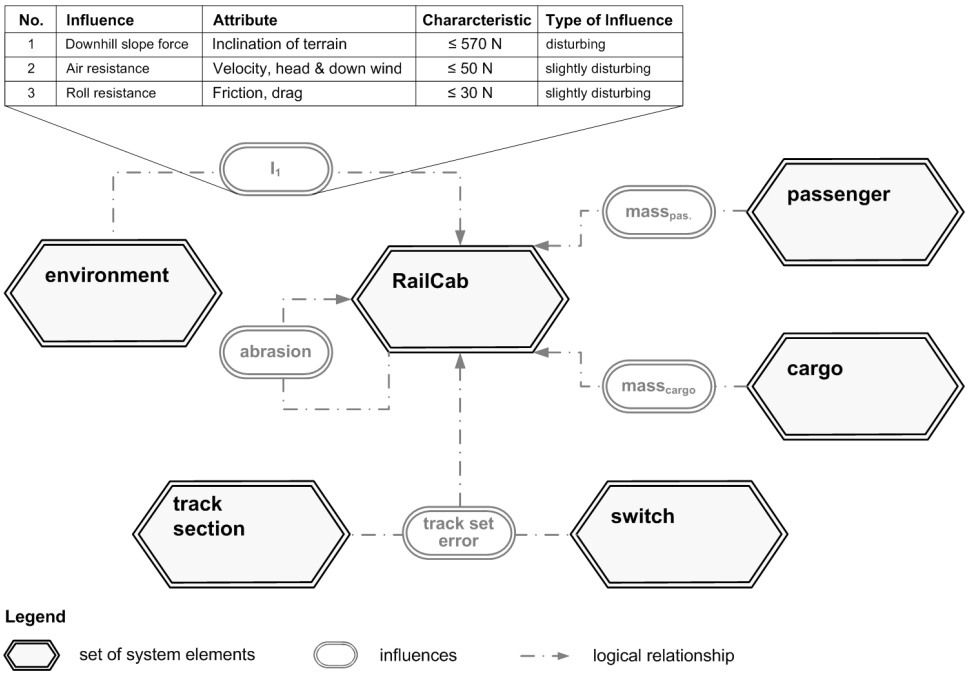


Figure 5-18: Environment of a RailCab (cut-out)

5.2.1.2 Application Scenarios

Figure 5-19 and Figure 5-20 exemplify the main application scenarios of an autonomous railway convoy. The first application scenario describes the merging process of two individually driving RailCabs into a convoy. On the contrary, the second application scenario describes the splitting process of a convoy into two individually driving RailCabs. The description of the partial development task for the application scenarios is stated at the upper part of Figure 5-19 and Figure 5-20 respectively.

The merging process to form a railway convoy in the first application scenario is the most safety critical operation. This is due to the fact of the unavoidable velocity differences when two RailCabs are driving together within small distances. Therefore the velocity of the RailCabs has to be limited during this process. A sketch about this application scenario is presented at the middle part of Figure 5-19.

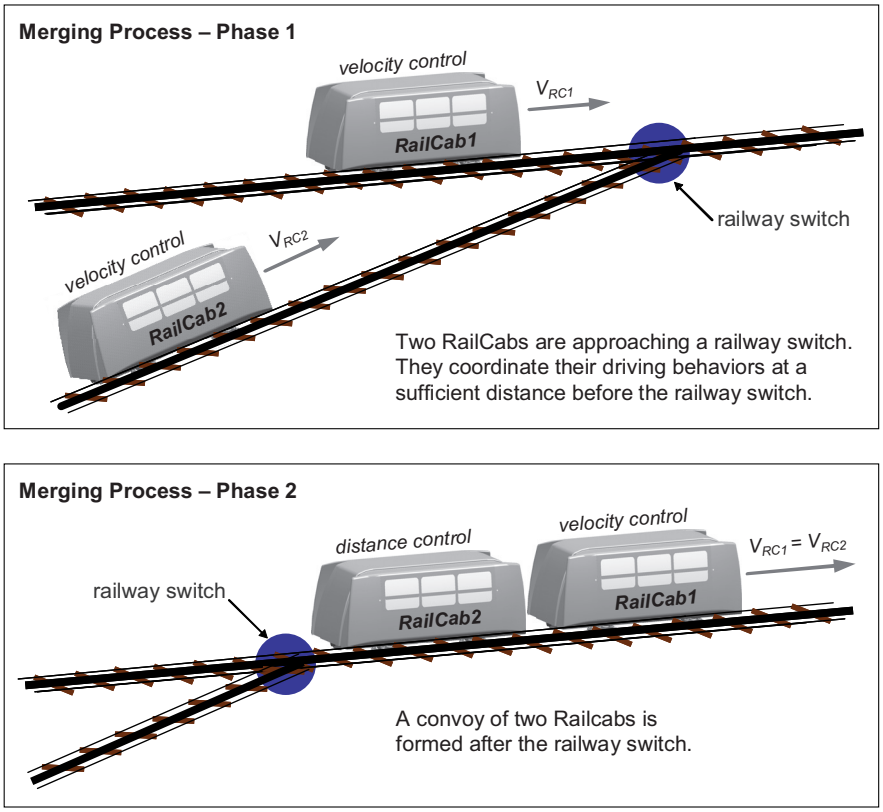
In the upper sketch, two RailCabs are approaching a railway switch. At that time, both RailCabs deploy velocity control. By means of message-based coordination, a decision regarding if a convoy should be formed has to be made. Upon the agreement for the formation of a convoy, they start to coordinate their driving behavior at a sufficient distance before the railway switch. It is denoted as the first phase of the merging process.

Date: 14. Dec 07	A	**Application Scenario** Forming a Convoy of two RailCabs	AS 1	Page 1

Description of partial development task

Two fully operational RailCabs in their individual operation are travelling towards a railway switch leading to a common route. The RailCabs have to drive in a convoy. As such, the driving behavior of the RailCabs has to be coordinated at a sufficient distance before the railway switch. The RailCab with the most energy reserve will drive in front. This RailCab plays the role of a leader while the other RailCab plays the role of a follower. A convoy of two RailCabs is formed. During the convoy operation, the distance between the RailCabs has to be constantly kept. This is done by means of distance control at the side of the follower RailCab. Nevertheless, velocity control is still employed by the leader RailCab. No collision should occur.

Sketch



Cross-references to application-specific partial models

Requirements | System of Objectives | Active Structure | Environment | Functions | Behavior

Figure 5-19: Application scenario for forming a convoy of two RailCabs (cut-out)

During the second phase of the merging process, the distance between the two RailCabs is getting critical. The velocity difference between the two RailCabs has to be reduced to such extent that no remaining damages will be caused in case of a collision [HTS+08b]. Subsequently, a convoy of two RailCabs is formed. As illustrated in the lower sketch, the leader RailCab maintains its ve-

locity control while the follower RailCab has to switch from velocity control to distance control. In a convoy operation, the follower RailCab has to keep a fixed distance from the leader RailCab independent of the velocity of the convoy.

The splitting process of a railway convoy in the second application scenario is similar with the merging process but executed in the reverse order.

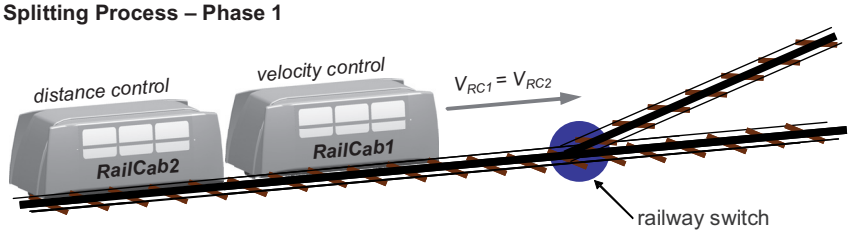
Date: 14. Dec 07	A	**Application Scenario** Dissolving a Convoy of two RailCabs	**AS 2**	Page 1

Description of partial development task

A convoy of two RailCabs is travelling towards a railway switch leading to two seperate routes. As the RailCabs have different final destinations, the convoy has to be dissolved. The RailCabs start to coordinate their driving behaviors as soon as the proposal to dissolve the convoy is accepted. This has to be done at a sufficient distance before the railway switch. During the convoy operation before the railway switch, the distance between the RailCabs has to be kept. As such, distance control is required for the follower RailCab. After railway switch, both RailCabs drive individually. Velocity control is employed for both RailCabs. No collision should occur.

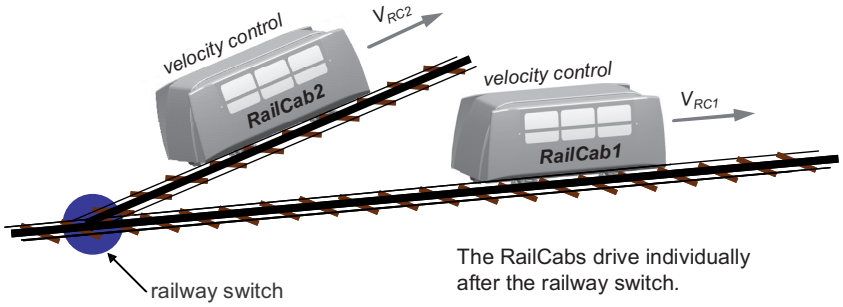
Sketch

Splitting Process – Phase 1



A convoy of two RailCabs is approaching a railway switch. They coordinate their driving behaviors at a sufficient distance before the railway switch.

Splitting Process – Phase 2



The RailCabs drive individually after the railway switch.

Cross-references to application-specific partial models

Requirements | System of Objectives | Active Structure | Environment | Functions | Behavior

Figure 5-20: Application scenario for dissolving a convoy of two RailCabs (cut-out)

5.2.1.3 Requirements

Figure 5-21 exemplifies a cut-out from the requirements list of a RailCab. As shown in the figure, the basic demands regarding the control of the longitudinal dynamics of a RailCab are listed. These demands are categorized under the main headings of kinematics, forces, and safety.

Date: 23.Sep 07			**Requirements List** RailCab (Overall System, Scale 1:2.5)	Page: 1
Changes	D/W[1]		Requirements	Responsible
		2	**Kinematics**	LEA/RtM
	D	2.1	Traveling directions: forward/backward	
	D	2.2	Traveling velocity: ≤10 m/s	
	D	2.3	Longitudinal acceleration on plane track section: ≤ 1.0 m/s²	
	D	2.4	Service braking: ≤ 1.0 m/s²	
	D	2.5	Emergency braking: 2.75 m/s²	
		3	**Forces**	LEA/RtM
	D	3.1	Vehicle mass: ≤ 1250 kg	
	D	3.2	Motor thrust: ≤ 1100 N	
		7	**Safety**	LEA/RtM
	D	7.1	To ensure safety by redundant measurements	
			● To use measurement devices based-on ultrasonic, infrared	
			and radar technology to cover areas from close range up	
			to large distances	
	D	7.2	To ensure safety during convoy formation and separation	
			● To limit the velocity difference between the RailCabs to 2 m/s	
			at non critical distances	
			● To limit the velocity difference between the RailCabs to 0.7 m/s	
			at critical distances	
	D	7.3	Employ hazard analysis and risk management	

1: D/W = Demand/Wish

Figure 5-21: Requirements list for the control of longitudinal dynamics of the RailCabs (cut-out)

Under the heading of kinematics, basic demands regarding the controlled motion of a RailCab are anticipated. A RailCab should be able to travel in both forward and backward directions on the rails. Considering the curves of the railway, the maximum travelling velocity of a RailCab is limited to 10 m/s. On plane track sections, the maximum longitudinal acceleration of a RailCab is limited to 1 m/s^2. During operation, the deceleration of a RailCab is limited to 1 m/s^2. In the case of emergency, a deceleration of 2.75 m/s^2 should be applied on the RailCab.

In corresponding with the requirements on the kinematics, further requirements regarding the action of forces that resulted in the controlled motion of a RailCab are specified. In this context, the mass of a RailCab is limited to 1250 kg while the thrust produced by the linear motor of the RailCab is limited to 1100 N. These requirements are categorized under the heading of forces.

Requirements to ensure the safety of an autonomous railway convoy are listed under the heading of safety. As shown in Figure 5-21, redundant systems are

used for signal detection where measurement devices based on ultrasonic, infrared, and radar technology are used to cover areas from close range up to large distances.

Redundant measurement itself is not sufficient. One of the most safety critical operations is the merging process of the individually driving RailCabs in order to form a convoy. A safety requirement is to limit the velocity of the RailCabs during this process. Within the initial phase of the approaching process, the velocity controller limits the velocity difference between the follower RailCab and the leader RailCab to 2m/s at non-critical distances [HTS+08b]. When reaching the relative braking distance, the velocity difference is limited to 0.7m/s. This speed difference is assumed to cause no remaining damages in case of a collision.

Furthermore, hazards occur during convoy operation may cause catastrophic results. As such, risk management is essential to reduce the likelihood of hazard occurrences and to restore the system into a safe state when the hazard occurs [HTS+08a, p. 48].

5.2.1.4 System of Objective

Figure 5-22 exemplifies the system of objectives of a RailCab. The main inherent, external, and internal objectives of the RailCab are described in the following paragraphs.

Inherent Objectives: Maximizing safety and reliability is an inherent objective of the RailCabs. It is essential to avoid fatal accidents and material damages. Safety and reliability have to be guaranteed in all modes of operation of the RailCabs. This includes the convoy operation as well as the individual operation. The risks faced by the system have to be identified and managed. The occurrence of system faults, for instance, the faults of radio communication between the RailCabs, have to be considered during the system design. In the case of a hazard occurrence, reference values for the controllers have to be adapted to restore the RailCab from an undesired state to a safe state. A proper control concept for a safe and reliable operation of the RailCabs is mandatory, especially to deal with cases of emergency. In this context, disturbances which can affect the safety and realiablity of the RailCabs have to be determined and integrated into the control concepts of the RailCabs.

External Objectives: Having ensured the safety of the system, the satisfaction of the passengers has to be met. It is specified as an external objective of the system. While driving on the rails, the reference velocity of the RailCabs can be determined based on the fare, comfort, and travelling time demanded by the passengers. A convoy operation of RailCabs reduces the energy demand per

passenger and thus reduces the fare. The comfort of the passengers onboard the RailCabs can be maximized by avoiding sudden jerks. The travelling time can be minimized by avoiding stopping, changing of RailCabs, travelling at modest top velocities, etc.

Internal Objective: The internal objectives of the RailCabs are derived from their external and inherent objectives. During the adaptation of system objective, safety and reliability of the RailCabs have to be given higher priority than the satisfaction of the passengers. The objective functions can be derived based on the internal objectives for optimization purposes.

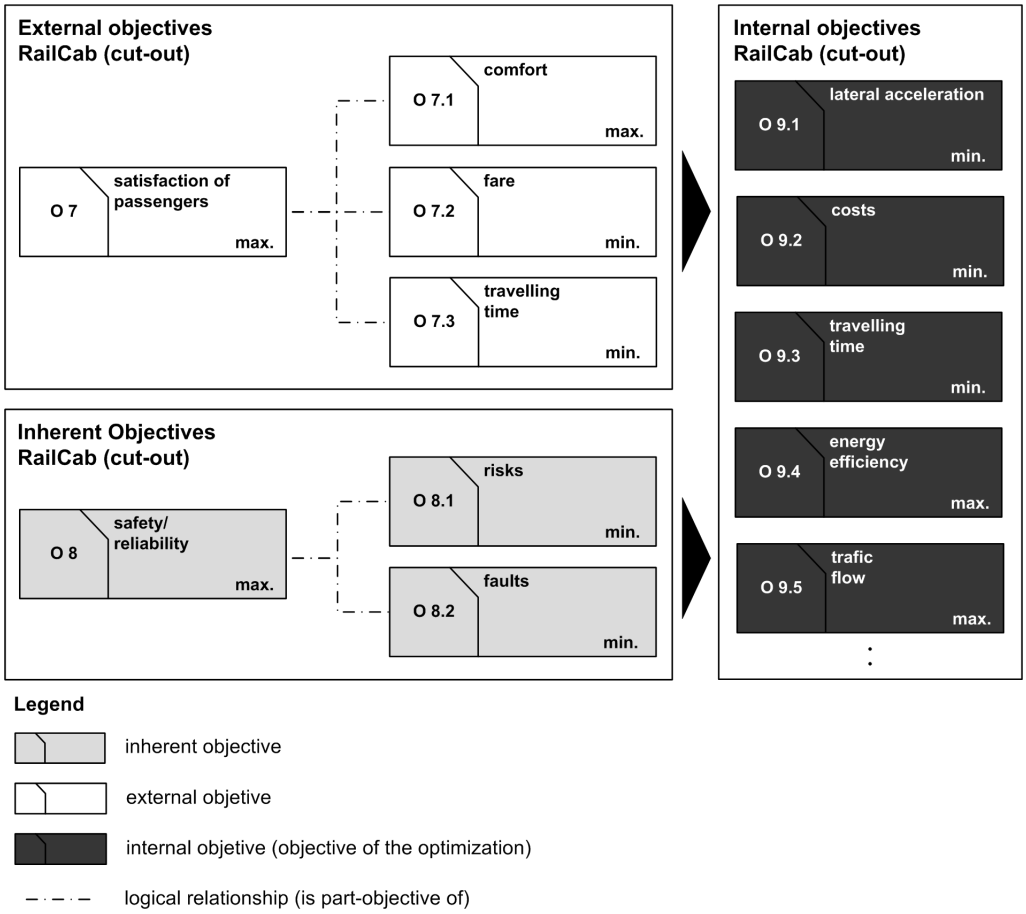


Figure 5-22: System of objectives of a RailCab (cut-out)

5.2.1.5 Functions

A convoy operation requires fully functional RailCabs. During the conceptual design phase, the essential functions required for the autonomous convoy operation of the RailCabs have to be carefully conceptualized. Figure 5-23 exemplifies a function hierarchy required for the convoy operation of the RailCabs.

At the top of the function hierarchy, the overall function required for an autonomous railway convoy is the ability to form and dissolve a convoy of Rail-

Cabs on the rails. On one hand, forming of a convoy involves the merging process where two RailCabs are driving approaching each other in order to form a convoy. On the other hand, dissolving a convoy involves the splitting process where a convoy is split into two individually driving RailCabs. These merging and the splitting processes can happen on the straight and curve rails as well as over a railway switch. The overall function to form and dissolve a convoy can thus be decomposed into two functions, one to control the longitudinal dynamics of the RailCabs, and the other one to control the lateral dynamics of the RailCabs. In this application example, only the control of the longitudinal dynamics of the RailCabs is concerned.

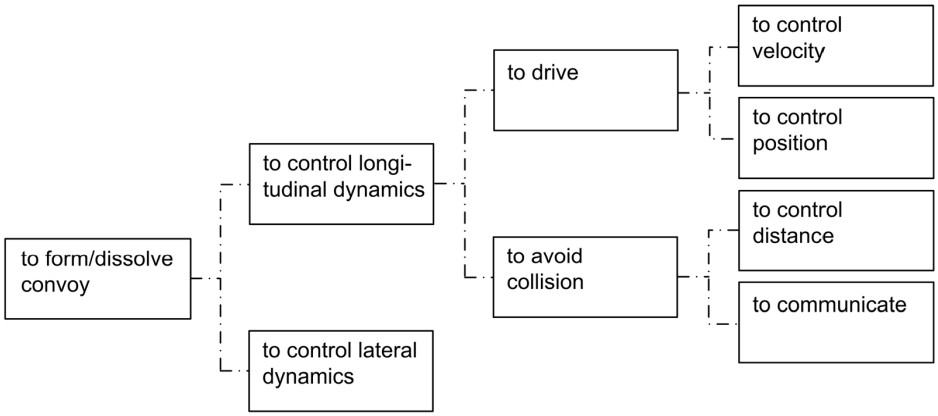


Figure 5-23: Function hierarchy for an autonomous railway convoy (cut-out)

At the subsequent level, the function to control the longitudinal dynamics of the RailCabs is further decomposed. Besides being able to drive on the rails autonomously, the ability of the RailCabs to avoid any possible collision with the other RailCabs is essential.

The function of driving involves the position and velocity control of the RailCabs. The position control enables a RailCab to arrive at a specific point, for instance, to stop at a specific platform at a train station. The velocity control enables a RailCab to accelerate or decelerate, for instance, slowing down when a RailCab is driving on a curved railway.

In order to avoid collision, the distance between the leading RailCab and the following RailCab has to be controlled. A minimum gap between the RailCabs has to be maintained to ensure a safe convoy operation. Besides that, effective communication between the RailCabs is imperative for the coordination of their driving behavior during the formation and separation of a convoy.

5.2.1.6 Active Structure

In this application example, a convoy which consists of only two RailCabs is concerned. They travel one after another on the rails without physical contact.

Figure 5-24 exemplifies the active structure of a convoy of two RailCabs. This is the simplest form of convoy operation at the hierarchical level of NMS. Such a convoy operation requires message-based coordination between the two RailCabs. The message-based coordination determines whether a convoy should be formed and when it should be dissolved [HTS+08a, p. 43]. The control of longitudinal dynamics enables the adjustment of the velocity of the RailCabs and thus the distance between the RailCabs. For the handling of different velocities and small distances between the two RailCabs, information is continuously exchanged between the RailCabs.

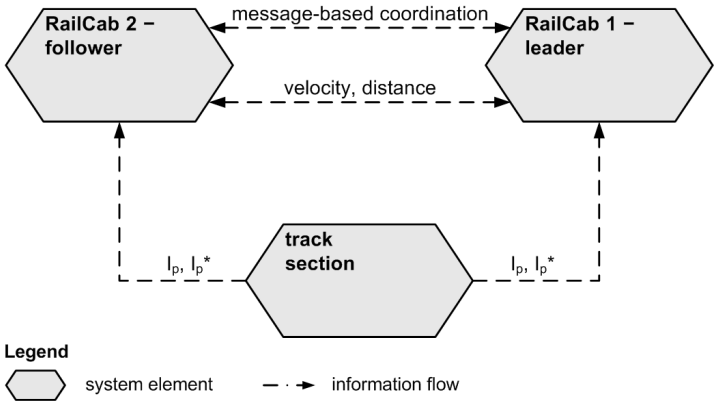


Figure 5-24: Active structure of a convoy of two RailCabs (cut-out)

Figure 5-25 exemplifies the RailCab as a system element at the hierarchical level of AMS. The RailCab consists of two driving modules at the two driving axles of the RailCab, which are specified as logical groups. Each of the driving modules contains the spring-and-tilt module, the active guidance module, and the drive-and-brake module. All these modules are at the hierarchical level of MFM. The propulsion of the RailCab is provided by the drive-and-brake module, which consists of a contact-free doubly-fed electromagnetic linear drive [ZS05] [ZBS+05]. Besides enabling the longitudinal motion, the linear drive allows power to be supplied to the RailCab without overhead lines or contact rails. The other conventional functions of the RailCabs such as supporting and guidance take place at the contact point between the wheel and the track. This allows the RailCabs to run on the existing railways. The active guidance module, with the front and rear steerable axles, enables the change of travelling direction when passing over a passive switch. In contrast to the conventional switching at the side of the railway, the directional switching of the RailCab takes place at the side of the RailCab. Besides that, the active spring-and-tilt module ensures high travelling comfort by damping out the excessive vibrations.

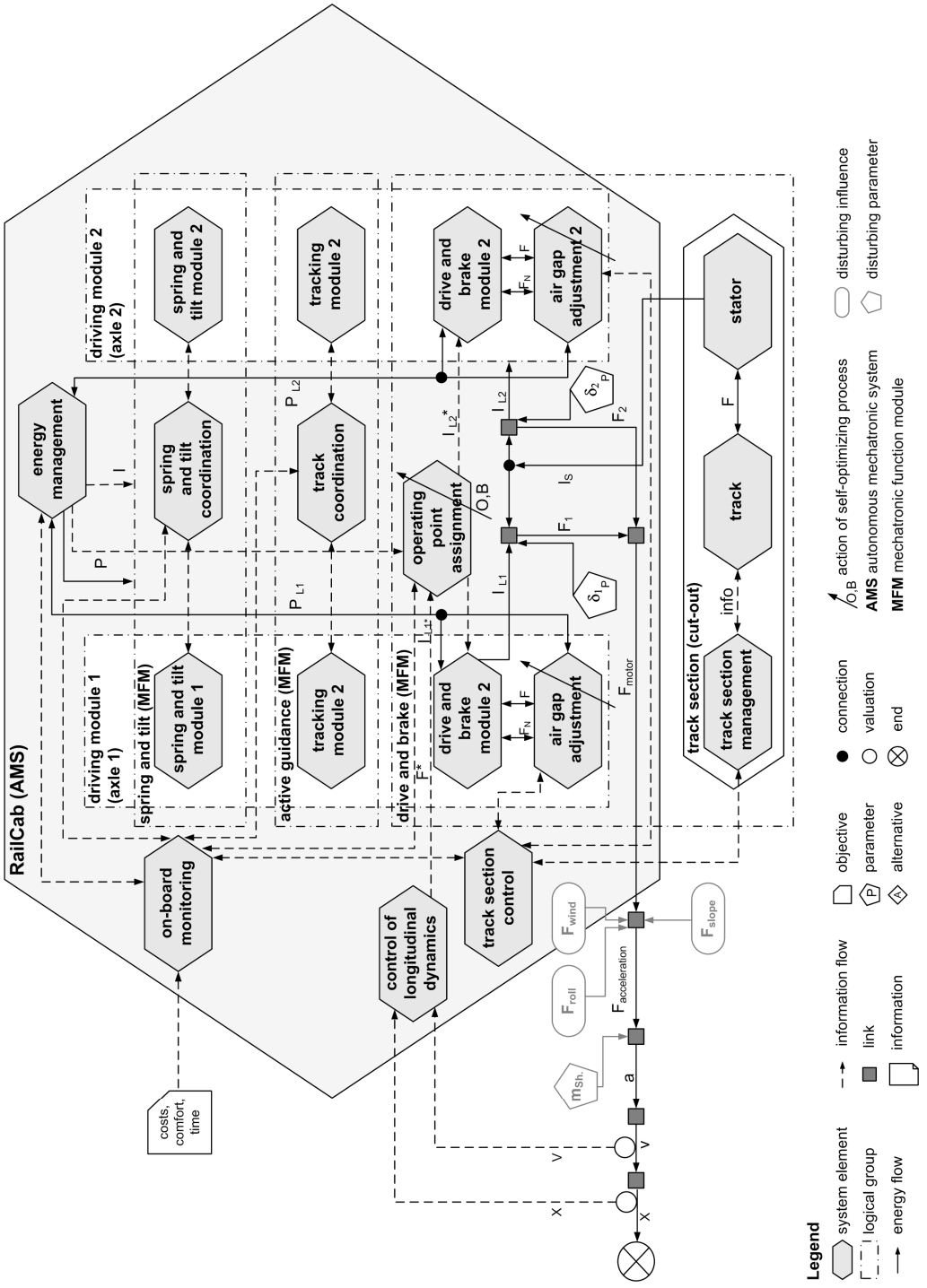


Figure 5-25: Active structure of a RailCab (cut-out)

The MFM that enables the motion and thus the convoy operation of the RailCabs is the drive-and-brake module. The linear drive consists of two parts: the rotors (secondary) mounted below the undercarriage and the stator (primary) mounted between the rails. Figure 5-26 further describes the active structure for the control of the longitudinal dynamics of the RailCabs. It involves both the stator and the rotor parts of the linear drive. The working principle of the linear drive has to be understood before the control concept can be specified.

The three-phase windings in the stator form an asynchronous magnetic field along the tracks. This is the stator field, which is interacting with the secondary magnetic fields of the rotors. The driving force is generated from the exact adjustment of the electromagnetic fields. As such, the RailCab can be accelerated or decelerated. Since the doubly-fed concept allows the variable adjustment of the secondary magnetic fields, several RailCabs can be operated on the same stator section with different velocities. The primary current of the stator is controlled separately.

The longitudinal dynamics of the RailCabs directly depends on the drive control. In order to effectively control the longitudinal dynamics of the RailCabs, a cascaded control structure as illustrated in Figure 5-26 is conceptualized. The innermost control loop regulates the secondary current of the rotor. The middle control loop regulates the velocity of the RailCab. The outermost control loop regulates the position of the RailCabs. Such a control structure distinguishes the different operating modes of the RailCabs by adapting the reference values. A reference generator is used to calculate the reference values required in the different modes of operation of the RailCabs.

In order to form a convoy, both the actual and reference distances between the RailCabs have to be taken into consideration. As such, a concept for distance control is required. In this context, the reference generator calculates the reference position of the follower RailCab based on the actual position of the leader RailCab and the reference distance demanded by the convoy formation [HFB06]. Besides that, the topology of the track has to be taken into account when calculating the references for distance control. This is marked as the alternative input A in the active structure.

When a RailCab is driving alone or as the leader in a convoy operation, distance control is not relevant. In this context, a concept for velocity control is required. The reference generator calculates the references for the velocity control of the RailCabs. While driving on the rails, the actual velocity of the RailCab is measured and transmitted to the velocity controller. The velocity controller compares the actual velocity with the desired velocity, and subsequently calculates the desired secondary current for the rotor. The velocity control of the RailCabs is marked as the alternative B in the active structure.

Whenever a convoy should be formed and when it should be separated, the configuration control activates the controller to be used in the actual mode of operation of the RailCab. As the mode of operation changes, a signal is generated to switch between the distance control and the velocity control of the RailCabs. The switching between these system elements leads to the behavioral adaptation of the RailCabs. As the convoy operation is safety critical, risk management is integrated into the configuration control of the RailCabs.

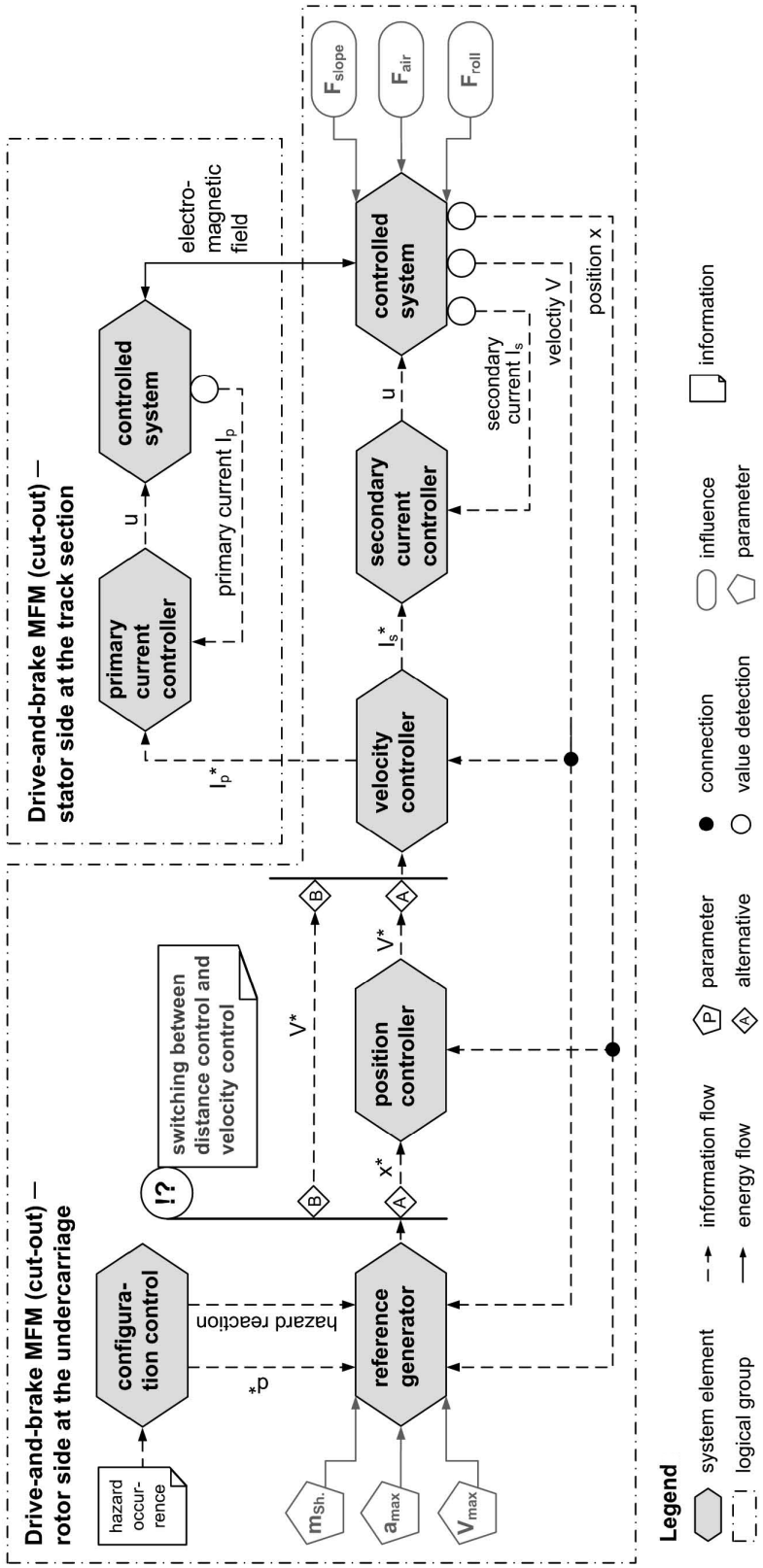


Figure 5-26: Active structure for the control of the longitudinal dynamics of RailCabs (cut-out)

5.2.1.7 Behavior – States

Figure 5-27 exemplifies the different states of an autonomous railway convoy which consists of two RailCabs. As illustrated in the figure, the RailCabs can either be in the off state, driving alone in the individual operation, coordinating their driving behavior in order to form a convoy, driving together in a convoy operation, or coordinating their driving behavior in order to dissolve a convoy.

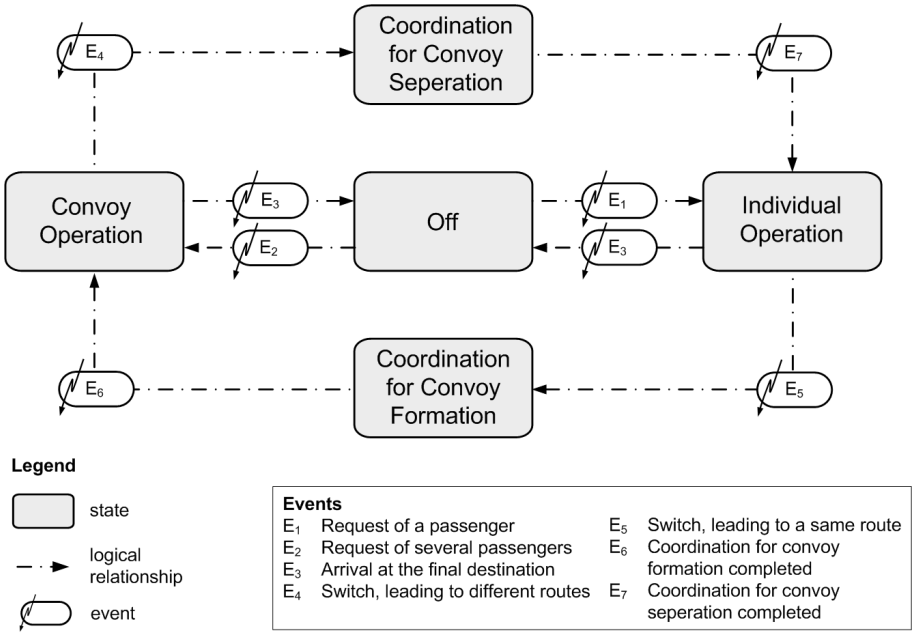


Figure 5-27: Different states of an autonomous railway convoy (cut-out)

Depending on the number of passengers, the operation of an individual RailCab or a convoy of two RailCabs can be initiated. As such, the RailCabs left their off state and start moving. Similarly, upon the arrival at their final destination, a transition into the off state is triggered.

When two RailCabs in their individual operations are approaching a railway switch leading to a common route, a convoy can be formed. The RailCabs leave the mode of individual operation and start to coordinate their driving behavior. The RailCab with the most energy reserve will drive in the front. This RailCab plays the role of a leader while the other RailCab plays the role of the follower. A convoy is formed.

Similarly, when a convoy of two RailCabs each with its own final destination is approaching a switch leading to different routes, they leave the mode of convoy operation and coordinate their driving behavior for convoy separation. Subsequently, a convoy is dissolved and the RailCabs are back to the mode of individual operation.

Figure 5-28 describes the coordination behavior of two RailCabs during the formation and separation of a convoy. The state diagram at the left describes the behavior of the RailCab at the rear of a convoy which plays the role of a follower whereas the main state at the right describes the behavior of the RailCab at the front of a convoy which plays the role of a leader. Message-based communication is applied for coordination during the formation and separation of a convoy. As indicated in the figure, distance control is required for the follower RailCab during the convoy operation, while the velocity control is required for the leader RailCab or when the RailCabs are driving alone.

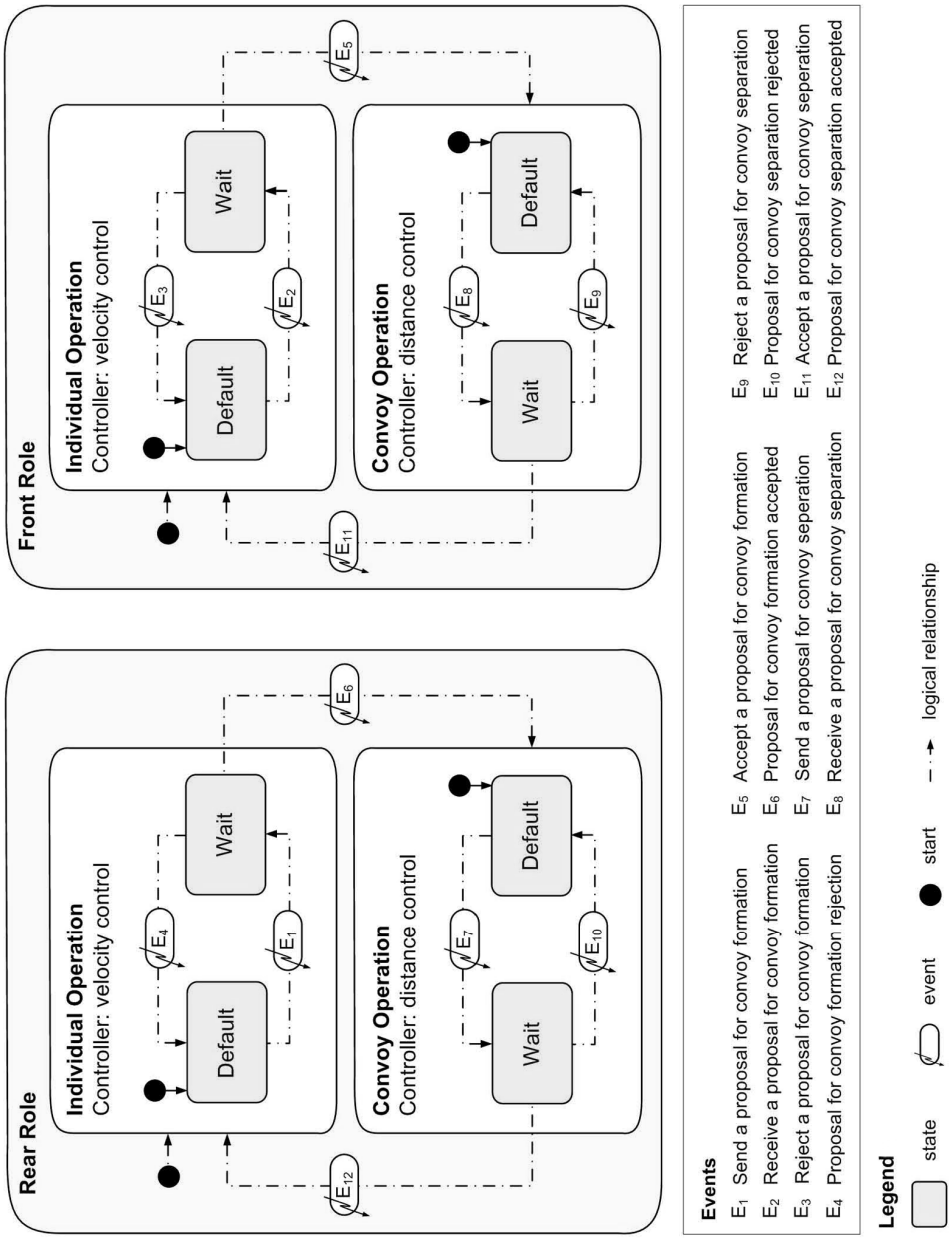


Figure 5-28: Different states during the formation and separation of a convoy (cut-out)

The coordination behavior is started by the RailCab at the rear of a convoy which sends a proposal for convoy formation to the RailCab at the front. The RailCab at the front either rejects or accepts this convoy proposal. If the proposal for convoy formation is rejected, both RailCabs stay in their default state of individual operation. If the proposal for convoy formation is accepted, a state transition from the individual operation into the convoy operation occurs for both RailCabs. For the follower RailCab, such state transition implies the switching from velocity control to distance control. However, the leader RailCab maintains its velocity control during the state transition. The coordination behavior for convoy separation is carried out in a similar way.

5.2.1.8 Behavior – Activities

Figure 5-29 exemplifies the activities for the control of longitudinal dynamics of the RailCabs. It includes the activities for the analysis of current situation, determination of system objectives, and adaptation of system behavior.

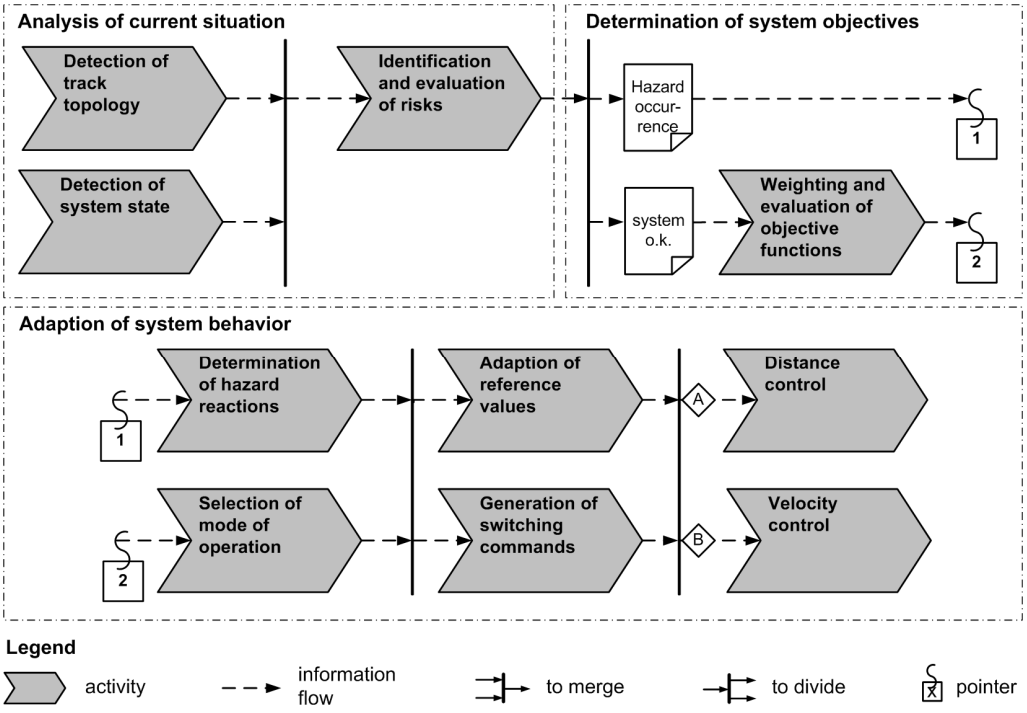


Figure 5-29: Activities for the control of the longitudinal dynamics of the RailCabs (cut-out)

Activities for the analysis of current situation include the detection of track topology and system state. The detection of system state includes the detection of position and velocity of every RailCab, detection of the distance from a follower RailCab to a leader RailCab, and fault detection. During the analysis of current situation, early identification of hazardous incidents [HTS+08a, p.45] is

desired for risk management. The evaluation of risks ensures the safety of the RailCabs during their autonomous convoy operations.

In the case of hazard occurrence, appropriate countermeasures have to be determined from a set of predefined hazard reactions to restore the RailCabs from an undesired state to a safe state. In the case of no hazard occurrence, the satisfaction of the passengers has to be met. This is done by optimizing the fare, comfort, and travelling time demanded by the passengers. Such optimization involves the weighting and evaluation of objective functions of the RailCabs.

Activities for the adaptation of system behavior are shown in the lower part of Figure 5-29. In a normal situation, the adaptation of system behavior involves the selection of the mode of operation of the RailCabs. Upon the selection of a mode of operation, the reference values required for a safe convoy operation have to be calculated. These reference values include the ideal distance between the RailCabs in a convoy operation as well as the reference values for the linear drive of the drive-and-brake module. Besides the calculation of reference values, generation of switching command is necessary as the mode of operation changes. The switching command switches between distance control and velocity control for a RailCab.

In a hazardous situation, reactions to handle the hazardous incidents result in adapted reference values or requested emergency brakes. In this context, the adaptation of system behavior can be, for instance, emergency stop, increasing distance between the RailCabs, or dissolving a convoy. The safety of the RailCabs has to be ensured in all modes of operations.

5.2.1.9 Cross-references between the Partial Models of the Principle Solution of an Autonomous Railway Convoy

Five types of cross-references between the partial models of the principle solution of an autonomous railway convoy are exemplified here.

Cross-references between active structure and functions: Figure 5-30 exemplifies the cross-references between the active structure and the function hierarchy pertaining to the control of the longitudinal dynamics of a RailCab. The control of the longitudinal dynamics of a RailCab involves the control of its position and velocity as well as the distance between the RailCabs. As illustrated in the figure, such cross-references link the control functions with the system elements assigned for the realization of the control functions. It has to be noted that the system element position controller is assigned for the realization of two control functions, i.e. the position and distance control. This is due to the fact that the distance between two RailCabs is actually the difference between the positions of the RailCabs.

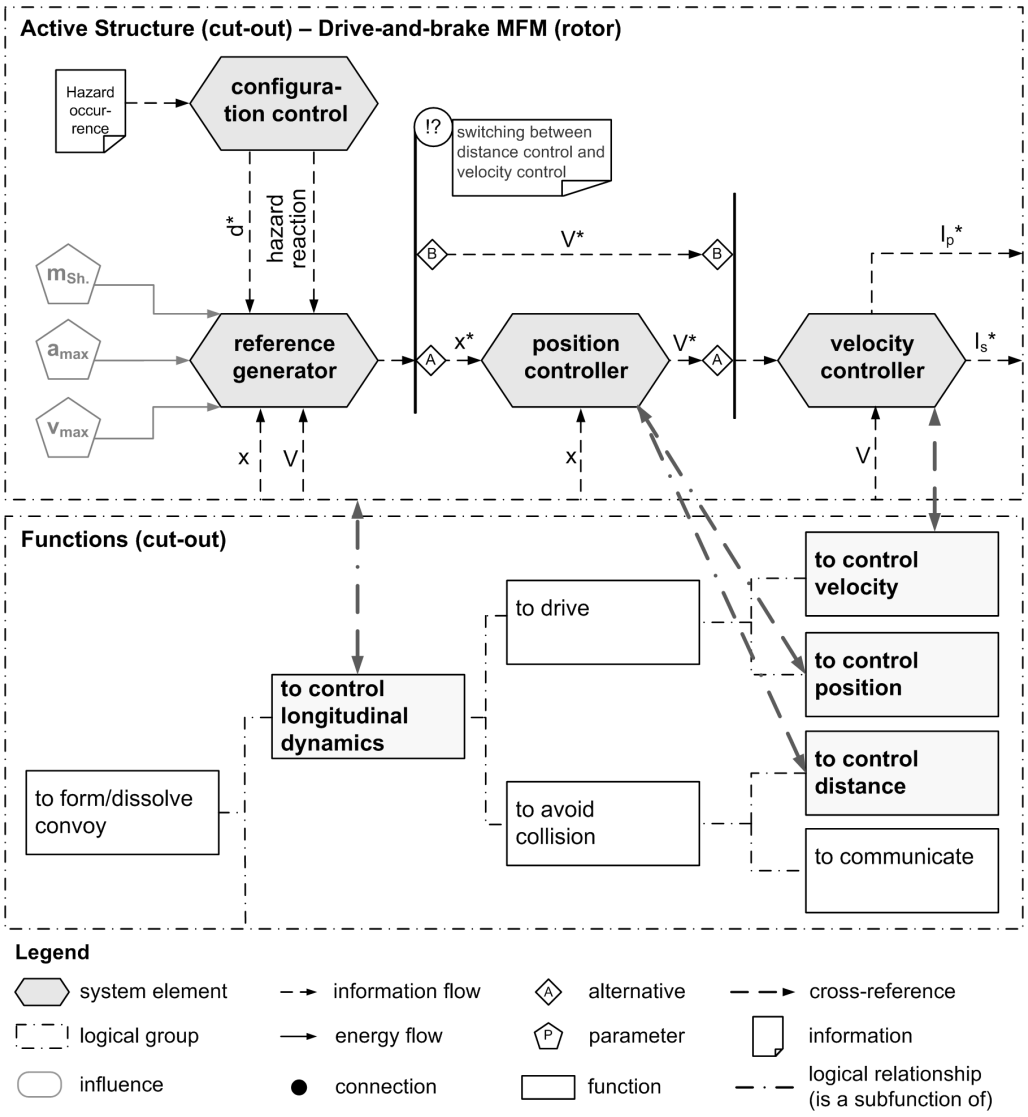


Figure 5-30: Cross-references between active structure and functions (cut-out)

Cross-references between active structure and environment: Figure 5-31 exemplifies the cross-references between the active structure and the environment pertaining to the control of the longitudinal dynamics of a RailCab. Such cross-references reveal further information about the influences acting on the controlled system as well as the source of information. As illustrated in the figure, the influences such as the downhill slope force, air resistance, and roll resistance specified in the active structure are linked with an influence table specified in the partial model environment. In the influence table, the attribute, characteristic, and type of the influences are listed. The source of these influences is the environment of the surrounding of a RailCab.

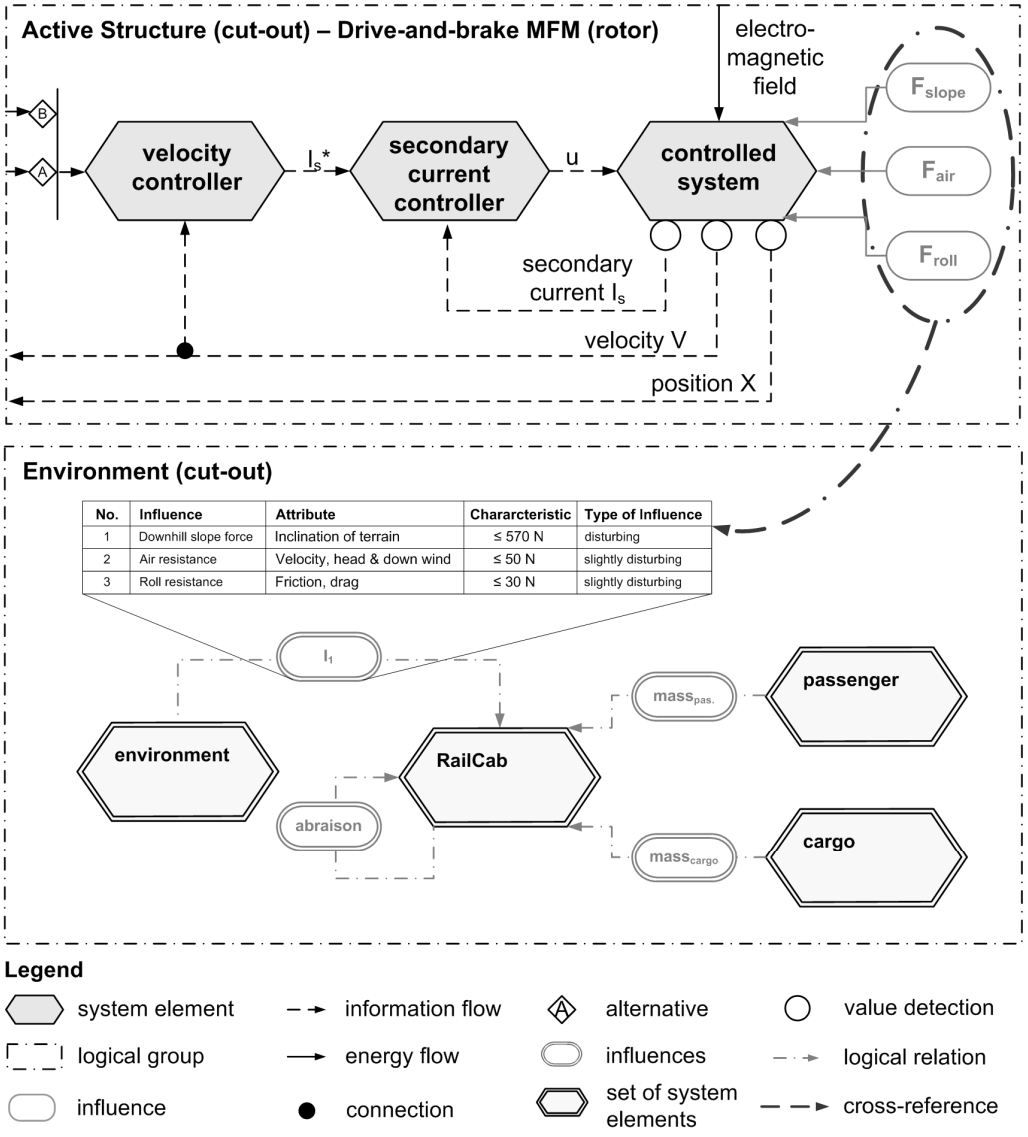


Figure 5-31: Cross-references between active structure and environment (cut-out)

Cross-references between active structure and requirements: Figure 5-32 exemplifies the cross-references between the active structure and the requirements pertaining to the control of the longitudinal dynamics of a RailCab. Such cross-references reveal the requirements to be fulfilled by the constructs specified in the active structure when carry out the control task. As illustrated in the figure, it is a demand that the configuration control carries out hazard analysis and risk management to ensure the safety of the autonomous railway convoys. Besides that, it is a demand that the velocity controller limits the travelling velocity of a RailCab to 10 m/s. Furthermore, the velocity controller is also demanded to limit the velocity differences between the follower and the leader RailCabs during the merging and the splitting process of the convoy operation. The last cross-reference reveals that redundant measurement of the position of the RailCab is required. Measurement devices based on ultrasonic, infrared,

and radar technology has to be used to cover areas from close range up to large distances.

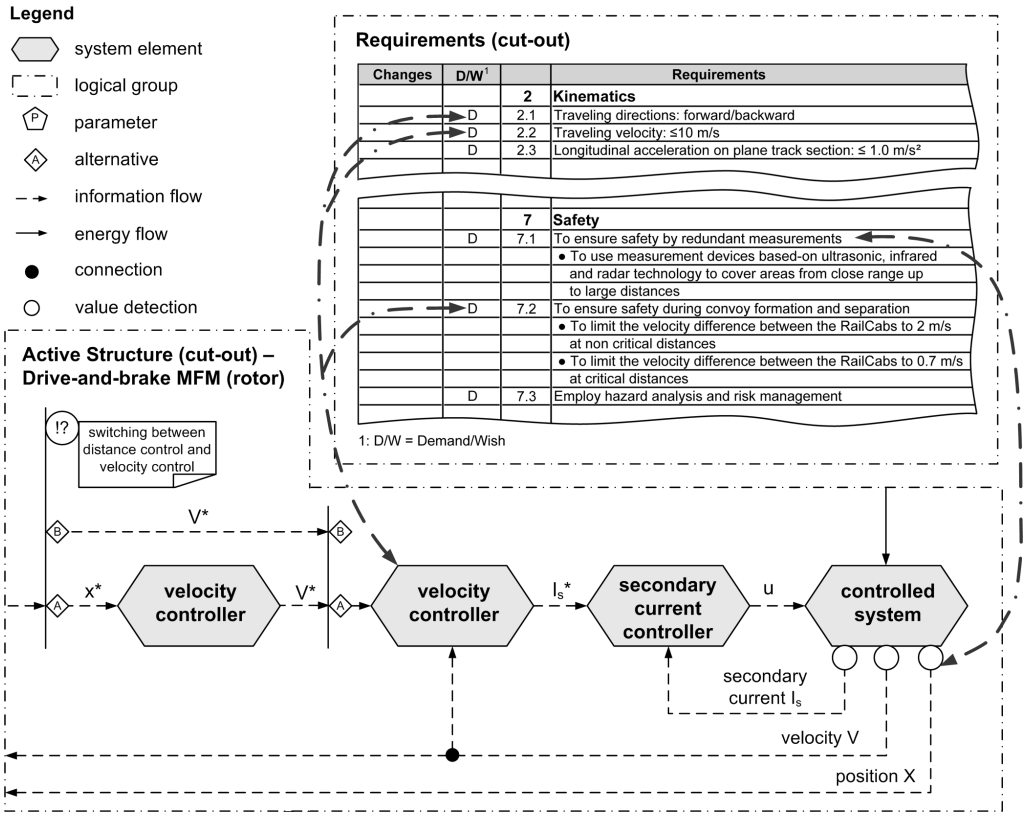


Figure 5-32: Cross-references between active structure and requirements (cut-out)

Cross-references between active structure and behaviour-states: Figure 5-33 and Figure 5-34 exemplify the cross-references between the active structure and the states of an autonomous railway convoy. Figure 5-33 exemplifies the cross-references that clarify the roles played by the RailCabs in a convoy operation. The leader RailCab plays the front role while the follower RailCab plays the rear role. Within the respective roles played by the RailCabs, the activation and deactivation of system elements when the RailCabs coordinate their driving behavior during the merging and splitting process has to be pointed out. As such, Figure 5-34 links the state transitions within a follower RailCab with the controller switching specified in the active structure. As illustrated in the figure, the position controller is activated during a convoy operation while the velocity controller is activated during an individual operation.

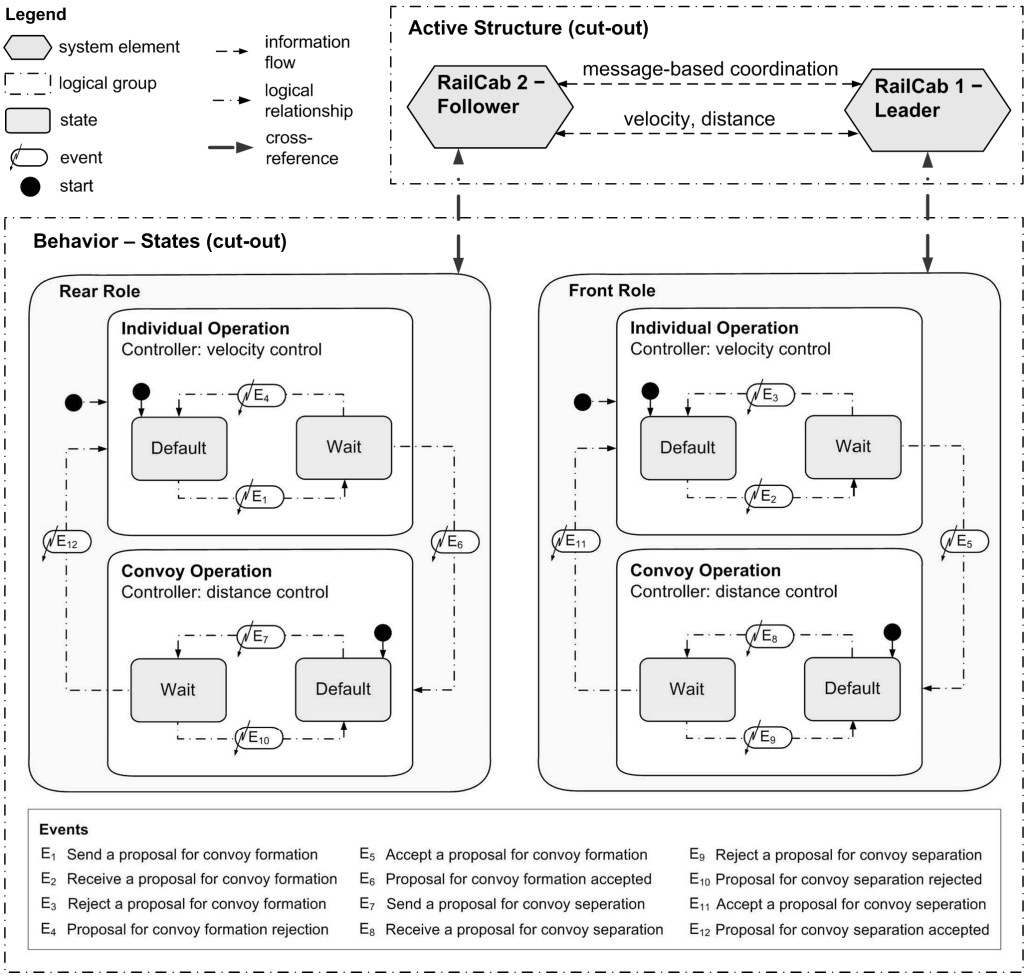


Figure 5-33: Cross-references between active structure and behaviour-states (a cut-out defining the role played by a RailCab in an convoy operation)

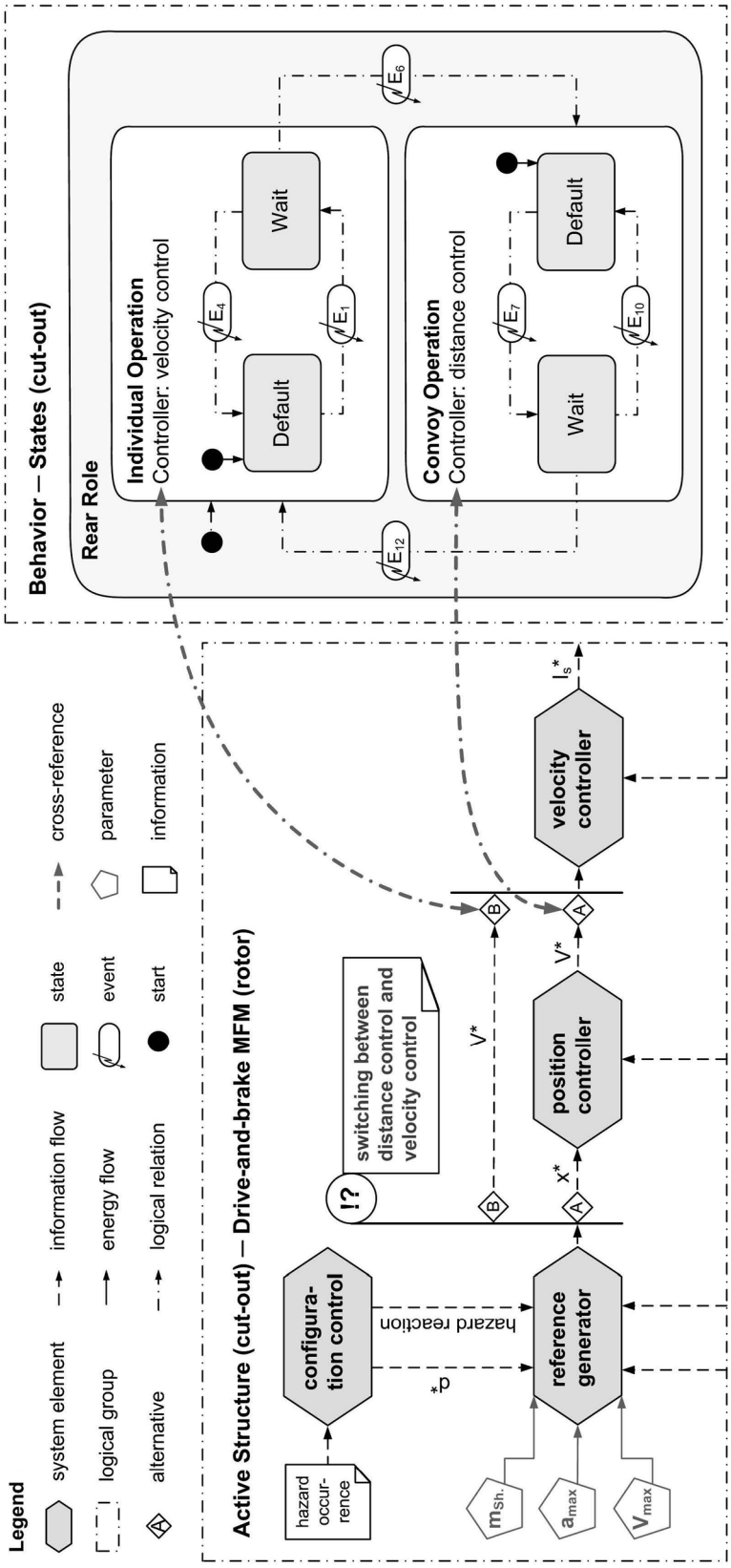


Figure 5-34: Cross-references between active structure and behavior-states (a cut-outlinking state transition with controller switching)

Cross-references between active structure and behavior-activities: Figure 5-35 exemplifies the cross-references between the active structure and the activities pertaining to the control of the longitudinal dynamics of a RailCab. As illustrated in the figure, a system element can execute more than one activity.

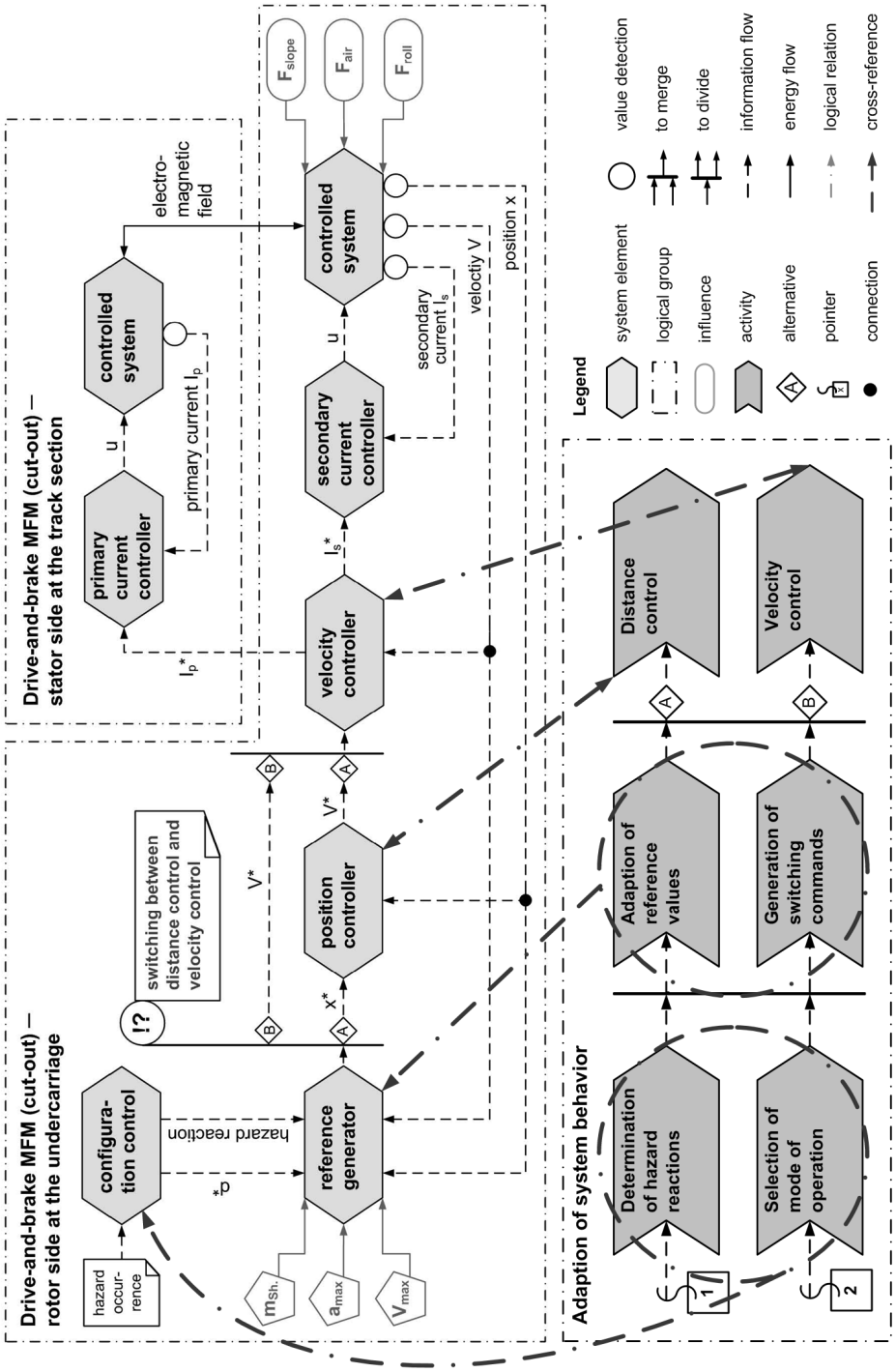


Figure 5-35: Cross-references between active structure and behavior-activities (cut-out)

The configuration control manages the risk by determining an appropriate hazard reaction in the case of hazard occurrence. Besides that, the configuration control selects the mode of operation of a RailCab. Based upon the current mode of operation, a reference generator is used to calculate the required set points as the reference values. For a follower RailCab, the reference generator calculates the reference distance from the follower RailCab to the leader RailCab. The reference position of the follower RailCab is calculated based on the current position of the leader RailCab and the reference distance. For a leader RailCab, the reference generator calculates the velocity references. Besides generating reference values, the reference generator generates a signal to switch between the distance control and the velocity control of the RailCabs.

5.2.2 Managing the Information Extraction from the Principle Solution for the Controller Design for an Autonomous Railway Convoy

Now the conceptual design phase has come to an end and the principle solution of an autonomous railway convoy is well specified. At the interface between the conceptual design phase and the concretization phase in the domain of control engineering, information has to be extracted from the principle solution for the design of controller pertaining to the longitudinal dynamics of the RailCabs. The procedural model to manage the information extraction during the transition from the principle solution towards the concretization of controller design as presented in Section 4.2 is exemplified here.

5.2.2.1 Extraction of Control Functions

Having formulated the principle solution, control functions have to be extracted from the principle solution. For this purpose, the system functionality of the autonomous railway convoy has to be interpreted. The understanding of the system functionality is used to guide the extraction of control functions required by the autonomous railway convoy**.**

System functionality: The system functionality of an autonomous railway convoy is the ability of the RailCabs to form and to dissolve a convoy on the rails as well as over a railway switch. For the interpretation of system functionality, references to the application scenarios (Figure 5-19 and Figure 5-20), system of objective (Figure 5-22) and the higher level functions of the system (Figure 5-23) are made. As the autonomous railway convoy is safety critical, the inherent objective regarding the safety and reliability of the RailCabs has to be prioritized at all time during the adaptation of system behavior.

Control functions: The convoy operation, as simple as just two RailCabs, requires accurate control of the longitudinal dynamics of the individual RailCabs. At the level of the Networked Mechatronic System, distance control is required to ensure a safe convoy operation. At the level of the Autonomous Mechatronic System, velocity and position control is needed to be able to accelerate, decelerate and arriving at a specific point. For the extraction of control functions, references to the function hierarchy (Figure 5-23) as well as the requirements (Figure 5-21) are made.

5.2.2.2 Outline of a Control Hierarchy

Having extracted the control functions, a control hierarchy for the control of the longitudinal dynamics of the RailCabs has to be outlined. In order to be able to outline a control hierarchy, the controlled variables have to be identified and the dependencies among the control functions have to be analyzed. In this phase, references to the function hierarchy (Figure 5-23), active structure (Figure 5-26), and the cross-references between them (Figure 5-30) are made.

Controlled variables: A closer inspection at the active structure as shown in Figure 5-26 provides information about the controlled variables. The controlled variables that can be identified are: ‘distance’, ‘position’, ‘velocity’, ‘primary current’ and ‘secondary current’. As shown in the active structure, the primary current of the stator has to be controlled independently [HYG01]. Subsequently, the only remaining actuating variable for thrust control is the secondary current of the rotor.

Interdependencies among control functions: The control functions extracted from the principle solution have to be structured into a hierarchy based on their interdependencies. First and foremost, the gap between the RailCabs has to be kept at a distance safe enough to avoid collision. For this purpose, information about the current position and velocity of the leader RailCab is required for the determination of the reference position for the follower RailCab [HFB06]. As the position of a RailCab changes when its velocity changes, the velocity of the RailCab has to be controlled. The thrust propelling a RailCab is the resultant of the adjustment of the electromagnetic field between the stator and the rotor. Therefore, the primary current of the stator and the secondary current of the rotor have to be controlled. Nevertheless, the primary current of the stator is controlled separately.

Control hierarchy: A control hierarchy as shown in Figure 5-36 is outlined for the control of the longitudinal dynamics of the RailCabs. The function ‘to control the longitudinal dynamics’ is first decomposed into the function ‘to control the distance’, then decomposed into the function ‘to control the position’, further decomposed into the function ‘to control the velocity’, and finally

decomposed into the function “to control the primary and secondary current”. The dependencies between the control functions are marked: ‘adjustment of relative position’, ‘adjustment of absolute position’, ‘adjustment of velocity’, and ‘adjustment of electromagnetic field’. The reference input and actual output at each hierarchical level are indicated in the control hierarchy. The distance control is integrated into the position control loop due to the fact that distance is actually a difference in position.

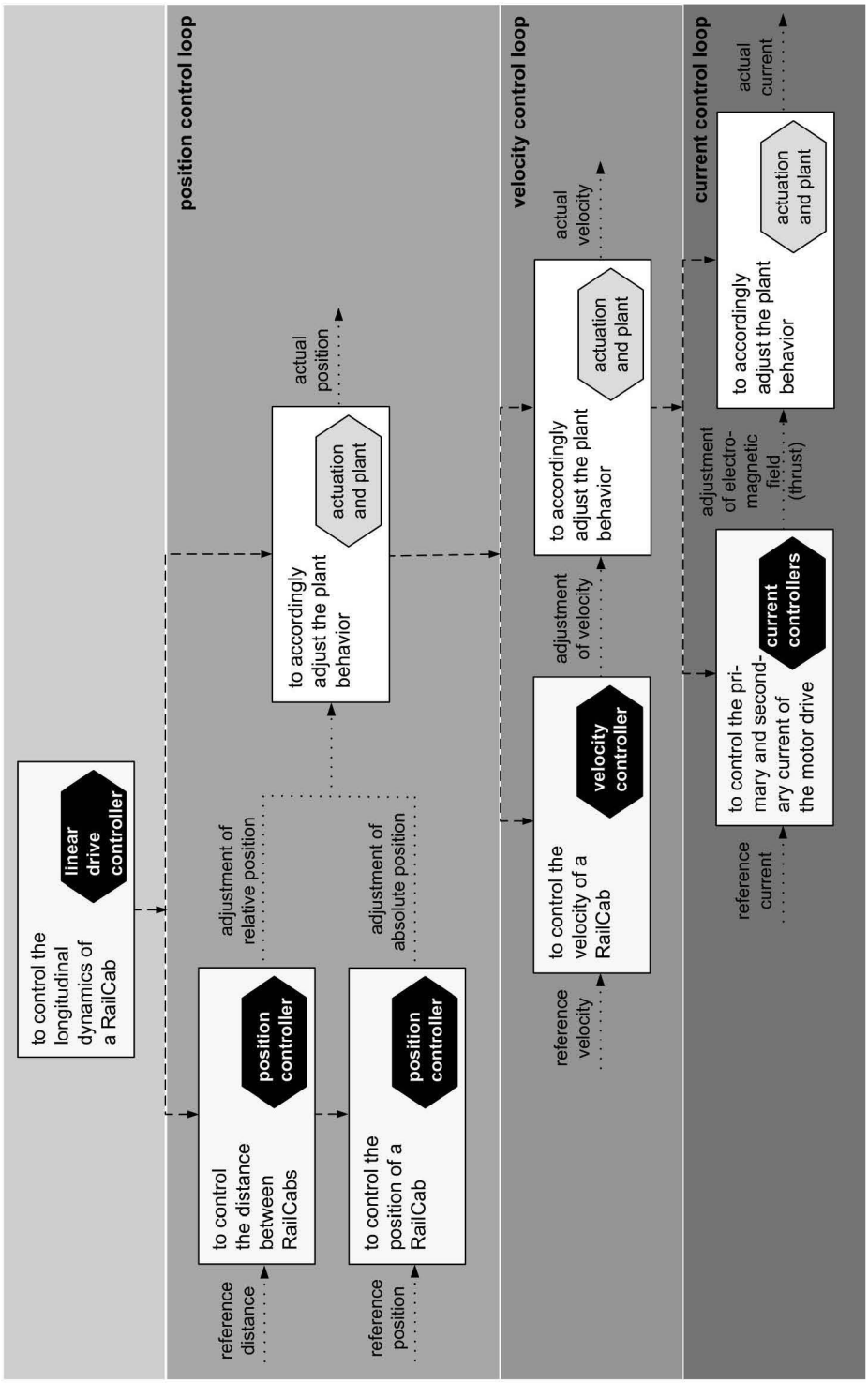


Figure 5-36: Control hierarchy for an autonomous railway convoy

5.2.2.3 Conception of Controller Design

Having extracted the control functions and outlined a control hierarchy, the preliminary block diagram for the control of the longitudinal dynamics of the RailCabs has to be outlined. It involves the organization of the blocks within the control loops and the analysis of the adaptation of the driving behavior of the RailCabs.

Organization of the blocks within the control loops: The preliminary block diagram for the control of the longitudinal dynamics of the RailCabs is shown in Figure 5-37. The blocks and the feedback loops are drawn based on the active structure. From the outer loop to the inner loop, the controlled variables involved correspond with those stated in the control hierarchy shown in Figure 5-36. Being at the lowest level of the control hierarchy, the secondary current is controlled at the innermost loop in the preliminary block diagram. Superimposing the current control loop is the velocity control loop while superimposing the velocity control loop is the position control loop. The distance control constitutes an extension to the position control loop. The reference generator determines the reference position x^* for the position controller.

By referring to the active structure, the feedback loops for the secondary current, the velocity, and the position can be drawn. Furthermore information about the environmental influences and the demands and wishes in carrying out the control task can be revealed by referring to the cross-references as shown in Figure 5-31 and Figure 5-32. Nevertheless, not all design considerations can be made during the conceptual design phase. For instance, the choice of the controller parameters can not be conceptualized, as it depends on the dead time of the converter and the time response of the linear drive.

Analysis of behavioral adaptation: Having organized the blocks within the control loops, the behavioral adaptation of the RailCabs in a convoy operation has to be analyzed. The task here is to make sure that the preliminary block diagram is conformal to the behavioral adaptation required by the RailCabs in a convoy operation. In this context, a controller switch is added for the switching between distance control and velocity control of the RailCabs. Besides that, a reference generator is added to calculate the reference position and the reference velocity for a RailCab. For this purpose, cross-references between active structure and the behavior ─ states (Figure 5-33 and Figure 5-34) as well as between active structure and behavior ─ activities (Figure 5-35) are referred.

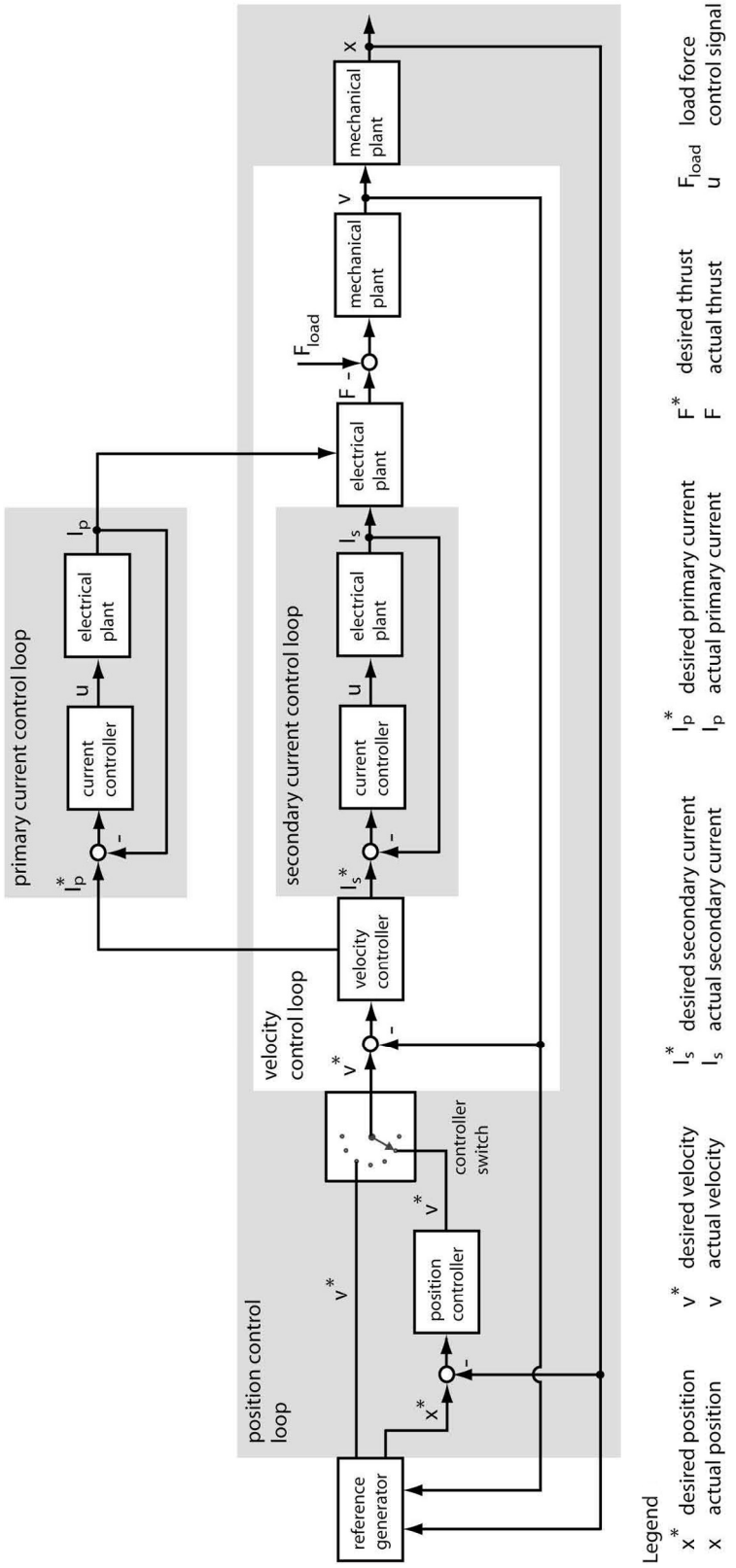


Figure 5-37: Preliminary block diagrams for the control of the longitudinal dynamics of a RailCab

Concretization of Controller Design

All blocks in the preliminary block diagrams shown in Figure 5-37 have to be supplemented with control algorithms or system equations. On one hand, the mechanical plants will contain equations describing the integration of velocity, the dynamics of a RailCab, and the generation of the driving thrust. On the other hand, the electrical plant will be added with equations describing the dynamics of the converters, the rotor, and the stator. The controllers will be added with, for instance, the P and the PI control algorithms. The downhill slope force F_{slope}, the air resistance F_{air}, and the rolling resistance F_{roll} are summed up here as F_{load}. Details about the disturbance variables can be taken from the environment model and incorporated into the system equations. Besides that, the system's behavior and the adaptation of behavior are described by the behavior-states and the behavior-activities. Basic design considerations include avoiding overshoot, maximum limit of acceleration as well as jerk in order to provide high riding comfort. With the doubly-fed linear motor design, the frequency and the amplitude of the secondary current can be regulated separately on each RailCab [HVB+05]. As such, it is possible to align the excitation field in the secondary as required, so that the force generation is optimized and several RailCabs can perform different thrust forces on the same primary segment [HG00]. As mentioned from the beginning, a test track and two RailCabs was built on a scale of 1:2.5 at the University of Paderborn.

5.3 Evaluation of the Method against the Requirements

The method presented here for managing the transition from the principle solution towards the controller design of advanced mechatronic systems successfully fulfill the requirements outlined in Section 2.6. The following paragraphs describe how the two interrelated approaches of the method fulfill each of the requirements.

R1 — A Holistic Principle Solution as a Starting Point for Concretization

In this work, the basic control concepts that have to be taken into account during the conceptual design phase of advanced mechatronic systems are identified. Besides that, an approach has been developed to point out the way to specify these basic control concepts within the principle solution of advanced mechatronic systems. The approach explicitly demonstrates how these basic control concepts can be specified in each partial models of the principle solution. Furthermore the fundamental cross-references which are essential for the control of advanced mechatronic systems are described. Such an approach successfully portrays a holistic picture of system design with the essential control concepts included. As exemplified in the application examples, the approach

effectively hones the role of the principle solution as a starting point for controller design of advanced mechatronic systems.

R2 ― Equal Treatment on the Basic Concepts from Different Domains

In this work, the basic control concepts for advanced mechatronic systems are specified using a specification technique spanning the different domains of mechatronics. These basic control concepts are specified in such a way that they can be easily interpretable even for the layman. As such, intuitive appreciation of the control concepts and thus the articulation of various concepts among engineers of different backgrounds are possible. As a result, the basic concepts from the domains of control engineering can be equally treated and intuitively integrated with the basic concepts of mechanics, electric/electronics, and software engineering within the principle solution of advanced mechatronic systems. As exemplified in the application examples, the basic concepts of different domains are seamlessly integrated, for instance, by means of logical relationship 'running on' or cross-references between the partial models.

R3 ― Systematic Extraction of Information from the Principle Solution

An approach has been developed in this work to manage the extraction of information from the principle solution for the controller design of advanced mechatronic systems in a systematic way. The approach points out the way to identify the control concepts specified within the principle solution, extract them out from the principle solution, and subsequently transform them into the preliminary block diagrams. As exemplified in the application examples, information is not only extracted from the individual partial models of the principle solution but also from the cross-references between the partial models. The results show that the control concepts specified within the principle solution can be transferred for the deployment of control engineers during the concretization phase without any information loss. As such, extraction of information from the principle solution for the controller design of advanced mechatronic systems is successfully managed in a systematic way.

R4 ― Integrating Top-Down and Bottom-Up Approaches

In this work, the top-down approach adopted when specifying the principle solution of advanced mechatronic systems and the bottom-up approach adopted for controller design are integrated by means of a control hierarchy. Referring back to the procedural model as shown in Figure 4-26, the approach taken for the extraction of control functions and backward progresses top-down. The transition from the top-down to the bottom-up approach happens right at the middle of the procedural model, i.e. during the analysis of the interdependencies among control functions in order to outline a control hierarchy. After that, from the conception of controller design and onward, bottom-up approach is

adopted. The first result out of the bottom-up approach is the preliminary block diagrams. As such, the method successfully integrates the top-down and bottom-up approaches which are respectively adopted during the conceptual design phase and the controller design phase of advanced mechatronic systems.

R5 ― Linking Semi-Formal and Formal Specifications

In this work, the semi-formal specification technique developed by FRANK et al to describe the domain-spanning principle solution of advanced mechatronic systems has to be linked with the formal specification of the block diagrams deployed for controller design. Transformation takes place between the system elements and the flows between them in the active structure to the blocks and the links between them in the preliminary block diagram. The preliminary block diagram is the first step towards the formal specification in the concretization phase in the domain of control engineering. The procedural model as shown in Figure 4-26 systematically points out the transitional phases and their respective activities during such transition. As exemplified in the application examples, the method successfully links the semi-formal and formal specifications deployed during the conceptual design phase and the controller design phase of advanced mechatronic systems.

6 Summary and Outlook

Summary

Mechatronics relies on the close interaction of mechanics, electric/electronics, control engineering, and software engineering. The current trend in mechatronics is led by the conceivable development of information technology which will enable mechatronic systems with inherent partial intelligence. They will be able to learn, to communicate, and to optimize their behavior autonomously in response to environmental changes. An interdisciplinary and integrative approach is crucial for the development of such systems.

In this work, the scope of research is placed within the early development phases of advanced mechatronic systems. During the conceptual design phase, the initial basic concepts from the various domains of mechatronics have to be intuitively integrated. Such a conceptual design results in the specification of a domain-spanning principle solution for the system to be developed. On the basis of this principle solution, further concretization takes place in each of the domains of mechatronics in a parallel way. The transition from the domain-spanning conceptual design towards the domain-specific concretization of advanced mechatronic systems has to be well managed to ensure a seamless development flow.

The analysis of the state-of-the-art shows that numerous design methodologies and specification techniques exist for the domain-spanning conceptual design of advanced mechatronic systems as well as for the domain-specific concretization in the domain of control engineering. However, literatures that address the transition from the domain-spanning conceptual design towards the domain-specific concretization are nearly nonexistent except for a few scattered thoughts that were written within the Collaborative Research Center 614 (CRC 614) “Self-Optimizing Concepts and Structures in Mechanical Engineering”.

Within the scope of this work, a method to manage the transition from the principle solution towards the controller design of advanced mechatronic systems has been developed. This method consists of two interrelated approaches, i.e. an approach to specify the basic control concepts within the domain-spanning principle solution of advanced mechatronic systems as well as an approach to manage the information extraction from the principle solution for the controller design of such systems.

The first approach describes how the basic control concepts required by advanced mechatronic systems should be specified within the principle solution of such systems during the conceptual design phase. It involves the partial

model environment, application scenarios, requirements, system of objectives, functions, active structure, behavior ─ states, and behavior ─ activities. As each of the partial models describes a particular aspect of the system, the control concepts specified within the different partial models of the principle solution have to be in conformity and supplement each other. As such, the fundamental cross-references between the partial models that are essential for the control of advanced mechatronic systems are described.

The second approach describes how information should be extracted from the domain-spanning principle solution of advanced mechatronic systems as the prerequisite for concretization in the domain of control engineering. For this purpose, a procedural model has been developed to systematicize the transitional phases, as well as their respective activities and results. In such a way, information can be extracted from the partial models of the principle solution as well as from the cross-references between them in a systematic way. The outcome is the preliminary block diagram for the controller design of such systems.

Two demonstrators of the CRC 614 are selected to validate the method proposed in this work. The demonstrators consist of a self-optimizing motor drive and an autonomous railway convoy. Both application examples are at the research forefront in their respective fields. On one hand, the control concepts of both the demonstrators are specified within their principle solution. On the other hand, the management of information extraction from the principle solution for the concretization of controller design for both demonstrators are demonstrated. As such, the aforementioned method is successfully validated.

The proposed method fulfills all the requirements outlined in this work. In a shut shell, the method developed in this work successfully bridges the gap between the domain-spanning conceptual design and the domain-specific controller design of advanced mechatronic systems.

Outlook

The method proposed in this work serves as a good starting point towards a semi-automatic transition between the domain-spanning conceptual design and the domain-specific concretization of controller design for advanced mechatronic systems. A computer-aided tool for the specification of the principle solution of advanced mechatronic systems is under development at the Heinz Nixdorf Institute, University of Paderborn. In the future, semi-automatic extraction of information leading towards the derivation of preliminary block diagrams based upon the principle solution of advanced mechatronic systems is desirable.

Besides the method presented in this work, methods for managing the transition from the principle solution of advanced mechatronic systems towards concretization in the domains of mechanics, electric/electronics, and software engineering are also required. Similar with the method proposed in this work, these methods should address approaches for specifying the basic concepts of mechanics, electric/electronics, and software engineering in the principle solution of such systems. Approaches to manage the extraction of information from the principle solution as the prerequisites for concretization in the domains of mechanics, electric/electronics, and software engineering should also be addressed.

Last but not least, the cognition ability of technical systems is an emerging research field. It is expected to significantly enhance the inherent partial intelligence of advanced mechatronic systems. Such a trend may demand new aspects to be met regarding their design methodologies and specification techniques especially within the conceptual design phase of such systems.

7 Bibliography

[AGK+06] AXENATH, B.; GIESE, H.; KLEIN, F.; FRANK, U.: Systematic Requirements-Driven Evaluation and Synthesis of Alternative Principle Solutions for Advanced Mechatronic Systems. In: Proceedings of the 14th IEEE International Requirements Engineering Conference (RE'06), September 11-15, 2006, Minneapolis/St. Paul, Minnesota, USA, 2006, pp. 159–168

[Ben05] BENDER, K. (HRSG.): Embeded Systems ─ qualitätsorientierte Entwicklung. Springer-Verlag, Berlin, 2005

[BGH+93] BLECK, A.; GOEDECKE, M.; HUSS, S.; WALDSCHMIDT, K.: Praktikum des modernen VLSI-Entwurfs. B. G. Teubner, 1996

[Bir80] BIRKHOFER, H.: Analyse und Synthese der Funktionen technischer Produkte. Dissertation, Technische Universität Braunschweig, 1980

[Bis93] BISHOP, R.H.: Modern Control System Analysis and Design using MATLAB®. Addison-Wesley Publishing Company, Reading, 1993

[Bla72] BLASCHKE, F.: The principle of field orientation as applied to the new transvektor closed-loop control system for rotating-field machines. Siemens Review, vol. 34, May 1972, pp. 217–220

[Bol03] BOLTON, W.: Mechatronics – Electronic Control Systems in Mechanical and Electrical Engineering. Pearson Prentice Hall, Harlow, 2003

[BSK+06] BÖCKER, J.; SCHULZ, B.; KNOKE, T.; FRÖHLEKE, N.: Self-Optimization as a Framework for Advanced Control Systems. IEEE Industrial Electronics, IECON 2006 - 32nd Annual Conference of the IEEE Industrial Electronics Society, 2006, pp.4671-4675

[Buc78] BUCKLEY, R.V.: Control Engineering – Theory, Worked Examples and Problems. The Macmillan Press Ltd, London, 1978

[Bur06] BURMESTER, S.: Model-Driven Engineering of Reconfigurable Mechatronic Systems. Dissertation. Logos Verlag, Berlin, 2006

[Buu89] BUUR, J.: A framework for mechatronics design methodology (ICED). In: Proceedings of the International Conference on Engineering Design. London Harrogate, London, Band 1, 1989

[Buu90] BUUR, J.: A Theoretical Approach to Mechatronics Design. Dissertation, Institute for Engineering Design, Technical University of Denmark, 1990

[BWW72] BAYER, K.; WALDMANN, H.; WEIBELZAHL, M: Field-oriented closed-loop control of a synchronous machine with the new transvektor control system. Siemens Review, vol. 39, 1972, pp. 220–223

[Cel91] CELLIER, F.E.: Continuous System Modeling. Springer-Verlag, Heidelberg, 1991

[DH02] DAENZER, W.F.; HUBER, F.: Systems Engineering ─ Methoden und Praxis. 11. durchgesehene Auflage, Zürich: Verlag Industrielle Organisation, 2002

[DIN19226] DEUTSCHES INSTITUT FÜR NORMUNG E.V. (DIN): Regelungstechnik und Steuerungstechnik. DIN-Norm 19226, Teil 1-6, Beuth-Verlag, Berlin, 1994

[Dör98] DÖRNER, D.: Thought and Design – Research Strategies, Single-case Approach and Methods of Validation. In: Frankenberger, E.; Badke-Schaub, P.; Birkho-fer, H. (Eds.): Designers. The Key to Successful Product Development. Springer-Verlag, London, 1998

[Ehr03] EHRLENSPIEL, K.: Integrierte Produktentwicklung. Carl Hanser Verlag, München, 2003

[FGK+04] FRANK, U.; GIESE, H.; KLEIN, F.; OBERSCHELP, O.; SCHMIDT, A.; SCHULZ, B.; VÖCKING, H.; WITTING, K.; GAUSEMEIER, J. (Hrsg.): Selbstoptimierende Systeme des Maschinenbaus - Definitionen und Konzepte. HNI-Verlagsschriftenreihe Band 155, Paderborn, 2004

[Föl08] FÖLLINGER, O.: Regelungstechnik - Einführung in die Methoden und ihre Anwendung. Hüthig Verlag, Heidelberg, 2008

[Fra06] FRANK, U.: Spezifikationstechnik zur Beschreibung der Prinziplösung selbstoptimierender Systeme. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Band 175, Paderborn, 2006

[Gau02] GAUSEMEIER, J.: From Mechatronics to Self-optimizing Concepts and Structures in Mechanical Engineering, in: CADFEM Users Meeting, Friedrichshafen, 2002

[Gau05] GAUSEMEIER, J.: From Mechatronics to Self-Optimizing Concepts and Structures in Mechanical Engineering: New Approaches to Design Methodology. International Journal of Computer Intergrated Manufacturing, Volume 18, Number 7, 2005

[GEK01] GAUSEMEIER, J.; EBBESMEYER, P.; KALLMEYER, F.: Produktinnovation - Strategische Planung und Entwicklung der Produkte von morgen. Carl Hanser Verlag, München, 2001

[GFD+08a] GAUSEMEIER, J.; FRANK, U.; DONOTH, J.; KAHL, S.: Specification Technique for the Description of the Principle Solution of Self-Optimizing Systems in Mechanical Engineering. Research in Engineering Design, Springer-Verlag, London (accepted), 2008

[GFD+08b] GAUSEMEIER, J.; FRANK, U.; DONOTH, J.; KAHL, S.: Spezifikationstechnik zur Beschreibung der Prinziplösung selbstoptimierender Systeme des Maschinenbaus ─ Teil 1. Konstruktion Juli/August, Heft 7/8, Springer-VDI-Verlag, Düsseldorf, 2008, S. 59─66

[GFD+08c] GAUSEMEIER, J.; FRANK, U.; DONOTH, J.; KAHL, S.: Spezifikationstechnik zur Beschreibung der Prinziplösung selbstoptimierender Systeme des Maschinenbaus ─ Teil 2. Konstruktion Juli/August, Heft 9, Springer-VDI-Verlag, Düsseldorf, 2008, S. 91─108

[GFL+07a] GAUSEMEIER, J.; FRANK, U.; LOW, C.; HENKE, C.: From Domain-Spanning Conceptual Design to Domain-Specific Controller Design of Self-Optimizing Systems. In: Proceedings of the Systems Engineering for Future Capability, February 12-13, Loughborough, UK, 2007

[GFL+07b] GAUSEMEIER, J.; FRANK, U.; LOW, C.; HENKE, C.: Synergistic Impact of Domain-Spanning Conceptual Design on Control of Self-Optimizing Systems. In: Proceedings of the 1st IEEE International Systems Conference - SysCon 2007, April 9-12, Honolulu, USA, 2007

[GFS05] GAUSEMEIER, J.; FRANK, U.; SCHULZ, B.: Domänenübergreifende Spezifikation der Prinziplösung selbstoptimierender Systeme unter Berücksichtigung der auf das System wirkenden Einflüsse. Mechatronik 2005, Innovative Produktentwicklung, 2005

[GFS06] GAUSEMEIER, J.; FRANK, U.; STEFFEN, D..: Specifying the principle solution of tomorrow's mechanical engineering products. In: Proceedings of the DESIGN 2006, 9th International Design Conference, Dubrovnik, Croatia, 2006

[GGS+07] GAUSEMEIER, J.; GIESE, H.; SCHÄFER, W.; AXENATH, B.; FRANK,U.; HENKLER, S.; POOK, S.; TICHY, M.: Towards the Design of Self-Optimizing Mechatronic Systems: Consistency Between Domain-Spanning and Domain-Specific Models. In: Proceedings of the 16th International Conference on Engineering Design, Desgin for Society (ICED 07). 28. - 30. August, Paris, France, 2007

[GKL+08a] GAUSEMEIER, J.; KAHL, S.; LOW, C.; SCHULZ, B.: From the Principle Solution towards Controller Design of Self-Optimizing Systems. In: Proceedings of the 7th International Heinz Nixdorf Symposium, February 20-21, Paderborn, Germany, 2008

[GKL+08b] GAUSEMEIER, J.; KAHL, S.; LOW, C.; SCHULZ, B.: Systematic Development of Controller Design Based on the Principle Solution of Self-Optimizing Systems. In: Proceedings of the 10th International Design Conference - DESIGN 2008, May 19-22, Dubrovnik, Croatia, 2008

[GKP08] GAUSEMEIER, J.; KAHL, S.; POOK, S.: From Mechatronics to Self-Optimizing Systems. 7th International Heinz Nixdorf Symposium „Self-optimzing Mechatronic Systems", February 20-21, 2008, Paderborn, HNI-Verlagsschriftenreihe, Volume 223, 2008

[GLS+08] GAUSEMEIER, J.; LOW, C.; STEFFEN, D.; DEYTER, S.: Specifying the Principle Solution in Mechatronic Development Enterprises. In: Proceedings of the 2nd IEEE International Systems Conference - SysCon 2008, April 7-10, Montreal, Canada, 2008

[GZD+08a] GAUSEMEIER, J.; ZIMMER, D.; DONOTH, J.; POOK, S.; SCHMIDT, A.: Conceptual Design of Self-Optimizing Mechatronic Systems. 7th International Heinz Nixdorf Symposium „Self-optimzing Mechatronic Systems", February 20-21, 2008, Paderborn, HNIVerlagsschriftenreihe, Volume 223, 2008, pp. 35–53

[GZD+08b] GAUSEMEIER, J.; ZIMMER, D.; DONOTH, J.; POOK, S.; SCHMIDT, A.: Proceeding for the Conceptual Design of Self-Optimizing Mechatronic Systems. In: Proceedings of the 10th International Design Conference - DESIGN 2008, May 19-22, Dubrovnik, Croatia, 2008

[GZF+07] GAUSEMEIER, J.; ZIMMER, D.; FRANK, U.; SCHMIDT, A.: Von der Mechatronik zur Selbstoptimierung. In: 3. Internationales Forum Mechatronik ifm 2007, 12.-13. September, Winterthur, Schweiz, 2007

[Hau01] HAUSE, M.: The SysML Modelling Language. 5th European Systems Engineering Conference, September 18-20, Edinburgh, Scotland, 2006

[HFB06] HENKE, C.; FRÖHLEKE, N.; BÖCKER, J.: Advanced Convoy Control Strategy for Autonomously Driven Railway Vehicles. IEEE Conf. on Intelligent Transportation Systems, Toronto, 2006

[HG00] HENKE, M.; GROTSTOLLEN, H.: Modelling and Control of a Long Stator Linear Motor for a Mechatronic Railway Carriage. In: Proceedings of the 1st IFAC Conference on Mechatronic Systems, Darmstadt, Germany, 2000, p. 353─357

[HTS+08a] HENKE, C.; TICHY, M.; SCHNEIDER, T.; BÖCKER, J.; SCHÄFER, W.: System Architecture and Risk Management for Autonomous Railway Convoys. In: Proceedings of the 2nd IEEE International Systems Conference - SysCon 2008, April 7-10, Montreal, Canada, 2008

[HTS+08b] HENKE, C.; TICHY, M.; SCHNEIDER, T.; BÖCKER, J.; SCHÄFER, W.: Organization and Control of Autonomous Railway Convoys. In: Proceedings of 9th International Symposium on Advanced Vehicle Control - AVEC'08, October 6─9, Kobe, Japan, 2008

[HVB+05] HENKE, C.; VÖCKING, H.; BÖCKER, J.; FRÖHLEKE, N.; TRÄCHTLER, A.: Convoy Operation of Linear Motor Driven Railway Vehicles. The Fifth International Symposium on Linear Drives for Industry Applications, LDIA 2005, Kobe - Awaji, Japan, 2005

[HYG01] HENKE, M.; YANG, B.; GROTSTOLLEN, H.: Linear Drive System for the NBP Railway Carriage. In: Proceedings of the World Congress on Railway Research - WCRR 2001, Köln, Germany, 2001

[ILM 92] ISERMANN, R.; LACHMANN, K.H.; MATKO, D.: Adaptive Control Systems. Prentice Hall International, Hertfordshire, 1992

[Kal98] KALLMEYER, F.: Eine Methode zur Modellierung prinzipieller Lösungen mechatronischer Systeme. Dissertation, Fakultät für Maschinenbau, Universität Paderborn, HNI-Verlagsschriftenreihe, Band 42, Paderborn, 1998

[Lan00] LANGLOTZ, G.: Ein Beitrag zur Funktionsstrukturentwicklung innovativer Produkte. Dissertation, Institut für Rechneranwendung in Planung und Konstruktion, Universität Karlsruhe, Shaker-Verlag, Band 2/2000, Aachen, 2000

[Lev96] LEVINE, W.S.: The Control Handbook. CRC Press, London, 1996

[LHL01] LÜCKEL, J.; HESTERMEYER, T.; LIU-HENKE, X.: Generalization of the Cascade Principle in View of a Structured Form of Mechatronic Systems, IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM 2001), Como, Italy, 2001

[LKS00] LÜCKEL, J.; KOCH, T.; SCHMITZ, J.: Mechatronik als integrative Basis für innovative Produkte. In: VDI-Tagung „Mechatronik – Mechanisch elektrische Antriebstechnik", 29. – 30. März 2000, Wiesloch, VDI-Verlag, Düsseldorf, 2000

[LSO01] LEE, K.D.; SUH, N.P.; OH; J.H.: Axiomatic design of machine control system, CIRP Annals - Manufacturing Technology, 50 (1), 2001, pp. 109-114

[Oga02] OGATA, K.: Modern Control Engineering. Prentice Hall, New Jersey, 2002

[OHG04] OBERSCHELP, O.; HESTERMEYER, T.; GIESE, H.: Strukturierte Informationsverarbeitung für selbstoptimierende mechatronische Systeme. In: Gausemeier, J.; Wallaschek, J. (Hrsg.): 2. Paderborner Workshop Intelligente Mechatronische Systeme, 25.-26. März 2004, HNI-Verlagsschriftenreihe, Band 145, Paderborn, 2004

[OMG03] OBJECT MANAGEMENT GROUP: UML for System Engineering Request for Proposal. Document ad/03-03-41, 2003

[OMG07] OBJECT MANAGEMENT GROUP: OMG Systems Modeling Language (OMG SysML™) V1.0 ─ OMG Available Specification, 2007

[PB96] PAHL, G.; BEITZ, W.: Engineering Design - A Systematic Approach, Second Edition, Springer Verlag, Berlin, Heidelberg, New York, 1996

[PBF+07] PAHL, G., BEITZ, W., FELDHUSEN, J., GROTE, K.-H.: Engineering Design – A Systematic Approach. Springer Verlag, London, 3rd edition, 2007

[Pet07] PETERS, W: FPGA-basierte Ansteuerung einer Leistungselektronik für Drehstromantriebe. Studienarbeit, Fachgebiet Leistungselektronik und Elektrische Antriebstechnik, Universität Paderborn, 2007

[Pet08] PETERS, W: Realisierung einer FPGA-basierte Antriebsregelung für einen Drehstrommotor. Diplomarbeit, Fachgebiet Leistungselektronik und Elektrische Antriebstechnik, Universität Paderborn, 2008

[Pik08] PIKA , S: Realisierung einer Antriebsregelung in einem CPU-FPGA System mit einem Echtzeitbetriebssystem. Masterarbeit, Fachgebiet Leistungselektronik und Elektrische Antriebstechnik, Universität Paderborn, 2008

[Rot00] ROTH, K.-H.: Konstruieren mit Konstruktionskatalogen. Springer-Verlag, Band 1 Konstruktionslehre, 3rd edition, Berlin, 2000

[Sch01] SCHRÖDER, D. : Elektrische Antriebe - Regelung von Antriebssystemen, Springer-Verlag Berlin Heidelberg New York, 2001

[SD00] SUH, N.P.; DO, S.H.: Axiomatic Design of Software Systems, Annals of CIRP 49 (No. 1), 2000, pp. 95–100

[SFB01] SFB 614: Einrichtungsantrag für den Sonderforschungsbereich 1799 (ab 1. Juli 2002: 614) 'Selbstoptimierende Systeme des Maschinenbaus'; Universität Paderborn, 2001

[SFB04] SFB 614 - Finanzierungsantrag für den Sonderforschungsbereich 614 "Selbst-optimierende Systeme des Maschinenbaus", 2005-2009. Universität Paderborn, 2004

[Som06] SOMMERVILLE, I.: Software Engineering 8. Addision Wesley, Harlow, 8th edition, 2006

[SPH+07] SCHULZ, B.; PAIZ, C.; HAGEMEYER, J.; MATHAPATI, S.; PORRMANN, M.; BÖCKER, J.: Run-Time Reconfiguration of FPGA-Based Drive Controllers. 12th European Conference on Power Electronics and Applications, Aalborg, Denmark, 2007

[Suh01] SUH, N.P.: Axiomatic Design ─ Advances and Applications. Oxford University Press, New York, 2001

[Suh04] SUH, N. P.: On functional periodicity as the basis for langterm stability of engineered and natural systems and its relationship to physical laws. In: Research in Engineering Design, Springer-Verlag, London, 2004

[Suh95] SUH, N.P.: Axiomatic Design of Mechanical Systems. Journal of Vibration and Acoustics, Volume 117, Issue B, June 1995, pp. 2-10

[Suh98] SUH, N .P.: Axiomatic Design Theory for Systems. In: Research in Engineering Design, No. 10, Springer-Verlag, London, 1998

[Sys04] SYSML PARTNER: System Modeling Language (SysML) Specification. Draft – Version 0.85R1. http//www.sysml.org, December 3, 2004

[Tec04] TECHNICAL BOARD INTERNATIONAL COUNCIL ON SYSTEMS ENGINEERING (INCOSE): Systems Engnineering Handbook. Version 2a, 2004

[TMV 06] TRÄCHTLER, A.; MÜNCH, E.; VÖCKING, H.: Iterative Learning and Self-Optimization Techniques for the Innovative RailCab-System. In: Proceedings of the 32nd Annual Conference of the IEEE Industrial Electronics Society – IECON`06, Paris, France, 2006

[Trä06] TRÄCHTLER, A.: RailCab – mit innovativer Mechatronik zum Schienenverkehrsystem der Zukunft. In: VDE Kongress 2006, Aachen, 2006

[Trä07] TRÄCHTLER, A.: Modellbasierter Entwurf mechatronischer Fahrwerkssysteme. 5. Paderborner Workshop „Entwurf mechatronischer Systeme“, 22.-23. März 2007, Paderborn, HNI-Verlagsschriftenreihe, Band 210, 2007, S. 3-13

[VDI2206] VEREIN DEUTSCHER INGENIEURE (VDI): VDI-guideline 2206 – Design methodology for mechatronic systems. Beuth-Verlag, Berlin, 2004

[Vöc03] VÖCKING, H.: Multirate-Verfahren und Umschaltstrategien für verteilte Reglersysteme. Master’s thesis, University of Paderborn, 2003

[Wei07] WEILKIENS, T.: Systems Engineering mit SysML/UML – Modeling, Analysis, Design. Morgan Kaufmann Publishers, 2007

[ZBS+05] ZIMMER, D.; BÖCKER, J.; SCHMIDT, A.; SCHULZ, B.: Elektromagnetische Direktantriebe im Vergleich. In: Antriebstechnik, number 2/2005, Vereinigte Fachverlage GmbH, Mainz, 2005

[ZS05] ZIMMER, D.; SCHMIDT, A.: Der Luftspalt bei Linearmotor-getriebenen Schienenfahrzeugen. In: Antriebstechnik Nr. 2, 2005

Appendix

A1 Remarks for the Self-Optimizing Motor Drive

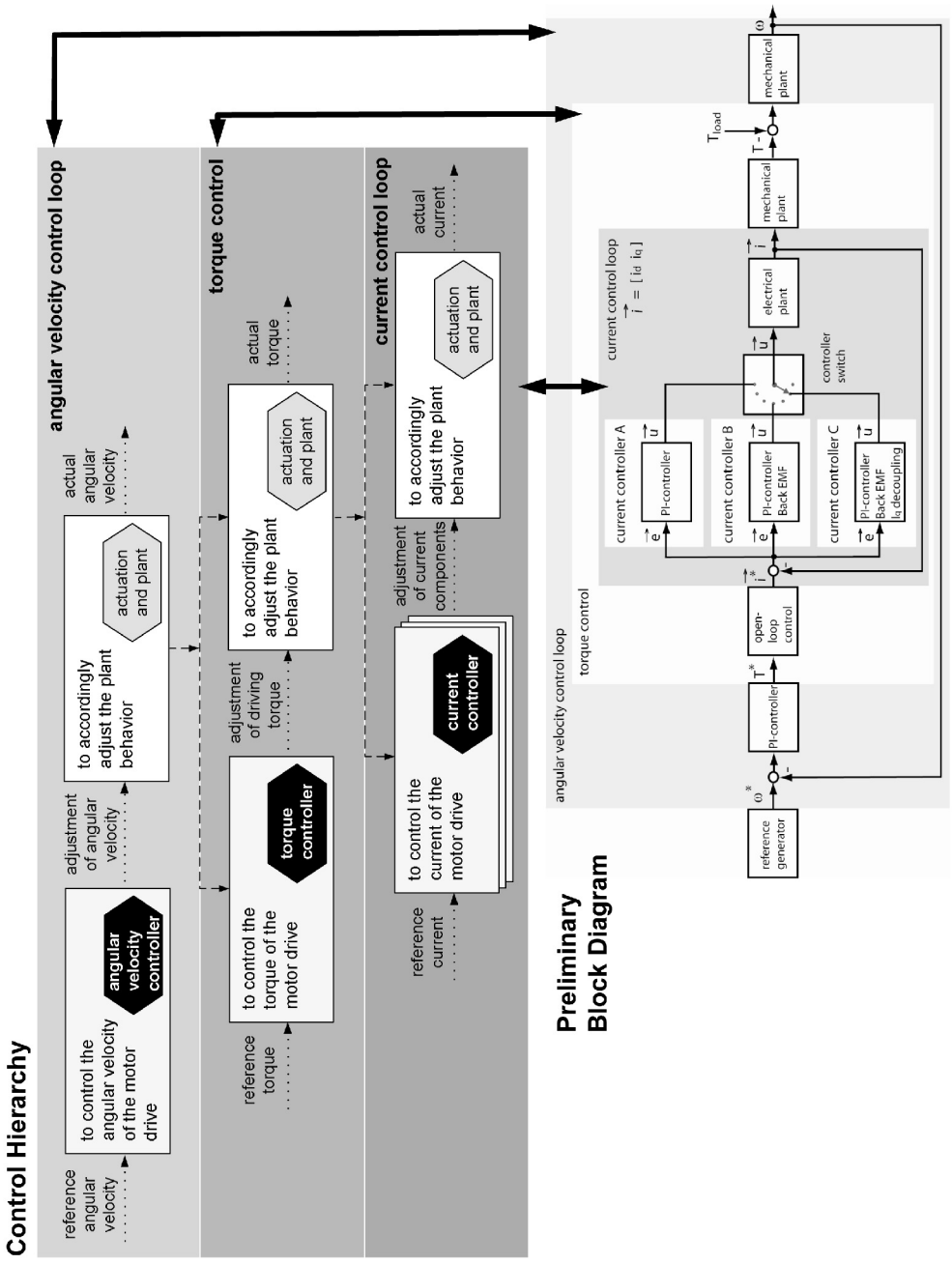


Figure A-1: Dependencies between the control hierarchy and the preliminary block diagram of a self-optimizing motor drive

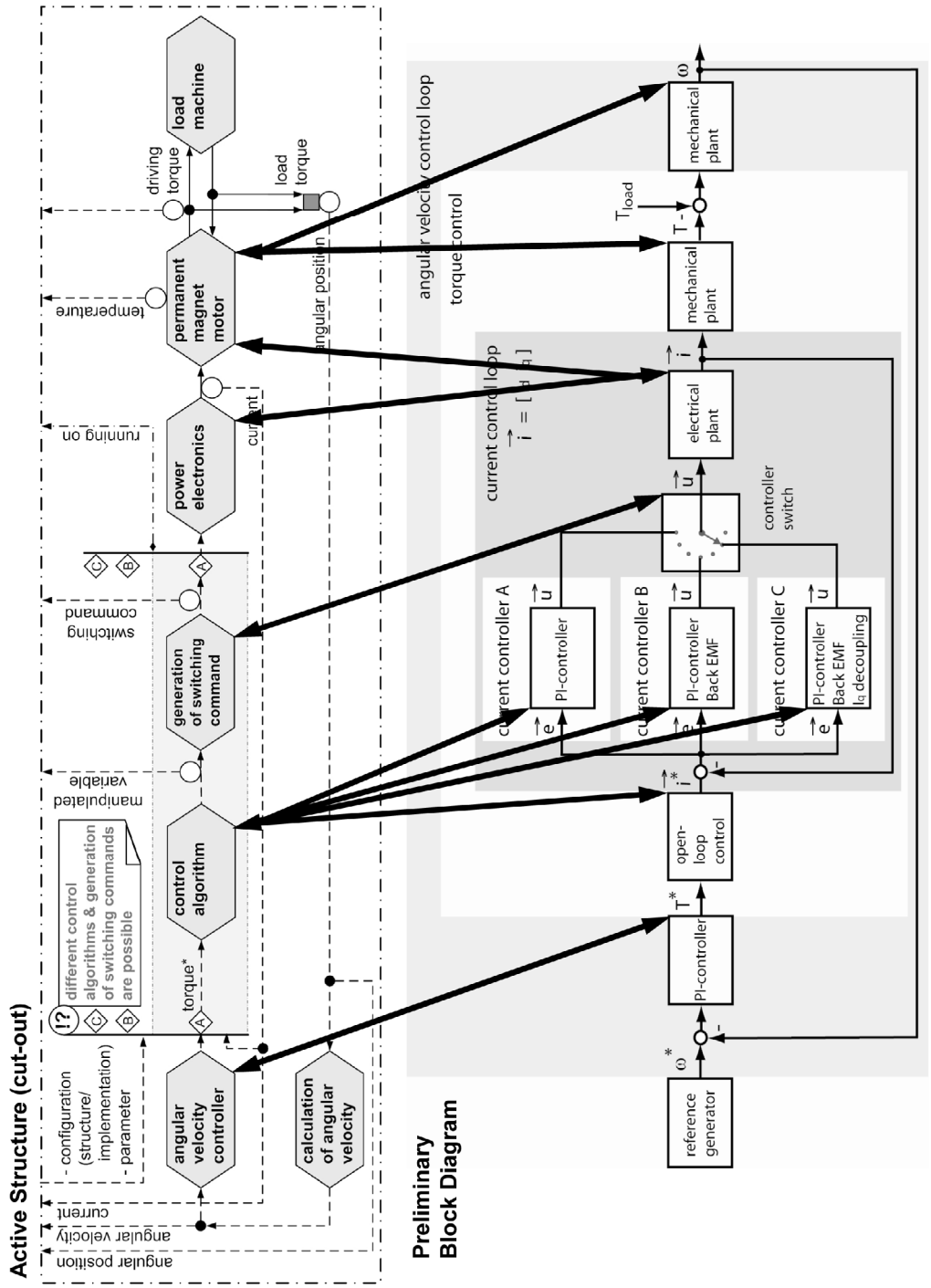


Figure A-2: Dependencies between the system elements of the active structure and the preliminary block diagram of a self-optimizing motor drive

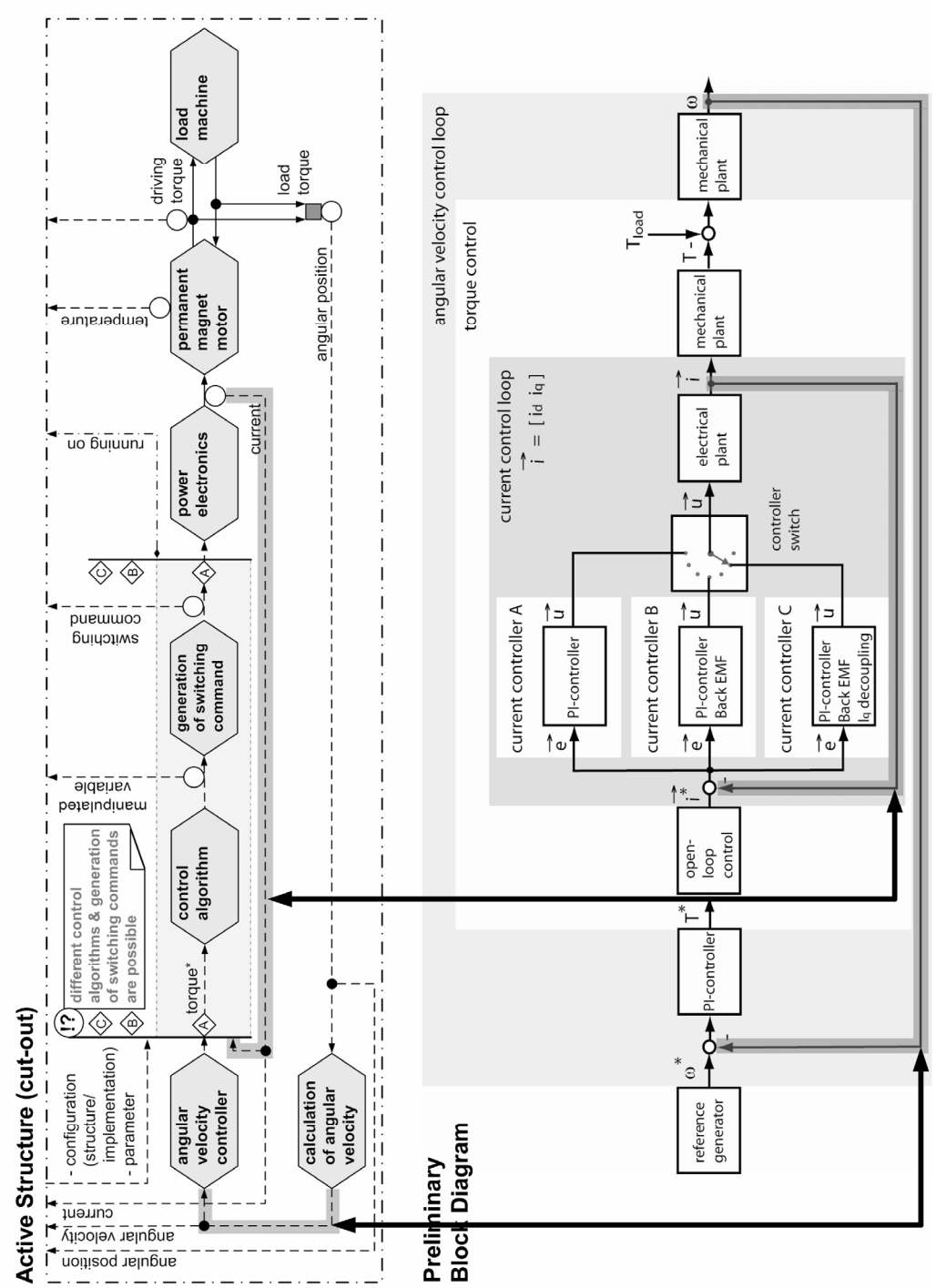


Figure A-3: Dependencies between the feedback loops of the active structure and the preliminary block diagram of a self-optimizing motor drive

A2 Remarks for the Autonomous Railway Convoy

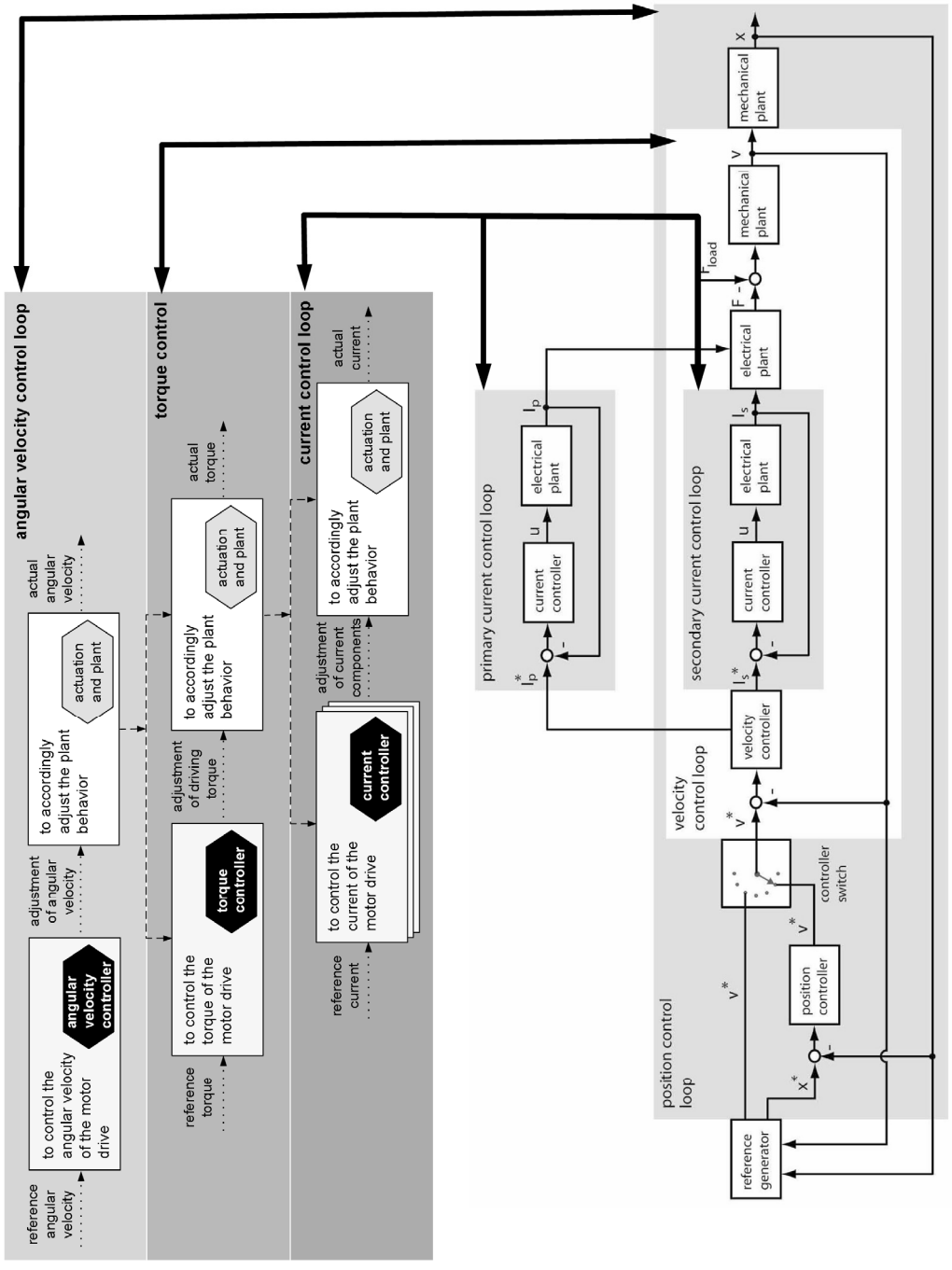


Figure A-4: Dependencies between the control hierarchy and the preliminary block diagram of an autonomous railway convoy

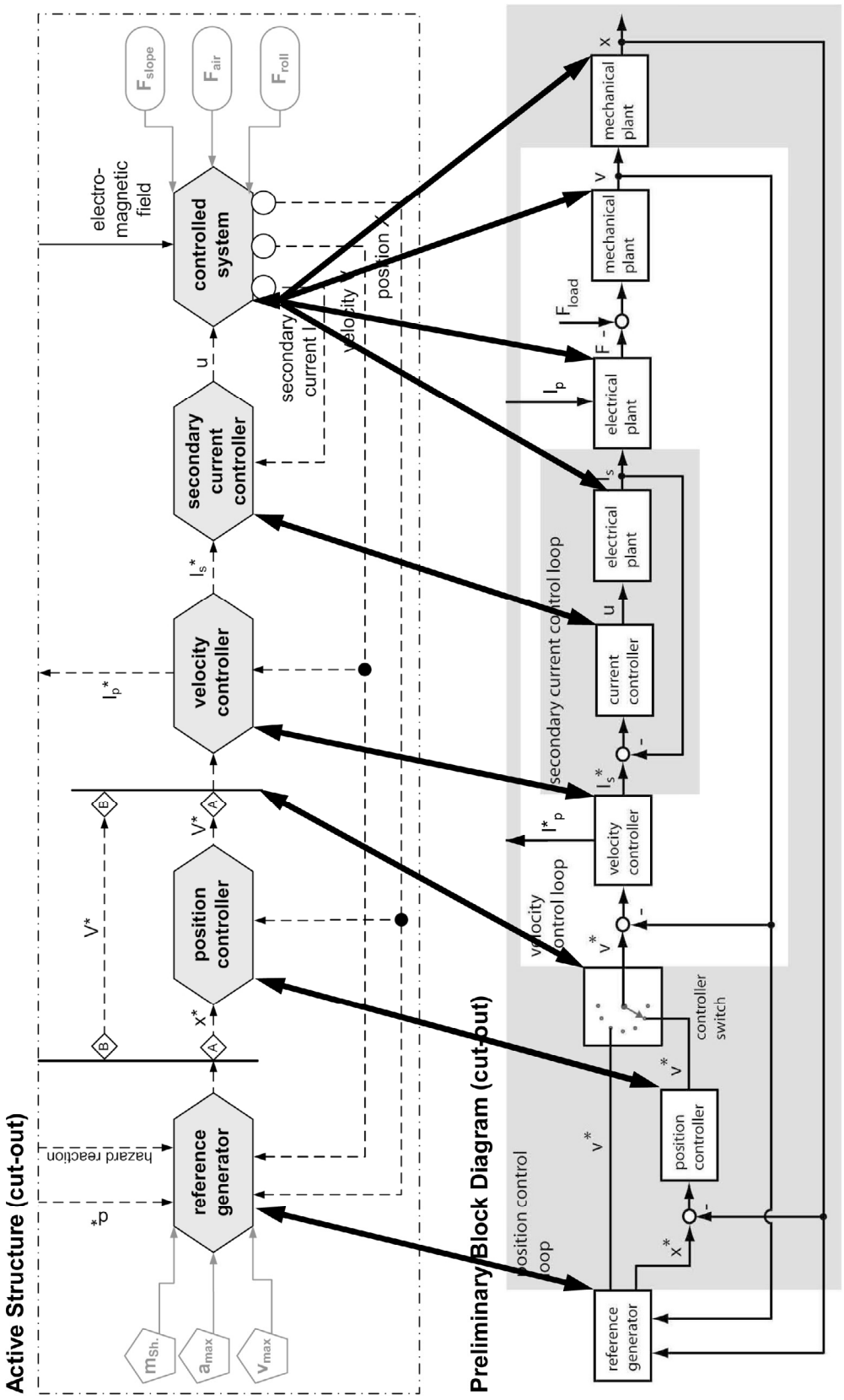


Figure A-5: Dependencies between the system elements of the active structure and the preliminary block diagram of an autonomous railway convoy

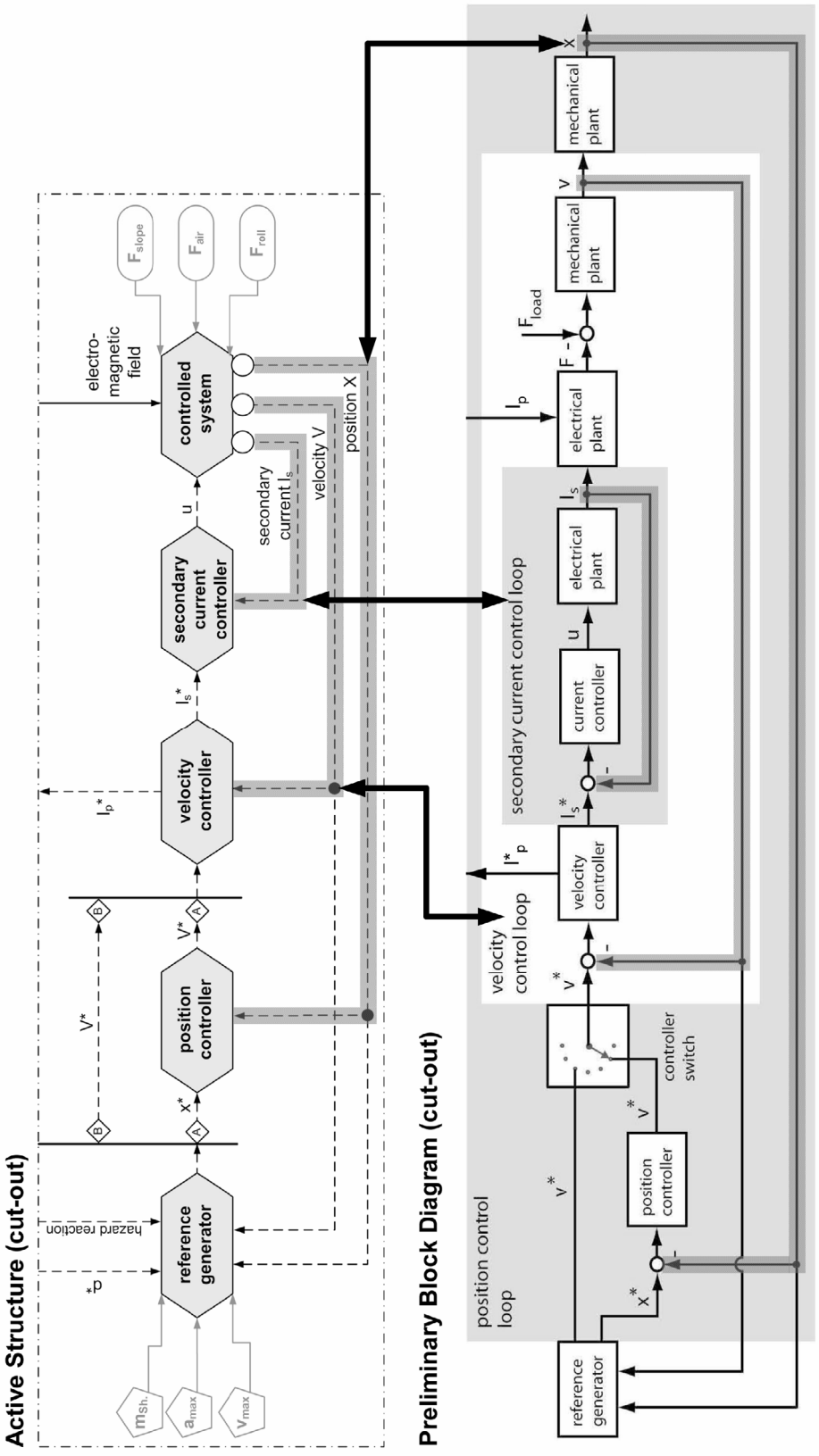


Figure A-6: Dependencies between the feedback loops of the active structure and the preliminary block diagram of an autonomous railway convoy