POLITECNICO DI MILANO
Facoltà di Ingegneria Industriale
Laurea Specialistica in Ingegneria Meccanica

"Computer-Supported Modeling Techniques to Analyze Goal Interrelations within Strategic Product Planning Systems"

Relatore: Prof. Gaetano Cascini
Correlatore: Dipl. Ing Clemens Hepperle

Simon Eichiner - 707363

Anno Accademico 2010/11

Technische Universität München

Diplomarbeit

Nr. 1212

Computer-Supported Modeling Techniques to Analyze Goal Interrelations within Strategic Product Planning Systems

Simon Eichiner

Ich versichere, dass ich die Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

PRODUKTENTWICKLUNG
Technische Universität München
o. Prof. Dr.-Ing. Udo Lindemann

Diplomarbeit
Nr. 1212

für Herrn Simon Eichiner
Matrikel-Nr.: 2714886

Thema: Computer-Supported Modeling Techniques to Analyze Goal Interrelations within Strategic Product Planning Systems

Initial situation:

Manufacturing companies face the challenge of enhancing their ability and productivity in innovating products due to an increasingly competitive environment which is characterised by rapidly changing technologies and dynamic market needs. In this context frontloading an integrated lifecycle understanding to the early stages of planning future products distinguishes itself as a promising approach. The ability to comprehend interrelations among and in-between future product and process potentials and demands increases the transparency of planned goods within corresponding innovation processes. This transparency allows anticipating possible goal conflicts and thus averts preventable changes along the future lifecycle. In consequence, companies avoid unnecessarily provoked lifecycle costs, which grow exponentially the later unintended changes are made within the lifecycle.

In this context, a detailed concept to analyse and elaborate on interrelations among anticipated demands and potentials along the whole lifecycle has been developed within the collaborative research center SFB 768 "Managing cycles in innovation processes". This concept, consisting of a framework for information flows and a method to handle this information in respect to analyse goal interrelation, needs to be expanded, both considering to embed the approach within planning processes and to implement these within respective computer supported tools.

Therefore, work carried in this thesis should primarily focus the conceptualization of computer-based tools for modeling interrelations within the context of analyzing goal interrelations. The currently existing approach should be extended to allow the analysis of multiple planning horizons and multiple scenarios for these planning horizons as well as corresponding qualitative and fuzzy information. Furthermore, the approach should consist of features to handle the update of respective models comfortably.

Taks and approach:

- Elaborating on research questions:
    - Dealing with current status of approach to analyze goal interrelations developed within the SFB 768 „Managing cycles in innovation processes"

  - Literature research concerning formalized modeling techniques to handle dependencies among elements deriving from different domains
  - Detailed presentation of research questions

- Analysis of further dimensions to be considered within approach by extending perspective on multiple time horizons and multiple scenarios
- Requirements analysis for developing concept of computer-supported modeling techniques
  - based on analysis of current advantages and disadvantages of existing modeling techniques
  - based on updated approach for analyzing goal interrelations
  - under consideration of possible updates of modeled elements
- Elaboration of formalized framework for computer-supported modeling of goal interrelations based on detected requirements
- Prototypic implementation of concept
  - using existing modeling techniques
  - providing insights into further development possibilities
- Embedding concept for computer-supported modeling techniques for the analysis of goal interrelations within the product planning process
- Validation of results based on an appropriate, industry-relevant example.
- Documentation of procedure and results in this thesis; reflection of results and presentation of future steps.

Die Arbeit bleibt Eigentum des Lehrstuhls.

betr. Assistent:	Clemens Hepperle
Partner in Industrie/ Forschung:	
ausgegeben am:	15.05.2011
abgegeben am:	

Garching, den

Computer-Supported Modeling Techniques to Analyze Goal Interrelations within Strategic Product Planning Systems

1 Introduction

1.1 Motivation

Since competitive pressure has in increased, manufacturing enterprises face the challenge to enhance their productivity in innovating products. This challenge is especially demanding in an environment, which is characterized by dynamic market needs, and rapidly changing technologies (COOPER & EDGETT 2005, p. 1). Consequently, enterprises need to be more efficient in their innovating processes and to avoid unnecessarily provoked lifecycle costs. A promising approach in this context consists in increasing the transparency of planned products within the corresponding innovation process. One way to achieve this is through methods supporting the ability to comprehend interrelations among and in-between potentials and demands of future products and processes (HEPPERLE ET AL. 2011a). This comprehension allows anticipation possible goal conflicts and thus reduces the amount of changes along the future lifecycle. These changes are directly linked to additional costs, which furthermore grow exponentially the later unintended changes are made within the lifecycle (EHRLENSPIEL ET AL. 2007, p. 242). However, having a grown complexity of developed products and process, these methods demand for more rigorous and formalized practices.

In response to this demand, along with advancements in computer technology, a common practice of engineering and product design is the fundamental transition from a document-based approach to a model-based approach. In the model-based approach, the emphasis shifts from producing and controlling documentation to producing and controlling a coherent model of the system with the support of computer technologies (FRIEDENTHAL ET AL. 2008, p. xi). Computers have advantages in representing, organizing, storing, and retrieving digital information, what makes them valuable tools. Thus, computers can especially be used to supporting the resolutions of problems or to carry out repeating tasks within product development. Moreover, with collaboration between humans and computers, extremely challenging tasks in product development for computers, e.g. creating a synthesis of a problem, can be overcome (SHEA 2010, p. 8). Consequently, the application of computer-supported methods can help managing the complexity, while at the same time improves design quality and cycle time, eases communications among a diverse development team, and facilitates knowledge capturing and evaluation.

1.2 Goals and Objectives

This thesis focuses on an especially developed approach, supporting the analysis and elaboration on interrelations among anticipated demands and potentials along the whole lifecycle. HEPPERLE ET AL. (2011a) and FÖRG (2010) are presenting this approach, allowing an early identification of goal interrelations and goal conflicts for the phase of product planning within the innovation process. Goal interrelations occur by applying essential product concepts for goals that are influencing the same parameters, which can lead to conflicts, in the case of incompatible circumstances. In order to handle these goal conflicts HEPPERLE ET AL. (2011a) and FÖRG (2010) are providing a graph and matrix-based approach, which links product goals among each other through possible solutions and their characterizing parameters. Figure 1-1 shows an abstract example for a graph-based representation of a product planning system and occurring goal interrelations. As it can be seen, demands are leading to goals, which are directly linked to a core function of a product or the development process. These core functions can be satisfied by various solutions, which are finally influencing the various parameters. Additionally, the approach considers an integrated lifecycle perspective and thus the various sources of demands and phases of a product's life and their complex interconnections.

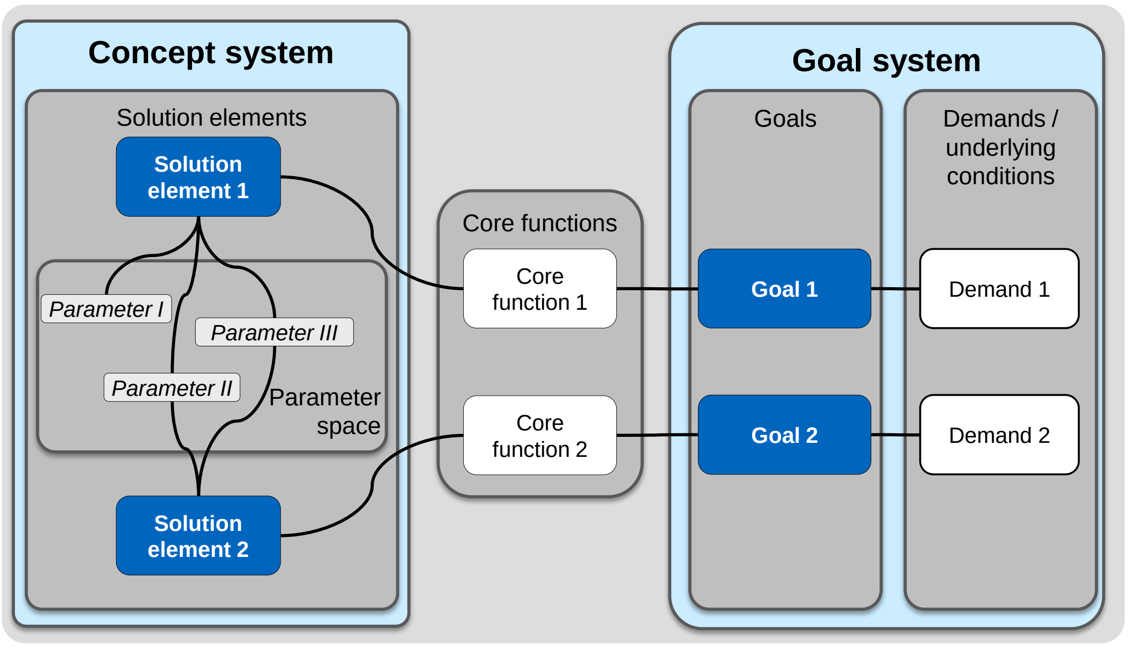


Figure 1-1 Abstract example of a graph-based representation of goal interrelations (HEPPERLE ET AL. 2011a)

A disadvantage of the approach is the limited amount of elements, which are manageable by humans. With increased product complexity, the number of elements and therefore the amount of stored information, which needs to be handled, raises significantly. Moreover, if several product generations and their future developments are object of the planning process, the number of considered elements and their interrelations grows even faster. Containing a large amount of stored information, the whole system arrives at a point, where humans spend more time on searching information than using it effectively (SHEA 2010, p. 8). Consequently, computer-techniques should be used to support information handling within the approach.

This paper aims to provide a concept for a computer-supported implementation of a goal interrelations analysis within a lifecycle-oriented product planning system through various product generations. Thus, based on the existing approach, possibilities for its enhancement as well as an implementation with various computer-supported modeling techniques need to be analyzed. The derived information should results in a specification of general requirements and a framework for the implementation of a computer-supported goal interrelation analysis.

Finally, a conceptual implementation is needed, showing a general outline for a software tool, which is based in previously achieved specifications.

1.3 Structures

Focusing on achieving the goal presented in chapter 1.2, the structure of this thesis is presented in this section. An overview on the general structure is shown in Figure 1-2, where the order of the various chapters as well as their relations can be seen. In order to provide a better understanding of the structure of this thesis, each chapter is further described in the following.

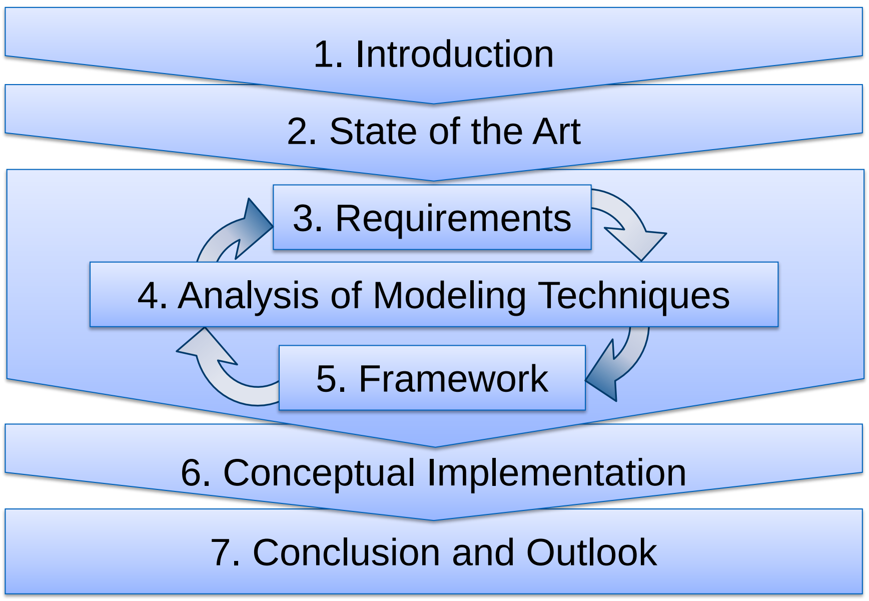

1. Introduction
2. State of the Art
3. Requirements
4. Analysis of Modeling Techniques
5. Framework
6. Conceptual Implementation
7. Conclusion and Outlook

Figure 1-2 Overview on the structure of this thesis

First, the thesis and its main topic are introduced in chapter 1, by primarily presenting the background motivation of this thesis. Based on this motivation the general goals and objectives are derived, followed by the description of the structure.

Chapter 2 provides the state of the art in related research. Therefore, it is divided into three parts: related work, related approaches and fundamental approach for the analysis of goal interrelations. The first part concerning related work regards several fields of research, as a general outline for this thesis. The following section about related approaches describes several different approaches, principles, methods and techniques, which are applicable to this research project. Since the approach for the analysis of goal interrelations presents the fundament of this paper, it is introduced separately in order to deliver a more detailed description. Having provided the background information on this chapter the main part of this paper becomes coherent.

As can be seen in Figure 1-2, the chapters 3, 4, and 5 that represent the main part of the thesis, are connected through an iteration loop. The iteration arrows are indicating that the content has not been developed in a linear way, but through various iterations. Therefore, the content of theses chapters is strongly linked with each other. The presented order, beginning with requirements, being followed by the analysis of modeling techniques and finally the definition of a general framework has been chosen to facilitate the reading of this paper.

Chapter 3 presents the collected requirements for a computer-supported goal interrelation analysis. These requirements are subdivided into three main parts: Requirements regarding the analysis of goal interrelations, requirements for the implementation of the related content, and requirements concerning the associated software implementation. All collected requirements have been improved by various iterations and present the final result. Having presented the detailed requirements to achieve the goal of this thesis, the knowledge gained strongly supports the general understanding of the following chapters.

Chapter 4 describes the analysis of several computer-supported modeling techniques and their aptitude for the analysis of goal interrelations. Therefore, the general conditions applied for the analysis of these techniques are specified. Based on these conditions, the four investigated techniques and their implementations to satisfy the requirements are presented. The gained results of the investigated implementations are finally confronted with each other and the requirements defined in chapter 3.

Based on the results discovered in chapter 4 and on the requirements listed in chapter 3, a framework for a general solution of a computer-supported goal interrelation analysis is shown in chapter 5. Hence, basic attributes and properties for the enhancement of the general approach are presented, as well as a systematical definition of all necessary elements. Furthermore, the main features and the requested structure of a software tool for goal interrelation analysis are defined within this section.

Chapter 6 finally presents an exemplary and conceptual implementation of a goal interrelation analysis, considering the requirements, the results of the analysis, and the framework. Therefore, a detailed example of lifecycle oriented product planning system is introduced, and used for an application of the approach. Furthermore, a conceptual implementation of a software tool is presented, satisfying the framework and the requirements.

Lastly, chapter 7 closes the thesis with a conclusion and a critical discussion of the results. Additionally an outlook is given, identifying necessary improvements, the need for an implementation and the application of the received results.

2 State of the Art

In this chapter, the fundamental background of the goal interrelation analysis and its application is provided. Therefore, the state of the art in related research is presented. This is done by firstly introducing the related work, being the sources of the request for the analysis of goal interrelations. Following the theoretical approach for analyzing goal interrelations, is presented. Finally related approaches are described, which include several principles, methods and techniques that are directly applied within this thesis. Having provided the background for this research project in this chapter, all necessary information has been delivered to understand the complete background of this thesis.

2.1 Related Work

Within this section, the related work to this thesis is introduced. In order to understand the application of a *Goal Interrelation Analysis* (GIA), it is vital to know the main expressions and potential fields of application. Consequently, strategic product planning, innovation management, product lifecycle management, complexity management, and their basic meanings are presented.

2.1.1 Strategic Product Planning

In general, the *Strategic Product Planning* (SPP) is seen as the initial phase of the product development process. However, a more detailed overview on the SPP phase, explaining the concept followed by the placement of the phase within the process to develop new products.

SPP implies the reference to a strategic behavior in the field of planning a new product. Commonly speaking ‘strategic’ characterizes long-term planning followed by the marshaling and allocation of resources to achieve pre-defined goals (REA & KERZNER 1997, p. 2). Transferred to the field of company behavior it characterizes the way to influence the relationship of an enterprise to its environment in order to secure long-term success. This can be enabled mainly by securing the competitive position and offering marketable products (TIETZE 2003, pp. 17ff).

With these properties, SPP is commonly placed in an early stage of the product innovation process. GAUSEMEIER (2001, p. 43) developed a model of the innovation process showing the three main cycles of the innovation process, presenting the SPP as the initial one (see Figure 2-1). Aim of this phase is to identify innovation projects that support broader business strategy which also account for changes in competitive and technological environments. Methods like market segmentation analysis, scenario technique, technological positioning, portfolio analysis and organizational analysis of resources and capabilities can be used. Having gained the relevant information, opportunities can be received and used as inputs for the following steps in the product development process (LIMBERG 2008, pp. 23–24; BRAUN 2005, pp. 17–18).

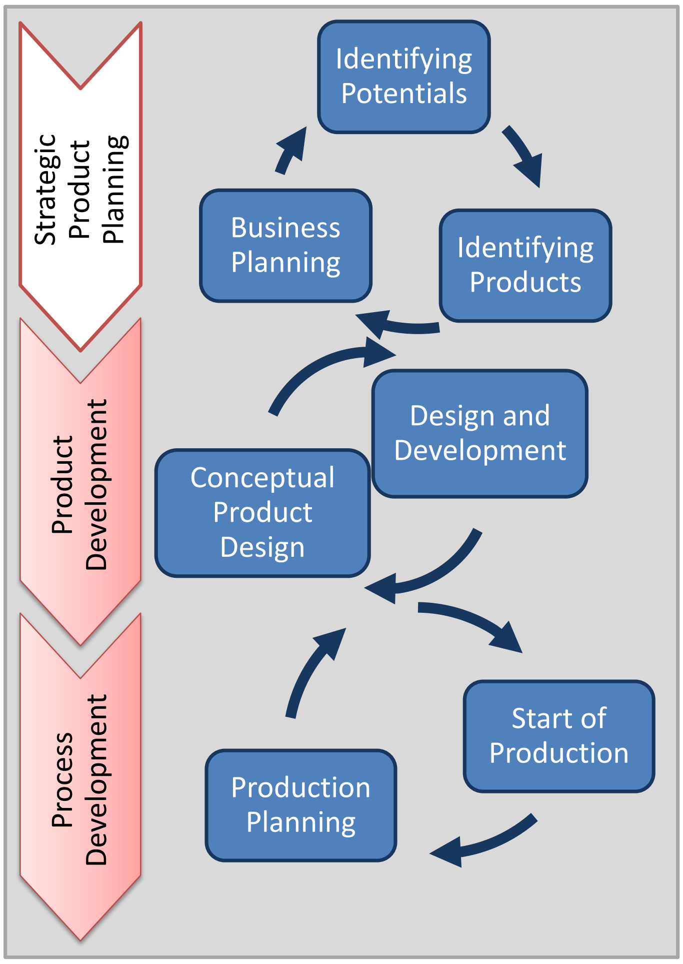


Figure 2-1 Strategic product planning as partial phase of the innovation process according to GAUSEMEIER *(2001, p. 43)*

To sum this up one can note that the main task of the SPP is to analyze possible solutions and potentials in order to create products, which satisfy demands within a long term business planning. These findings can be supported by a consequent analysis of goal interrelations in order to identify possible conflicts and strongly interrelated goals.

2.1.2 Innovation and Innovation Management

Another related field of work is innovation management. Companies can gather significant advantages in a competitive environment by inventing and offering innovative and creative products. Unfortunately, these products also imply the risk of a failure. Therefore, innovation management helps to minimize risk and to design successful products with substantial benefits for customers and, as an optimum, uniqueness (SCHWANINGER 2005, p. 40; LAMBERTZ ET AL. 1996, pp. 30–32). To introduce innovation management, firstly the expressions are defined and classified, followed by the management process for them.

Innovation is defined as qualitatively novel product or procedure, which use new knowledge to distinguish noticeable from the state of the art (HAUSCHILDT & SALOMO 2011, p. 3; AFUAH 2003, p. 13). It is not just a straight improvement on a technological problem. It is a more complex development (HAUSCHILDT & SALOMO 2011, p. 4). Innovations are the combination of new technological and market knowledge through companies' competences and assets in order to achieve new products that customers will want (see Figure 2-2). Thus, a product is new in that its cost is lower, it has improved attributes and attributes it never had before (AFUAH 2003, pp. 5–6).

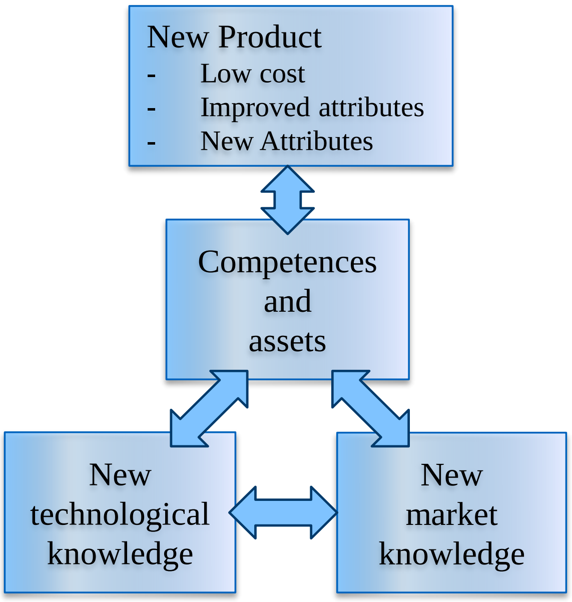


Figure 2-2 Composition of an innovation (AFUAH 2003, p. 5)

HAUSCHILDT & SALOMO (2011, pp. 5–23) further provide several dimensions to classify an innovation shown in Figure 2-3. Therefore, it is possible to ascertain something as innovative by looking at its content, intensity, subject and process. The content describes thereby what is new such as a product or process, an attribute, a technical, organizational, business related or a postindustrial innovation. The intensity refers to how new an innovation is, either by a fact or by a certain degree. The subject regards for whom it is new which can be experts, management, branches, nations or humanity. Moreover, it may be considered where, along the process of creating innovations, the innovation can be found. Having determined these dimensions, innovations can be compared in order to become manageable and therefore successful.

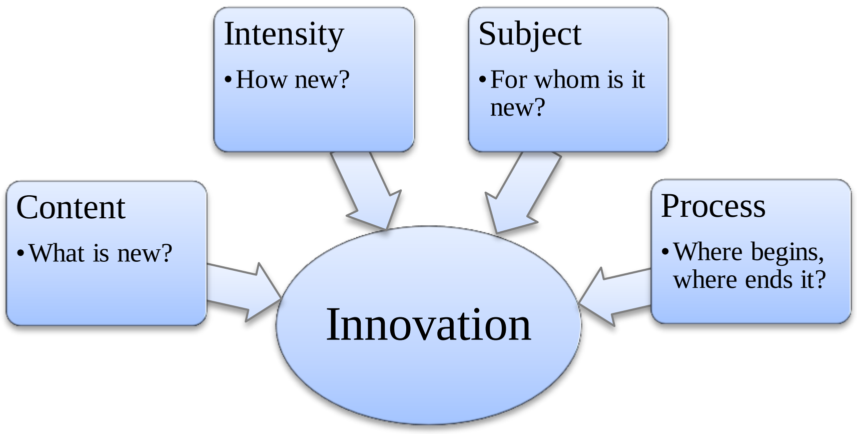


Figure 2-3 Dimensions of an innovation according to H*AUSCHILDT* & S*ALOMO (2011, pp. 5–23)*

Finally, innovation management can be described as an active organization of innovations and the involved system. This includes not only managing the process of an innovation, but the entire institution, where the processes occurs (HAUSCHILDT & SALOMO 2011, p. 29). Therefore, the innovation management is focused towards understanding and controlling the appearance and diffusion of innovations within several areas of influence (HERRMANN 2010, p. 238). Innovation management has to consider the inter-divisional character and the single phases of the innovation process while working on strategic, tactical and operative tasks. Further information in this field is affected by complexity dynamic and non-transparency (SEIDEL 2005, pp. 14f.).

The product itself and the process of the product design within innovation management are relevant for this thesis. Limiting the view towards a new product, innovation management equals strategic product planning (HERRMANN 2010, p. 242). The presented concept in this thesis is used to identify goal interrelations during product planning process that can also be applied as a method in the innovation management.

2.1.3 Product Lifecycle and Product Lifecycle Management

The product lifecycle and product lifecycle management are in two different ways connected to this thesis. Firstly, the product lifecycle is a significant part of the method for integrated product planning. It is used as a source for the identification of demands and considered interrelations chains are classified according to their occurrence. Furthermore, product planning is a part of the product lifecycle and therefore the presented concept can be used as a method within it. Hence, the product lifecycle and *Product Lifecycle Management* (PLM) are presented in the following section.

A product lifecycle includes all phases in the life of a product and helps considering important influences and requirements of a product. These cycles can be seen, analogue to biological or natural systems with limited lifespans, as every phase from the first idea of a product, over research, product design, construction, production, sales and service until the disposal (ARNOLD ET AL. 2005, p. 13; HERRMANN 2010, pp. 63ff). Having included the whole life of a product into the planning process sharpens the system understanding and therefore possible interrelations can more accurately be identified (EHRLENSPIEL ET AL. 2007, pp. 242ff). The detection of goal conflict can be supported by anticipating interrelations, company processes, and context factors, leading to a reduction of changes (GEORGANTZAS & ACAR 1995, pp. 13ff; GAUSEMEIER ET AL. 2004, pp. 46f.; HEPPERLE ET AL. 2011b). Supporting this analysis, various models of lifecycles exist according to the relevant subject.

Within the context of this work, especially the integrated lifecycle model defined by HEPPERLE ET AL. (2009) has to be mentioned. The model is a synthesis of various models from different areas of product design and can be seen in Figure 2-4. The integrated perception of a lifecycle model enhances the classical linear market cycle and disposal cycle by a design cycle, an observation cycle and parallel phases (HEPPERLE ET AL. 2009; HOECK 2005, pp. 35ff.; HERRMANN 2010, p. 71). As a result the lifecycle model is built of superordinate phases of 'Product planning', 'Product development and design', 'Production process preparation', 'Production', 'Distribution', 'Utilization', 'Maintenance', 'Modernization lifecycle' and 'Product disposal'. If requested, these superordinate phases can be split further down to more than 25 working phases and more than 15 product states (HEPPERLE ET AL. 2011b).

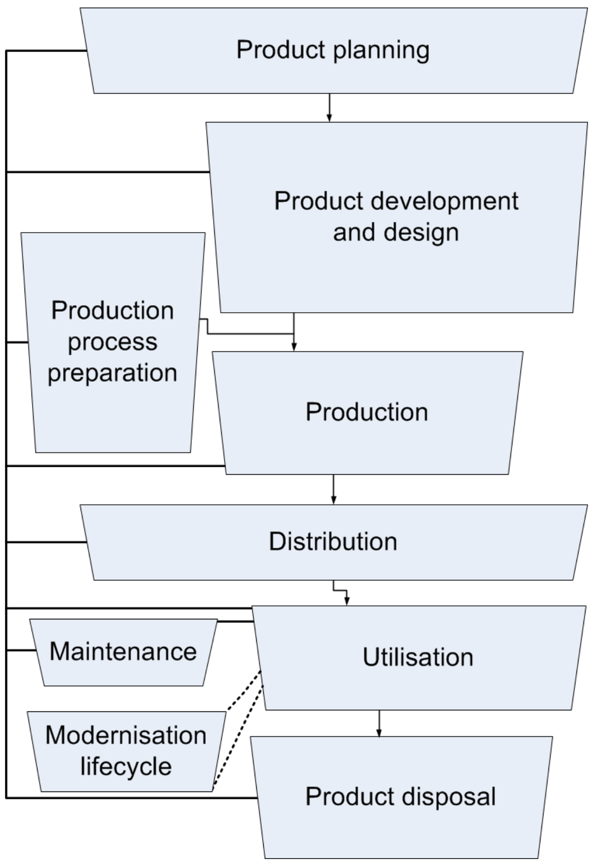


Figure 2-4 Integrated product lifecycle model according to HEPPERLE ET AL.(2009)

PLM, besides, is a knowledge-based, integrated concept to control, manage, handle and supervise the implementation of a lifecycle-oriented product planning process (ARNOLD ET AL. 2005, p. 14; FELDHUSEN & GEBHARDT 2008, p. 20). Moreover, with the 'Liebensteiner Theses', a general explanation of product lifecycle management has been defined (SENDLER 2009, p. 27):

- PLM is a concept, no system and no (completed) solution.
- In order to implement/realize a PLM concept solution components are needed. These include CAD, CAE, CAM, VR, PDM and other application for the product design process.
- In addition, interfaces to other applications like ERP, SCM or CRM are components of the PLM concept.
- PLM providers offer components and/or services for the implementation of the PLM concept.

A graphical classification of PLM into typical processes of a company is provided by (ARNOLD ET AL. 2005, p. 14) (see Figure 2-5). While the enterprise resource planning system addresses mainly the production process, the product lifecycle management focuses orthogonally to it the product design and connected processes. Furthermore, the implementation of the integrated PLM processes is enabled by specialized software systems, methods and information supporting interconnection between all phases.

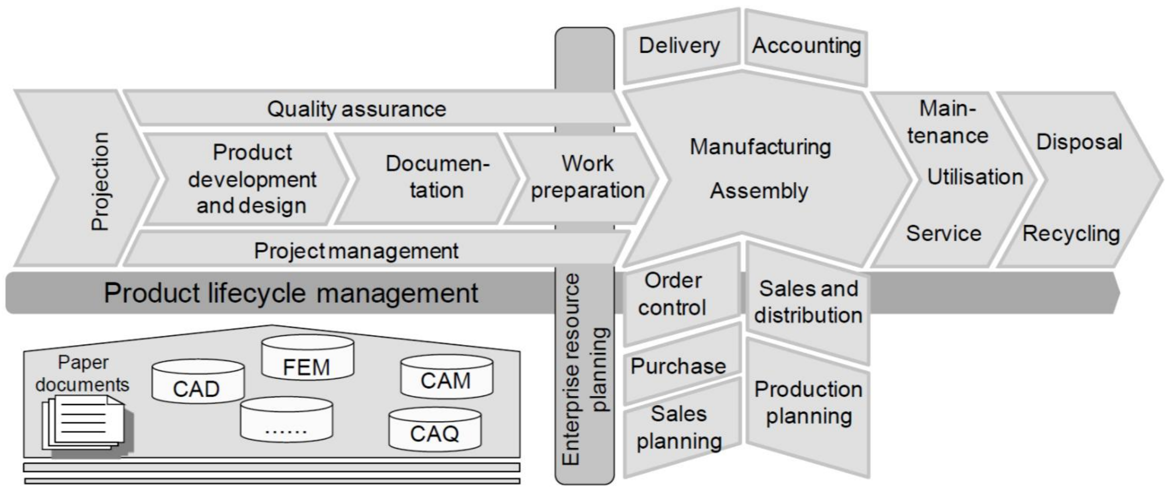


Figure 2-5 Concept of product lifecycle management according to ARNOLD ET AL.(2005, p. 14)

2.1.4 Complexity and Complexity Management

As already mentioned, there are some factors, which are limiting product design. Among them is the complexity of products, which makes product design often difficult to handle without the support of special techniques, e.g. computers (LINDEMANN & KIEWERT 2005, p. 415). In order to understand what complexity is and how it can be handled, the definition of complexity of technical systems is regarded and an introduction to management of complexity is given.

In general, the definition of complexity in literature is not consistent. Every definition focuses on certain aspects according to their discipline (LINDEMANN ET AL. 2009, pp. 25ff). Complexity of technical systems is depending upon the quantity of system elements and their relations, the intransparency, the number of variables and their dynamic behaviors (LINDEMANN ET AL. 2009, pp. 25ff; LINDEMANN 2005, p. 287; EHRLENSPIEL ET AL. 2003, pp. 54f). Consequently, the complexity of technical systems can generally be defined as the property of a system to have many different states and behaviors (SCHWANINGER 2005, p. 31; LINDEMANN ET AL. 2009, pp. 25ff).

In the case of product design, complexity can be generated by different sources. Especially the number of disciplines involved and the distribution of work influences its degree (GAUSEMEIER & REDENIUS 2005, pp. 554f). As can be seen in Figure 2-6 complexity in product design exists within the product, the process, the organization and the market. These fields can be divided into internal and external sources, but just internal complexity can be actively managed (LINDEMANN ET AL. 2009, pp. 25ff).

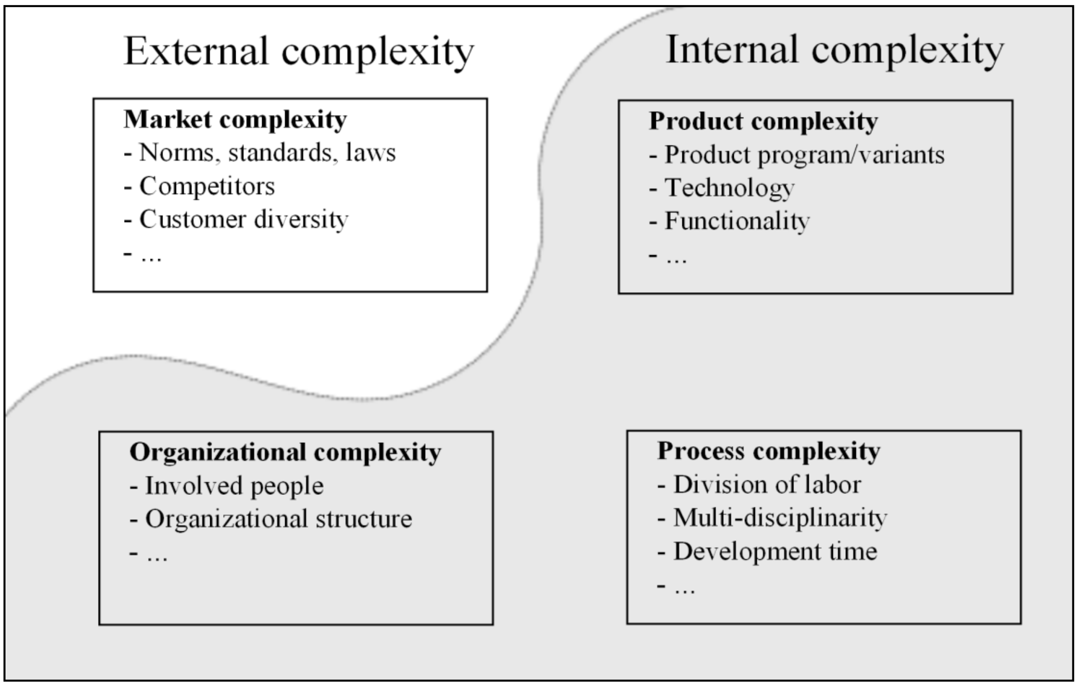


Figure 2-6 Four fields of complexity in product design and associated sources (LINDEMANN ET AL. 2009, p. 27)

A high degree of complexity can be observed in almost every relevant phase of the product lifecycle. This can be observed especially in products, which merge multiple disciplines like mechatronic and individually adapted ones (Gausemeier & Redenius 2005, pp. 554f; BULLINGER ET AL. 2003, p. 18). Because of grown complexity in products and processes, special adapted strategies are needed. These strategies are referred to as complexity management. (CLARK & FUJIMOTO 2005, pp. 129ff; LINDEMANN ET AL. 2009, pp. 31ff). Hence, complexity management describes a method to control, guide and develop a company, its products and the environment. It allows achieving a requested state under complex circumstances and the consideration of typical problems in handling complexity (SCHWANINGER 2005, p. 31). (LINDEMANN ET AL. 2009, pp. 31ff) specifies two fundamental strategies to handle complexity shown in Figure 2-7. Thus, the basic strategy is the acquisition and evaluation of complex systems, which is required for both successive strategies in order to address complex problems.

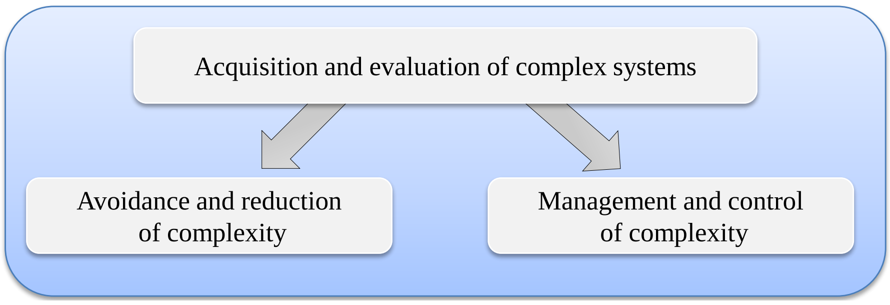

Acquisition and evaluation of complex systems
Avoidance and reduction of complexity
Management and control of complexity

Figure 2-7 Complexity management strategies according to Lindemann et al (2009, pp. 31ff)

The first fundamental step of successful complexity management is generating a meaningful model of the complex system. Most importantly, one has to identify all objects as well as all existing side effects instead of an extract of the system (DÖRNER 2010, pp. 288ff). These items have to be evaluated and systematically sorted to identify structural characteristics. Several methods of representation exist, which have to be chosen according to the requested management goal, but especially matrix based approaches like the multiple-domain matrix or graph-based approaches are commonly applied in this field (LINDEMANN ET AL. 2009, pp. 33ff).

Based on the generated model the strategy of avoiding and reducing complexity can be performed. In general, it is an effective and easy strategy to handle complexity but has to be applied reasonably. By identifying and eliminating unnecessary elements and relations while keeping the existing system's functionality, the complexity of each system can be reduced (LINDEMANN ET AL. 2009, p. 34). One disadvantage of this method is that often the elements or relations, which are the main reason for the complexity of the system, are directly associated to important attributes for the client. In the worst case the reduction of complexity can lead directly to a reduced competitive capability and is consequently in these cases not applicable (LINDEMANN & MAURER 2007)

Another strategy is to control and manage the complexity. It can be described as the ability to handle a system at an optimal level of complexity. Thus, the advantages of the system can be used while its side effects can be controlled without endangering the requested goals (LINDEMANN ET AL. 2009, p. 34; LINDEMANN & MAURER 2007; PUHL 1999). Therefore, the possibility to manage and control complexity offers a high, often unused, entrepreneurial potential. It enables several opportunities, like highly individual products, increased protection from product plagiarism and improved prediction of the impact of change (PINE 2008, pp. 31ff; WILDEMANN ET AL. 2007, p. 50; CLARKSON ET AL. 2004).

In order to achieve objectives and actively develop complex products, several methods have been developed (LINDEMANN ET AL. 2009, pp. 39–42). Among these methods especially the visualization of information, the multiple-domain matrix and the computational approaches are within the focus of this thesis.

2.2 Approach for the Analysis of Goal Interrelations

Main task of this work is to present a concept for a computer-supported implementation of the approach provided by HEPPERLE ET AL. (2011a) and FÖRG (2010). The approach was developed to anticipate and systematically analyze demands and corresponding product goals as well as trends concerning future solutions by linking product goals among each other via possible solutions and their characterizing parameters.

This chapter gives an introduction to the approach. This is done by presenting firstly the aim upon which the approach is based, the elements considered within, and the context analysis needed to identify them. Based on these, the graph-based approach for identifying and analyzing goal interrelation is described. Finally, it is shown how the approach is enhanced to reflect lifecycle-oriented product planning.

2.2.1 Aims Concerning the Approach

HEPPERLE ET AL. (2011a) and FÖRG (2010) are presenting an approach, which supports the early phases of the innovation process by providing a life-cycle oriented approach to identify and analyze goal interrelations among future demands and potential. To assure covering the right aspects he bases his approach on a systematic clarification of requirements. These are gathered by reflecting the weaknesses and strengths of related work as well as planning-specific requirements for developing new approaches as can be seen in Figure 2-8.

- Focusing on information relevant and manageable for planning phases
- Compatibility to existing planning procedures and applicability to various branches, products, levels of innovation (radical and incremental), etc.
- Pursuing a modular approach to allow the partial use according to the respective planning task
- Compatibility to existing organisational structures within an enterprise (both considering upstream (e.g. information acquisition) and downstream (e.g. product development) processes))
- Allowing consideration of process and product related potentials and demands arising along different lifecycle phases
- Allowing traceability of causal chains among abstract context factors to concretised solution opportunities
- Increasing transparency in causal networks of product potentials and demands by integrating structural complexity management approaches
- Consideration of both Market-Pull and Technology-Push approaches to detect and set trends
- Providing a framework to deduce consistent product concepts by increasing transparency concerning product goal interrelations
- Enhancing innovativeness by solution neutral goal formulation as well as openness for functionally equivalent and better solutions independent from known existing solutions

Figure 2-8 Aims of the approach to handle goal interrelations (HEPPERLE ET AL. 2011a; FÖRG 2010)

These requirements are setting the framework for the graph-based method to represent goal interrelations. Thus, it considers product potentials and corresponding product concepts as well as demands arising from the company, market and environmental context. The corresponding approach is described in the next chapters.

2.2.2 Considered Elements within the Approach

This section gives an overview on the different elements and their coactions being relevant in lifecycle oriented planning with the aim of identifying goal interrelations according to HEPPERLE ET AL. (2011a). In Figure 2-9 a graphical visualization of the framework can be seen.

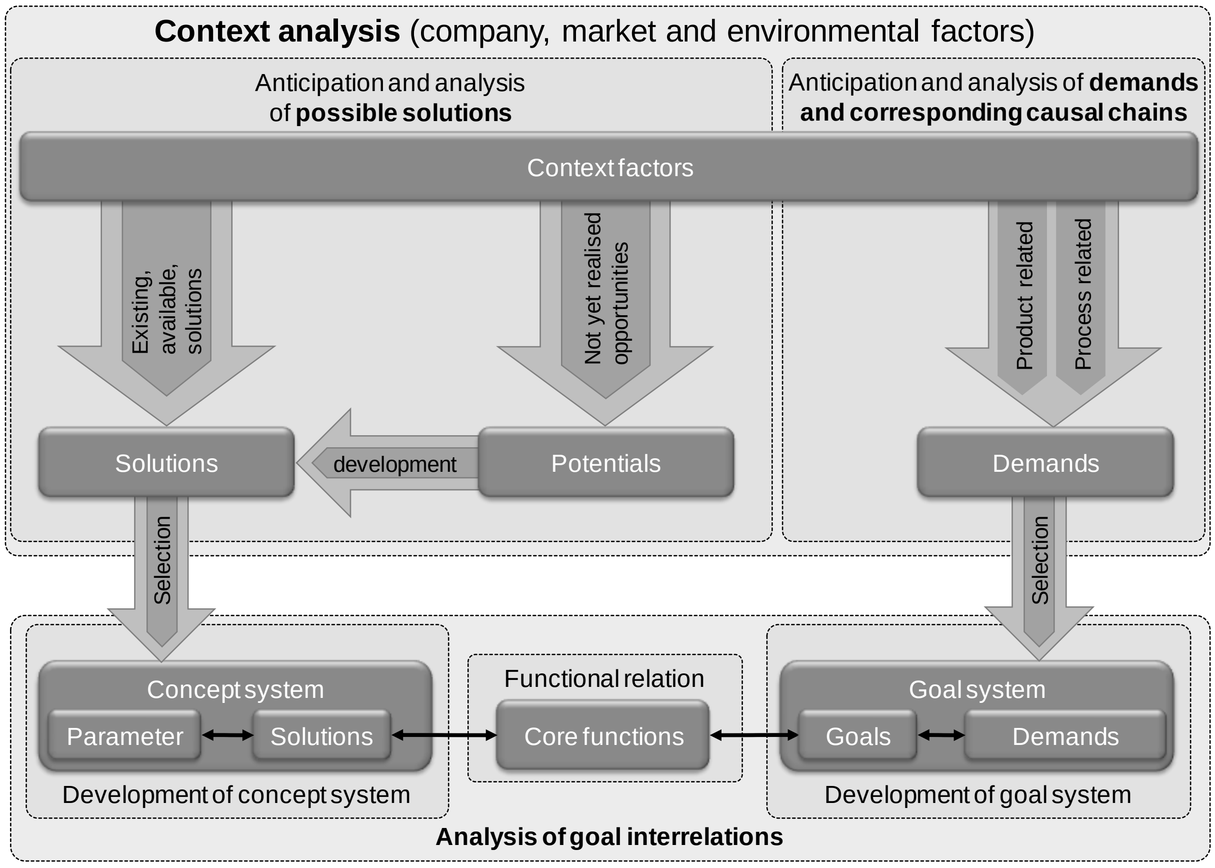

Context analysis (company, market and environmental factors)
Anticipation and analysis of **possible solutions**
Anticipation and analysis of **demands and corresponding causal chains**
Context factors
Existing, available, solutions
Not yet realised opportunities
Product related
Process related
Solutions
development
Potentials
Demands
Selection
Selection
Concept system
Parameter
Solutions
Functional relation
Core functions
Goal system
Goals
Demands
Development of concept system
Development of goal system
Analysis of goal interrelations

Figure 2-9 Framework for analysis of goal interrelations according to HEPPERLE ET AL. *(2011a)*

The superordinate task of identifying future potentials and demands is illustrated in the upper box. The context factors can be analyzed with the model provided by LANGER & LINDEMANN (2009b), described in the following section. Thus, important issues from the environmental, market and company context can be detected. These, shown on the right side of the figure, can lead to opportunities and demands, referred to as a market pull. On the left side, where future solutions are anticipated and analyzed, the opportunities are referred to as technology push. These solutions can be subdivided into two types: on the one hand, already existing and available solutions that can be used in future solutions (e.g. products transferable from other branches). On the other potential solutions being not yet implemented into products (e.g. findings in research). These potentials are needed to be developed and projected to product solutions in order to be later considered within the analysis of goal interrelations (HEPPERLE ET AL. 2011a). As a next step, a first selection of solutions and demands should be carried out in order to allow manageable analysis of goal interrelations and to focus on the task of planning products. This step should be performed systematically for example by considering the background of the company strategy and economic aspects (HEPPERLE ET AL. 2011a).

The confrontation of the collected demands on the right side and solutions on the left side can be seen in the lower box of Figure 2-9. In the figure, the alignment of the elements is based on the Means-End-Chains, which is later described in chapter 2.3.1.2. At the goal system (right side), the chain starts with demands (e.g. deriving from external sources like legal restrictions) and connects them to product goals (e.g. satisfying the legal restriction). On the left side, the concept system is developed based on the preselected and anticipated solutions. Furthermore, the solutions are concretized by planning relevant parameters. Both sides, the goal system and the concept system, are connected through a functional relation, described by the core function. Core functions are directly related to either features of the planned product or tasks within the product planning process and are serving to organize the various chains. Thus, a complete chain from the demand via goals and core functions to solutions with their parameters can be provided. Within the approach, the information acquisition of goal and concept system can be carried out concurrent, since both sides are first independent and linked afterwards by core functions. Another advantage of the framework is the flexible support of Technology-Push and Market-Pull approaches. This means that product concepts can be deduced starting from both sides, the concept system (Technology-Push) as well as the goal system (Market-Pull) (HEPPERLE ET AL. 2011a). However, for the identification of the necessary factors a method for the analysis of the context in needed, and therefore presented in the following section.

2.2.3 Context Analysis

New products are influenced by many different factors from within or outside of the company. One task within product development is to identify all possible sources of influences and consider their specific requirements. Also in the case of a GIA, it is important to identify all elements of the product planning system. Not identifying all influences can, in the worst case, lead to a complete product failure, e.g. not considering a certain law. Especially external factors are often difficult to identify. Therefore, it is necessary to perform a systematic analysis of the integrated context of a company, instead of an intuitive identification of requirements. This can be done by using a model for the classification of context factors (LANGER & LINDEMANN 2009a).

A framework for the systematical analysis of context factors is provided by Langer and Lindemann (2009a) and can be seen in Figure 2-10 Model for classification of context factors. First, it distinguishes between elements from the development process, the company, the company interfaces, the market and the environment. With the market being considered as an element of the overall environment, the company interfaces are part of the market, and therefore the connection between company and market. Within the company, the overall development process takes place, thus laying emphasis on this model. Additionally the differentiation between purchases and sales is introduced in the model to provide new perspectives for the allocation of factors. The application context is bundling fields of context, which are possibly affecting the application, and usage of future products (like e.g. legislation, energy). These elements are crossed with the elements from technology/knowledge, socio-economics, and politics/legislation to identify all influencing factors. Further, the four classes can be subdivided by twelve sub-elements leading to a complete framework for the classification of context factors (HEPPERLE ET AL. 2011a; LANGER & LINDEMANN 2009a).

Initially this framework was established for the developments of products. However, it can also be applied in earlier planning tasks, like the GIA, because it describes factors being in general interest for innovating products. Thus, it is an important part of the integrated identification of relevant demands and potentials to point out what results need to be further processed to identify interrelations between product demands and potentials (HEPPERLE ET AL. 2011a). In the next section it is shown how, based on the framework, how the described elements are used within a graph-based approach to analyze goal interrelations.

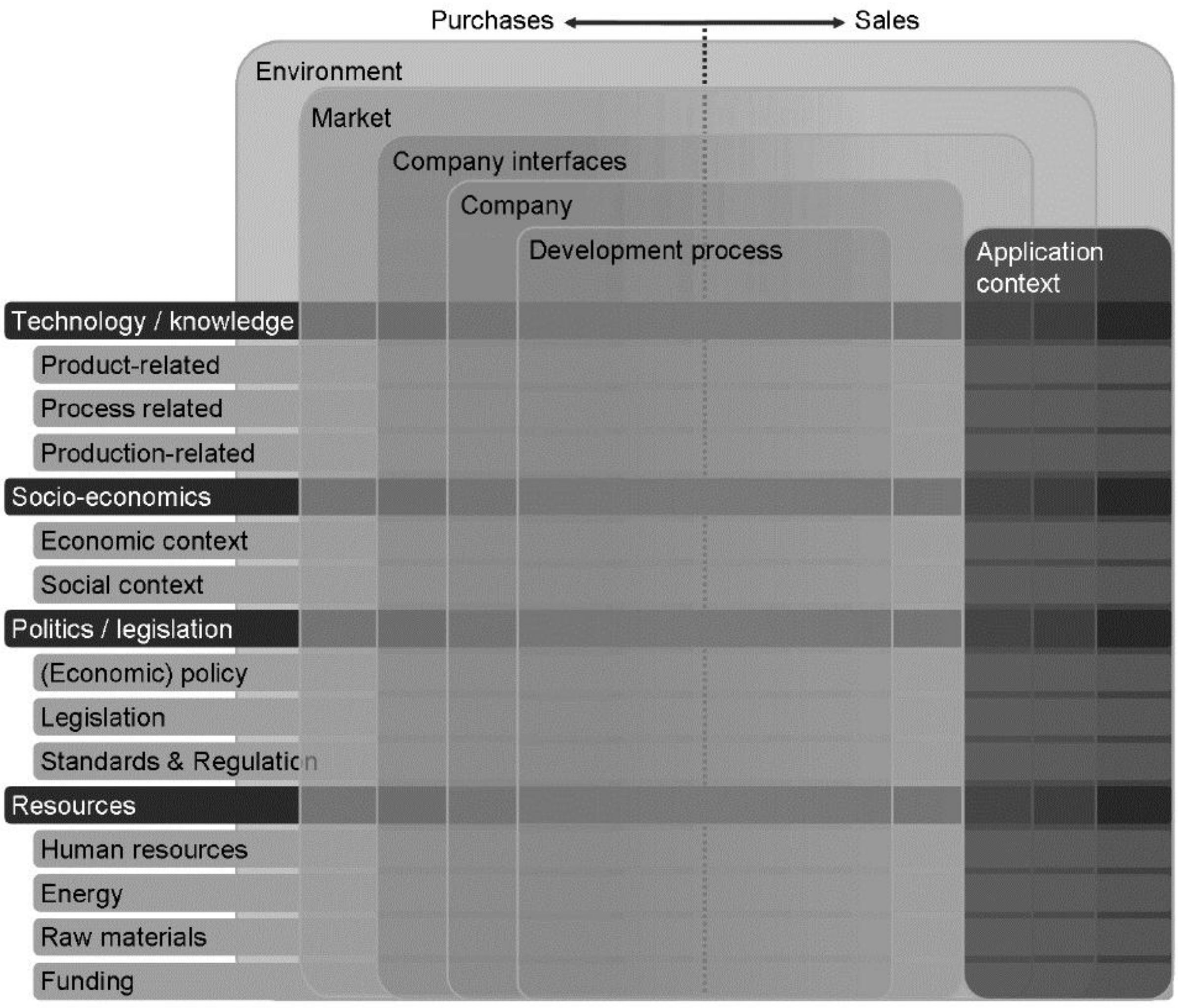


Figure 2-10 Model for classification of context factors (LANGER & LINDEMANN 2009a)

2.2.4 Graph-Based Approach for Identifying Goal Interrelations

The implementation of the described framework by a graph representation is based on a definition by (HEPPERLE ET AL. 2011a) describing which kind of goal interrelations exist and why the two goals interrelate:

- *"First, goal interrelations can occur, because two goals follow the same characteristic, but with different values (e.g. customer 1 prefers speed > 150 km/h,* *customer 2 prefers speed < 120 km/h). This goal interrelation and corresponding conflict can already be identified by comparing and handled by prioritising the different available values."*

- *"Another type of goal interrelation can occur if two product goals are linked by the possible solution(s) to achieve the goals. E.g. customer 1 demands goal 1 (energy consumption when operating the car < 4 litres/100 km) and at the same time demands goal 2 (acceleration from 0 to 100 km/h in less than 6s). There is no direct goal conflict between these two goals. Still, if taking solution A (specific combustion engine) to achieve goal 1, goal 2 cannot be achieved and vice versa."*

The approach and its graph representation limit themselves to the second type of goal interrelation. This means that achieving the considered product goal requires a consistent set of possible and compatible goals. Vice versa, if it is not possible to detect a consistent set of solution, goals have to be prioritized or further possible solutions have to be found and new iterations of the goal interrelation analysis have to be carried out.

The implemented graph representation of the framework for a goal interrelation analysis is presented in Figure 2-11 as a simplified, abstract example by HEPPERLE ET AL. (2011a). The example shows the elements that were already presented in chapter 2.2.2. Thereby on the right side, a demand is connected to a goal, whereas on the left side, a potential is linked to a solution, which has several connections to parameters. Finally, both sides are connected through the core function. Since the example is simplified, just connections between two single elements are considered and no multiple ones. However, in practical examples this representation technique can also express multiple connections. For example, several solutions can be linked to one core function or several demands can be linked to one goal etc.

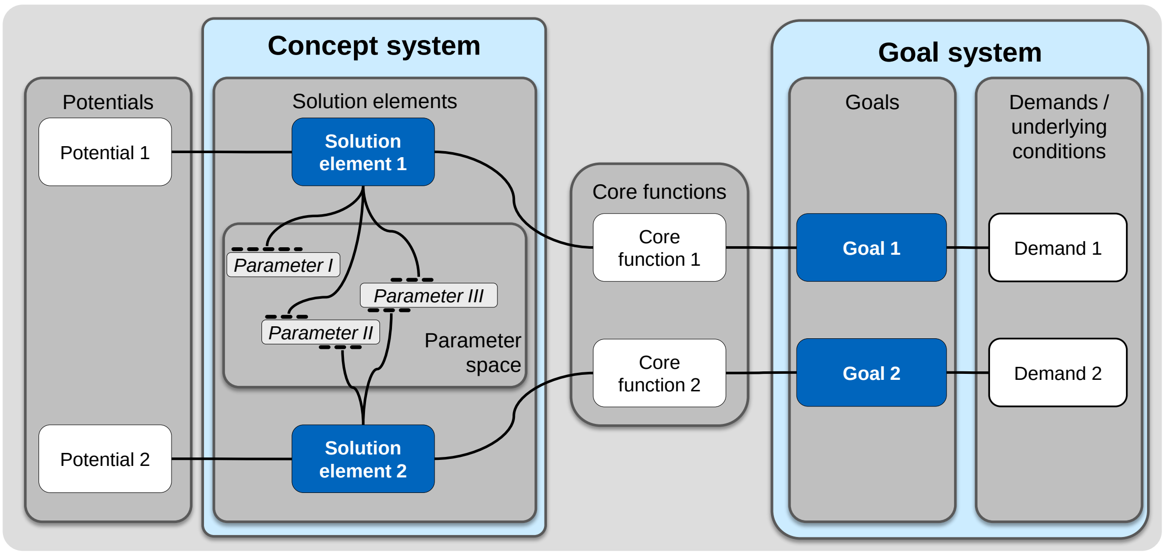


Figure 2-11 Simplified, abstract example of graph-based representation of goal interrelations (HEPPERLE ET AL. 2011a)

As can be seen in Figure 2-11, within the graph representation, the ‘parameters’ are the characterizing elements for the possible solutions. As multiple possible solutions can be characterized by the same parameter, these solutions are therefore indirectly linked through it (see ‘Parameter 2’ and ‘Parameter 3’ in Figure 2-11) (HEPPERLE ET AL. 2011a). Following the connected chains of elements, from the solution to the goal, an interrelation between ‘Goal 1’ and ‘Goal 2’, in Figure 2-11, can be detected. On these connection-chains, the goal interrelation analysis, with the aim to identify goal conflicts, is based.

Hepperle et al. (2011a) and FÖRG (2010) are further presenting an approach for identifying goal conflicts by showing alternatives of linking product parameters. In the abstracted example Figure 2-11 a goal conflict can be identified through ‘Parameter 2’, where two connected solutions are pointing at different ranges of the parameter. In contrast, the two connected solutions of ‘Parameter 3’ have overlapping ranges and therefore no goal conflict can be identified. Several types of connection between solution and parameter exist, which can be seen in the overview of alternatives for linking product parameters in Figure 2-12. It shows how the scale for characterizing a parameter can differ: For very precise information about the possible solution, the quantitative scale can be considered. Nevertheless often, the information about the anticipated future solutions is very vague and qualitative; hence, for the product planning the qualitative scale or the even less detailed binary scale is of particular interest. Additionally it can be distinguished between the type of parameter interrelation, which means if the solution is connected to only one discrete value or a range of parameters. With having several different possibilities for describing connection between solutions and a parameter, which can occur also at the same time in one system, it is important that solutions sharing one parameter should be described similarly in order to allow detecting the interrelation and the possible conflict between them.

Moreover, the approach also allows linking several parameters with each other. Since in practice often the influence of a parameter implies the direct influence of another parameter, it is possible to link parameters directly. Often the connection is based on a physical law or equation. Consequently, interrelations between two solutions can also be found via two or more parameters (HEPPERLE ET AL. 2011a). For example if a car is heavier, the kinetic energy to accelerate it raises proportional, based upon the law for kinetic energy $E = \frac{1}{2}mv^2$.

The approach and the graph presentation according to HEPPERLE ET AL. (2011a) and FÖRG (2010) allows identifying goal interrelations via possible solutions within a future product concept. Connected to it, also goal conflicts can be detected by analyzing the values of the parameter. Besides, the approach is directed on an integrated lifecycle perspective, why its usage against the background considering manifold lifecycle phases is presented in the next section.

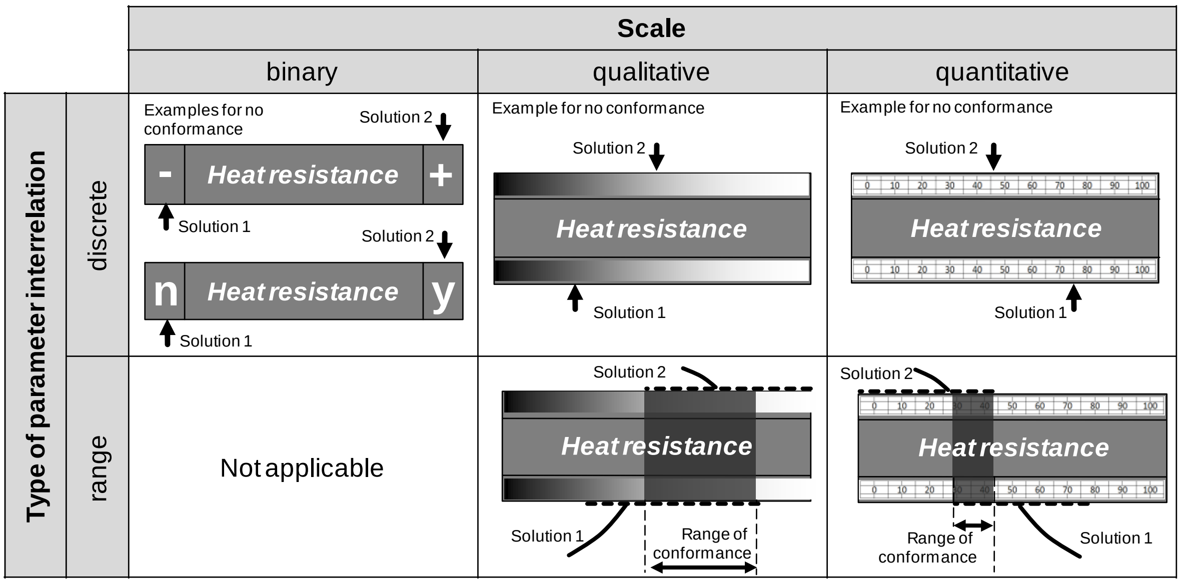


Figure 2-12 Alternatives of linking product parameters (e.g. heat resistance) (HEPPERLE ET AL. 2011a)

2.2.5 Lifecycle-Oriented Goal Interrelations

Planning new products through all lifecycle phases is an advantage for companies to differ from competitors, since the company's strengths and weaknesses may be assigned to other lifecycle phases. With the background of presenting an approach for the integrated lifecycle perspective of product planning, HEPPERLE ET AL. (2011a) and FÖRG (2010) are introducing, how the approach is extended to consider lifecycle phases. To start with, this is done by applying the approach to the various lifecycle phases at the same time. Additionally, it is suggested to develop the framework for the every lifecycle phases in an extra box, so transparency can be maintained at a high level (See Figure 2-13). Therefore, a new product can be described by all related parameters derived from the various lifecycle phases, which can also be linked to each other.

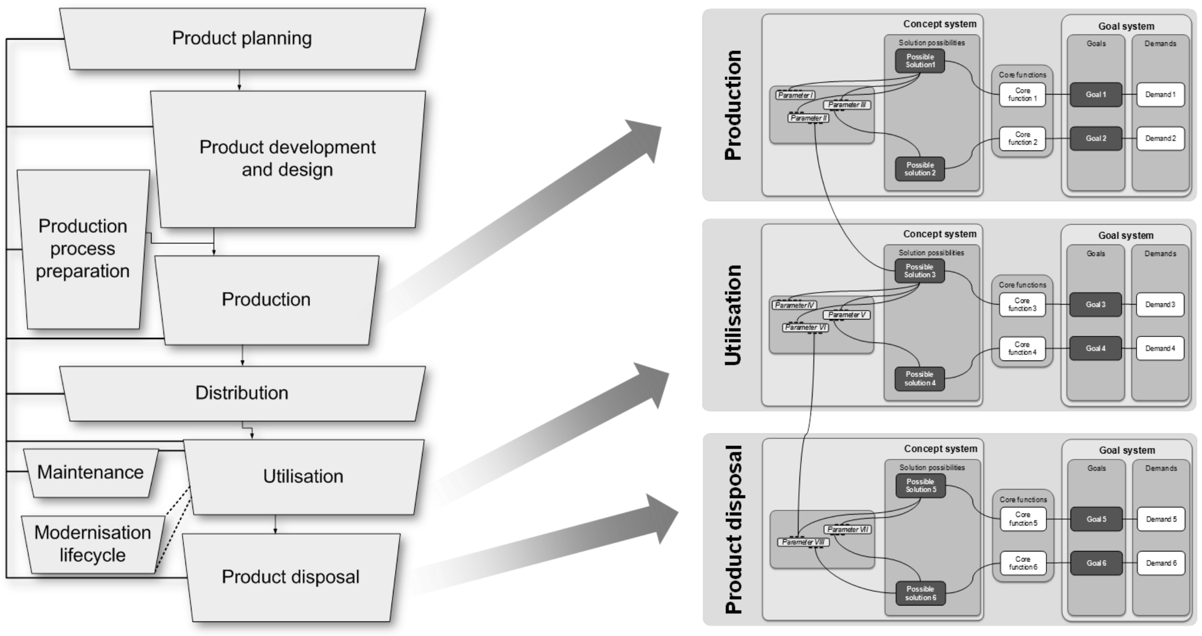


Figure 2-13 Graph-Based goal interrelation analysis with consideration of product lifecycle

2.3 Related Approaches

In order to receive a deeper understanding of this thesis's content, the related approaches need to be regarded. Giving an introduction into these, an overview on the fundamental principles, the applied methods, the usage of models, as well as the computer-supported modeling techniques is presented.

2.3.1 Fundamental Principles in Product Design

In this section, some basic principles, being often applied in systematical product design and related to the GIA, are presented. These principles are general procedures of how to handle certain circumstances, leading to improved results. Consequently, the general principles to visualize systems as well as mapping information are introduced, which both are also applied in the approach to analyze goal interrelations.

2.3.1.1 System Visualization

Making system data visually available is a fundamental feature of many research disciplines and an important principle of systematical management. The general objective of representing information visually is to give users a global overview and a better system understanding by focusing views on specific aspects (LINDEMANN ET AL. 2009, pp. 39f.). Furthermore, the use of graphics and diagrams helps to overcome communication issues and saves time for transferring information (DOW & TAYLOR 2009, p. 179). There are several methods for visualizing systems which are each specialized on their objectivities, but in product design graphs and matrices are the most common (LINDEMANN ET AL. 2009, pp. 39f.; WARE 2004, pp. 210ff).

The application of graph theory for representing system is a basic property of many methods in product design, where the focus lies on elements and their interdependencies. In general, a graph describes a start node and an end node, which are connected through an edge. Figure 2-14 shows a simple example for graph representation, where several components of a product are connected to parameters by lines. Having a component as a start node, a parameter as an end node, and a line as an edge, it is a typical graph representation, which therefore can help to identify components and depending parameter. Other examples, which can be traced back to the usage of graph theory are the critical path method or the project scheduling as well as numerous algorithmic problems (LINDEMANN ET AL. 2009, pp. 47.; GROSS & YELLEN 2006, pp. 493ff). Consequently, graph theory provides the basis to characterize system constellations like identifying implied substructures or structural attributes (LINDEMANN ET AL. 2009, pp. 127ff; GROSS & YELLEN 2006).

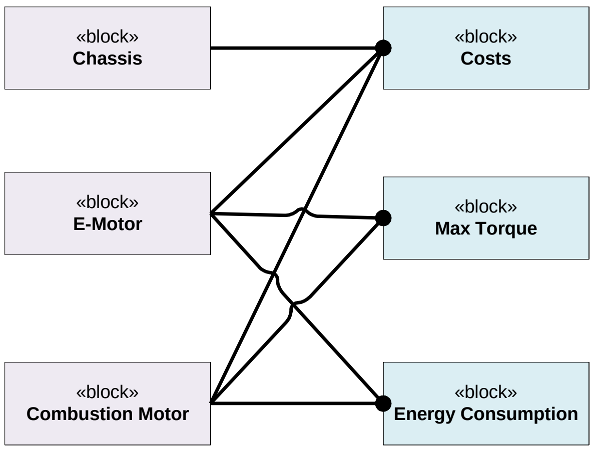


Figure 2-14 Assignment of components to parameters with graphs

Matrix-based approaches can be seen as an alternative to graph-based approaches, since in general it is possible to transform graphs to matrices. However, it has to be considered, that every representation has own advantages for certain problems including the possibility to lose some explicit information during the transformation (BROY 1998, pp. 167ff; LINDEMANN ET AL. 2009, pp. 39f.). There is a large variety of matrix-based approaches in product design, which can be classified by the quantity of elements involved and executed computations. Figure 2-15 depicts the four types of general matrix systems. At first, the intra-domain matrices (depiction a) in Figure 2-15) are matrices which represent relations within one domain, e.g. *Design Structure Matrices* (DSM). Secondly, the inter-domain matrices (depiction b) in Figure 2-15) are considering the relations between elements belonging to two different domains, e.g. *Design Mapping Matrix* (DMM). Combined intra- and inter-domain matrices (depiction d) in Figure 2-15) are using the advantages of both systems, as it is used in the House of Quality introduced later in chapter 2.3.2. Finally the *Multiple-Domain Matrix* (MDM) uses the combination of inter and intra domain matrices plus the computation of information stored in other subsets and will be introduced in the following section.

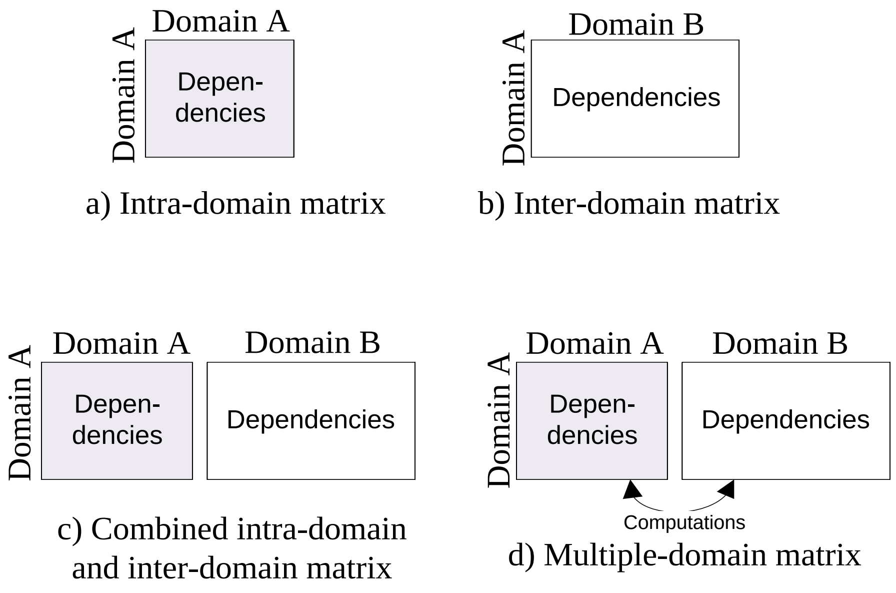


Figure 2-15 Classification of matrix-based methods (LINDEMANN ET AL. 2009, p. 50)

2.3.1.2 Mapping Information

Focus of this research project is the analysis of goal interrelations. An essential basis of it is the linkage of information from various domains. Since the confrontation of information from various domains is required for most of cross-domain analysis disciplines, multiple approaches exist (LINDEMANN & MAURER 2007). Among them, this section regards the MDM as well as the Means-End-Chain.

Firstly, the MDM will be presented, whose origins can be seen in the complexity management (see chapter 2.1.4). The MDM is a graph- and matrix-based approach to analyze the interrelations of different domains (e.g. functions, processes, components, etc.) and to detect structure in the system in order to manage its complexity (LINDEMANN & MAURER 2007; LINDEMANN ET AL. 2009, pp. 39ff). It combines the functionality of a *Design Structure Matrix* (DSM) and a *Domain Mapping Matrix* (DMM) (see Figure 2-16) and enhances it with computation possibilities. The DSM and the DMM are methods supporting systems engineering by confronting system components from one or two domains in a matrix form (FRIEDRICH 2010, pp. 3ff; ZWERENZ 2009, p. 30; LINDEMANN ET AL. 2009, pp. 39ff). The MDM is providing a comprehensive and visual representation of a system, which is applicable on various design and development challenges. It offers an excellent basis for analyzing structures and structural patterns that numerous have researched in the recent years (BROWNING 2001; HEPPERLE ET AL. 2011a).

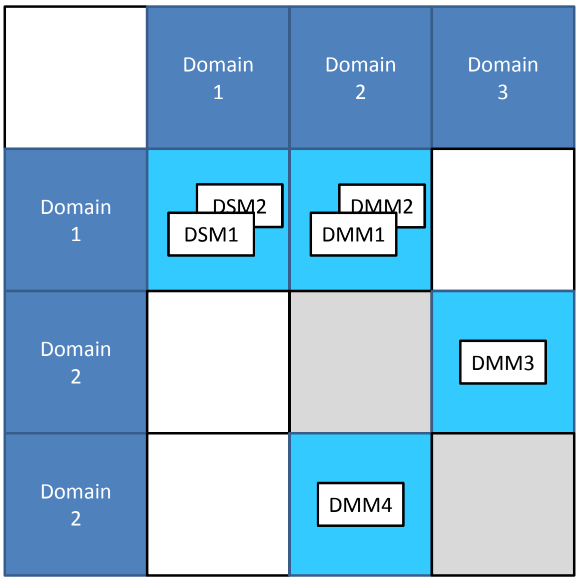


Figure 2-16 Multiple-Domain Matrix according to L*INDEMANN ET AL*. *(2009, pp. 69ff)*

Another approach for linking information is a Means-End-Chain, which is derived from psychological and marketing research (HERRMANN & HUBER 2009, p. 179). Means-End-Chains are used to analyze the connection between certain product attributes and customer's personal values (GRÖNING 2010, p. 4). The general Means-End-Chain links certain product attributes over benefits to personal values as can be seen in Figure 2-17 (KUSS 2000, p. 67; PARRY 2005, p. 4). However, the specific marketing algorithm is not directly relevant for goal interrelation analysis and strategic product planning. The interesting detail of the approach is the general idea of addressing customers' demands on different abstraction levels, which can be regarded as a basic method for structuring future demands and potentials (HEPPERLE ET AL. 2011a).

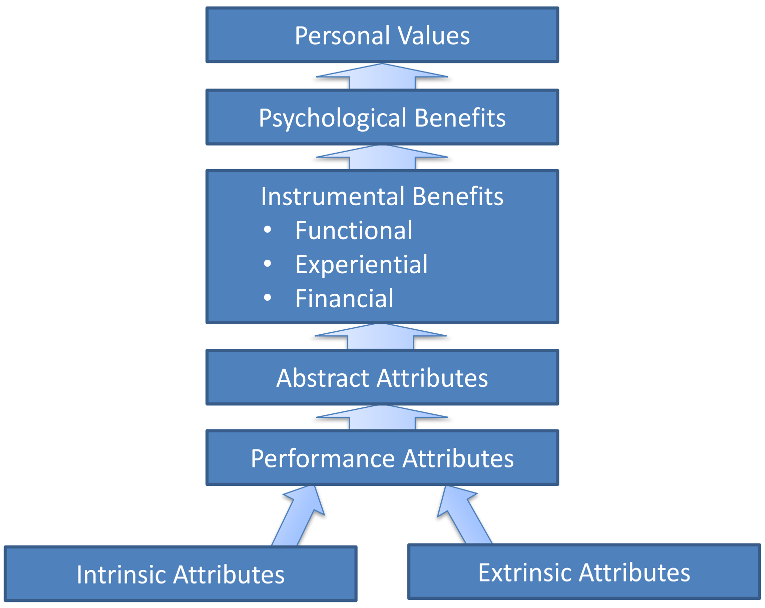


Figure 2-17 A general Means-End Chain model according to P*ARRY 2005, p. 4)*

2.3.2 Applied Methods in Product Design

This section introduces into several applied methods from the field of product design, which is relevant in the context of this thesis. Thus, an introduction into the *Quality Function Deployment* (QFD) is given, which provides a confrontation of solution and demands in production, similar to the one in the GIA. Furthermore, an approach for the consideration of temporal aspects within product design as well the scenario-technique for the consideration of several trends or scenario is shown.

An approach which connects customers' requirements and technical product specifications and which is widely accepted in industrial practice is QFD (AKAO 1990, pp. 3ff; HEPPERLE ET AL. 2011a). Integrating customers' demands into the product design is the main focus of QFD. This is done by building the *House of Quality* (HoQ) which is a matrix based diagram combining several charts (Figure 2-18). Within the HoQ, customers' requirements are compared to other products and to the companies' engineering knowledge. Further interdependencies between engineered attributes and their attachments to product goals are listed (JAINTA 2009, p. 4; WEBBER & WALLACE 2008). Summarizing all necessary information, the HoQ can be seen as a blueprint for product design (MADU 2006, p. 19)

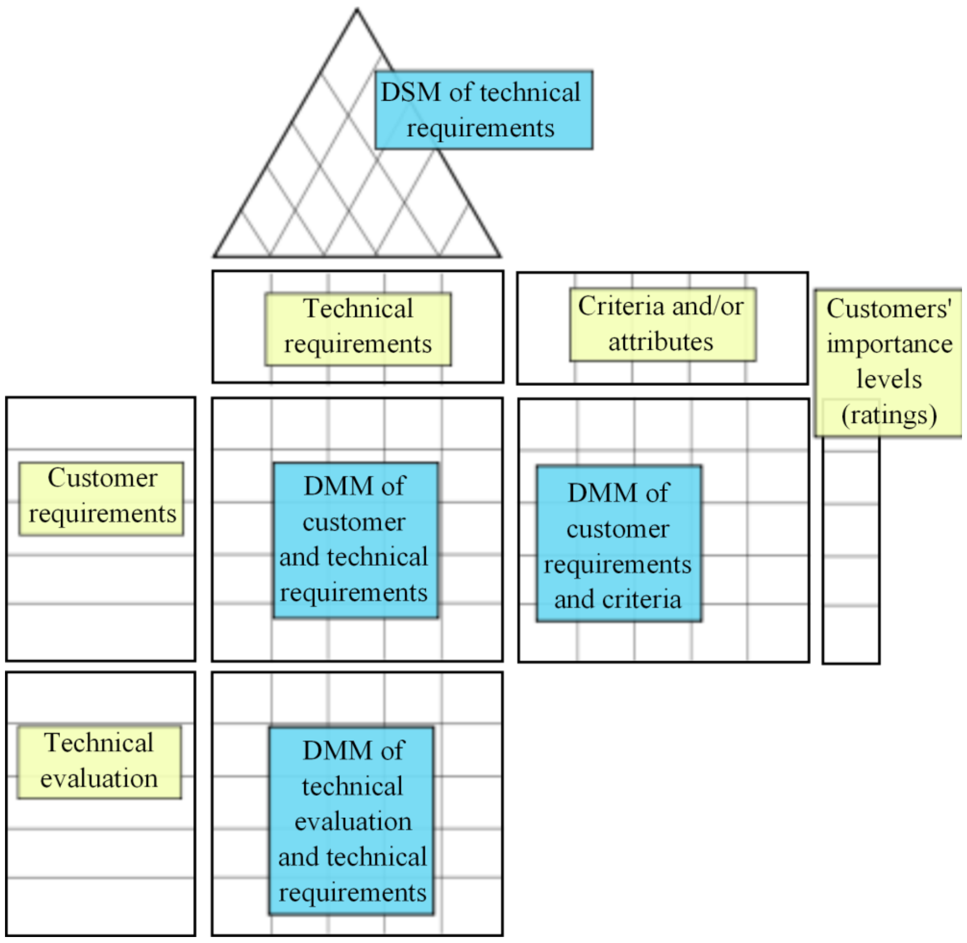

DSM of technical requirements
Technical requirements
Criteria and/or attributes
Customers' importance levels (ratings)
Customer requirements
DMM of customer and technical requirements
DMM of customer requirements and criteria
Technical evaluation
DMM of technical evaluation and technical requirements

Figure 2-18 Main components of the House of Quality according to WEBBER & WALLACE *(2008, p. 348)*

Anticipating the whole lifecycle of future products allows fast responses to influences from the environment. Thus, it is an essential task to consider temporal aspects in the product planning process. With this knowledge, it is possible to adapt the portfolio of offered products to the latest condition concerning content and time (UGHANWA & BAKER 1989, pp. 212f; HEPPERLE ET AL. 2011b). As an introduction into this topic, the temporal aspects in lifecycle oriented planning of product-service systems as well as the scenario technique are presented.

HEPPERLE ET AL. (2011b) identify several planning-relevant temporal aspects concerning singular lifecycle phases as well as lifecycle constellations of singular and multiple product generations. Therefore he provides a scheme based on a simplified product lifecycle, consisting of only four superordinate phases ‘Development’, ‘Production’, ‘Utilization’ and ‘Recycling/Disposal’ (see Figure 2-19). Besides multiple product generations, it also describes the industry-relevant topic of ‘facelifts’. This scheme supports the understanding of temporal aspects concerning lifecycles of one product generation and multiple product generations and therefore the systemically description of temporal patterns.

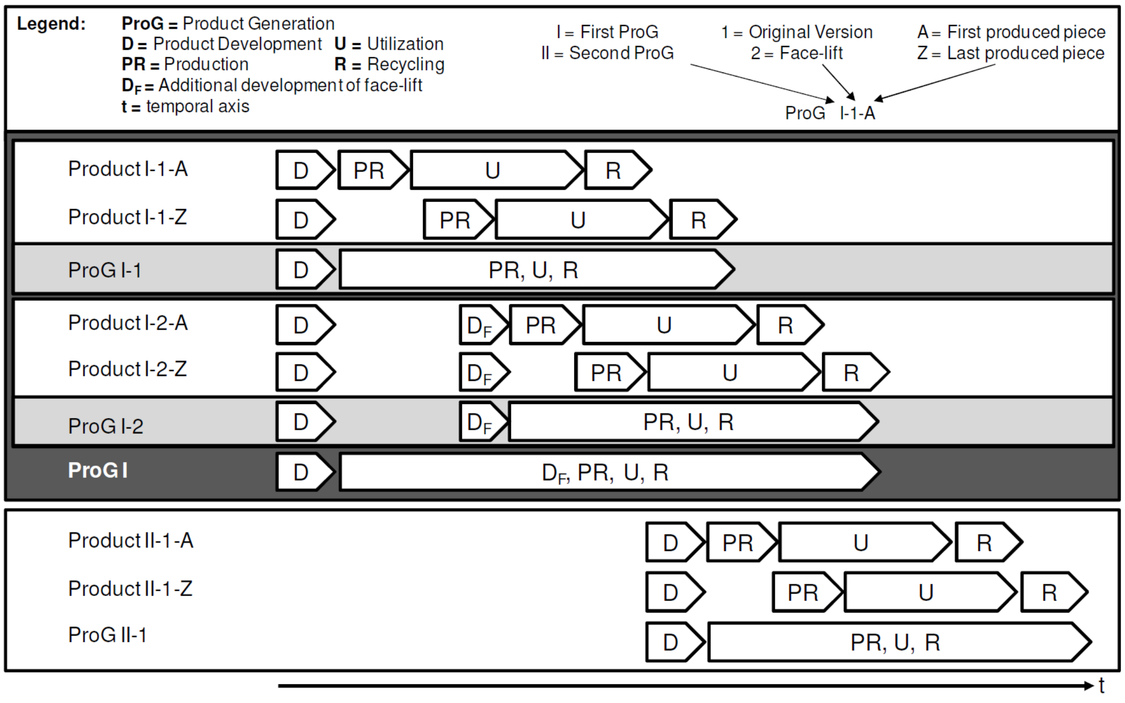


Figure 2-19 Scheme of multiple subsequently following lifecycles and product generations (HEPPERLE ET AL. 2011b)

Another method, which is relevant for this research project, is the scenario technique. The goal of the scenario technique is an early identification of the dynamic development-paths of a system in order to derive conclusions on appropriate actions (WILMS 2006, p. 68). The approach combines the principles of ‘multiple futures’ and ‘cross-linked thinking’ and is generally based upon the scenario as a thinking model for strategic management (GAUSEMEIER ET AL. 2009, pp. 60ff). The idea of a scenario is visualized graphically in Figure 2-20 with the scenario-cone. Starting point for a scenario is always the current time. Depending on several decisions and disturbance, the development arrives at another point in the future. With longer development time, future conditions get less certain, which results in more variance and possible developments. Consequently, the scenario-cone spreads with the time and contains multiple development lines for scenarios (MIßLER-BEHR 1993, p. 3; GAUSEMEIER ET AL. 2009, p. 62). Thus, it can be seen that for planning future products, several scenarios have to be considered, in order to anticipate necessary trends for new products.

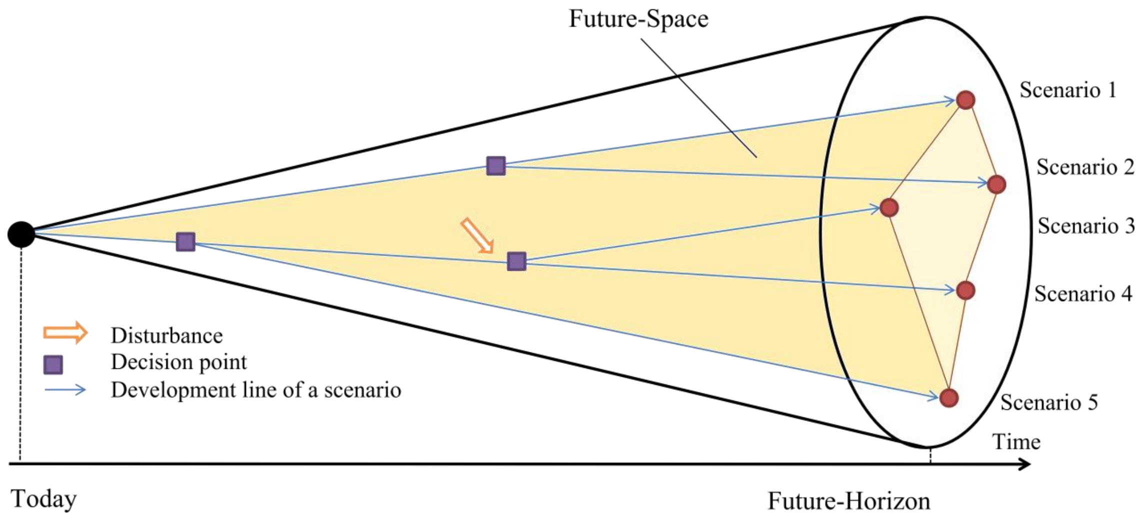


Figure 2-20 Scenario-Cone according to G*AUSEMEIER ET AL. (2009, p. 62)*

2.3.3 Usage of Models in Product Design

With the increased need of developing complex products for highly dynamic markets, working with simplified models of products is an essential technique in product development (SCHWANINGER 2005, p. 33). In these models, all information gathered during the development process may be collected and used as sources for further analysis through all phases of the product development (KOCH & MEERKAMM 2001, p. 218; GAUSEMEIER ET AL. 2005, pp. 607ff). As an introduction to this topic, a short definition of models is given, followed by the general usage of models in engineering disciplines. Finally the model-based systems engineering approach is presented, which is also applied within this research project.

In general, a model is a (simplified) image of a real or thought system, transferred into another system (BUNGARTZ 2009, p. 5; SCHILLER 2009, p. 23). Models are built according to the original, with changes to support the achievement of the intended aim (GAUSEMEIER & REDENIUS 2005, p. 562; KASTENS & KLEINE BÜNING 2005, p. 16). Since the quality of a result is directly connected to its underlying model, it needs to be chosen, how to model the system and which techniques are used to describe it. Consequently, some elements or properties that occur as unimportant can be neglected to reduce complexity and ease handling. In general, models can be characterized as material or immaterial while immaterial can further be verbal, mathematical, algorithmically or graphical (SCHILLER 2009, pp. 23, 50). Within product design especially CAx systems and systems engineering is based on the usage of models.

CAx systems are helping to make relevant decisions about a product as early as possible in the product design process. This is done by calculating and simulating several alternatives of the product as close to reality as possible. The expression CAx summarizes computer-supported systems in product development (VAJNA 2005, p. 433). An example of various tools and the phases they are commonly used in can be seen in Figure 2-21. As can be seen, the conceptual stage of product planning is hardly supported by specialized creating tools like CAD, but more general tools like the PLM or common office software.

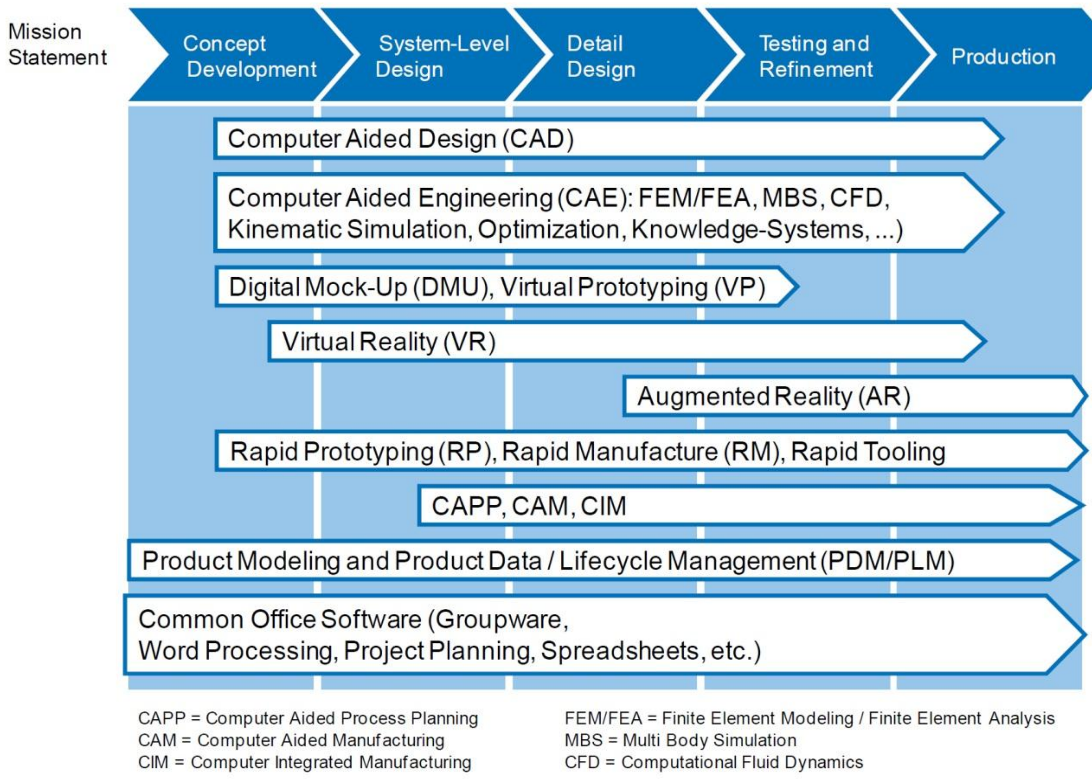

Mission Statement
Concept Development
System-Level Design
Detail Design
Testing and Refinement
Production
Product Launch
Computer Aided Design (CAD)
Computer Aided Engineering (CAE): FEM/FEA, MBS, CFD, Kinematic Simulation, Optimization, Knowledge-Systems, ...)
Digital Mock-Up (DMU), Virtual Prototyping (VP)
Virtual Reality (VR)
Augmented Reality (AR)
Rapid Prototyping (RP), Rapid Manufacture (RM), Rapid Tooling
CAPP, CAM, CIM
Product Modeling and Product Data / Lifecycle Management (PDM/PLM)
Common Office Software (Groupware, Word Processing, Project Planning, Spreadsheets, etc.)
CAPP = Computer Aided Process Planning
CAM = Computer Aided Manufacturing
CIM = Computer Integrated Manufacturing
FEM/FEA = Finite Element Modeling / Finite Element Analysis
MBS = Multi Body Simulation
CFD = Computational Fluid Dynamics

Figure 2-21 CAx tools in product development (SHEA 2010, p. 25)

Nevertheless, since product models used in CAx have evolved during several years, the trend towards integrated, knowledge-based models can be seen. Goal of this trend is to integrate all information needed during the whole product lifecycle into one model. Figure 2-22 shows this evolution. It started with conventional CAD models, which only use fixed geometric values, evolving to parametric ones, containing elements with variable constraints, leading to upcoming, knowledge-based CAD models (GAUSEMEIER ET AL. 2004, pp. 609f; VAJNA 2005, p. 437). Within these models, all relevant information is described by features. Through features, it is possible to model all properties of a product with considering all necessary circumstances. This is leading to an integrated product model, which allows managing the increased amount of collected and requested information (KOCH & MEERKAMM 2001, p. 218; VAJNA 2005, p. 437). Consequently, having a complete and consistent product model is one of the most important tasks in a modern product development environment and has to be supported also in the earliest phases of product planning processes.

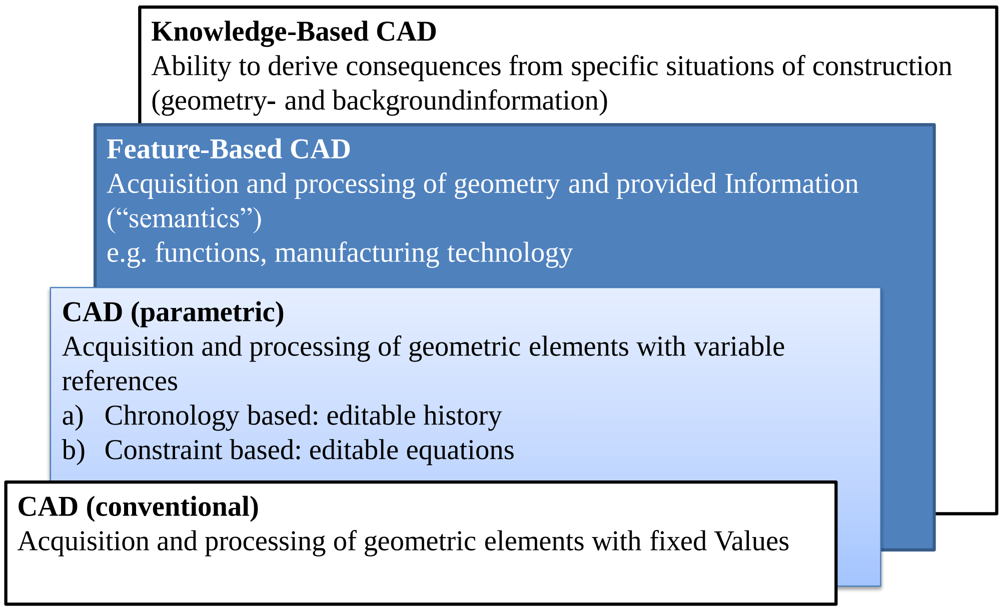

Knowledge-Based CAD
Ability to derive consequences from specific situations of construction (geometry- and backgroundinformation)
Feature-Based CAD
Acquisition and processing of geometry and provided Information ("semantics")
e.g. functions, manufacturing technology
CAD (parametric)
Acquisition and processing of geometric elements with variable references
a) Chronology based: editable history
b) Constraint based: editable equations
CAD (conventional)
Acquisition and processing of geometric elements with fixed Values

Figure 2-22 Development of CAD systems (VAJNA 2001, p. 8, 2005, p. 438)

The *Model-Based Systems Engineering* (MBSE) approach is a formalized application of modeling to support systems engineering (FRIEDENTHAL ET AL. 2008, p. 17). Systems engineering is an approach to use procedures and methods leading to an optimal development or redesign of any system (WALTER 2010, p. 4). Thereby it concentrates on the definition and documentation of system requirements, the design analysis, verification and validation activities from the conceptual design phase throughout development and all later lifecycle phases (WEILKIENS 2006, p. 11). The origins of MBSE can be seen in the CAD development while it was created to overcome limitations of document-based product planning and development. These are among others the inefficiencies in finding errors and stress points, testing both performance and timing behavior in one or more competing designs, and providing actionable information for trade studies and design reviews. Additionally there is a need to conduct tests before the first prototype has been completed (BUEDE 2009, p. 26; FRIEDENTHAL ET AL. 2008, p. 17). The output of the MBSE is therefore a coherent model of the system, where the emphasis is placed on evolving and refining the model using model-based methods and tools (FRIEDENTHAL ET AL. 2008, p. 17). One possibility to provide means to capture the systems modeling information is the *Systems Modeling Language* (SysML) (FRIEDENTHAL ET AL. 2008, p. 30).

SysML is a general-purpose graphical modeling language, which is developed to be a language for systems engineering. It supports the analysis, specification, design, verification, and validation of complex systems. These systems may include hardware, software, data, personnel, procedures, facilities, and other elements of man-made and natural systems. The language is based on the *Unified Modeling Language* (UML), which is established in software engineering and adapted to the needs of general system development and optimization. Looking at the nine provided diagrams, which can be seen in Figure 2-23, SysML can represent the following aspects of systems, components, and other entities (WEILKIENS 2006, pp. 160ff; FRIEDENTHAL ET AL. 2008, pp. 30ff):

- Structural composition, interconnection, and classification.
- Function-based, message-based, and state-based behavior.
- Constraints on the physical and performance properties.
- Allocations between behavior, structure, and constraints (e.g., functions allocated to components).
- Requirements and their relationship to other requirements, design elements, and test cases.

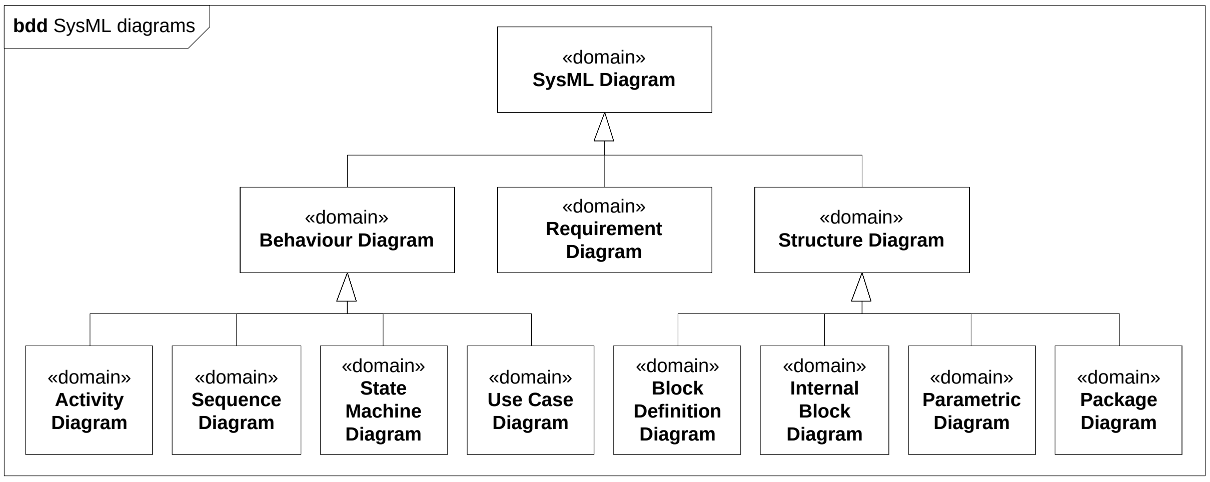

bdd SysML diagrams
«domain» **SysML Diagram**
«domain» **Behaviour Diagram**
«domain» **Requirement Diagram**
«domain» **Structure Diagram**
«domain» **Activity Diagram**
«domain» **Sequence Diagram**
«domain» **State Machine Diagram**
«domain» **Use Case Diagram**
«domain» **Block Definition Diagram**
«domain» **Internal Block Diagram**
«domain» **Parametric Diagram**
«domain» **Package Diagram**

Figure 2-23 SysML diagrams (WEILKIENS 2006, p. 160; FRIEDENTHAL ET AL. 2008, p. 30)

2.3.4 Computer-Supported Modeling Techniques

Since this thesis regards the application of computer-supported modeling techniques for goal interrelation analysis, it is necessary to get an overview on these techniques. This section provides an introduction to the four modeling techniques, which are within the focus of this research project. These are the matrices, graph grammars, relational databases, and sematic webs. All of them are presented by showing their advantages, their common application, their potentials, and, when available, an introduction into a supporting formal language.

2.3.4.1 Matrices

Matrix-Based methods are widely used for the modeling and analysis of systems (LINDEMANN ET AL. 2009, p. 49). A general introduction to them has already been given in chapter 2.3.1.1 and a more specific one for the multiple-domain matrix in chapter 2.3.1.2. As an overview about the usage of matrices for computer-supported modeling, the limitations of the classical usage of matrices are shown, followed by the possibilities and the support of matrices in current software tools to overcome these.

Since with growing systems, the connected matrices grow even faster, resulting in a limited readability and potential for analysis of them. Therefore, techniques to represent a system need to be more advanced than the classical pen and paper method (KÖNIG ET AL. 2008; LINDEMANN ET AL. 2009, pp. 49:119ff). While a matrix can be used to express all information of a system, it often makes more sense to convert only those parts being in the focus of interest. Also purely modeling the whole system would not be rational, as most users will be unable to understand the model (KÖNIG ET AL. 2008). Furthermore analyzing the structure of the nodes and edges, the subset or the whole system, represented in matrices, leads to complex computations, which can hardly be handled manually. Figure 2-24 gives an overview of basic analysis criteria for the structural characterization of the system. Especially if comprehensive systems in particular are examined, software support is required, because computational efforts for these analyses increase rapidly with the number of nodes (LINDEMANN ET AL. 2009, pp. 49:135).

Analysis criterion	**Explanation**	**Illustrate application**
Banding	Enhancement of partitioning, identification of elements that can be executed in parallel or sequentially	Identification of an optimized hierarchical order of components allows the determination of a process sequence for minimizing iterations caused by change impact
Clustering	Identification of subsets with a high internal amount of edges (compared to external links to the surroundings)	Clusters are appropriate for module building because adaption of one component of cluster often causes a cluster-internal change impact
Degree of connectivity	Percentage of existing edges compared to the theoretically possible quantity of edges	An exceptionally high or low degree of dependency suggests deficiencies in information acquisition
Distance matrix	Specifies the distances between all node parings in a structure and represents them in a matrix depiction	Ordering component blocks in the matrix by length of distance identifies the degree of change impact between component groups
Matrix of indirect dependencies	Specifies the quantity of indirect dependencies between every node pairing	Plausibility check during information acquisition: many indirect change impacts between components can suggest the existence of a direct one
Partitioning, triangularization, sequencing	Sequential or block order of nodes	Best adaptation sequence concerning effects of change propagation; minimization of iterations due to backward change impacts

Figure 2-24 Basic matrix analysis criteria for the structural characterization of systems (LINDEMANN ET AL. 2009, p. 137)

Since matrix-based computations are widely applied to complex systems, there is a large demand for software tools. The internet page http://www.dsmweb.org shows an overview on established software tools, which are using matrices-based computations and consequently matrices as computer-supported modeling goals (LINDEMANN 2011). Among them, exist research tools, which are for example macros for Excel or Matlab, as well as commercial tools like *Loomeo* (http://www.teseon.com) specialized in working with matrices. These software tools share the ability to model and to analyze matrices and thus to help keeping the system transparent and manageable. In the case of this paper, the software tool *Loomeo* is tested for an implementation of computer-supported goal interrelations in chapter 4.2.1.

2.3.4.2 Graph Grammars

Graph grammars are providing a foundation for synthesizing graph-based models as well as for modeling their structural change. The basic features of graphs were already introduced in chapter 2.3.1.1, enabling the representation of systems by using nodes and edges and their combinations. Graph grammars have an intuitive nature and are very efficient computationally. Thus, they are very effective in modeling structure and their dynamic changes (HELMS ET AL. 2009).

PAVLIDIS (1972) defines graph grammars as the quadruple of sets. These sets contain nonterminal elements, terminal elements, rewriting rules and initial elements. A set of valid elements can contain nodes, edges and their combination to form graphs, while *P* is a set of production rules, which are applied to achieve transformations of matching graphs. Thus a set of elements, needs to be defined that can be used during further operations (HELMS ET AL. 2009).

HELMS ET AL. (2009) furthermore take an object-oriented approach to define a metal model, considering the formal graph-grammar definition. In the approach the meta-model represents a class model from which instances are derived that constitutes the elements of the graph. Figure 2-25 shows on the left side a SysML like meta model that defines the general classes ‘circle’ and ‘rectangle’, the abstract classes ‘black’ and ‘grey’, and their descendants (special classes) which obtain their properties via inheritance. This principle of inherited properties works for both, nodes and edges. On the right side of the figure, the set of rewrite rules is presented, which encompass all valid operations for creating or modifying nodes and edges. These are sufficient to enable the modeling structural change according to EBEN ET AL. (2008). Thus, graph grammars fulfill all requirements to create holistic models of systems and their structural change.

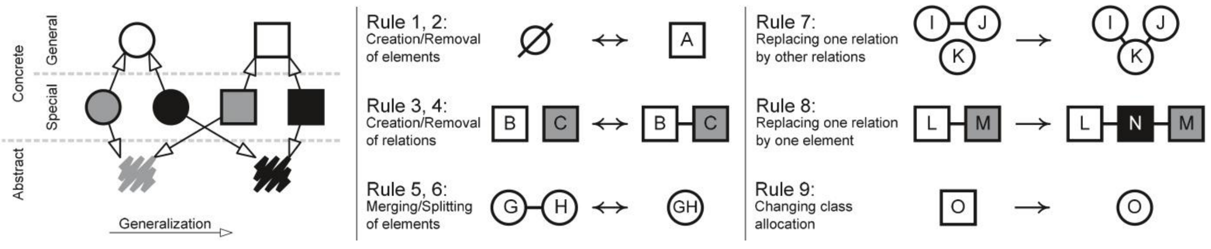

Concrete
General
Special
Abstract
Generalization
Rule 1, 2:
Creation/Removal of elements
A
Rule 3, 4:
Creation/Removal of relations
B
C
Rule 5, 6:
Merging/Splitting of elements
G
H
GH
Rule 7:
Replacing one relation by other relations
I
J
K
Rule 8:
Replacing one relation by one element
L
M
N
Rule 9:
Changing class allocation
O

Figure 2-25 Meta-Model and rule set for graph grammars (HELMS ET AL. 2009)

Considering the object-oriented approach by HELMS & SHEA (2009, p. 3) the procedure of defining computational design synthesis study consists of two parts, the definition and the application, as can be seen in Figure 2-26. In the first part, the vocabulary and the rules are defined, enabling the creation of metamodels and rule sets. This defines the grammar of a system, which results, after compilation, in an executable grammar. The second part describes the application of the compiled grammar in order to perform the requested tasks. Thus, with the definition of a rule sequence, based on several rule sets, a logic or strategy of the application can be encoded. Applied on an initial graph, the graph is transformed accordingly, returning a resulting graph. This graph can be evaluated or further processed, e.g. through simulation, as an initial graph for subsequent transformation or stored as an example in a design archive.

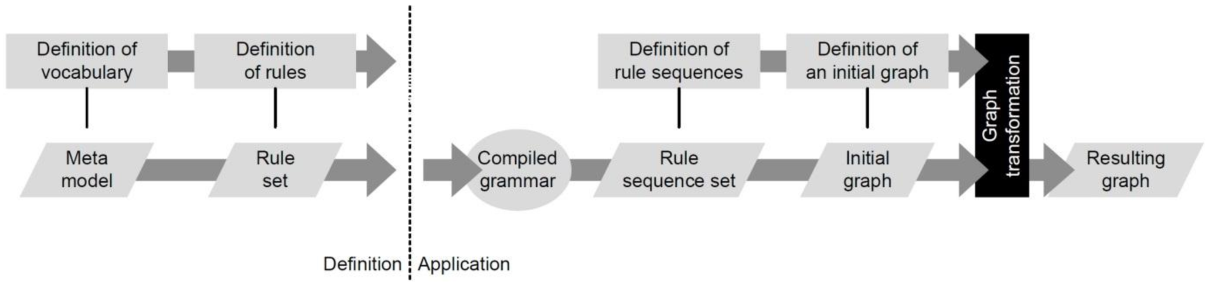


Figure 2-26 Definition of a graph-grammar and object-oriented computational design synthesis study (HELMS & SHEA 2009, p. 3)

Concluding graph grammars can be used to build models with a pre-defined vocabulary upon which implemented rules can be applied. In the case of this thesis, the various elements and rules for the goal-interrelation analyses can therefore be implemented as a meta-model and applied for the various cases. This is tested in chapter 4.2.2 with the research software tool *Booggie* (http://www.*booggie*.org/) which is based on the generative programming system for graph rewriting system, GrGen.NET (JAKUMEIT ET AL. 2010). In general, the GrGen.Net supports directly the creation of graph models, matching of patterns, rewriting them, applying rules and controlling them (BLOMER ET AL. 2011).

2.3.4.3 Relational Databases

The amount of information generated and needed in industry and administration has increased significantly in recent years. Consequently, the demand is created for powerful computers with mighty programming systems to take over the tasks to store and administrate data. Nowadays mostly *Database Management Systems* (DBMS) are used to fulfill this task and among them, the relational database is the most common (KLEINSCHMIDT 2005, pp. 1, 7). Consequently, the technique of using relational databases is also interesting as the basis for an implementation of a computer supported goal interrelation analysis. As an overview on this technique the main structure of a database environment, the fundamental relational model, as well as the query language *Structured Query Language* (SQL) are briefly presented.

A DBMS consists of a database and a set of programs access that data. Thus, a database is a well-organized collection of data that are related in a meaningful way, which can be accessed in different logical orders. The storage of information is hereby of primary importance (SUMATHI & ESAKKIRAJAN 2007, p. 3). Figure 2-27 shows a simplified model explaining the main components of a database environment. It can be seen that a DBMS receives an input from a user or a programmer, which will be processed by two levels, the computing level and the data access level. With this infrastructure, a DBMS can manage the amount of stored data, the consistency, the security and easy accessibility (KLEINSCHMIDT 2005, pp. 3ff). Furthermore every DBMS is based on a database model, with the relational database being the most common model and consequently in the focus of this work.

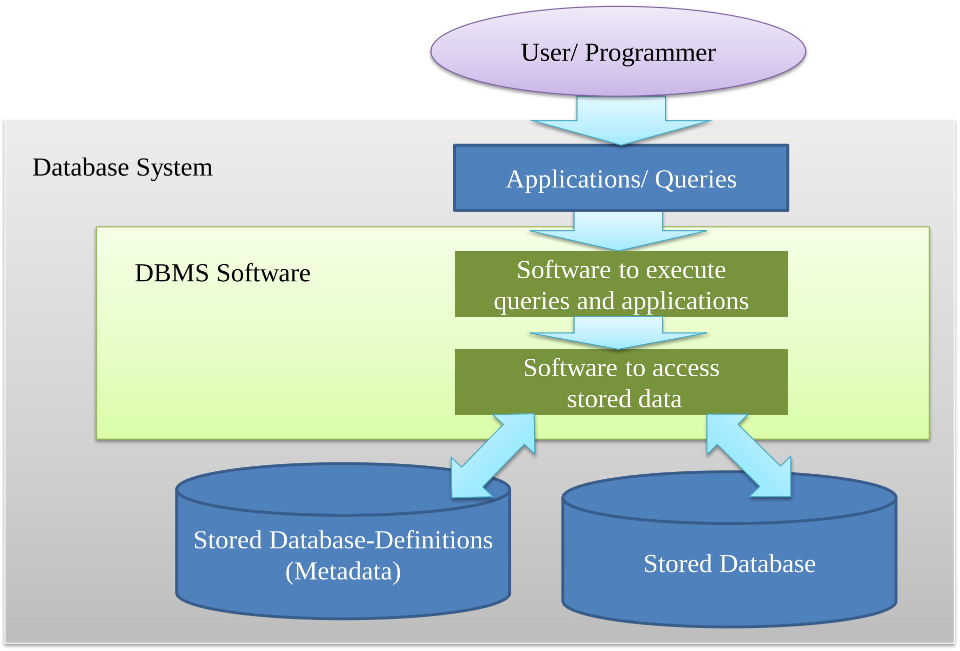

User/ Programmer
Database System
Applications/ Queries
DBMS Software
Software to execute queries and applications
Software to access stored data
Stored Database-Definitions (Metadata)
Stored Database

Figure 2-27 Simplified model of a database environment (KLEINSCHMIDT 2005, p. 4)

The relational model is based on the principle that both data and their relationships are collected within tables, being also called relations (SUMATHI & ESAKKIRAJAN 2007, p. 70; KLEINSCHMIDT 2005, p. 7). A database's structure is defined by a collection of relations, the integrity is maintained by using primary as well as foreign keys, and the relational algebra and relational calculus can be used to manipulate its data. Figure 2-28 shows an example for a relational database. In it four tables, which are named 'Core Functions', 'Solutions', 'Field of Solutions', and 'FoS to Core Assertion', can be seen. Each table defines certain attributes, which are represented by the columns, whereas the underlined attribute is the primary key. A primary key is used to identify unique sets of data in the corresponding relation and therefore maintains integrity. Each table contains multiple sets of data, being written in each row, also called tuples, where a value for a primary key occurs only once. Furthermore, a possibility to access, manipulate and define these relations, according to the user's need, is provided through SQL being the standard-database language.

Core Functions

CoreID	**Core Function**
1	Budget Aquisition
2	HR Planning
3	Motor Production
4	Chassis Production
5	Engine Specification
6	Chassis Specification

FieldOfSolutions

FoS ID	**Field of Solutions**
1	Motor
2	Chassis
3	Motor Production
4	Sponsors
5	Human Resources
6	Chassis Production

Solutions

Solution	**Field of Solutions**
Hybrid Monocoque+Spaceframe	Chassis
Spaceframe	Chassis
Sponsor Garage	Chassis Production
TuFast Garage	Chassis Production
Equal Distribution	Human Resources
Biased Distrubution	Human Resources
2*100KW Motor	Motor
4cyl Motor	Motor
2cyl Motor	Motor
TuFast Garage	Motor Production
External Production	Motor Production
A,C,D,E,F	Sponsors
B exclusively	Sponsors

FoS to Core Assertion

CoreID	**FoS ID**
1	4
2	5
3	3
4	6
5	1
6	2

Figure 2-28 Example for a relational database

The Structured Query Language (SQL) is a tool for organizing, managing, and retrieving data stored by a relational database. It is therefore a comprehensive language for controlling and interacting with a DBMS. SQL is not a procedural computer language like PASCAL or C, but a descriptive one with a collection of statements specialized for database management tasks. SQL also resembles English sentences, which makes it relatively easy to learn (WEINBERG ET AL. 2010, p. 3; KLEINSCHMIDT 2005, pp. 22ff). An example for an SQL request to retrieve the relation between solutions and core functions is shown in Figure 2-29. This SQL query returns a list of the matching attributes 'Solution' and 'Core Function' by joining the four tables according to their matching tuples. With the SQL statements, it is possible to perform almost all necessary operations for a database management. A more detailed explanation of these statements can be found in further reading, for example WEINBERG ET AL. (2010) or KLEINSCHMIDT (2005).

```
SELECT [Solutions].[Solution], [Core Functions].[Core Function]
FROM ([FieldOfSolutions]
       INNER JOIN ([Core Functions]
                INNER JOIN [FoS to Core Assertion]
                  ON [Core Functions].[Core Function]
                     = [FoS to Core Assertion].[CoreID])
         ON FieldOfSolutions.[Field of Solutions]
            = FoStoCoreAssertion.[FoS ID])
     INNER JOIN Solutions
       ON FieldOfSolutions.[Field of Solutions]
          = Solutions.[Field of Solutions];
```

Figure 2-29 SQL request to retrieve interrelations between solutions and core functions

2.3.4.4 Semantic Web

Another technique for computer-supported modeling is semantic web, which has been developed as a technique for optimized knowledge management in the *World Wide Web* (WWW). In general, semantic is the theory of the meaning of sign, thus making signs and words understandable. Overall aim of semantic techniques is therefore to make systems interpretable by machines (INSTITUT AIFB 2011). As an introduction for the semantic technologies in the field of system modeling, the main expressions of semantic techniques are explained, followed by a presentation of its considered elements, and concluding a demonstration of a representation language, the *Web Ontology Language* (OWL).

In this field, the three main expressions, 'semantic network', 'semantic web' and 'ontology' are commonly used. Thereby semantic network is the broadest term, which describes universally all technologies for machine-interpretable systems. Furthermore semantic web highlights with its name its relation to internet technologies and the usage of semantic technologies as an extension of the WWW making its information handling more efficient. Ontologies, lastly, are representations of specific systems in their context, described by their elements using semantic techniques (HABERER 2007, pp. 9ff; INSTITUT AIFB 2011).

Within a semantic network, several basic elements are defined to represent a system: classes, individuals, object relations and data relations. Thus, classes describe groups of entities with same properties; individuals also called instances or entities are existing elements, which are not further dividable; object relations represent connections between entities and finally data relations are connections between entities and specific data. In general, a system is described by multiple rules, expressing the relations between these elements. These rules to describe interrelations in a semantic network are commonly referred to as axioms. Furthermore, each element can have specific properties, classes and relations, which can be organized, using hierarchies and inheritances. Consequently, the management of all necessary elements and relations is supported, which are needed for modeling a reasonable system (HABERER 2007, pp. 9ff; KASHYAP ET AL. 2008, pp. 23ff; KNUBLAUCH ET AL. 2004).

Demonstrative for the assembly of a semantic web with the mentioned elements, a simplified product planning system of an electric car can be seen in Figure 2-30. There, the single individuals, 'Combustion Engine', 'Electric Engine', 'Powerful Motor', and 'No Gas'. These elements are further part of the two classes 'Solution' and 'Demand' and therefore classified as solution or demand. Additionally the object relations 'satisfies' and 'conflict' connect the solutions to the demands. As a final point the two green data relations 'HP' can be seen, which connect the different solutions to integer values. Concluding it can be seen that with these elements a complete system with all its interrelations can be described.

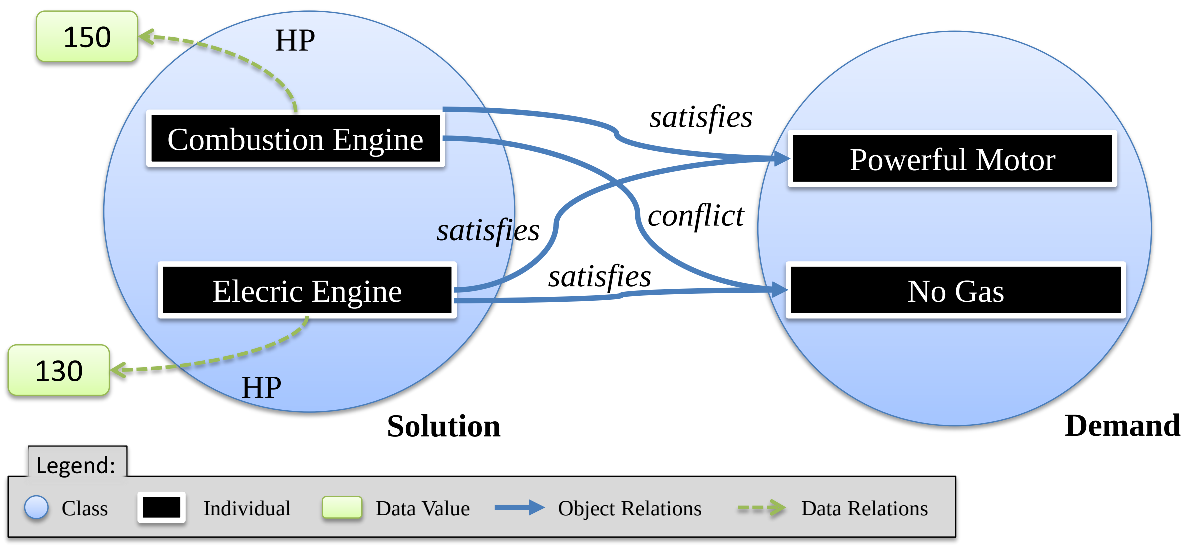


Figure 2-30 Demonstration of a semantic web for a product planning system

In order to generate content with semantic and consequently machine-interpretable information, a grammar is needed which specifies certain rules. Above all, the *World Wide Web Consortium* (W3C) has set fundamental standards with the Resource Description Framework (RDF) and lately the *Web Ontology Language* (OWL). OWL provides an expressive language for describing ontologies as well as methods for generating implicit knowledge. It enables the possibility to express all axioms, like class assertions, subclasses or object property assertions etc. There is also a syntax based on the *Extensible Markup Language* (XML) available, the OWL-XML-Syntax, which makes ontologies exchangeable (HABERER 2007, pp. 126ff; KASHYAP ET AL. 2008, pp. 35ff).

Being based on XML, the OWL-XML-Syntax is easily understandable as can be seen in the given introduction. To begin with, the creation of an ontology it is necessary to specify an address, called *Uniform Resource Identifier* (URI), which makes the ontology unique. Furthermore, all classes, individuals, and relation can be defined. How to define the individual 'Combustion_Engine' and the class 'Solution' is shown in Figure 2-31. Moreover, an object property assertion is regarded in Figure 2-32, which is the connection between two individuals through an object property. In this case, the connection between the two individuals 'Combustion_Engine' and 'Powerful_Motor' through the relation 'satisfies' is defined.

```
<owl:NamedIndividual rdf:about="http://TUfast.de#Combustion_Engine"/>
<owl:Class rdf:about="http://TUfast.de#Solution"/>
```

Figure 2-31 Definition of an individual and a class in OWL

Similarly, also other axioms can be defined, but a description of the whole syntax at this point would be too extensive and can be found in further reading, for example (HABERER 2007, pp. 17ff), (KASHYAP ET AL. 2008, pp. 37ff), and (MOTIK ET AL. 2009). Anyway, a good understanding of the syntax is helpful but not necessary since there are many software tools supporting OWL.

```
<owl:NamedIndividual rdf:about=" http://TUfast.de#Combustion_Engine">
<TUfast:satisfies rdf:resource="http://TUfast.de#Powerful_Motor"/>
</owl:NamedIndividual>
```

Figure 2-32 Definition of an object relation in OWL

Since sematic web is a rapidly growing field of interest, there is already a large variety of existing programs. These programs simplify and enrich the handling of semantic networks while being essential for a wide adaption of the new technology (KNUBLAUCH ET AL. 2004). Among them are application interfaces, applications, editors, parser, project software, reasoner and viewer, but especially the open platform *Protégé*, is within the focus of this thesis (HABERER 2007, pp. 126ff; KASHYAP ET AL. 2008, pp. 137ff). *Protégé* started with being a simple ontology editor, but was extended to combine all necessary components for the whole process of working with ontology (KNUBLAUCH ET AL. 2004; KASHYAP ET AL. 2008, p. 139). Therefore, this software is the basis for the implementation in chapter 4.2.4.

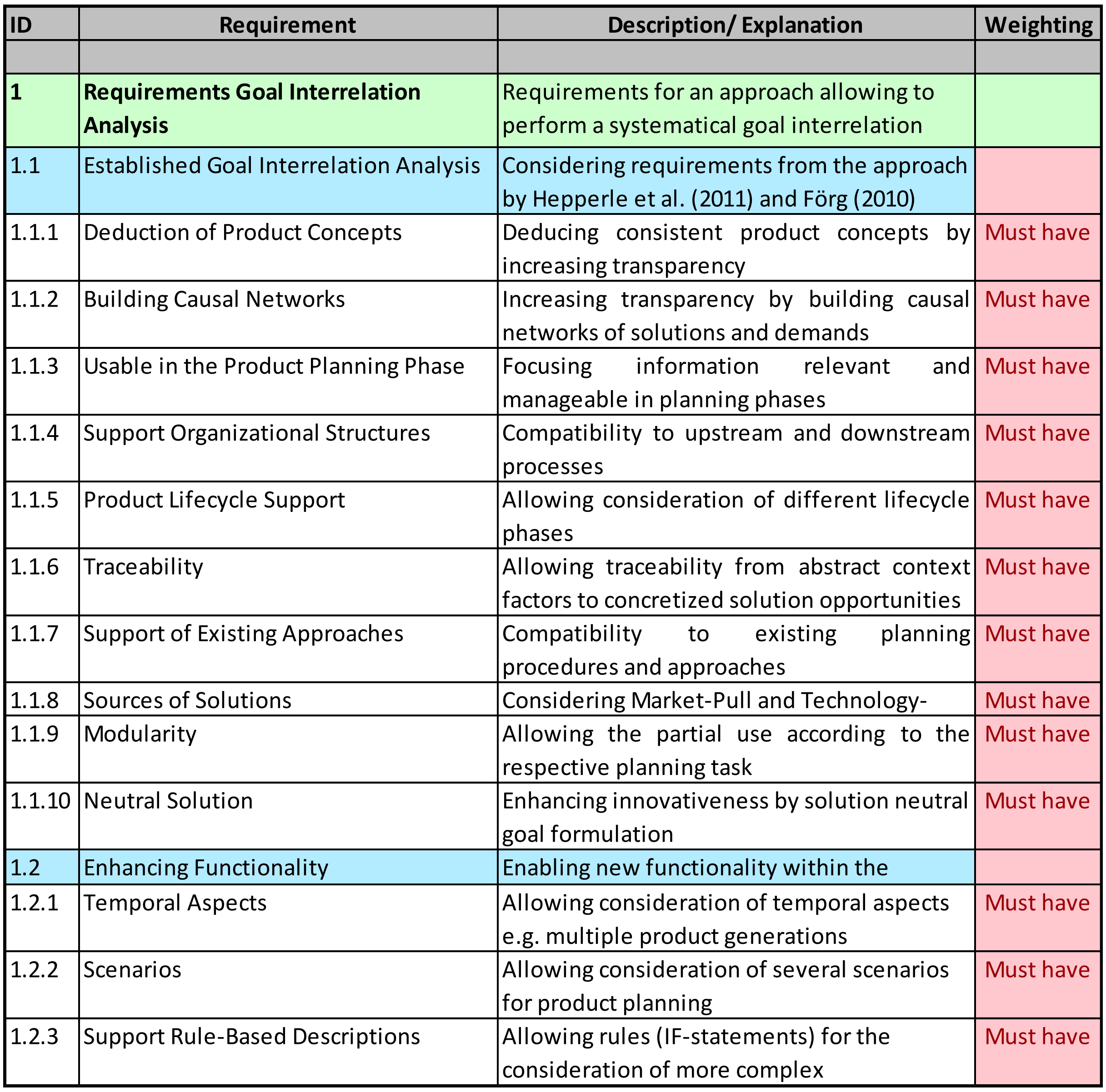
3 Requirements for Computer-Supported Goal Interrelation Analysis

A first step towards the implementation of computer-supported *Goal Interrelation Analysis* (GIA) is the definition of the general requirements for it. The requirements were elaborated by researching related work and related approaches from the systematical product development. Furthermore, these results were extended, improved, approved, and weighted in several iteration loops. These include interviews and workshops with various stakeholders as well as creating implementation with several established computer-supported modeling techniques for a profound analysis. These implementations are further presented in chapter 4. Additionally it was identified, that in general the requirements can be allocated to three different groups. The first one describes the requirements for the general approach to analyze goal interrelation in the context of a product planning process. The second group defines the requirements for the implementation of the content of a GIA, thus ‘what’ a software tool needs to handle. Finally, the last group deals with the requirements for the implementation of a GIA supporting software tool, describing what features it has to offer and how it has to work. Hence, the finally achieved version of the requirements is presented in three different groups in this section. All of the requirements are listed in tables, telling their unique identification number, the name of the requirement, a more detailed description and a weighting, showing if it is mandatory (Must have), medium (Should have), or optional (Nice to have).

3.1 Requirements for a Goal Interrelation Analysis

To begin with, the general requirements for an approach concerning the goal interrelation analysis within the product planning process are given. These can be seen in Figure 3-1 and are subdivided into two major points: the established GIA and the enhanced functionality. Since this thesis takes the graph-based approach of Förg (2010) and Hepperle et al. (2011a) as the underlying work, the main features of it must be fulfilled within this concept. Therefore, they are listed in the first sub point. Since they are already discussed in chapter 2.2, they are not further explained within this chapter.

Furthermore, research has identified the need for additional functionality of the graph-based approach in order to enable an increased usability of it. Consequently, a consideration of temporal aspects as well as scenarios is requested in order to support directly several product generations and trends. For the description of more complex relations and circumstances, the support of rule-based knowledge is also added. Summarizing all these requirements a holistic view upon the main features of the enhanced graph-based approach, which serves as the basis for the computational implementation, is given.



ID	Requirement	Description/ Explanation	Weighting
1	**Requirements Goal Interrelation Analysis**	Requirements for an approach allowing to perform a systematical goal interrelation	
1.1	Established Goal Interrelation Analysis	Considering requirements from the approach by Hepperle et al. (2011) and Förg (2010)	
1.1.1	Deduction of Product Concepts	Deducing consistent product concepts by increasing transparency	Must have
1.1.2	Building Causal Networks	Increasing transparency by building causal networks of solutions and demands	Must have
1.1.3	Usable in the Product Planning Phase	Focusing information relevant and manageable in planning phases	Must have
1.1.4	Support Organizational Structures	Compatibility to upstream and downstream processes	Must have
1.1.5	Product Lifecycle Support	Allowing consideration of different lifecycle phases	Must have
1.1.6	Traceability	Allowing traceability from abstract context factors to concretized solution opportunities	Must have
1.1.7	Support of Existing Approaches	Compatibility to existing planning procedures and approaches	Must have
1.1.8	Sources of Solutions	Considering Market-Pull and Technology-	Must have
1.1.9	Modularity	Allowing the partial use according to the respective planning task	Must have
1.1.10	Neutral Solution	Enhancing innovativeness by solution neutral goal formulation	Must have
1.2	Enhancing Functionality	Enabling new functionality within the	
1.2.1	Temporal Aspects	Allowing consideration of temporal aspects e.g. multiple product generations	Must have
1.2.2	Scenarios	Allowing consideration of several scenarios for product planning	Must have
1.2.3	Support Rule-Based Descriptions	Allowing rules (IF-statements) for the consideration of more complex	Must have

Figure 3-1 General Requirements for Goal Interrelations Analysis

3.2 Requirements for an Implementation of the content

As a next point, the requirements for the implementation of the approach's content for goal interrelation analysis are given in Figure 3-2. Therefore, the table is divided into three major points: 'Types of Elements', 'Element Relations', and 'Element Handling'. In this case, the expression element considers all objects, classes or entities within the implementation.

ID	Requirement	Description/ Explanation	Weighting
2	**Requirements for the Implementation of the Approach**	Requirements concerning the implementation of the approach to analyze goal interrelations	
2.1	Types of Elements	Considering specific types of elements for the implementation of the approach	
2.1.2	Support all Considered Elements of a GIA	Supporting the main elements of GIA: Demands, Goals, Core Functions, Fields of Solutions, Solutions, Parameters	Must have
2.1.3	Support of PSS-Lifecycle Stages	Supporting the consideration of the various lifecycle phases	Must have
2.1.4	Support of Temporal Classification	Supporting temporal classification	Must have
2.1.5	Support of Scenarios	Supporting the mapping of elements to various scenarios	Must have
2.2	Element Relations	Supporting relation between elements	
2.2.1	Goals to Demands	Supporting the relation between goals and demands	Must have
2.2.2	Core Functions to Goals	Supporting the relation between core function and goals	Must have
2.2.3	Core Functions to Fields of Solutions respectively Solutions	Supporting the relation between solutions and core functions	Must have
2.2.4	Solutions to Parameters	Supporting the relation between solutions and parameters	Must have
2.2.7	Fuzzy Values for Parameters	Allowing the consideration of fuzzy information by enabling connections to a range of values of a parameter	Nice to have
2.2.8	Various Parameter Connections	Allowing various types of relations between solution and parameters e.g. binary, quantitative, qualitative, or customized	Should have
2.3	Element Handling	Supporting type and situation specific handling of the elements	
2.3.1	Unique Classification of Elements	Limiting elements to be classified as an single type of basic element.	Must have
2.3.2	Simultaneous Classification of Elements	Supporting elements to be classified as part several lifecycle stages, scenarios and times	Must have
2.3.3	Global Parameter Definition	Supporting the global definition of parameters by their name, type and preferred value	Must have
2.3.4	Field of Solutions	Supporting the collection of interchangeable solutions, which are connected to the same parameters and the same core functions, in fields of solutions	Must have
2.3.5	Core Functions Optionality	The usage of core functions is not necessary, but needs to be congruent for the whole system	Nice to have
2.3.6	Support of Rule-Based Knowledge Content	Enable certain elements or relation or changing values by optional added IF-Statements describing rules	Nice to have

Figure 3-2 Requirements for Implementation of the Approach for Goal Interrelation Analysis

The first point in the table regards the support of the various types of elements. Therefore, an implementation needs to consider the main elements of a GIA, which are demands, goals, core functions, fields of solutions, solutions, and parameters. Additionally various lifecycle phases, a temporal classification, as well as the consideration of scenarios should be supported for the mapping of elements. The second point regards all necessary relations between these elements and their properties. This means to consider the relation between demands and goals, between goals and core function, and core functions to solutions respectively fields of solutions. Furthermore, the solutions need to be connected to their parameters, if possible with fuzzy values and various types of relations related to the kind of interaction between them. Lastly, the third point describes further requirements for handling elements in order to enable a GIA. Thus, it is specified that an element can be only of the type demand, goal, core function, field of solutions, or solution, whereas it can be classified to several lifecycle stages, scenario and temporal aspects. Furthermore, it is requested to support a global definition of parameters, as well as field of solutions being collections of various interchangeable solutions. The last two requirements are voluntary and describe the functionality to make core functions optional, as well as the support of rule-based knowledge content. Concluding with the implementation of all the presented requirements the main functionality for analyzing goal interrelations by a software tool can be secured.

3.3 Requirements for a Software Implementation

The last table of requirements regards the field of software implementation. These requirements are especially addressing the development of the software tool, describing requested functionality and features in order to offer a high usability. They are shown in Figure 3-3 and Figure 3-4, divided into a mandatory and an optional list.

The mandatory requirements, which can be seen in Figure 3-3, are describing the main and needed functions for the implementation of a software tool. For a better overview, the table is subdivided into four major points, specifying the handling of data, the data acquisition, the query for conflicts and the data organization. Among these points, the first point ‘data handling’, is responsible for considering the requirements for the implementation of the content, described in chapter 3.2. The second point, data acquisition, deals with the feature to enter new data and information into the system. This means, the software should generally allow entering new data into the system. This should be enabled in easily understandable forms, which are especially adapted to the considered element. In addition, the possibility to import data directly from other external sources should be supported as well as an interactive and live forecast of conflicts during the data acquisition. The third point, query for conflicts, regards the functionality to identify goal interrelations and conflicts of these. Thus, the software should identify interrelations between goals, which are indirectly connected through parameters. Furthermore, it should detect conflicts between solutions, which are connected to the same parameter but with incompatible values. This should lead to complete relation chains, to get an overview of interconnections in the entire system. Lastly it should also be enabled to filter the results according to their classifications, perform a partially query and to save the query. Finally the last point, data organization and maintenance, specifies the requirements for the handling of the entered and stored data. Therefore, it should be possible to store the acquired data in a common file format. The software should also support plausibility and consistency checks as well as a general update of large sets of data.

ID	Requirement	Description/ Explanation	Weighting
3	**Software Requirements**	Requirements concerning the software implementation of the goal interrelations planning method	
3.1	Data Handling	Handling all data required to perform goal interrelation analysis, defined in point 2	
3.2	Data Aquisition	Acquiring and entering of new data sets	
3.2.1	Data Acquisition	Allowing to enter new data about the product	Must have
3.2.2	Fast and Easy Data Acquisition	Enabling data Input in a single form for one data set, and the inheritance of attributes from parent elements	Should have
3.2.3	Import Functionality	Allowing to import already existing data from external sources like QFD, Excel, etc.	Nice to have
3.2.4	Forecast of Conflicts	Checking "live" the new goal interrelations and shows conflicts	Nice to have
3.3	Query for Conflicts	Detecting conflicts between goals through parameters	
3.3.1	Identify Interrelations of Goals	Checking the relations between Goals through their related parameters	Must have
3.3.2	Conflict Detection within Parameters	Detecting possible conflicts between solutions which occur when they are both connected to the same parameter	Must have
3.3.3	Detecting Complete Relation Chains	Detecting chain connections between all elements of the system order to understand all interrelations	Must have
3.3.4	Filters	Supporting the filtering of results (scenario, time, specific entity etc.)	Must have
3.3.5	Partial Query	Enabling query to be performed on a partial system for performance reasons	Nice to have
3.3.6	Saving Queries	Allowing to save specific queries and their results	Should have
3.4	Data Organization/ Maintenance	Organizing and maintaining the stored information	
3.4.1	Data Storage in File	Storing data in an known, widely spread data format	Must have
3.4.2	Plausibility and Consistency Checks	Enabling the possibility to check the data for plausibility and consistency	Should have
3.4.3	Data Update	Enabling the update of whole sets of data should (Like in SQL)	Should have

Figure 3-3 Mandatory Software Requirements for Computer-Supported Goal Interrelation Analysis

In the following Figure 3-4, the optional requirements for the software implementation are specified. These are subdivided into three points, the visualization, the computational optimization, and the ergonomic requirements. The first point, visualization deals with a graphical representation of the goal interrelation analysis, in order to provide a better overview to the user. Consequently, the software should provide a graphical representation of the entire system. This representation should be further filterable, in order to focus on specific lifecycle phases, scenarios, temporal aspects, etc. For a large and complex system, the graphical representation should be using multiple dimensions. Lastly, also the results of the interrelation and conflict identification should be shown within the visualization. The second point, computational system optimization, specifies the potential of deriving optimal solutions for the *Product Planning System* (PPS) by the computer. Hence, it should be possible to compute optimal combinations of goals and their solutions. Furthermore, an interface should be provided to access all elements from external programs in order to optimize them. Finally the last point, 'ergonomically' describes demanded features to facilitate the handling of the software. Thus, the software should provide an intuitive graphical user interface. Additionally it should be possible to interact with the system in different levels of detail, related on the goal to receive a vague overview or to see all details of the system. This level of abstraction should be also indicated to the user in order to allow a consistent level for the entered elements. Summarizing this section, a complete definition for all important features for the implementation of a software tool is given, allowing a complete support of the application of a GIA.

Concluding, this whole section is describing the requirements for the implementation of a computer-supported GIA. The requirements are split into three sections, the general methodical, the content for the implementation, and the software features. These requirements are further used as a basis for the implementations in order to analyze the various computer-supported modeling techniques and leading to a specification of the framework for computer-supported goal interrelation in chapter 5.

ID	Requirement	Description/ Explanation	Weighting
3	**Software Requirements**	Requirements concerning the software implementation of the goal interrelations planning method	
3.5	Visualization	Visualizing the stored information	
3.5.1	Graphical Representation of the System	Representing the PPS in a net/graph form according to Förg (2011) and Hepperle (2011)	Should have
3.5.2	Filters for Visualization	Supporting filters for the visualization to see just specific parts of the system(e.g. entity type, scenarios, time, lifecycle stage)	Should have
3.5.3	Multidimensional	Presenting the interconnections between elements in the visualization using multiple dimensions	Nice to have
3.5.4	Showing Results/Conflicts	Indicating the conflicts in the system visually	Nice to have
3.6	Computational System Optimization	Supporting the identification of an optimal solution for the PPS	
3.6.1	Best Combination Check	Identifying the best solution by comparing the various solutions	Nice to have
3.6.1	Optimization Interfaces	Providing an application interface to perform further optimizations	Nice to have
3.7	Ergonomically	Considering ergonomically aspects	
3.7.1	Intuitive Graphical User Interface	Providing a graphical user interface which enables an intuitive handling of the software	Should have
3.7.2	Hierarchical Adapted Result Presentation	Presenting a quick overview of the results by showing in different level of details: Temporal, Scenario, Lifecycle, Group of Solution and Solution	Should have
3.7.3	Representation of the Abstraction Level of the Product	Indicating the required level of abstraction to the user, which is required for the consistent inputs	Should have

Figure 3-4 Optional Software Requirements for Computer-Supported Goal Interrelation Analysis

4 Analysis of Computer-Supported Modeling Techniques

In this chapter, the possibilities to analyze goal interrelations with established computer-supported modeling techniques and their related software is tested and discussed. The analyzed techniques are representing a system using the multiple-domain matrix, graph grammars, relational databases, or a semantic web. In order to provide equal conditions for the analysis of the various test conditions, their proceedings and the applied example of a *Product Planning System* (PPS) are initially presented. Based on these, the four selected computer-supported modeling techniques are applied for an implementation of the example and evaluated in the end of this section. Since the implementations are part of the iterations to define the requirements for computer-supported *Goal Interrelation Analysis* (GIA), they were improved with every iteration cycle. Therefore, this chapter presents a mature state of the implementations that can serve for further analysis.

4.1 Conditions for the Analysis

In order to provide equal conditions for the analysis of the various modeling techniques, the proceedings of the analyses as well as the example for the tested implementation of the PPS is presented. Thus it is shown which aspects of the various solutions are investigated and where the focus of the analysis is set.

4.1.1 Proceeding of the Analysis

A general proceeding for the analyses of the four different computer-supported modeling techniques is defined to allow the achievement of comparable results. Consequently, for every solution the related software tool is introduced, followed by an implementation of the exemplary PPS and a presentation of how the tool can support the analysis of the interrelation of the various goals of the system. Summarizing, all results are evaluated in a final section, which is confronting the presented techniques with the main requirements presented in chapter 3.

The introduction of the applied software tool is the first step to receive an overview and understand the aspects of the presented implementations. The usage of software tools for the analysis of the various modeling techniques is necessary, since the modeling techniques are providing only solutions how to represent a system, but not for the creation of the model. Consequently, a software tool is chosen which supports the various modeling techniques, the design of the system representation, as well as further analysis. Problematic with the various tools is their difference in professionally and maturity. Therefore, the functionalities, the stability, the usability, the diffusion of the supported file formats, etc. is hardly comparable. Nevertheless, the programs are introduced by an outline of their basic characteristics and by an explanation of their basic process. This process implies the creation of a system, the adding of information, and finally the possibilities to support further analysis.

The second step is the implementation of the PPS with the various modeling techniques. Therefore, a solution for the representation of the exemplary PPS presented in the next section is generated using the various modeling techniques. The solution needs to consider all relevant elements of the PPS like goals, demands, etc. Subsequently, the relations between the elements, as well as further attributes like the belonging product lifecycle, season, or scenarios need to be modeled. Additionally to the elements and interrelations, the implementation needs to fulfill some further requirements in order to represent a complex PPS. These are based on the requirement presented in chapter 3.1 for enhancing the existing approach through considering temporal aspects, scenarios as well as rule-based content. Thus, all elements of the system should be classified to specific scenarios or product generations. Furthermore, specific rules have to be implementable considering more complex relations between the various elements, allowing changing values of parameters or disabling specific elements. This can be achieved by adding the according attributes to the elements of the PPS. Finally the kind of relation between the solutions and their influenced parameters needs to be considerable. For this example, the three types of relations have been identified: leads to, requires and affects. In this case, if a solution leads to a parameter, it sets the parameter to the specific value without interacting with any other solutions. If a solutions requires a value of a parameter, it needs minimum the define value in order to be applicable. Lastly if a solution affects a parameter, it changes the value of the parameter by adding the solution's value for it.

The next step is to analyze the possibilities to perform a GIA based on the various techniques. To begin with, the generally offered methods as well as limitations to analyze systems are presented and applied on the exemplary PPS. This is followed by results and general advantages in supporting a GIA are demonstrated. Consequently with the whole proceeding of analyzing the various computer-supported modeling techniques all important aspects of these are shown, which serve as a basis for the final evaluation in the last chapter.

4.1.2 Applied Example of a Product Planning System

As already mentioned in the previous section, an example for the application of a PPS is needed. Therefore an example is generated, which is based on the Formula Student Competition and *TUfast*. In order to present this example, firstly, the general background of it is introduced, followed by the derived example of a PPS.

4.1.2.1 Introduction to Formula Student Competition and *TUfast*

The Formula Student is a competition, where students build single seat formula racecar competing with these against teams from all over the world in multiple disciplines. Since the competition is held yearly with newly adapted rules for each season, the time for the entire development process is limited to around ten months. The competitive criteria consist of several disciplines, which are divided into static and dynamic ones. In the static ones, the students show experts in commerce their constructive solutions, defend their calculated production costs and present a business-plan. On the race track however, the prototypes must prove their capabilities concerning acceleration, skidpad, handling, endurance and fuel-efficiency. Thus the competition is not limited to a single race, but considers the entire development process as well. Since 2010 there exist two competitions in the Formula Student:

the classic Formula Student Combustion, with combustion engines, and the Formula Student Electric with electric engines (LINDEMANN ET AL. 2009, pp. 12f; TUFAST E.V. 2011b; VDI-SOCIETY FOR AUTOMOTIVE AND TRAFFIC SYSTEMS TECHNOLOGY 2011b).

Since 2002 the TU München is participating at the Formula Student with its project team *TUfast*. For the competition the team is constructing their racecar powered by a 600 ccm motorbike engine. Therefore many components, from the drive to the suspension, have to be adapted for the application scenario or completely developed and produced. This is done by a team with around 30 voluntary members, mostly mechanical engineering students, which work two days average per week. For the production of the racecar, they have a general project plan, which can be outlined as the follows: Initially the general development is determined through a concept workshop. Subsequently, the component design starts, which is terminated by the final specification of the component design. Simultaneously the production starts and lasts until the racecar is available and in a roadworthy status (VDI-SOCIETY FOR AUTOMOTIVE AND TRAFFIC SYSTEMS TECHNOLOGY 2011b; TUFAST E.V. 2011b).

Since 2011 *TUfast* is also active at the Formula Student Electric with its division 'e-technology'. Thus, a new scenario for the development has to be considered, requiring the usage of an electric motor together with accumulators as engine of the car. In general, the rules differ hardly from the regular Formula Student, except from some changes due to the difference of combustion and electric cars. Consequently it is possible to interchange experiences, processes and constructions from both division of *TUfast* (VDI-SOCIETY FOR AUTOMOTIVE AND TRAFFIC SYSTEMS TECHNOLOGY 2011a; TUFAST E.V. 2011a).

4.1.2.2 Derived Example for the Analysis

This section presents the requested example which is based on the development process of a formula student racecar with a combustion engine in the season 2010. The example shows a simplified PPS in order to serve for an implementation and further general analysis of the various modeling techniques. In general the example has its focus on the project planning of the product, resulting in resource-based parameters, but the method is equally applicable on examples addressing technical issues.

Figure 4-1 shows the schematic representation of the considered example. The illustration is based on the graph-based representation, presented in chapter 2.2.4. On the right side, it can be seen that two phases of the product lifecycle are regarded, the utilization as well as the production phase and a general project planning phase. This general phase was added, since the example focuses on the project planning of a product. These phases are opposed to the global parameters of the PPS being the budget, the person days for the motor or chassis division and the production quality. Every lifecycle phase consist of two corresponding demands which are connected to two goals. The goals are furthermore connected to core functions of the product planning process. These again are related to the fields of solutions each having two solutions as members. Every field of solution is related to several parameters indicating that all including solutions are influencing the same parameters as well, but with different values. Since the solutions and their specific values are key elements to analyze interrelation between the connected goals they need to be regarded more closely.

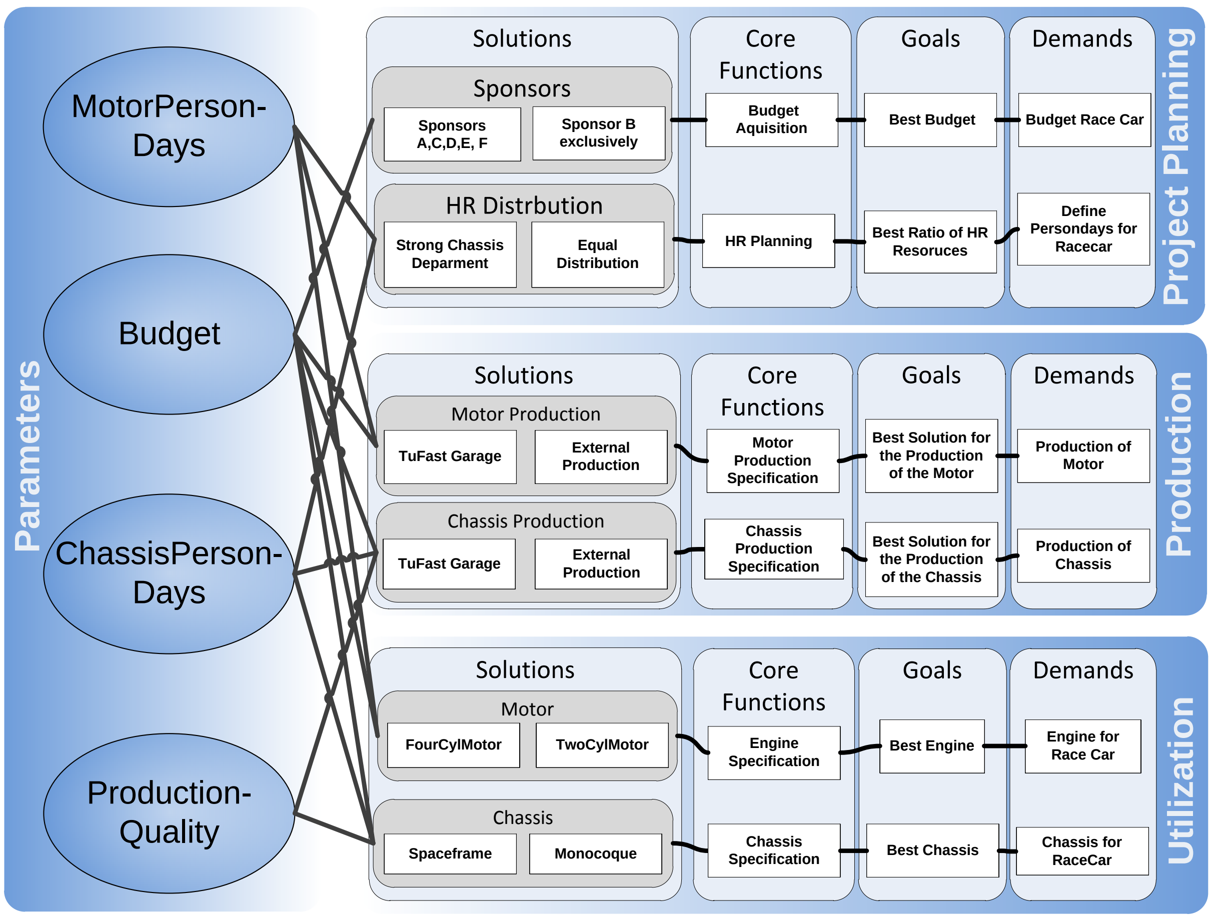


Figure 4-1 Simplified example for the PPS of a racecar

For the presented example every field of solutions includes two interchangeable solutions with different values but with equal behavior on the way to influence the various parameters. Therefore the sponsors can be either sponsor A,C,D,E, and F which are setting the budget to 70.000€ or the sponsor B exclusively which sets the budget to 60.000 €. The distribution of the human resources can be either equal, setting the person days of the chassis and the motor division to 1.000 person days equally, or a strong chassis department with 1250 person days while the motor department has only 750 person days. The production of the motor can be either done in the garage of *TUfast* which needs 5.000€ of the budget and 400 person days of the motor division or externally, needing 20.000€ from the budget but only 100 person days. In addition, the production of the chassis can be done either at the *TUfast* garage needing 10.000€ from the budget, 600 person days from the chassis division and leading to a medium quality of the production. Otherwise the external production demands 30.000€, 100 person days and leads to a very high production quality. For the motor of the racecar a four or two cylinder motor can be selected. The four cylinder motor costs 10.000€ from the budget and needs 100 person days from the motor division while the two cylinder motor costs only 5.000€ but needs 300 person days due to longer maintenance times. Finally for the chassis the two solutions, space frame or monocoque, are available. The space frame requires only a low production quality, costs 5.000€, and needs 300 person days from the chassis division. The monocoque instead requires a very high production quality, costs 20.000€, and needs 500 person days. Concluding, this section a complete PPS for a racecar, with all necessary elements and interrelations, is presented.

Furthermore, regarding the example, the applied limitation needs to be presented. Firstly it needs to be said, that the values for the different solutions are just exemplary values, which not necessarily represent real values. Additionally the number of parameters, influenced by the various solutions, is strongly minimized in order to provide an example which is simple enough to be used for a general analysis. Thus the advantages of some solutions are not obvious, but this is not necessary to be seen in order to provide a representation of a complete PPS. In general the example is biased towards parameters which represent resources of a product planning process and not technical one, which is also considered by the introduction of the additional phase 'Project Planning' which is not a common phase of the product lifecycle. However, since also technical solutions are influencing parameters, as can be seen for example with the parameter 'ProductionQuality', the application with more technical PPS is equally possible. Nevertheless it shows strength of the approach, because it is not limited to one domain of the product planning but universally applicable.

4.2 Analysis of Computer-Supported Techniques

In this section, the application of multiple computer-supported modeling techniques to support the analysis of goal interrelations is presented. Therefore the four techniques, 'MDM', 'graph grammars', 'relational databases' and 'semantic web' are applied for an implementation, based on the conditions defined in the previous section. Consequently their aptitude for providing a computer-supported goal interrelation analysis is analyzed within this section.

4.2.1 Multiple-Domain Matrix

The *multiple-domain matrix* (MDM) is a systematically defined matrix which can be used as a representation of systems. Additionally with the transfer into a computer-based form, various computations and automated analyses can be applied (see chapter 2.3.4.1). Thus it supports the basic requirements for the implementation of a computer-supported goal interrelation analysis. This implementation is presented by firstly introduction into the used software, *Loomeo*, secondly by showing an approach for a product planning system followed by the application of a goal interrelation analysis.

4.2.1.1 Introduction into the software tool *Loomeo*

Loomeo is a software tool, which works with MDM and was developed by the Teseon GmbH (http://www.teseon.de) in order to support the understanding of complex systems and their dependencies. Thus, it is especially designed for program planning, product design, change management, variant management, market structure analysis, and organization planning.

Within these fields, it enables to manage the occurring complexity through identification of interrelations and potentials for interaction in networks, as well as by analyzing structures and their behavior (MAURER 2011). To support these, the software tool is focused on three tasks, the acquisition, the visualization, and the analysis of system architectures.

The first step towards managing complexity of system architectures is the information acquisition. Therefore *Loomeo* provides an import function, where data can be directly received from external sources. Further a dedicated acquisition and documentation mode is included, which supports the data collection in workshops by special graphical interfaces and coloring. Additionally the tool can also derive dependencies with matrix computation techniques and all information can live reedited within the program.

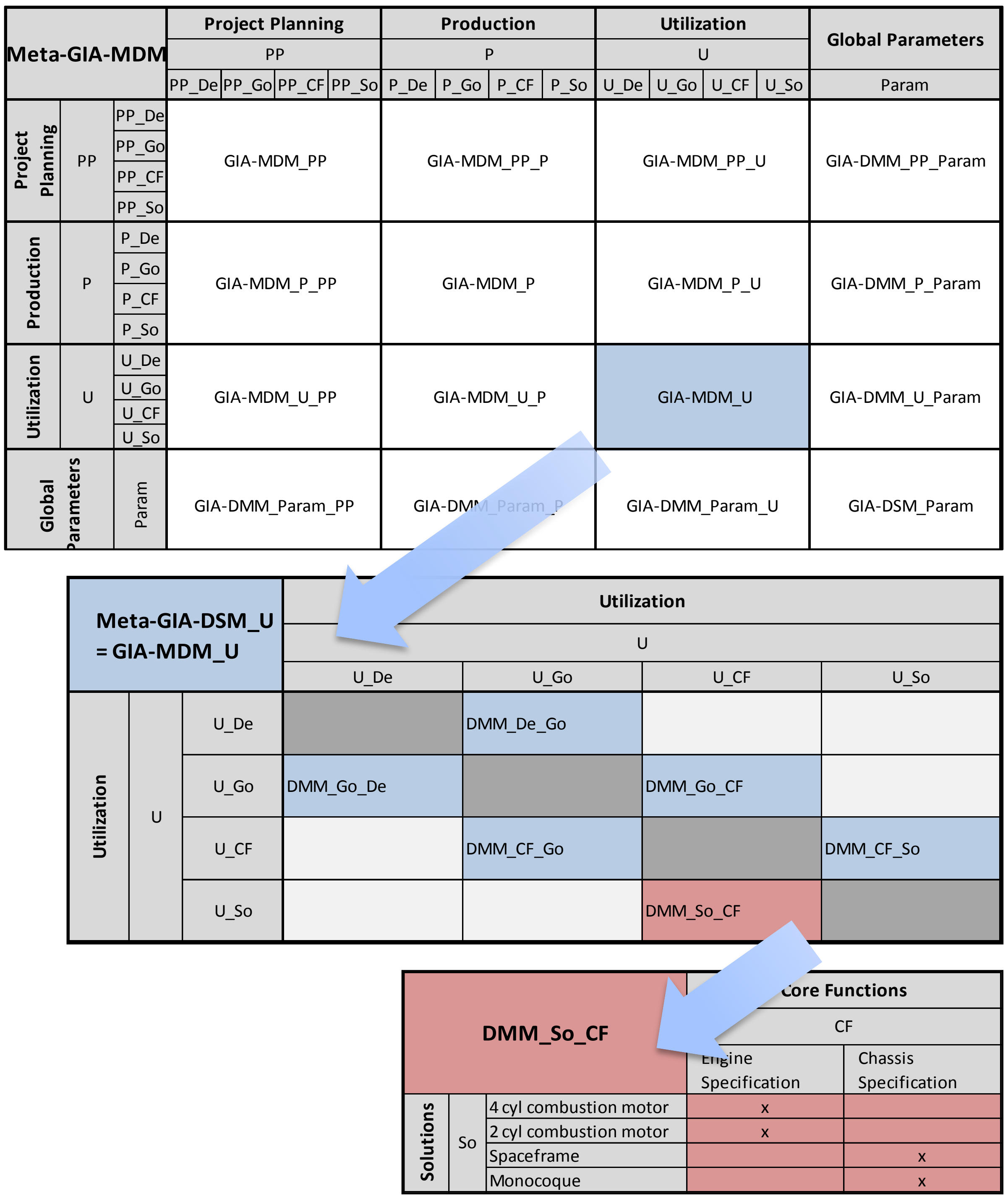
After acquiring the information about the system, *Loomeo* can visualize the system with in a matrix or a graph form. Matrices thereby provide an ordered view on the system, with elements on both axes and their connections marked in the main field with crosses or numbers according to their weighted relations. On the other hand, graphs are used to represent the system, connecting the elements with direct lines. Graphs can be arranged manually, hierarchically or automatically and dynamically according to their connections to other elements. With these representations a first overview on the system is given and first graphical analysis can be performed.

Another focus of *Loomeo* is the computational analysis of system. Thus it can deduce indirect dependencies, create clusters of strongly dependent elements. It can also identify cycles, paths, and special elements. Another feature is the calculation of degree metrics of the matrix and automatically draws statistical graphs upon it. Results of these analyses often can be shown directly in the graphical representation with highlighted colors. For this work especially the path analysis is important, which is a basic feature to identify interrelations between goals.

4.2.1.2 Implementation of a Product Planning System

In order to performing a goal interrelation analysis based on a MDM, firstly an implementation of a product planning system has to be done. Within this thesis, the implementation is based on the approach by (FÖRG 2010, pp. 95ff). This approach is used and adapted in order to create a MDM-based representation of a product planning system, the *Meta-Goal Interrelation Analysis-MDM* (Meta-GIA-MDM).

The Meta-GIA-MDM is a matrix representation of a product planning system, considering demands, goals, core functions, solutions, and global parameters over all lifecycle phases. The general buildup of the Meta-GIA-MDM for the development of a racecar can be seen in Figure 4-2. Thus the Meta-GIA-MDM is composed by four meta-domains, 'Project Planning', 'Production', 'Utilization', and 'Global Parameters' setting altogether twelve MDMs. These have the domains 'Demand' (De), 'Goal' (Go), 'Core Function' (CF), 'Solution' (So), or 'Global Parameters' (Param). The example in Figure 4-2 shows further an extract of the MDM connecting the domains from the utilization phase, being labeled GIA_MDM_U. This MDM can theoretically be subdivided into twelve DMMs and four DSMs, which can be seen in the figure in the middle matrix. However, from these matrices just six DMMs are relevant, since the other would represent relation, which are not considered in the PPS. Within these DMMs the relations between two elements are directly marked. This can be seen in the lower matrix for the DMM between the solutions and the core functions from the utilization phase. The 'Global Parameters' are not listed within a lifecycle phase, because they are independent and better readable, when they are written separately. Summarizing, with this definition of a Meta-GIA-MDM a complete PPS can be represented and used as an input matrix to be analyzed with *Loomeo*.



Meta-GIA-MDM			Project Planning	Production	Utilization	Global Parameters
			PP	P	U	
			PP_De, PP_Go, PP_CF, PP_So	P_De, P_Go, P_CF, P_So	U_De, U_Go, U_CF, U_So	Param
Project Planning	PP	PP_De, PP_Go, PP_CF, PP_So	GIA-MDM_PP	GIA-MDM_PP_P	GIA-MDM_PP_U	GIA-DMM_PP_Param
Production	P	P_De, P_Go, P_CF, P_So	GIA-MDM_P_PP	GIA-MDM_P	GIA-MDM_P_U	GIA-DMM_P_Param
Utilization	U	U_De, U_Go, U_CF, U_So	GIA-MDM_U_PP	GIA-MDM_U_P	GIA-MDM_U	GIA-DMM_U_Param
Global Parameters		Param	GIA-DMM_Param_PP	GIA-DMM_Param_P	GIA-DMM_Param_U	GIA-DSM_Param

Meta-GIA-DSM_U = GIA-MDM_U			Utilization			
			U			
			U_De	U_Go	U_CF	U_So
Utilization	U	U_De		DMM_De_Go		
		U_Go	DMM_Go_De		DMM_Go_CF	
		U_CF		DMM_CF_Go		DMM_CF_So
		U_So			DMM_So_CF	

DMM_So_CF			Core Functions	
			CF	
			Engine Specification	Chassis Specification
Solutions	So	4 cyl combustion motor	x	
		2 cyl combustion motor	x	
		Spaceframe		x
		Monocoque		x

Figure 4-2 Definition and buildup of the Meta-GIA-MDM based on several MDMs representing an entire product planning system

Transferring the exemplary PPS of a racecar accordingly the systematic of a Meta-GIA-MDM representation, it results in a matrix which is presented in Figure 4-3. As can be seen the matrix has the size of 35 per 35 and is a weakly filled symmetric matrix. Since a relation, indicated by an 'X' in the according field, represents a general, unbiased connection between two elements, the matrix is filled completely symmetric. Additionally it can be seen that elements are mainly linked within their lifecycle phases or to phase-neutral parameters, resulting in many marked fields close to the diagonal line, on the right side, and on the lower side of the matrix. However, this matrix notation allows a complete representation of a product planning system which can serve as a basis for a systematical goal interrelation analysis.

Meta-GIPS-MDM				1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35
Project Planning	Demands	Budget Acquisition for Tufast	1	▨		X																																
		Define Resources	2		▨		X																															
	Goals	Best Budget for one Year	3	X		▨		X																														
		Best HR Ratio Between Divisions	4		X		▨		X																													
	Core Functions	Budget Acquisition	5			X		▨		X	X																											
		Resource Planning	6				X		▨			X	X																									
	Solutions	Sponsor ACDEF	7					X		▨																										X		
		Sponsor B	8					X			▨																									X		
		Strong Chassis Division	9						X			▨																						X			X	X
		Equal Distribution	10						X				▨																					X			X	X
Production	Demands	Production of Motor	11											▨		X																						
		Production of Chassis	12												▨		X																					
	Goals	Best Solution for Motor Production	13											X		▨		X																				
		Best Solution for Chassis Production	14												X		▨		X																			
	Core Functions	Motor Production Specification	15													X		▨		X		X																
		Chassis Production Specification	16														X		▨		X		X															
	Solutions	Tufast Garage for Motor	17															X		▨																X	X	
		Tufast Garage for Chassis	18																X		▨													X	X	X		
		External Production for Motor	19															X				▨														X	X	
		External Production for Chassis	20																X				▨											X	X	X		
Utilization	Demands	Engine for Racecar	21																					▨		X												
		Chassis for Racecar	22																						▨		X											
	Goals	Best Engine	23																					X		▨		X										
		Best Chassis	24																						X		▨		X									
	Core Functions	Engine Specification	25																							X		▨		X	X							
		Chassis Specification	26																								X		▨			X	X					
	Solutions	4 cyl Combustion Motor	27																									X		▨						X	X	
		2 cyl Combustion Motor	28																									X			▨					X	X	
		Spaceframe	29																										X			▨		X	X	X		
		Carbon Monocoque	30																										X				▨	X	X	X		
Global Parameters		ChassisPersonDays	31									X	X								X		X									X	X	▨				
		ProductionQuality	32																		X		X									X	X		▨			
		Budget	33							X	X									X	X	X	X							X	X	X	X			▨		
		MotorPersonDays	34									X	X							X		X								X	X						▨	
		AdminPersonDays	35									X	X																									▨

Figure 4-3 Filled Meta-GIA-MDM for the development of a racecar

4.2.1.3 Goal Interrelation Analysis

Analyzing how goals of the product planning are connected with each other is the main aim of this implementation. Since *Loomeo* does not support the GIA natively, integrated methods have to be adapted and used in order achieve results. Nevertheless, a general analysis of interrelations can be performed directly in the matrix. Thus, the manual analysis is presented, followed by an introduction into the path analysis provided by *Loomeo* which can support the identification of interrelations. However, the conflict detection with MDM is very limited, since no values or different types of relations can be stored within the matrix.

Meta-GIPS-MDM				1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35
Project Planning	Demands	Budget Acquisition for Tufast	1			X																																
		Define Resources	2				X																															
	Goals	Best Budget for one Year	3	X				X																														
		Best HR Ratio Between Divisions	4		X				X																													
	Core Functions	Budget Acquisition	5			X				X	X																											
		Resource Planning	6				X					X	X																									
	Solutions	Sponsor ACDEF	7					X																												X		
		Sponsor B	8					X																												X		
		Strong Chassis Division	9						X																									X			X	X
		Equal Distribution	10						X																									X			X	X
Production	Demands	Production of Motor	11													X																						
		Production of Chassis	12														X																					
	Goals	Best Solution for Motor Production	13											X				X																				
		Best Solution for Chassis Production	14												X				X																			
	Core Functions	Motor Production Specification	15													X				X		X																
		Chassis Production Specification	16														X				X		X															
	Solutions	Tufast Garage for Motor	17															X																		X	X	
		Tufast Garage for Chassis	18																X															X	X	X		
		External Production for Motor	19															X																		X	X	
		External Production for Chassis	20																X															X	X	X		
Utilization	Demands	Engine for Racecar	21																							X												
		Chassis for Racecar	22																								X											
	Goals	Best Engine	23																					X				X										
		Best Chassis	24																						X				X									
	Core Functions	Engine Specification	25																							X				X	X							
		Chassis Specification	26																								X					X	X					
	Solutions	4 cyl Combustion Motor	27																									X								X	X	
		2 cyl Combustion Motor	28																									X								X	X	
		Spaceframe	29																										X					X	X	X		
		Carbon Monocoque	30																										X					X	X	X		
Global Parameters		ChassisPersonDays	31									X	X								X		X									X	X					
		ProductionQuality	32																		X		X									X	X					
		Budget	33							X	X									X	X	X	X							X	X	X	X					
		MotorPersonDays	34									X	X							X		X								X	X							
		AdminPersonDays	35									X	X																									

Figure 4-4 Meta-GIA-MDM-Based goal interrelation analysis

Firstly, the procedure of a manual identification of goal interrelations is given. An example for it can be seen in Figure 4-4, which shows the identification of all goals which are connected to the parameter 'ChassisPersonDays'. Therefore the parameter is found in the line 31. Taking the corresponding column, all solutions which are connected to the parameter can be identified, being element 9, 10, 18, 20, 29, and 30. Starting from the parameter's column, one can go left in any of these lines from the solution, to find the next corresponding 'X'. Based on this result, it is possible to go up in the matrix, to the next 'X' which directly indicates the corresponding goal. Consequently it can be identified, that the goals ‘Best HR Ration Between Divisions’, ‘Best Solutions for Chassis Production’, and ‘Best Chassis’ are interrelated through the parameter ‘ChassisPersonDays’. However this way of identifying goal interrelations may be possible for small systems, but with an increased number of system elements it becomes too complex. Consequently a computer supported analysis is requested

Moreover, the path analysis provided by *Loomeo* can be used to automatically identify all paths between two elements. Therefore the start and the end nodes have to be selected and the software tool returns a list with all possible paths and their lengths. The results can further be shown in the graph or the matrix representation of the system, using different colors to highlight them. Thus, for the identification of relations between two goals, these have to be selected as start and end node. The path has to follow further elements from the domain ‘Core Functions’, ‘Solutions’ and ‘Parameters. In order to gain valid results, it is important that there are no closed loops and no element occurs several times in the path. An example of the result presentation for the path analysis between two goals of a product planning system can be seen in Figure 4-5. In the Figure a matrix and a graph representation of the PPS can be seen. The graph system is aligned automatically according to the connection of the elements, so that five branches can be identified with the demands as the leaf elements and the density of their interrelations to each other. Thus the goal ‘Best Chassis’ and ‘Best Solution for Chassis Production’ as well as ‘Best Engine’ and ‘Best Solution for Engine Production’ can be recognized as highly linked. Additionally the result of the path analysis between the goal ‘Best Budged for one Year’ and ‘Best Solution for Engine Production’ is highlighted in the colors, green, yellow, orange, and red. As can be seen there are twelve different paths to connect the two goals, all passing the parameter ‘budget’. Also in the matrix representation the results are highlighted, although they are much harder to understand. Anyway with this method all possible connections between two goals can be identified, serving for a further analysis of their conflicts.

Based on these results it necessary to compare the values through which both goals are connected to the parameter in order to identify conflicts between two solutions. The problem with this is that these values are not represented directly in the Meta-GIA-MDM. This is theoretically possible since an additional domain with values could be added, but practically not manageable, since all qualitative and quantitative values require an own entry. Consequently values have to be retrieved from an external source. These values can further be confronted on the identified paths leading to valid or incompatible combinations of goals.

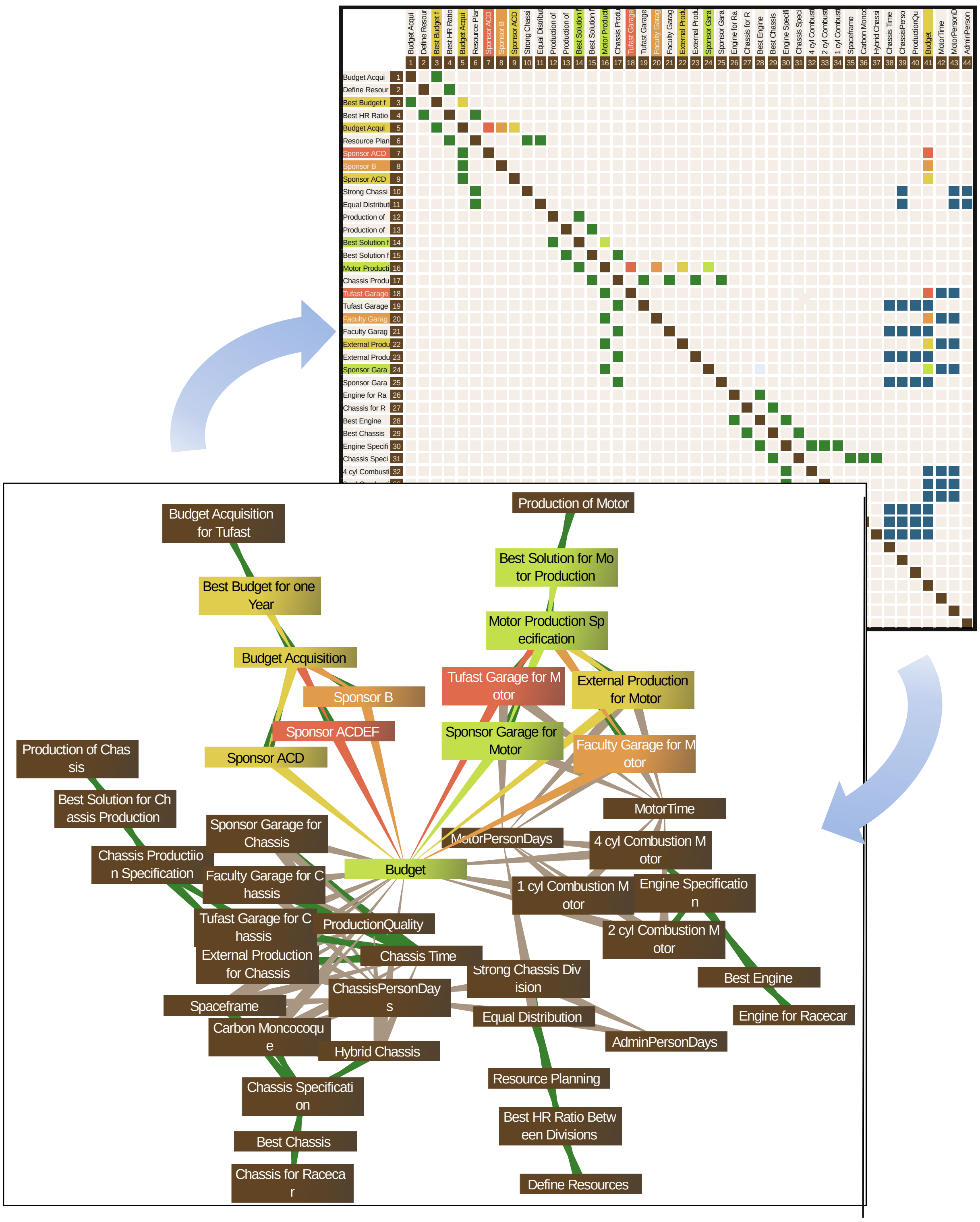


Figure 4-5 Graphical representation of the results for a path-analysis between two goals by Loomeo

4.2.2 Graph Grammar

As the second established computer-supported modeling technique, graph grammars are regarded. With graph grammars, it is possible to describe a system and its behavior based on the usage of graphs and underlying rules (see chapter 2.3.4.2 for further details). These representations can be used for further computational analyses and therefore supports also the basic requirements for the implementation of a computer-support goal interrelation analysis. The implementation is presented by firstly introducing into the used software, *booggie*, followed by an approach for the representation of a product planning system based on graph grammars and the potentials for application of a goal interrelation analysis.

4.2.2.1 Introduction into the software tool *booggie*

In order to get a general overview on the strength and weaknesses of graph grammars in representing systems and to be an applicable technique for a goal interrelation analysis, these are tested with the software tool *booggie* (http://www.boggie.org). *booggie* is currently developed by the Virtual Product Development Group at the Technische Universität München (HELMS & SHEA 2010). *booggie* is an abbreviation for ‘bring object oriented graph grammar into engineering’ and thus a graph definition and transformation tool especially designed for engineering purposes. In its core *booggie* combines two fully developed tools, the GrGen.NET of the University Karlsruhe which provides the graph transformation engine and Tulip which is used for the visualization (JAKUMEIT ET AL. 2010; AUBER 2003). *Booggie* allows the definition of elements in meta models, as well as rules and rule sets for the graph transformation. Additionally it supports the definition of ports in order to model specific connections between elements and attributes to add extra information to the elements. With these features, it enables all main requests for the implementation of a goal interrelation analysis. The generic process of working with *booggie* begins with the definition of the elements with their ports, followed by the creation of rules and rule sequences which are finally applied to achieve the requested results.

The first step towards representing a system with *booggie* is the definitions of all elements of the system, creating a metamodel of the system. Therefore, the fundamental elements are created by naming them and grouping them hierarchically. Additionally ports, which are the points to connect two elements and thus describing their kind of connections, can be defined and hierarchically sorted. These ports can be added to the elements, representing what kind of connection an element can handle. Elements can be changed in in their appearance, meaning their shape, color or texture. Moreover elements can be described by additional attributes, which are defined by the type, e.g. integer, string, etc., and their value. All these properties of an element, ports, appearances and attributes, are inherited by elements of a hierarchical lower level, but can be changed in the values individually. Summarizing, with elements, having specific ports and attributes it is possible to generate a metamodel, which provides all basic elements necessary to create a holistic representation of a system by graphs.

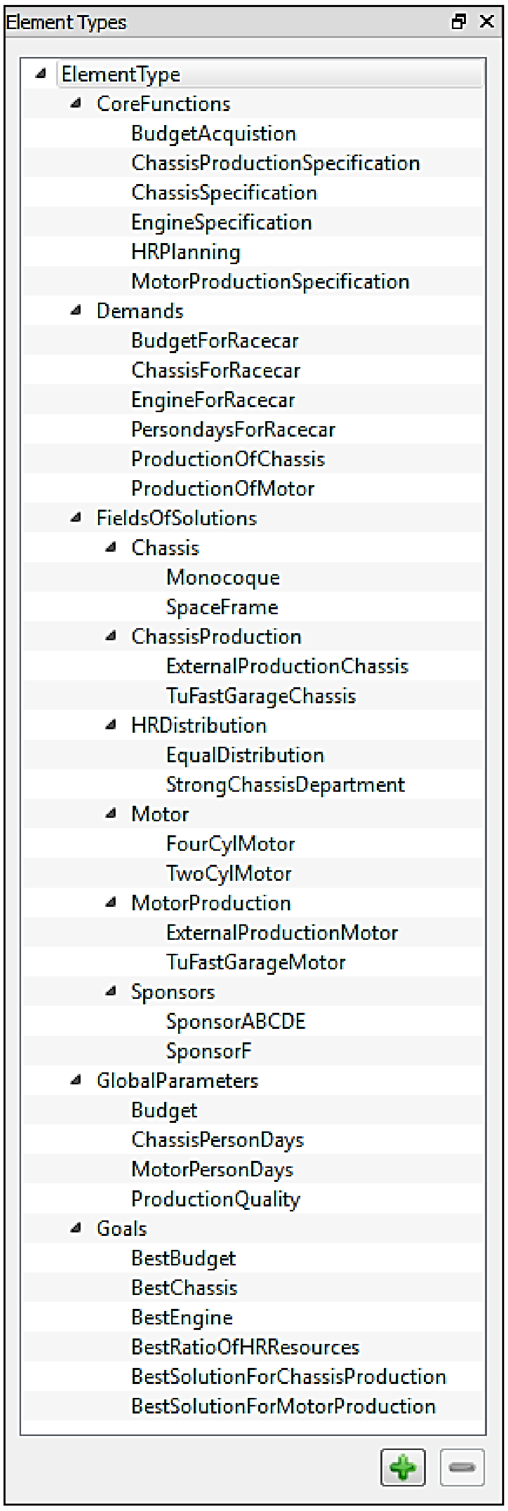
The next step is to create a graph based representation of the system. Therefore a new space for the graph representation has to be created, where all elements, previously defined in the metamodel, can be dragged into. These elements can be arranged freely in the three-dimensional space and are drawn there according to their defined appearance and with all attached ports. Further ports can be connected with each other, representing linked elements. While drawing these connections, *booggie* monitors if they are valid in their directions and types of linked ports are compatible. With elements being specific nodes and the connections between the ports being named edges, a graphical representation of a system based on the graph theory can be provided.

Finally the creation of rules and their application on the previously defined graph representation of a system is the last fundamental feature of *booggie*. The rule system provided by the software is very powerful. Rules can be created using the syntax of GrGen.Net or Phyton. GrGen.Net offers a declarative language for graph modeling, pattern matching, and rewriting, as well as rule control (BLOMER ET AL. 2011, p. 2), whose main feature are introduced in chapter 2.3.4.2. Phyton furthermore, is a powerful object-oriented programming language (WEIGEND 2010, p. 19). Consequently, with the support of GrGen.Net and additionally Phyton, the rules implemented in *booggie* can handle almost all circumstances. Additionally several rules can be connected to rule sequences, making the easily repeatable. Finally these rules and rule sequences can be applied directly upon a selected graph representation of a selected system. With all the described features *booggie* offers a broad basis for the implementation of a GIA as well as many possibilities for extension through to its rule system.

4.2.2.2 Implementation of a Product Planning System

The implementation of a PPS with *booggie* is the first step towards performing a GIA based on graph grammars. As explained in the previous section, all elements within the product planning system have to be defined in a metamodel, with their hierarchical groups, their ports and their attributes. This complete metamodel is further being used to build a graph-based representation of a product planning system.

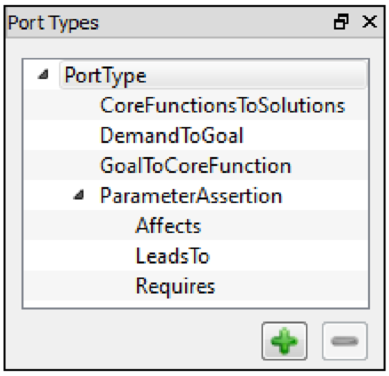
The definition of a metamodel begins with the generation of a list of all basic elements and ports. A list of elements for the exemplary development of a race can be seen in Figure 4-6 ordered hierarchically. Assembling the list hierarchically helps generating a consistent metamodel since properties of top-level elements are automatically inherited by elements of a lower level. In the figure the five groups of basic elements, according to the approach presented in chapter 2.2.2, can be seen: ‘CoreFunctions’, ‘Demands’, ‘FieldsOfSolutions’, ‘GlobalParameters’, and ‘Goals’. The ‘FieldsOfSolutions’ are further subdivided into the included solutions of the field. For example the solution ‘ExternalProductionChassis’ is part of the field ‘ChassisProduction’ which is classified as a ‘Field of Solution’. Figure 4-7 shows the list of defined port types. Accordingly, the port ‘CoreFunctionsToSolutions’ is implemented to connect core functions and solutions, the port ’DemandToGoal’ for demands and goals, and the port ‘GoalToCoreFunction’ for goals and core functions. The ports to link solutions to their parameters are called ‘ParameterAssertion’ and are additionally subdivided into the three port types describing how the solution influences the parameter; if it ‘Affects’ the value, ‘LeadsTo’ a fix value, or ‘Requires’ a specific value. These elements and ports are combined with attributes in a next step in order to achieve the definition of all elements within the metamodel.



Element Types

- ElementType
  - CoreFunctions
    - BudgetAcquistion
    - ChassisProductionSpecification
    - ChassisSpecification
    - EngineSpecification
    - HRPlanning
    - MotorProductionSpecification
  - Demands
    - BudgetForRacecar
    - ChassisForRacecar
    - EngineForRacecar
    - PersondaysForRacecar
    - ProductionOfChassis
    - ProductionOfMotor
  - FieldsOfSolutions
    - Chassis
      - Monocoque
      - SpaceFrame
    - ChassisProduction
      - ExternalProductionChassis
      - TuFastGarageChassis
    - HRDistribution
      - EqualDistribution
      - StrongChassisDepartment
    - Motor
      - FourCylMotor
      - TwoCylMotor
    - MotorProduction
      - ExternalProductionMotor
      - TuFastGarageMotor
    - Sponsors
      - SponsorABCDE
      - SponsorF
  - GlobalParameters
    - Budget
    - ChassisPersonDays
    - MotorPersonDays
    - ProductionQuality
  - Goals
    - BestBudget
    - BestChassis
    - BestEngine
    - BestRatioOfHRResources
    - BestSolutionForChassisProduction
    - BestSolutionForMotorProduction

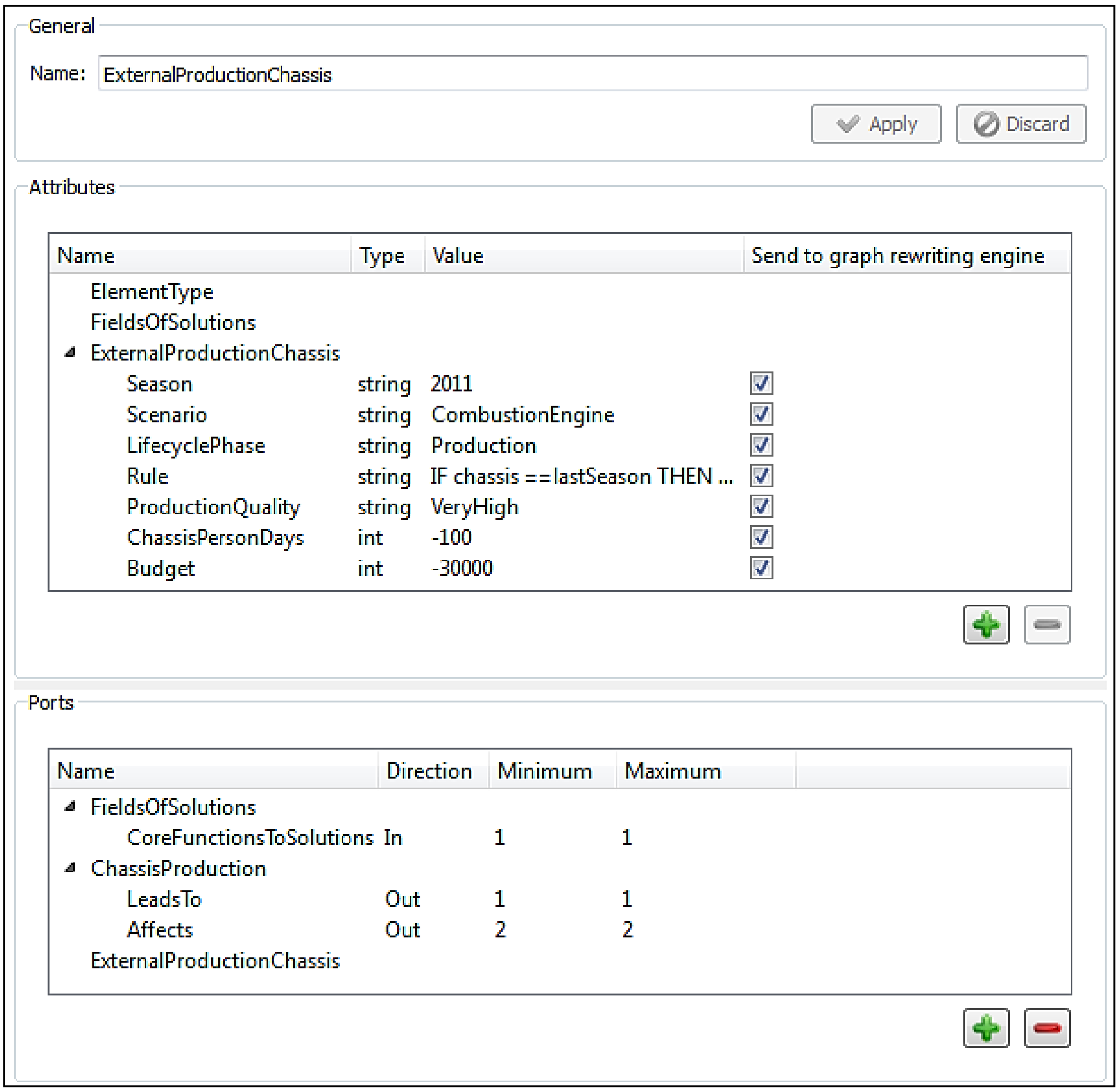
Figure 4-6 Hierarchical implementation of the metamodel with its elements for the development of a racecar


Port Types

- PortType
  - CoreFunctionsToSolutions
  - DemandToGoal
  - GoalToCoreFunction
  - ParameterAssertion
    - Affects
    - LeadsTo
    - Requires

Figure 4-7 Definition of ports used in the product planning system

Furthermore all elements are being defined more detailed by adding ports and extra information through attributes to them. An exemplary definition for the element ‘ExternalProductionChassis’ can be seen in Figure 4-8. There, it is shown that all auxiliary information for the elements can be added in the attributes. Thus, the element belongs to the car developed in season 2011, for the scenario of a combustion engine. It also belongs to the lifecycle phase ‘Production’ and has an included rule, which occurs if the chassis is the same as last season. Further, the three parameters ‘ProductionQuality’, ‘ChassisPersonDays’, and ‘Budget’ are assigned to a specific value for the solution. All these attributes are inherited from the top level elements and changed in their value according to the element, so a consistency in attributes is supported. On the lower part of the figure the ports added to the element can be seen. Therefore ‘ExternelProductionChassis’ inherits the port to connect to core function from the generic element ‘FieldsOfSolutions’ and two ports which are affecting an parameter as well as one port leading to a value of parameter, inherited from the field of solution ‘ChassisProduction’. Consequently the element is completely defined in all its properties and possible types of connections.



General

Name: ExternalProductionChassis

Attributes

Name	Type	Value	Send to graph rewriting engine
ElementType			
FieldsOfSolutions			
ExternalProductionChassis			
Season	string	2011	✓
Scenario	string	CombustionEngine	✓
LifecyclePhase	string	Production	✓
Rule	string	IF chassis ==lastSeason THEN ...	✓
ProductionQuality	string	VeryHigh	✓
ChassisPersonDays	int	-100	✓
Budget	int	-30000	✓

Ports

Name	Direction	Minimum	Maximum
FieldsOfSolutions			
CoreFunctionsToSolutions	In	1	1
ChassisProduction			
LeadsTo	Out	1	1
Affects	Out	2	2
ExternalProductionChassis			

Figure 4-8 Definition of an exemplary element with attributes and ports

Finally the graph representation of product planning system can be created based on the previously defined meta model. Figure 4-9 shows the representation of a planning system for a racecar. For a better understanding of the whole scheme, the elements are shaped differently and the ports have different colors. Thus the four global parameters ‘MotorPersonDays’, ‘Budget’, ChassisPersonDays’, and ‘ProductionQuality’ can be seen in the middle of the figure shaped like a hexagon, colored light grey. Additionally they have three differently colored ports, making them linkable to solutions which affects (green), requires (yellow), or leads to (orange) a value. Furthermore the demands are shaped like triangles, the goals like ellipses, the core function like squares, and the solutions like circles. Consequently a complete chain from the demand to the parameter can be generated. For example the demand ‘ProductionOfChassis’ is connected through the goal ‘BestSolutionForChassisProduction’ and the core function 'ChassisProductionSpecification' to the two solutions 'ExternalProductionChassis' and '*TUfast*GarageChassis'. Additionally it can be seen that the two solutions require a value from the parameter 'ProductionQuality', and affect the values of the parameters 'Budget' and 'ChassisPersonDays'. Summarizing, with the graph representation of how the elements are linked and the attributes defined in the metamodel, a holistic representation of a product planning system can be provided.

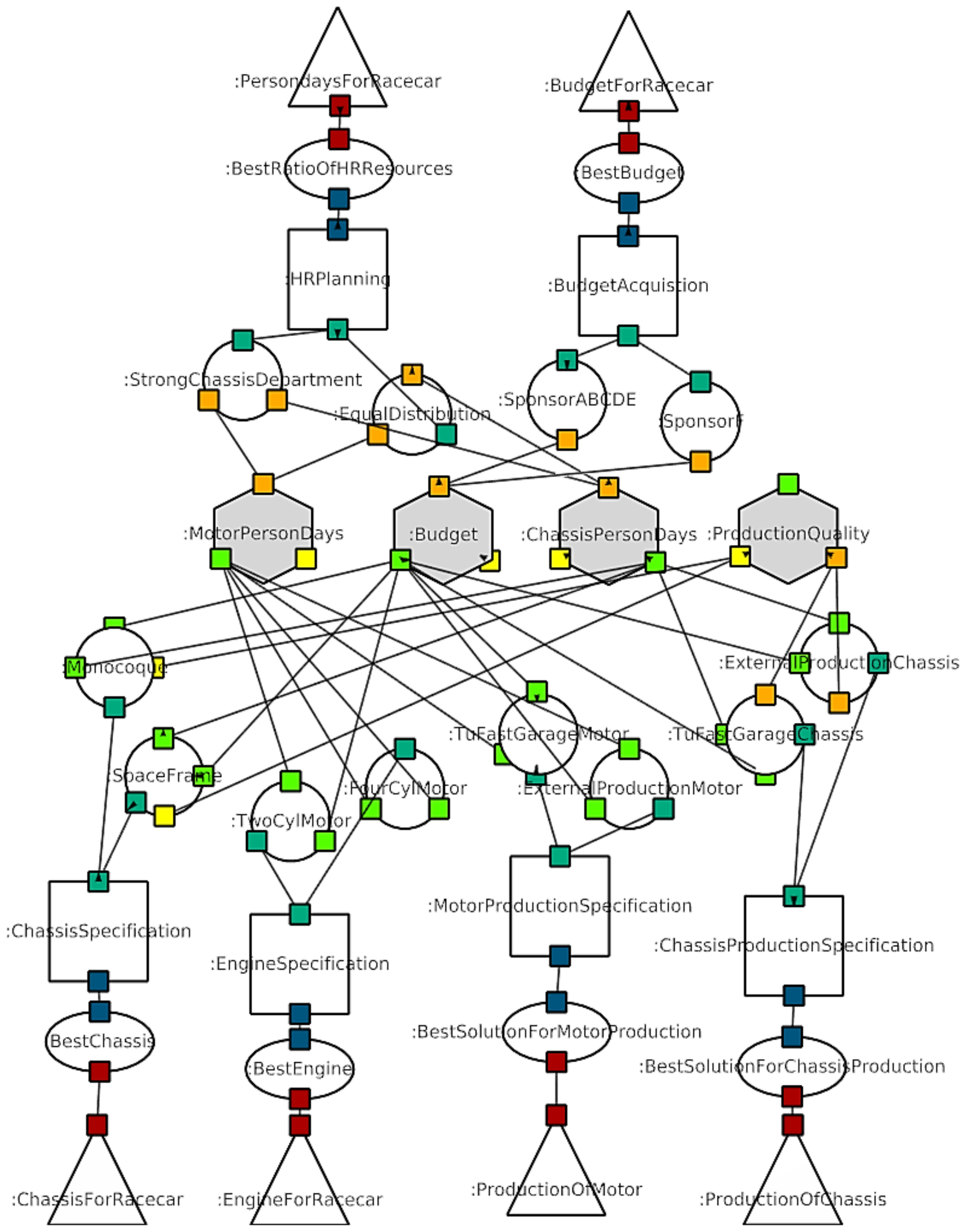


Figure 4-9 Graph-based representation of a product planning system for a racecar

4.2.2.3 Goal Interrelation Analysis

In general, it is not possible to perform a GIA or a part of it directly within *booggie*, since the software does not provide any fundamental functionality for the analysis of the implemented system representations. Nevertheless with the support of user defined rules, based on GrGen.Net or Phyton, an interface for a GIA is provided. Since it is not part of this thesis to provide an implementation of a GIA, just the main functions of the required rules are discussed.

A GIA consists of several analysis steps, performed upon the representation of a product planning system. The analysis is for the most part implementable directly in the graph grammar language, GrGen.Net, but moreover into the programming language Phyton, which both are natively supported by *booggie*. To begin with, the relevant goals have to be identified. This can be done, since all goals are sub elements of the generic element goal in the hierarchical list of elements. Additionally they can be identified by the attributes, describing lifecycle phase, season and scenario. Starting from the selected goal, the connected core functions can be found through the port ‘GoalToCoreFunction’ and the solutions for the goal through the port ‘CoreFunctionsToSolutions’. To check whether two goals are interrelated, these two are needed to be chosen and the connection through core function, solutions and assigned parameters are checked. If there is an intersection between the parameters, a possible interrelation between two goals has been detected. Furthermore this connection chain has to be investigated by the various types and values of the parameters assignments, if they are compatible among each other. For example the goal ‘Best Chassis’ and ‘BestSolutionForChassisProduction’ are connected through the three parameters ‘Budget’, ‘ChassisPersonDays’, and ‘ProductionQuality’. A conflict e.g. can be detected between the solution ‘Monocoque’ and the solution ‘*TUfast*Garagechassis’ since the second one leads to the parameter ‘ProductionQuality’ with the value ‘High’ while the ‘Monocoque’ requires the value ‘VeryHigh’ for the ‘ProductionQuality’.

Summarizing it can be said that a GIA is not yet performable with *booggie*, but with its support to GrGen.Net and Phyton and the ability to handle all necessary information from the product planning system in the included graph representation, it offers all the required possibilities to implement it. Thus it can be detected whether two goals are interrelated and if their solutions have a conflict. Nevertheless also a global detection of compatible solutions and their connected goals is possible, supporting the strategic product planning process fundamentally.

4.2.3 Relational Databases

Relational databases are the third computer-supported modeling technique, which is investigated in this thesis. Relational databases are designed to handle a large amount of information through ordering them in tables and to derive further knowledge from it (see chapter 2.3.4.3 for more details). Since a system representation is a collection of information about the system, it can be also represented by a relational database, upon which a deeper analysis can occur. Therefore an implementation of a goal interrelation analysis is generally possible. The application is presented by firstly introducing into the used software tools, followed by an approach for the implementation of a product planning system based on tables and the potentials for the goal interrelation analysis.

4.2.3.1 Introduction of the software tool *Access*

The application of relational databases for representing a product planning system and a GIA is tested with the software tool *Access* in order to get a general overview on the strength and weaknesses of the technique. *Access* is a database-management program that enables the maintenance of data arranged according to a fixed structure in databases. It is part of the widely spread and in several generations improved Microsoft Office suite, resulting in a good usability of the interface and a good compatibility and support of the program. With *Access* it is possible to create and maintain databases as well as selecting, sorting and displaying the information stored in a database (MICROSOFT CORPORATION 2011). The implementation of a database is generally done by specifying the tables and their connections, followed by filling them with data directly in the tables or through a freely designable interface. Is the information acquired, it can be accessed and analyzed through the query system (YOUNG ET AL. 2010, pp. 9ff).

Building a representation of a system with *Access* begins with the definition of the tables in the database and their interrelations among each other. The tables are defined by their names, their fields and a primary key. The fields are furthermore named individually and assigned to a specific data type like text or integer. Nevertheless these data types can also be references to other tables, allowing a field of one table can be directly connected to another table. Consequently referred fields can only have values previously added to the referred table, leading to a better consistency of the database representation. Additionally the primary key has to be selected. It is the key for the unique identification of every single dataset of a table. Hence, the key can be a special identification number (ID) or fields, where all datasets are individual, like an unique element name. With the definition of all tables and their interrelations the data of a system can be acquired to build a holistic representation of the system, being the basis for further analysis.

The acquisition of information in *Access* can occur either directly in the tables, through a freely adaptable interface or imported from external sources. When entering the information directly in the tables, each row represents a dataset whereas the primary key has to be unique. The adaptable interfaces can be used to enter information to several tables by merging them in one input screen, which eases the handling of a large amount of interrelated tables. Additionally *Access* supports a direct import of information from various sources and formats, like text-files, Excel tables, or external databases. The acquired information can now be used as the basis for further analysis and queries.

Queries are a way to receive requested information from the acquired information in the database. Queries can process data from related tables, build subsets of them according to specific criteria, sort and alphabetize data and create new calculated fields (YOUNG ET AL. 2010, pp. 9ff). In *Access*, queries can be created by a supporting assistant system, by a dedicated interface, or by a SQL implementation. These can search the tables of the databases, join them, order them, or perform more complicated operations with values from them. The results are then returned also as tables, which can change dynamically with changing data in the database. The queries are designed to process information very effectively and fast, why they are especially helpful with large database. With all the described features *Access* offers a good basis for the implementation of a GIA, which has its special strength upon the good compatibility, the mature software, the good usability and its special design to process a large amount of data.

4.2.3.2 Implementation of a Product Planning System

The implementation of a product planning with *Access* is the first step towards performing a GIA based on a relational database. As explained in the previous section, in the beginning, the tables with their fields, the primary keys and their interrelations need to be defined. Thus, a basic representation of a product planning system is created which can be filled with the information about the exemplary development of a racecar.

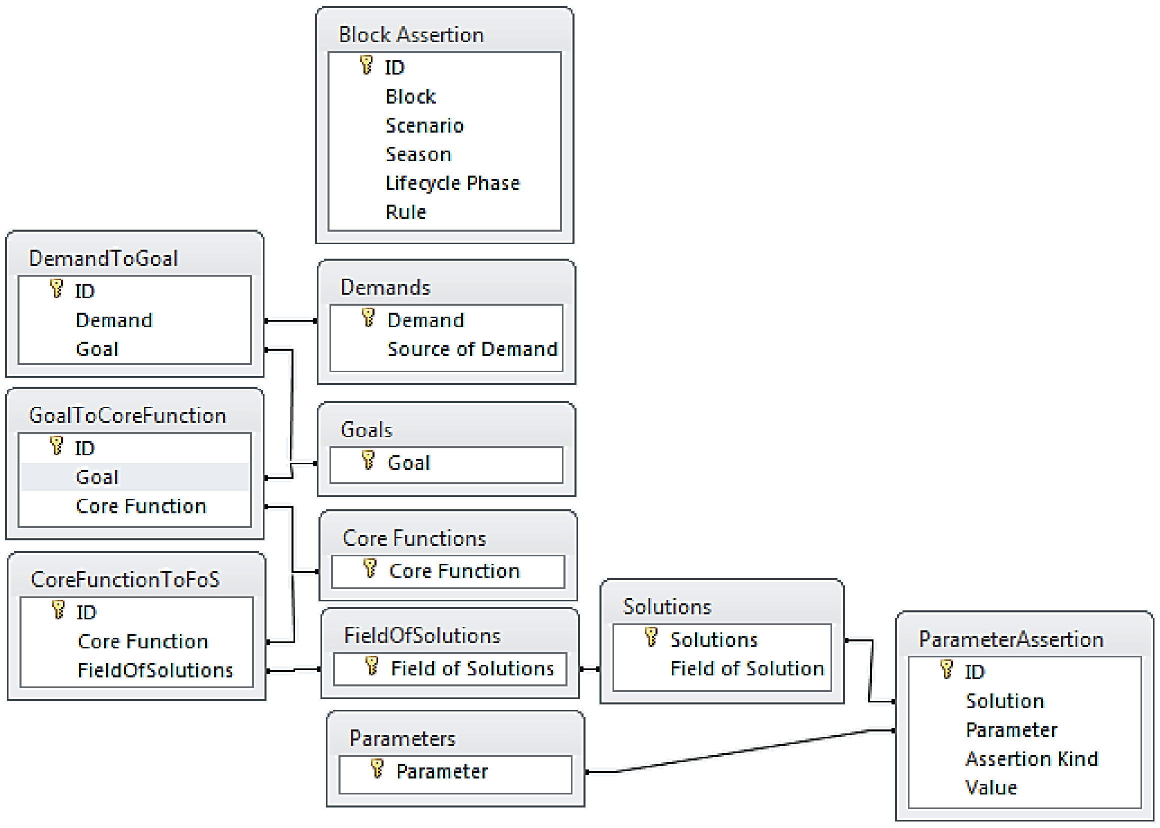
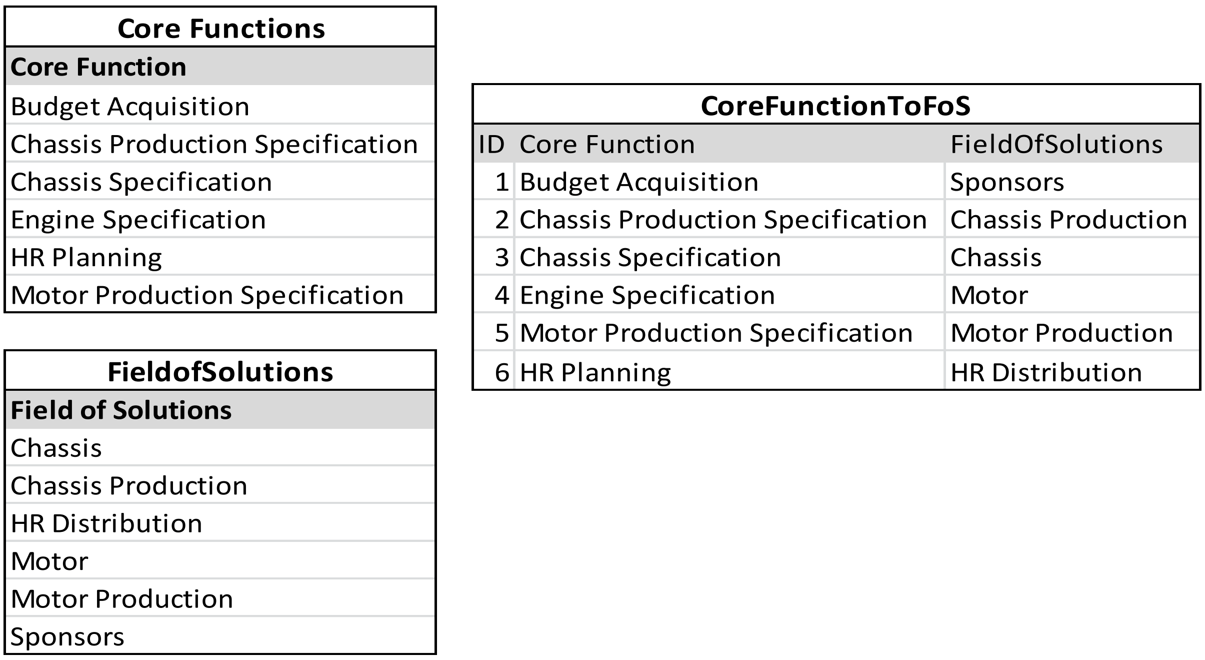


Figure 4-10 Scheme of the relational database for a product planning system

The exemplary implementation of a database scheme for the PPS can be seen in Figure 4-10. In the figure, every block represents a table, where the various fields are listed in the inner box and the primary key is market with a key icon. Additionally it is shown by direct lines, how the fields are connected to each other. Thus, the demands are listed in the table ‘Demands’ together with their sources. Accordingly, goals, core functions, fields of solutions, solutions, and parameters are written in their referring table. Further the table ‘DemandToGoal’, ‘GoalToCoreFunction’, and ‘CoreFunctionToFoS’ are showing how the particular elements are connected among each other. The table ‘ParameterAssertion’ describes how solutions are linked to their parameters and the value for the connections. Finally the table ‘BlockAssertion’ is defined in order to add further properties to all basic elements, demands, goals, core functions, and solutions, the scenarios, the seasons, the lifecycle phases and an enabling rules. Having presented the complete database scheme of a generic PPS, the information about the development of a racecar can be added in order to receive a representation of the example.



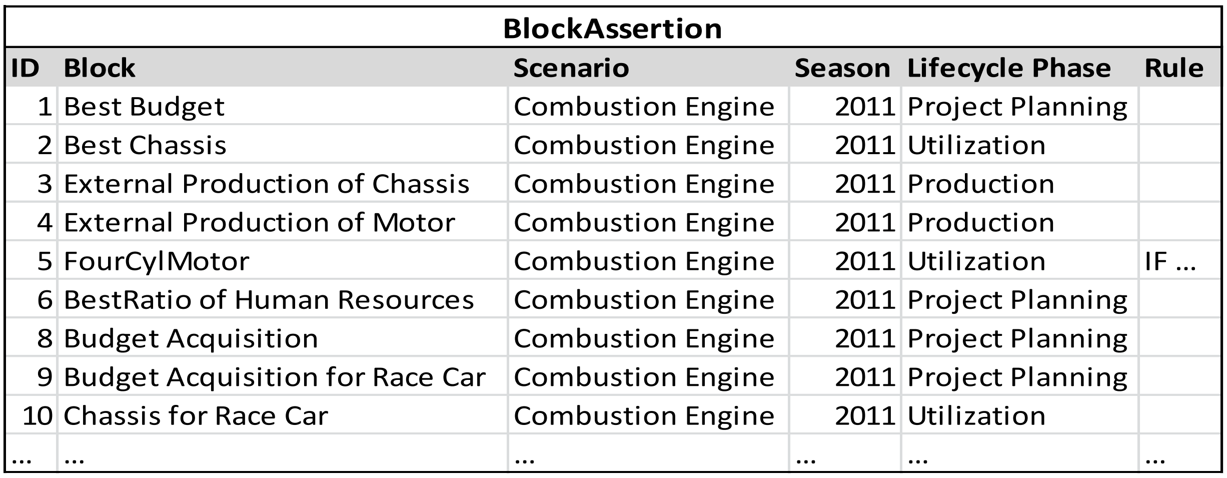
Core Functions
Core Function
Budget Acquisition
Chassis Production Specification
Chassis Specification
Engine Specification
HR Planning
Motor Production Specification

FieldofSolutions
Field of Solutions
Chassis
Chassis Production
HR Distribution
Motor
Motor Production
Sponsors

CoreFunctionToFoS		
ID	Core Function	FieldOfSolutions
1	Budget Acquisition	Sponsors
2	Chassis Production Specification	Chassis Production
3	Chassis Specification	Chassis
4	Engine Specification	Motor
5	Motor Production Specification	Motor Production
6	HR Planning	HR Distribution

Figure 4-11 Tables 'CoreFunctions', 'FieldofSolutions', and 'CoreFunctionsToFoS'

All the presented tables have to be filled with datasets as a next step in order to achieve a complete representation of the PPS of the example. In this implementation the datasets have been entered directly into the tables. An example for filled tables can be seen in Figure 4-11. In the figure, the tables ‘Core Functions’ and ‘FieldofSolutions’ can be seen, which are both filled with the six core functions or fields of solution from the exemplary PPS. Further the table ‘CoreFunctionToFoS’ lists which core function is connected to which field of solutions. For example under the field ‘ID’ with the value six, it is shown that the core function ‘HR Planning’ is connected to the field of solution ‘HR Distribution’. Both, the field ‘CoreFunction’ and ‘FieldOFSolutions’ are directly linked to the referring tables. Hence, the table ‘CoreFunctionToFoS’ allows only dataset entries which are already defined in the other tables. Another more complex table can be seen in Figure 4-12 in an extract. The table ‘BockAssertion’ adds further information to all elements. Thus the field ‘Block’ is connected through a query to all elements from the tables ‘Demands’, ‘Goals’, ‘CoreFunctions’, ‘FieldsOfSolutions’, and ‘Solutions’. These elements can be selected and information about their scenario, season, lifecycle phase, and rule is added. For example under ‘ID’ five the block ‘FourCylMotor’ is assigned to the scenario ‘Combustion Engine’, the ‘Season’ is 2011 and the ‘Lifecycle Phase’ is ‘Utilization’. Finally the field ‘Rule’ contains a written rule which checks certain conditions followed by the specified consequences.

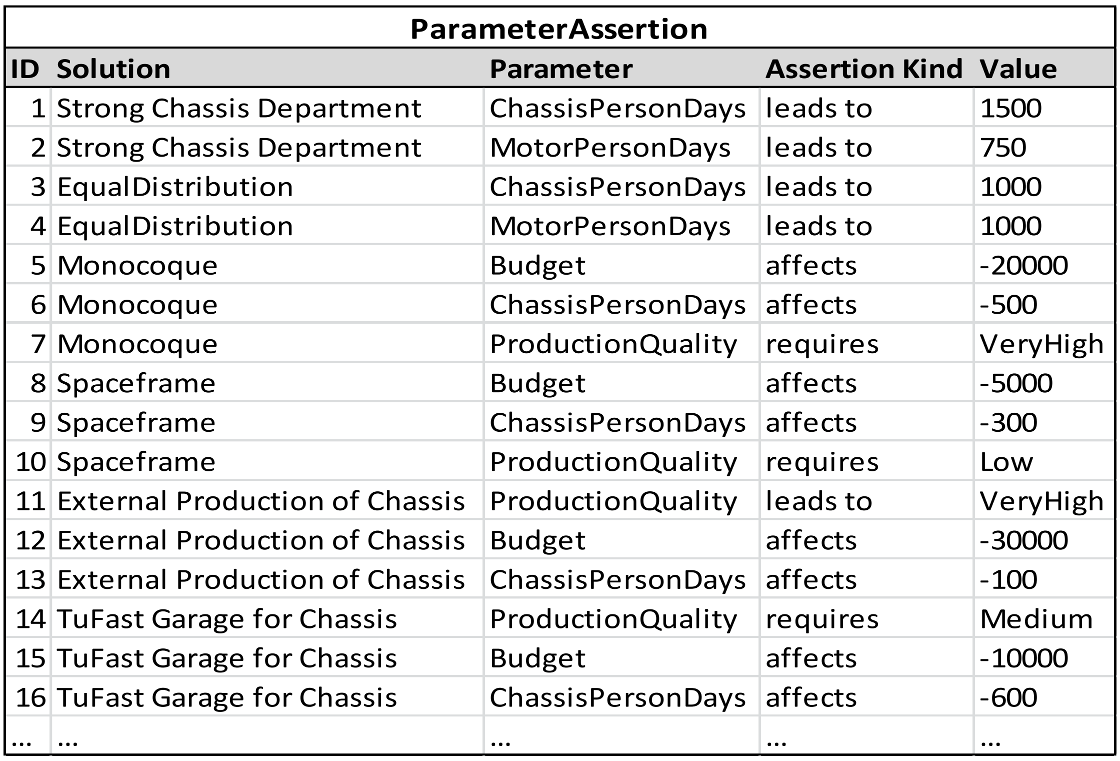


BlockAssertion

ID	Block	Scenario	Season	Lifecycle Phase	Rule
1	Best Budget	Combustion Engine	2011	Project Planning	
2	Best Chassis	Combustion Engine	2011	Utilization	
3	External Production of Chassis	Combustion Engine	2011	Production	
4	External Production of Motor	Combustion Engine	2011	Production	
5	FourCylMotor	Combustion Engine	2011	Utilization	IF ...
6	BestRatio of Human Resources	Combustion Engine	2011	Project Planning	
8	Budget Acquisition	Combustion Engine	2011	Project Planning	
9	Budget Acquisition for Race Car	Combustion Engine	2011	Project Planning	
10	Chassis for Race Car	Combustion Engine	2011	Utilization	
...	...	...	...	...	...

Figure 4-12 Extract from the table 'BlockAssertion'

Furthermore, Figure 4-13 shows an extract of the most relevant table for the GIA, the ‘ParameterAssertion’. There, it is described which solution is connected to which parameter, by what kind of relation and the value of it. In the table each connection is described in a single dataset, so a solution can occur various times in the table. The elements for the solutions and the parameter fields are directly received from the according table, while the ‘Assertion Kind’ can be either ‘leads to’, ‘affects’, or ‘requires. For example the solution ‘Spaceframe’ can be seen in the table occurring three times. Once it is connected to the parameter ‘Budget’ through the relation ‘affect’ and the value ‘-5000’. Additionally it ‘affects’ the parameter ‘ChassisPersonDays’ by the value ‘-300’ and it requires a ‘Productionquality’ which is at least low. Consequently with the presented scheme the whole PPS for the racecar is modeled and can serve as a basis for the application of the GIA.



ParameterAssertion				
ID	**Solution**	**Parameter**	**Assertion Kind**	**Value**
1	Strong Chassis Department	ChassisPersonDays	leads to	1500
2	Strong Chassis Department	MotorPersonDays	leads to	750
3	EqualDistribution	ChassisPersonDays	leads to	1000
4	EqualDistribution	MotorPersonDays	leads to	1000
5	Monocoque	Budget	affects	-20000
6	Monocoque	ChassisPersonDays	affects	-500
7	Monocoque	ProductionQuality	requires	VeryHigh
8	Spaceframe	Budget	affects	-5000
9	Spaceframe	ChassisPersonDays	affects	-300
10	Spaceframe	ProductionQuality	requires	Low
11	External Production of Chassis	ProductionQuality	leads to	VeryHigh
12	External Production of Chassis	Budget	affects	-30000
13	External Production of Chassis	ChassisPersonDays	affects	-100
14	TuFast Garage for Chassis	ProductionQuality	requires	Medium
15	TuFast Garage for Chassis	Budget	affects	-10000
16	TuFast Garage for Chassis	ChassisPersonDays	affects	-600
...	...	...	...	...

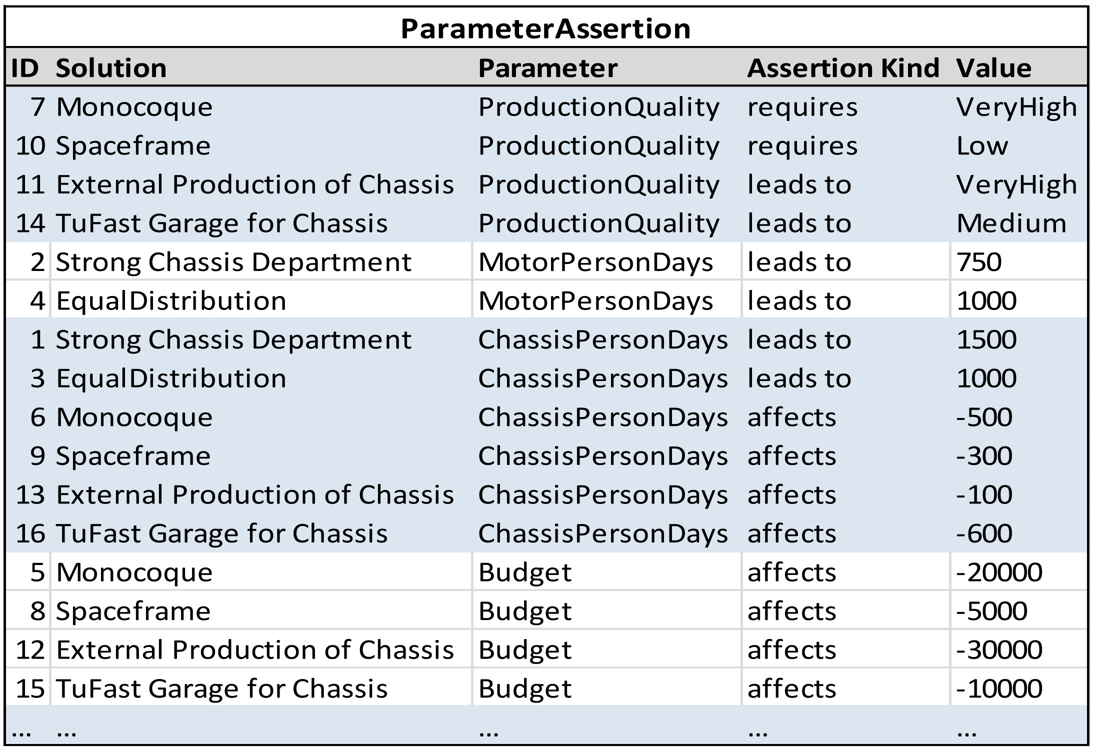
Figure 4-13 Extract from the table 'ParameterAssertion'

4.2.3.3 Goal Interrelation Analysis

Performing a GIA with relational databases can be supported by *Access* mainly through the table representation and the query system. For further support of the GIA, multiple *Application Programming Interfaces* (API) for relational databases are available (SUMATHI & ESAKKIRAJAN 2007, p. 515). These are enabling direct access to the stored data with higher programming languages, which can be used for the implementation of the missing functionalities. Nevertheless provides access with the table representation and the query assistant which can generate SQL queries enough functions for a partially realization of a GIA.

The representation of a system with tables is supporting a GIA in several ways. Therefore, *Access* offers several filters, like masking out specific datasets, and a sorting system for tables which change the order of the datasets. Applying these can change the focus of a view and therefore can help analyzing the represented systems. For example the table in Figure 4-14 is a rearranged version of the table shown in Figure 4-13, which is sorted according to the name of the parameter. In the rearranged table, the solutions which are connected to the same parameter are more clearly identifiable. For example the four solutions 'Monocoque', 'Spaceframe', 'External Production of Chassis', and '*TUfast* Garage for Chassis' are written together, since they are all connected to the parameter 'Production Quality'. In addition, conflicts can be identified in this view more easily, like the solution '*TUfast* Garage for

Chassis' leads to a medium production quality, while the 'Monocoque' requires a very high production quality. Consequently it can be seen that already a simple rearrangement of a table can support a GIA.



ParameterAssertion				
ID	**Solution**	**Parameter**	**Assertion Kind**	**Value**
7	Monocoque	ProductionQuality	requires	VeryHigh
10	Spaceframe	ProductionQuality	requires	Low
11	External Production of Chassis	ProductionQuality	leads to	VeryHigh
14	TuFast Garage for Chassis	ProductionQuality	leads to	Medium
2	Strong Chassis Department	MotorPersonDays	leads to	750
4	EqualDistribution	MotorPersonDays	leads to	1000
1	Strong Chassis Department	ChassisPersonDays	leads to	1500
3	EqualDistribution	ChassisPersonDays	leads to	1000
6	Monocoque	ChassisPersonDays	affects	-500
9	Spaceframe	ChassisPersonDays	affects	-300
13	External Production of Chassis	ChassisPersonDays	affects	-100
16	TuFast Garage for Chassis	ChassisPersonDays	affects	-600
5	Monocoque	Budget	affects	-20000
8	Spaceframe	Budget	affects	-5000
12	External Production of Chassis	Budget	affects	-30000
15	TuFast Garage for Chassis	Budget	affects	-10000
...	...	...	...	...

Figure 4-14 Extract from the table 'ParameterAssertion', rearranged according to the parameter

Furthermore, with an application of the query system, a more complex analysis of the system is possible. Therefore either SQL queries are implemented directly or the assisting system of *Access* can be used. The queries are returning tables as a result, which can be individually composed from the information available in the relational database. In the table, several fields can be selected from various tables which are joined on their related fields. These can be further grouped and checked if they fulfill certain condition as well as ordered or used for further calculations. An example for a generated SQL query of the *Access* assistant, which is designed to return the interrelations between solutions, is shown Figure 4-15. As introduced in chapter 2.3.4.3, the SQL query is written according to a specific scheme, which divides the request into several parts. Thus, initialized through the 'SELECT' statement, the requested fields are listed. In this case these fields are the fields 'Solution', 'Value', 'Assertion', 'Parameter' 'Assertion 2', 'Value 2', and 'related Solutions' is requested. The 'FROM' statement defines the sources of the requested fields, which are the according tables fields being joined on the related field. For example in the first two lines of the 'FROM' statement, it is written that the table 'FieldOfSolutions' and the table 'Solutions' should join when the value from the field 'Field of Solutions' is equal in both tables. Furthermore the 'GROUP BY' statement groups the results accordingly to avoid double datasets the in results. Additionally, with the 'HAVING' statement, the query checks whether values from the fields

'Solutions' and 'rel. Solutions' are different and are not asserted to the same field of solutions (solutions from the same field of solutions never can interrelate since they cannot be applied at the same time). Finally the 'ORDER BY' statement sorts the results alphabetically according to the solution and the field of solutions.

```
SELECT ParameterAssertion.Solution AS Solution,
       ParameterAssertion.Value AS Value,
       ParameterAssertion.[Assertion Kind] AS Assertion,
       ParameterAssertion.Parameter AS Parameter,
       ParamAssert2.[Assertion Kind] AS [Assertion 2],
       ParamAssert2.Value AS [Value 2],
       ParamAssert2.Solution AS [related Solution]
FROM ((FieldOfSolutions INNER JOIN Solutions ON
     FieldOfSolutions.[Field of Solutions] = Solutions.[Field of Solution])
     INNER JOIN (ParameterAssertion INNER JOIN ParameterAssertion AS
     ParamAssertion2 ON ParameterAssertion.Parameter = ParamAssertion2.Parameter)
     ON Solutions.Solutions = ParameterAssertion.Solution) INNER JOIN Solutions AS
     Solutions_1 ON ParamAssert2.Solution = Solutions_1.Solutions
GROUP BY ParameterAssertion.Solution, FieldOfSolutions.[Field of Solutions],
         Solutions_1.[Field of Solution], ParameterAssertion.Value,
         ParameterAssertion.[Assertion Kind], ParameterAssertion.Parameter,
         ParamAssert2.[Assertion Kind], ParamAssert2.Value, ParamAssert2.Solution
HAVING (((Solutions_1.[Field of Solution])
       <>[FieldOfSolutions].[Field of Solutions]) AND
       ((ParamAssert2.Solution)<>[ParameterAssertion].[Solution]))
ORDER BY FieldOfSolutions.[Field of Solutions], ParameterAssertion.Parameter;
```

Figure 4-15 SQL query to receive interrelations between the solutions of the PPS

Having applied the query to identify interrelations between solutions, *Access* returns a table which can be used for further analysis. Figure 4-16 shows an extract of the resulting table which is filtered to the solution 'FourCylMotor'. On the left side of the table, it can be seen to which parameter the solution is connected, the type of assertion and the value. On the right side all other solutions which are connected to the same parameter are listened, how they are related and by which value. This view is a good support to identify conflicts between solutions, since it shows all relevant solutions and their connections which needs to be checked. Thus the four cylinder motor uses 10000 from the budget and 100 person days from the motor division and needs to be confronted to several other solutions. Consequently, for the parameter 'MotorPersonDays' no conflict with other solutions can be detected, while the budget can be identified as a critical parameter.

Solution	Value	Assertion	Parameter	Assertion 2	Value 2	related Solution
FourCylMotor	-10000	affects	Budget	affects	-20000	Monocoque
FourCylMotor	-10000	affects	Budget	affects	-5000	Spaceframe
FourCylMotor	-10000	affects	Budget	affects	-10000	TuFast Garage for Chassis
FourCylMotor	-10000	affects	Budget	affects	-30000	External Production of Chassis
FourCylMotor	-10000	affects	Budget	affects	-20000	External Production of Motor
FourCylMotor	-10000	affects	Budget	affects	-5000	TuFast Garage for Motor
FourCylMotor	-10000	affects	Budget	leads to	60000	Sponsor F
FourCylMotor	-10000	affects	Budget	leads to	70000	Sponsor ABCDE
FourCylMotor	-100	affects	MotorPersonDays	leads to	1000	EqualDistribution
FourCylMotor	-100	affects	MotorPersonDays	leads to	750	Strong Chassis Department
FourCylMotor	-100	affects	MotorPersonDays	affects	-100	External Production of Motor
FourCylMotor	-100	affects	MotorPersonDays	affects	-400	TuFast Garage for Motor

Figure 4-16 Filtered results for the query to identify interrelations between solutions

Finally the queries can be extended from the related solutions to the connected goals. Thus direct interrelations between goals can be investigated. An example for the analysis of goals can be seen in Figure 4-17. The table shows a list of all connected goals and the number of relations, which were detected in the database. Thus it can be identified that especially the goals ‘Best Chassis’ and ‘Best Solution for Chassis Production’ are strongly interrelated while the goals ‘Best Budget’ and ‘Best Ratio of Human Resources’ do not have a common parameter. In the case of this example, the number of relation is always a multiple of four, since every goal can be achieved by two solutions, leading to a minimum of four ways to connect two goals. Consequently with this result a good general overview is given, helping to understand which goals are more critical and need to be under surveillance.

Goal A	Goal B	#Relations
Best Budget	Best Chassis	4
Best Budget	Best Engine	4
Best Budget	Best Solution for Chassis Production	4
Best Budget	Best Solution for Motor Production	4
Best Chassis	Best Engine	4
Best Chassis	Best Solution for Chassis Production	12
Best Chassis	Best Solution for Motor Production	4
Best Chassis	Best Ratio of Human Resources	4
Best Engine	Best Solution for Chassis Production	4
Best Engine	Best Solution for Motor Production	8
Best Engine	Best Ratio of Human Resources	4
Best Solution for Chassis Production	Best Solution for Motor Production	4
Best Solution for Chassis Production	Best Ratio of Human Resources	4
Best Solution for Motor Production	Best Ratio of Human Resources	4

Figure 4-17 Results of a query which identifies the number of interrelations between goals in the PPS

Summarizing all presented possibilities of the GIA based on relation databases, it offers already a good support for further analysis of the implemented PPS. Especially SQL is useful to create queries which are returning dynamically information about specific properties of the system. Nevertheless SQL does not last for a complete GIA. Since it does not support the consideration of rules for elements or the type of parameter relation, an extension with higher level programming languages needs to be implemented in order to perform a complete GIA.

4.2.4 Semantic Web

Finally the usage of semantic web for the modeling of a PPS and the offered possibilities to perform a GIA is regarded. Therefore an implementation based on the *Web Ontology Language* (OWL) is presented. OWL is developed for an efficient knowledge management system by adding semantic meanings to stored data. This is done by modeling information through the usage of classes, properties, individuals and annotations which can have further semantic functionalities (see chapter 2.3.4.4 for more details). Consequently sematic web offers the general functionality which is requested in this thesis and which will be analyzed in the following

4.2.4.1 Introduction of the Software Tool *Protégé*

In order to analyze advantages and disadvantages of semantic web technology in this context, the software tool *Protégé* (http://protege.stanford.edu/) is used. *Protégé* is a free, open source platform which offers a suite of tools to develop domain models and knowledge-based applications with semantic web technologies. At its core, *Protégé* implements a rich set of knowledge-modeling structures and actions that support the creation, visualization, and manipulation of ontologies in various representation formats. Additionally with the provided plug-in architecture and a java-based API, the software is continually improved and extended by a growing user community (STANFORD CENTER FOR BIOMEDICAL INFORMATICS RESEARCH 2011). A representation of a system based on OWL is generally created by describing classes, properties, individuals and formal semantic attributes. Editing these formal semantic attributes is supported comfortably in *Protégé* with a graphical interface, where they can be directly selected and manipulated. Having designed a complete ontology, these OWL formal semantic attributes allow deriving logical consequences which can be used to check the consistency of a system as well as serve for further processing and analysis of the stored information.

The initial step for creating a representation of a system with *Protégé* commonly begins with the definition of classes in a hierarchical list. Classes serve as logical collections of individuals. Thus, classes may be defined by a reasonable name and further annotations to understand which individuals are member of it. An example would be the class ‘Goals’ which collects all individuals, which are goals. Additionally classes can be described by formal semantics, which specify for example equivalent classes, super classes or disjoint classes. Disjoint in this case means, that members of one class cannot be member of the disjoint class (See PRUD'HOMMEAUX & SEABORNE 2008 for further details). Consequently, with a complete collection of classes, all individuals in an ontology can be classified accordingly while the semantic attributes help to maintain the consistency of the system.

Furthermore the various types of properties are needed to be defined. Properties can be either object properties, which are connecting two individuals, or data properties, which are connecting individuals to data. Thus, they are needed to be defined in advance in order to be applied later during the definition of individuals. Both can be created in a hierarchical list and described by further semantic characteristics. Firstly, for data properties these semantic attributes can contain information about the domain of the connected individuals, the range of data types it connects to, the super properties, the disjoint properties and whether the data property is functional can be added. For object properties otherwise, information about the domain of individuals, the range of individuals, equivalent object properties, super properties, disjoint properties and property chains can be added. Additionally the object properties can be characterized whether they are functional, inversion functional, transitive, symmetric, asymmetric, reflexive, or irreflexive. Further information about the meaning of the various types of characterizing semantic attributes can be found in PRUD'HOMMEAUX & SEABORNE (2008).

The last step towards representing a system in an ontology is the definition of individuals and their assignment to classes and properties. Individuals or instances are non-dividable elements which represent existing elements of a system (PRUD'HOMMEAUX & SEABORNE 2008).In order to initialize them they can be created in a non-hierarchical list by specifying the name and adding annotations. In a next step, these individuals can be assigned to the according classes. Additionally the property assertions can be set, by adding information to which other individual through object properties or to specific data values through data property the individuals are connected. Finally it can be specified which individuals are identical or explicit different to the current. Summarizing, with individuals representing system elements, being part of various classes and connected among each other with object properties or to data objects with data properties, a holistic representation of a system can be built. This can be approved and checked for inconsistency with a software tool called ‘reasoner’.

Based on the created ontology, *Protégé* supports also further analyses of it. This can be done either with the simple integrated query system, a plug-in supporting more complicated queries or the open API of *Protégé*. The integrated query system is designed to quickly test definitions of classes to see that they subsume the appropriate subclasses, or checking class memberships of arbitrary description. The plug-ins mainly base on SPARQL which is a query language especially designed for the semantic web. It offers capabilities for querying required and optional patterns along with their conjunctions and disjunctions as well as extensible value testing and constraining queries (PRUD'HOMMEAUX & SEABORNE 2008). For an even more complex analysis *Protégé* also offers an API, based on the OWL-API which enables direct access to the information stored in the ontology with java and therefore a higher programming language. With these options it may be possible to implement all requested types of analysis of the represented system.

4.2.4.2 Implementation of the PPS

The first step towards analyzing goal interrelations, based on a semantic web, is to implement a PPS using the technique. In the case of *Protégé*, as explained in the previous section, at first the classes and properties need to be defined. Subsequently individuals are added to the system and assigned to the according classes, object or data properties with their specific values. Thus an ontology based on OWL is created which can represent an entire system and enables the possibility to be tested for consistency by its semantic characteristics.

An implementation of the classes and object properties in order to represent the PPS of the example can be seen in Figure 4-18. On the left side of the figure the list of the implemented classes is presented. In this definition, the classes are divided into parameters, scenarios, seasons and *Strategic Product Planning* (SPP) elements. Consequently, the class parameter is designed to group all instances of parameters. The classes for scenarios as well as the one for seasons are further subdivided, containing specific scenarios or seasons to which the various individuals can be assigned. Moreover the class 'SPP-Elements' contains all basic element classes for a strategic product planning system: the demands, the goals, the core functions and the field of solutions. The last ones are further subdivided into the various fields of solutions which are collecting the instances representing specific solutions. Since all necessary types of individuals can be classified with this scheme, it is an applicable solution for the representation of a PPS.

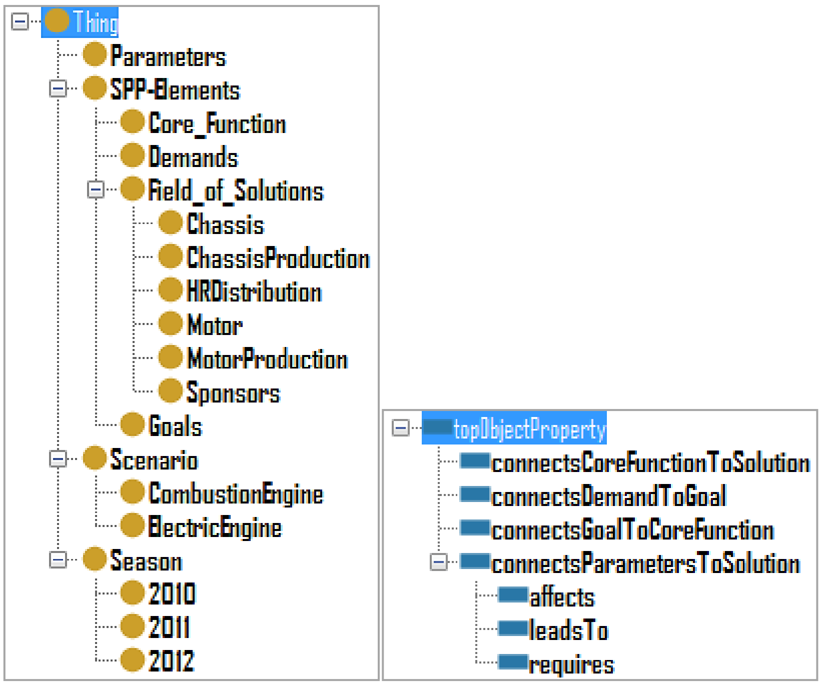


Figure 4-18 Classes and object properties for the PPS of a racecar development

Furthermore on the right side of the Figure 4-18, the implemented object properties can be seen. Hence, there are properties defined to link demands to goals, goals to core functions, and core functions to solutions and solutions to their parameter. The connection between solutions and parameters can be subdivided to classify the type of connection. Therefore it is distinct if the solutions affects, requires or leads to a value of a parameter. Based on the implemented scheme of classes and object properties the individuals can be defined and asserted accordingly.

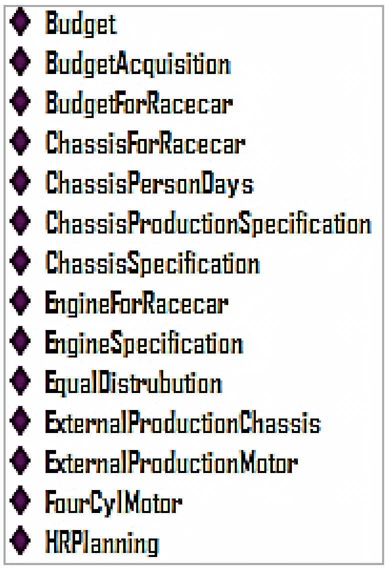
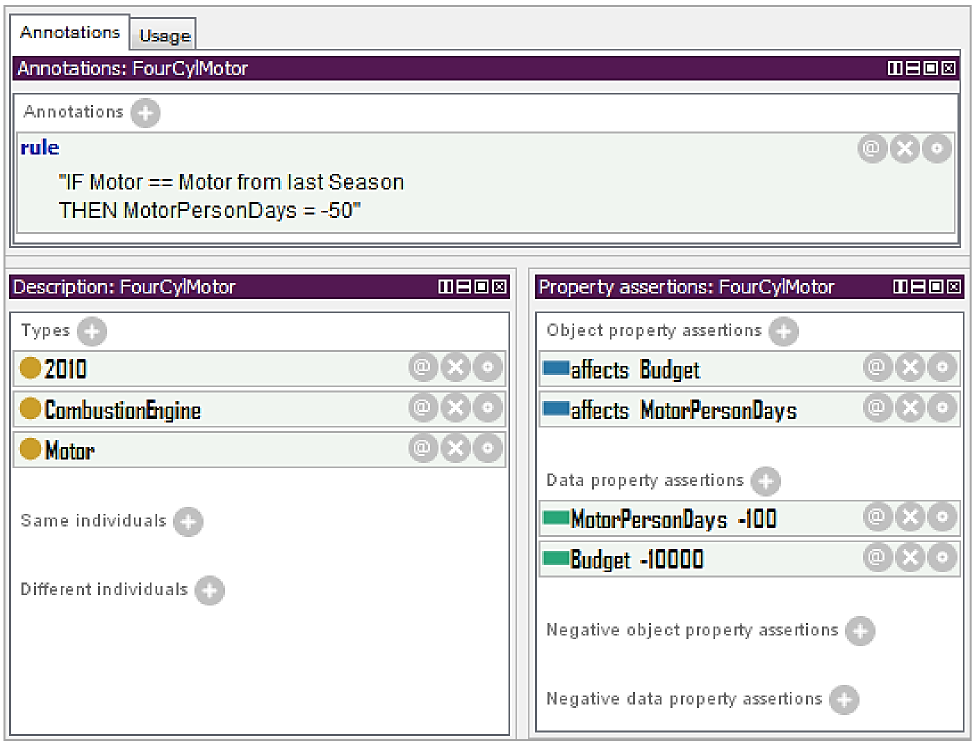

◆ Budget
◆ BudgetAcquisition
◆ BudgetForRacecar
◆ ChassisForRacecar
◆ ChassisPersonDays
◆ ChassisProductionSpecification
◆ ChassisSpecification
◆ EngineForRacecar
◆ EngineSpecification
◆ EqualDistrubution
◆ ExternalProductionChassis
◆ ExternalProductionMotor
◆ FourCylMotor
◆ HRPlanning

Figure 4-19 Extract of the list of individuals of the exemplary PPS

The specific elements of a PPS can be represented in OWL, by defining according individuals. An extract of the individuals defined in the ontology can be seen in the Figure 4-19. Hence individuals can be specific goals, core functions, demands, parameters or solutions. As already mentioned these individuals are further characterized by adding annotations about them and assigning them to classes and properties. The full characterization of the individual ‘FourCylMotor’ can be seen in Figure 4-20. On the lower left side of the figure the descriptions of the individual are shown, describing its type as a member of the classes ‘2010’, ‘Combustion Engine’, and ‘Motor’ which represents its season, scenario and field of solutions. Moreover neither ‘Different Individuals’ nor ‘Same Individuals’ are defined. On the lower right side of the figure, the property assertions are listed, showing that ‘FourCylMotor’ affects the individuals ‘Budget’ by -10000 and ‘MotorPersonDays’ by -100. Finally a rule is added in the annotations in the upper box of the figure, saying that the value of ‘MotorPersonDays’ changes if the same motor from last season is used. Summarizing, with a complete and consequent characterization of all the implemented individuals a representation of the entire system can be built with all the included elements, their specific attributes and their interrelations.


Annotations | Usage

Annotations: FourCylMotor

Annotations

rule
"IF Motor == Motor from last Season
THEN MotorPersonDays = -50"

Description: FourCylMotor

Types
- 2010
- CombustionEngine
- Motor

Same individuals

Different individuals

Property assertions: FourCylMotor

Object property assertions
- affects Budget
- affects MotorPersonDays

Data property assertions
- MotorPersonDays -100
- Budget -10000

Negative object property assertions

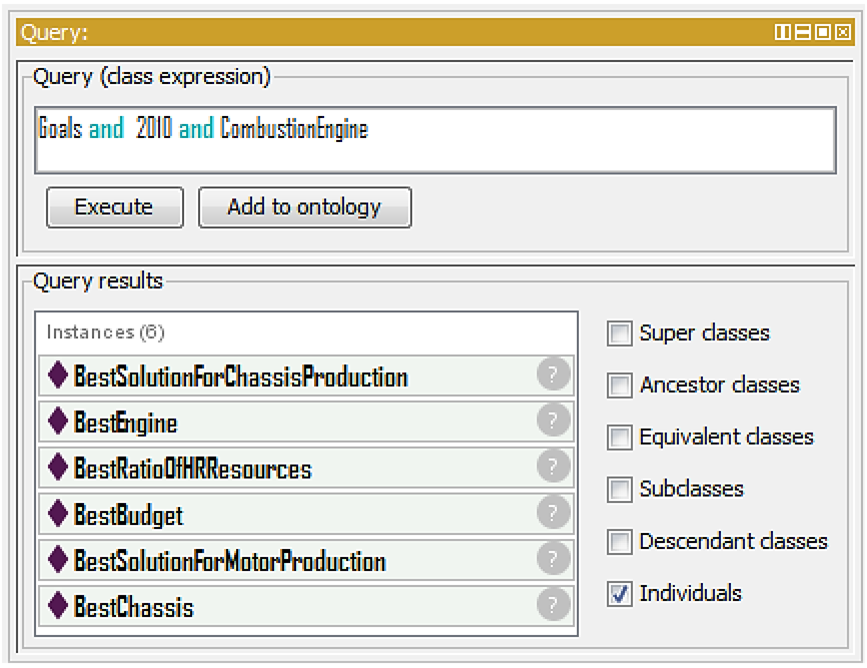
Negative data property assertions

Figure 4-20 Definition of the characteristics of the individual 'FourCylMotor'

4.2.4.3 Goal Interrelation Analysis

Analyzing goal interrelations with *Protégé* in a PPS which is based on a semantic web is only partially supported by the program. Thus, with the integrated query, *Protégé* can help identifying individuals, which are members of specific classes as well as their combination. Furthermore the query for relationships and paths between various individuals or their values can be solved with plug-ins based on SPARQL, which a query language designed for the semantic web. Nevertheless for more complex analysis *Protégé* offers an open API which enables direct access to all implemented elements, their assertions, annotation, value, etc. with the higher programming language Java.

The query integrated in *Protégé* is a simple query which supports the identification of requested elements. Thus, an expression, representing the requested combination of classes, has to be executed in the query and the program returns the results. These results can contain the super, ancestor, equivalent, sub-, or descendant classes as well as individuals which are assigned to. Figure 4-21 shows an example of the integrated query, where all individuals being member of the class 'Goals', '2010' and the scenario 'CombustionEngine' are identified through the class expression 'Goals and 2010 and CombustionEngine'. Since the query processes only class expression, more complex relations cannot be found. Consequently the integrated query system helps identifying individuals, being in the focus of a GIA, but does not support any further analysis.


Query:

Query (class expression)

Goals and 2010 and CombustionEngine

Query results

Instances (6)

- BestSolutionForChassisProduction
- BestEngine
- BestRatioOfHRResources
- BestBudget
- BestSolutionForMotorProduction
- BestChassis

- [ ] Super classes
- [ ] Ancestor classes
- [ ] Equivalent classes
- [ ] Subclasses
- [ ] Descendant classes
- [✓] Individuals

Figure 4-21 Protégé query for instances of the classes 'Goals', '2010', and 'CombustionEngine'

In order to get results for queries concerning classes, properties, individuals and their interrelations a more powerful query system is needed. An example is the OWL2Query plug-in for *Protégé* (KŘEMEN & KOSTOV 2011). It is based on SPARQL and offers a possibility to draw the requested query with graphs and translates them to SPARQL expressions. These expressions and graphs need to define a specific scheme, also called tuple, containing start node, edge and end node. Executing these expression returns a list of the requested variables and the found elements.

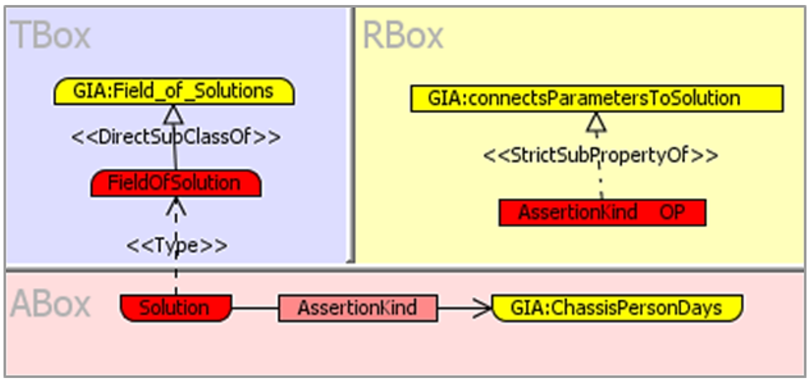


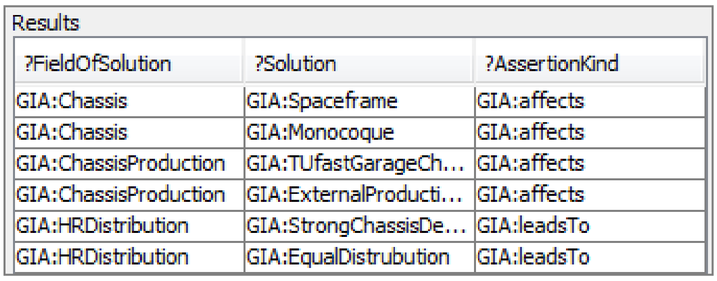
Figure 4-22 Query for all solutions connected to 'ChassisPersonDays' using the graph system of OWL2Query

Figure 4-22 shows an example for a graph representation of a query. It requests all individuals, which are from the class ‘Solutions’ and connected through any sub property of the object property ‘connectsParametersToSolution’ to the individual ‘GIA:ChassisPersonDays’. The graph system is good to understand with yellow boxes being specific elements of the ontology and red boxes being variables. Subsequently the plug-in translates the request to a SPARQL query, which is presented in Figure 4-23. It shows that the SPARQL syntax is close to the SQL syntax. Thus in the ‘SELECT’ statement, all the requested variables for the results are written. Further in the ‘WHERE’ statement, the constraints of the query are defined with tuples. For example the first block defines that elements considered for the variables ‘?AssertionKind’ need to be of the type object property and a strict subclass of the object property ‘connectsParametersToSolutions’. Thus an equivalent query is possible to be presented in a graph and a written form.

```
SELECT ?FieldOfSolution ?Solution ?AssertionKind

WHERE

 {    ?AssertionKind <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
      http://www.w3.org/2002/07/owl#ObjectProperty> ;
      <http://pellet.owldl.com/ns/sdle#strictSubClassOf>
      <http://pe.tum.de/TUfastSPP.owl#connectsParametersToSolution> .

      ?Solution  ?AssertionKind
      <http://pe.tum.de/TUfastSPP.owl#ChassisPersonDays>;
      <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>  ?FieldOfSolution .

      ?FieldOfSolution  <http://pellet.owldl.com/ns/sdle#directSubClassOf>
      <http://pe.tum.de/TUfastSPP.owl#Field_of_Solutions> . }
```

Figure 4-23 SPARQL query for all solutions connect to 'ChassisPersonDays'

Based on the command, the program returns the solution in form of a table which can be seen in Figure 4-24. It shows the three different fields of solutions, and the six belonging solutions with their kind of connection to the parameter ‘ChassisPersonDays’. Summarizing with this query system it is possible to identify interrelations between elements which can serve as a basis to analyze the goal interrelation of a PPS. Nevertheless to compute the results and to derive further information about the system, the query system is not powerful enough. It needs to be supported by a dedicated software implementation which can use the offered open *Protégé*-API to directly access all elements with the higher programming language java.


Results

?FieldOfSolution	?Solution	?AssertionKind
GIA:Chassis	GIA:Spaceframe	GIA:affects
GIA:Chassis	GIA:Monocoque	GIA:affects
GIA:ChassisProduction	GIA:TUfastGarageCh...	GIA:affects
GIA:ChassisProduction	GIA:ExternalProducti...	GIA:affects
GIA:HRDistribution	GIA:StrongChassisDe...	GIA:leadsTo
GIA:HRDistribution	GIA:EqualDistrubution	GIA:leadsTo

Figure 4-24 Results of the SPARQL query for solutions connected to the parameter 'ChassisPersonDays'

4.3 Evaluation of the Analysis

In order to receive a complete overview on the presented solutions for modeling a PPS and analyzing goal interrelations, they are evaluated confronting the defined requirements. Therefore the solutions are compared with the relevant requirements, which are concerning the implementation of the approach and the software (see chapter 3.2 and 0). The results of these confrontations are further compared among each other and presented according to the main points of the requirements list. A qualitative classification of the results can be seen in Figure 4-25, showing the main points of the requirements list and how the various techniques are supporting the requested feature. Hence a full circle indicates a very good support whereas an empty circle represents a bad support of them.

ID	Requirement	MDM	GG	relDB	semWeb
2	**Requirements for the Implementation of the Approach**				
2.1	Types of Elements	◕	●	●	●
2.2	Element Relations	◕	●	●	●
2.3	Element Handling	◔	◕	◕	◕
3	**Software Requirements**				
3.1	Data Handling	◑	◕	◕	◕
3.2	Data Acquisition	◑	◔	◑	◔
3.3	Query for Conflicts	◔	◔	◑	◑
3.4	Data Organization/ Maintenance	◔	◑	◕	◕
3.5	Visualization	◑	◑	○	◔
3.6	Computational System Optimization	○	○	○	○
3.7	Ergonomically	◔	◔	◔	◔

Figure 4-25 Qualitative classification of the results confronting the requirements

To begin with the requirements concerning the implementation of the approach are regarded. These concern mainly the implementation of the PPS which serves as a basis for the GIA. The point 2.1 regards the various types of elements which are needed to be supported. Here it can be observed that only the MDM has disadvantages implementing scenarios and temporal aspects since the complexity of the matrices is increasing strongly with implementing them. Point 2.2 describes the requested relations between the elements. In general all solutions can handle the basic relations between the elements, but have issues with the support of fuzzy parameter assertion. Additionally the solution based on the MDM does not directly support various types of relations between solutions and their parameters. Finally point 2.3 considers the handling of the various elements, supporting type and situation specific requirements. As a result of the analysis it can be seen that the MDM has issues implementing limitations to unique or simultaneous classification of elements, field of solution, an optional core function and a rule-based attribute to elements. The other solutions just have issues considering the optional core functions while the other requirements are supported. Summarizing the results for the implementation of the approach a disadvantage of the MDM can be seen, while graph grammars, relational databases and semantic web are equally good modeling techniques for it.

The second part of the requirements is addressing the software with its computational features and its properties. Thus the software should support every phase of a GIA and meanwhile provide a good usability to the user. The first point 3.1 reflects the results from the first block where the graph grammar, the relational database, and the semantic web fulfill the data handling better than the MDM. The data acquisition, point 3.2, is otherwise supported best by the MDM followed by the relational database. This is because both have a very formal method to acquire the data, while with the graph grammar and the semantic web it is necessary more complicated operations have to be performed. In addition, the import is better supported with the MDM and the relational Database, since both can directly import tables. The point 3.3, query for conflicts, is best solved through the relational database and the semantic web. Both have a query language which allows identifying specific elements, their interrelations as well as access to the values of the parameters. The graph grammar offers these possibilities as well, but the language is not especially designed for queries and therefore worse applicable. The MDM allows identifying paths, showing possible conflicts, but since values are not directly available, no further analysis is possible. For the point 3.4, data organization and maintenance the analysis shows that the semantic web is more convenient than the relational database and the graph grammars followed by the MDM. This can be deduced to the fact that all solutions can store the data in a file, but the plausibility and consistency is better maintained due to the formal semantic specification in the semantic web. The relational databases as well as the graph grammars are supporting the consistency through to predefined limitations for the relations while the MDM can be filled freely with information. Updating data is solved best with the relational database, since SQL directly supports the update, whereas all other solutions require manual intervention. Nevertheless the visualization, point 3.5, is supported best by the MDM and the graph grammar, whereas the relational database has no directly implemented visualization feature. The MDM can use graphical filters and colors to underline its analysis' results. The graph grammars can, due to their graph nature; show the system in three dimensions and also highlight certain elements and connection manually with colors. In general, the semantic web can also be represented using graphs and therefore offers all possibilities for a good visualization, but since it is not directly implemented into *Protégé* it could not be tested. The computational optimization, point 3.6, is not yet available in any of these solutions. Nevertheless graph grammars, relational databases and sematic webs are offering APIs for higher programming languages which can be used for its implementation. Finally point 3.7 deals with ergonomic properties of the solutions which are strongly dependent on the software and therefore an advantage for the maturity of *Access*. Summarizing it can be said that the software requirements are generally fulfilled much worse than the requirements for the implementation of the approach. However the semantic web and the relational database are offering the best solutions for the requirements, followed by the graph grammars and finally the MDM.

Regarding the whole evaluation some trends for the results can be identified. As a general result it can be seen that every modeling technique has advantages and disadvantages in specific fields of the overall solution. Having confronted the solutions to all requirements defined in chapter 3, the relational database and the semantic web are more applicable for the implementation of a GIA. Comparing the relational database and the semantic web especially the maturity of the techniques and the applied software tools differs between both. Thus the relational database and particularly *Access* offers a high reliability and a good support through external programs. Nevertheless, being comparable to the more mature solution of relational databases, the relatively young technique semantic web and the open software program *Protégé* offers a lot of potential for future trends and developments.

5 Framework for a Computer-Supported Goal Interrelations Analysis

This chapter presents a general framework for a computer-supported for *Goal Interrelation Analysis* (GIA). It is deduced from the previously defined requirements (see chapter 3) and the results from the analysis of established computer-supported modeling techniques and adapted through various iterations. Therefore the framework is presented accordingly to the requirements in three sections. Consequently, in the first one, the requirements for the enhancement of the approach for goal interrelation analysis according to (HEPPERLE ET AL. 2011a) and (FÖRG 2010) are satisfied through a classification of temporal aspects and scenario as well as the introduction to rule-based description. Furthermore the requirements for the implementation of the approach are satisfied by a formal definition of the structure of the *Product Planning System* (PPS) for analyzing goal interrelations. Finally the requirements for the software implementation are considered by a general use case diagram as well as the definition of a framework for the structure of the software tool. Subsuming all specifications, a general framework for a succeeding implementation for a software tool which supports a goal interrelation analysis is presented.

5.1 Enhancement of the Approach for Goal Interrelation Analysis

Since a general need for an improvement of the approach for analyzing goal interrelations Hepperle et al. (2011a) and Förg (2010) was identified, this chapter describes some newly adapted aspects which enhance the existing approach. Hence, a classification for the consideration of temporal aspects in the product development as well as a classification for the consideration of several scenarios is presented. Furthermore, the application of rule-based descriptions is introduced.

To begin with the consideration of a temporal aspect for the classification of elements within a PPS is presented. This is necessary in order to analyze the complex relations between several product generations or facelifts. An overview on the temporal enhancement of the approach is shown in Figure 5-1. Consequently the existing approach is combined with an adapted version of the approach for the consideration of temporal aspects of product-service-systems by Hepperle et al. (2011b), which is presented in chapter 2.3.2. Accordingly, an additional classification for the elements and their interrelations of a PPS is introduced. Hence, they can be described by their belonging lifecycle phase, product generation and facelift. In the figure for example, two product generations and one facelift of the first generation is shown. Additionally the parameters are not further connected directly to a specific lifecycle. They are defined globally, minimizing the risk of confusion or passing of parameters in large models of PPS considering several product generation. Summarizing an enhancement of the existing approach in order to consider several product generations or facelifts can be done by an additional classification of the related elements.

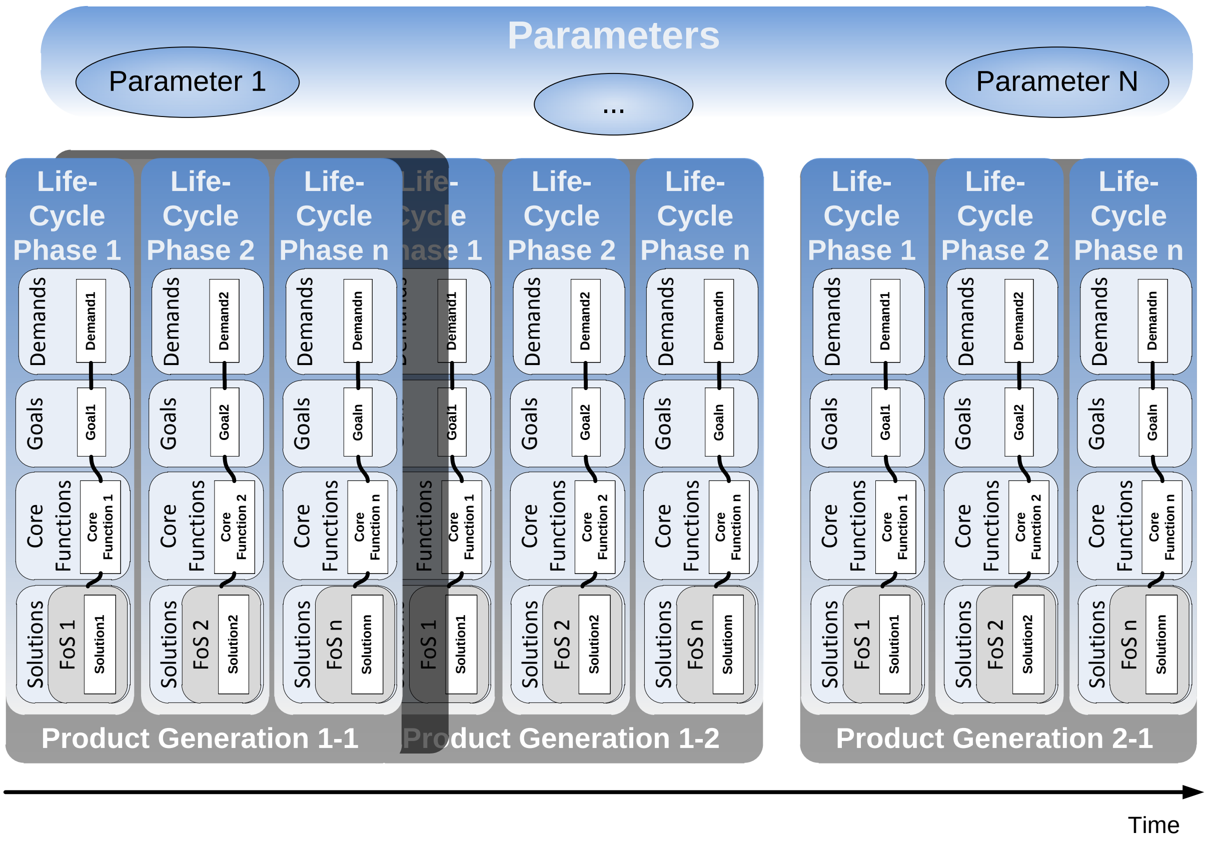


Figure 5-1 Classification of several PPS according to their temporal aspect

Another requested feature for a GIA is the consideration of several scenarios. This supports the planning of future products, since besides the current scenario also other trends can be investigated. A general overview on the application of scenarios for the classification of several PPS is shown in Figure 5-2. The classification is based on the scenario-technique, which is presented in chapter 2.3.2, and applied to the approach like the classification for the consideration of a temporal aspect. Consequently the basic elements of the PPS are categorized by their type, product lifecycle phase, product generation and additionally the conferring scenario. As an example for the classification Figure 5-2 shows a red line being scenario A which splits after the third generation to the green line being scenario B. The belonging scenarios are further written in the attributes of the PPS allowing a precise identification. Concluding with the introduction of the additional attribute to classify the elements of a PPS it is enabled to including several scenario-based PPS in an overall strategic product planning.

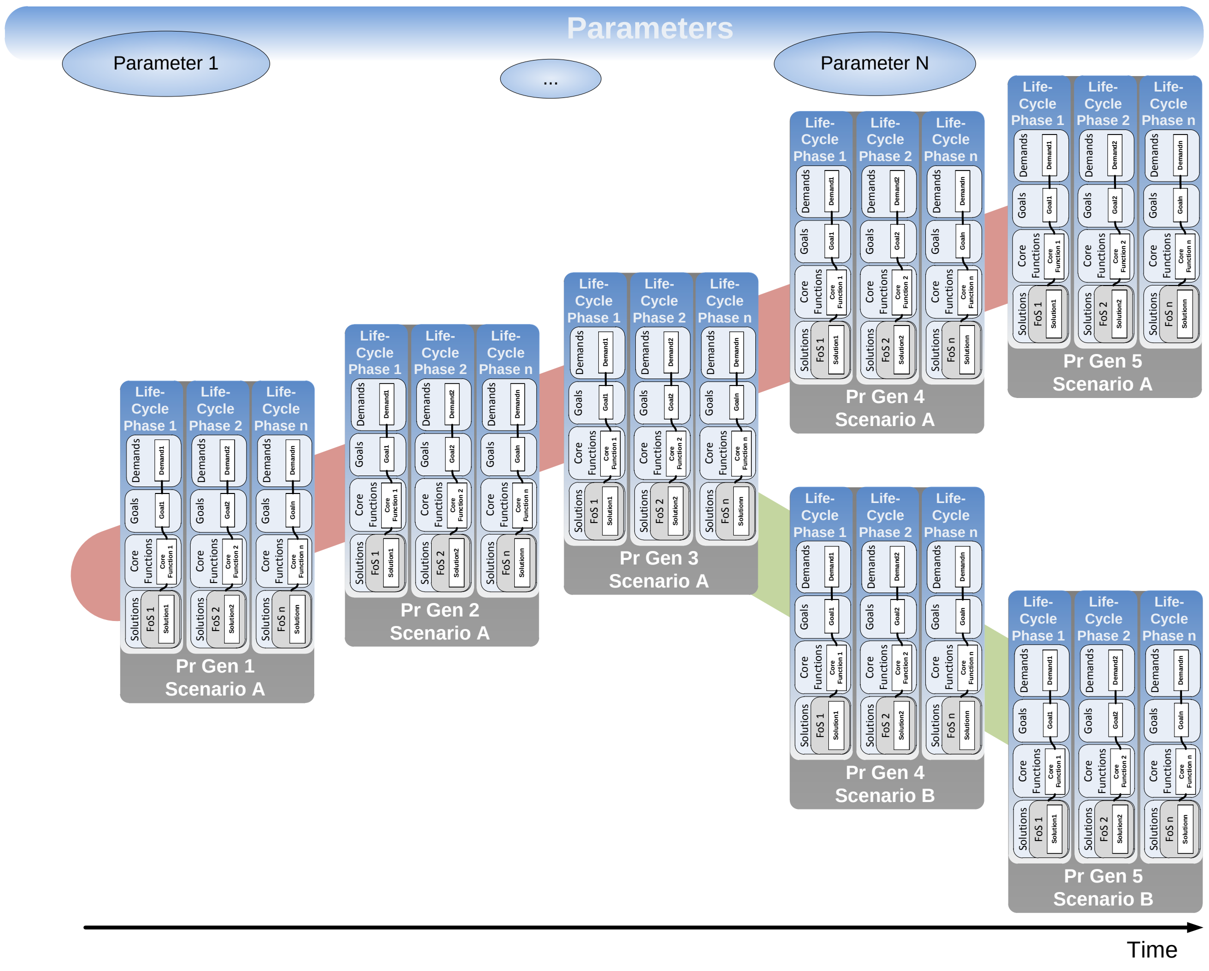


Figure 5-2 Classification of several PPS considering several scenarios

Finally an introduction of a possibility to apply rule-based knowledge representation within the PPS is requested. Integrating rule-based knowledge into the approach allows describing more complex relation than the model- and graph-based approach. An example would be a changed value of a solution's parameter if another specific solution is chosen. Therefore an attribute with the rule needs to be added as an additional attribute to the elements. In general these rules have a basic structure, divided into two parts. The first part is usually introduced by the keyword 'IF' describing the requirement or premise for the rule. The second part is describing the action or conclusion and introduced by the keyword 'THEN'. With this construction the actual knowledge can be directly implemented in the rule (MÜNZER 2011, p. 8). Consequently with adding an attribute for rule-based content to the elements of a PPS further knowledge can be represented by enabling the description more complex interrelations within these.

Recapitulating, the requirements for the enhancement of the approach to analyze goal interrelation by (HEPPERLE ET AL. 2011a) and (FÖRG 2010) can be satisfied by adding new classification possibilities as well as attributes to the existing elements of the PPS. Thus, these elements can be classified by their product generation, facelift and the related scenario. Furthermore an attribute can be added to the elements if necessary containing a rule-based description of complex knowledge. With these enhancements of the approach it is possible to create a holistic representation of a PPS which can serve for an integrated analysis of goal interrelation.

5.2 Goal Interrelation Analysis System

In order to allow accurate implementation of the approach a precise framework for the considered elements is needed. Therefore the relevant elements of the approach are defined with their structure, attributes and interrelation. This is done by using the description language sysML to create a block definition diagram as well as and internal block diagram. Through these diagrams it is possible to receive a complete overview on the relevant elements for the implementation of the approach.

Firstly all relevant elements of a system for goal interrelation analysis are defined within the block-definition diagram presented in Figure 5-3. There are seven elements defined: the general GIA-element, demands, goals, core functions, field of solutions, solutions and parameters. The general GIA-Element contains further several attributes which can be used to describe the name of the element, the according lifecycle, the temporal classification, the scenario to which the elements belongs, as well as a rule for additional complex knowledge. These attributes are inherited by the field of solution, core function, goal, and demand which consequently can be described by the same ones. It is also shown that a Field of Solution is composed by several solutions, which consequently can be described by the same attributes. Furthermore parameters and their attributes are also defined. These are the name, the type of the parameter for example integer or string, the rule for rule-based knowledge representation and the maximum as well as the preferred value. Summarizing this diagram presents a complete definition of the main elements of a GIA-System with the all necessary elements and their attributes for the implementation of the approach.

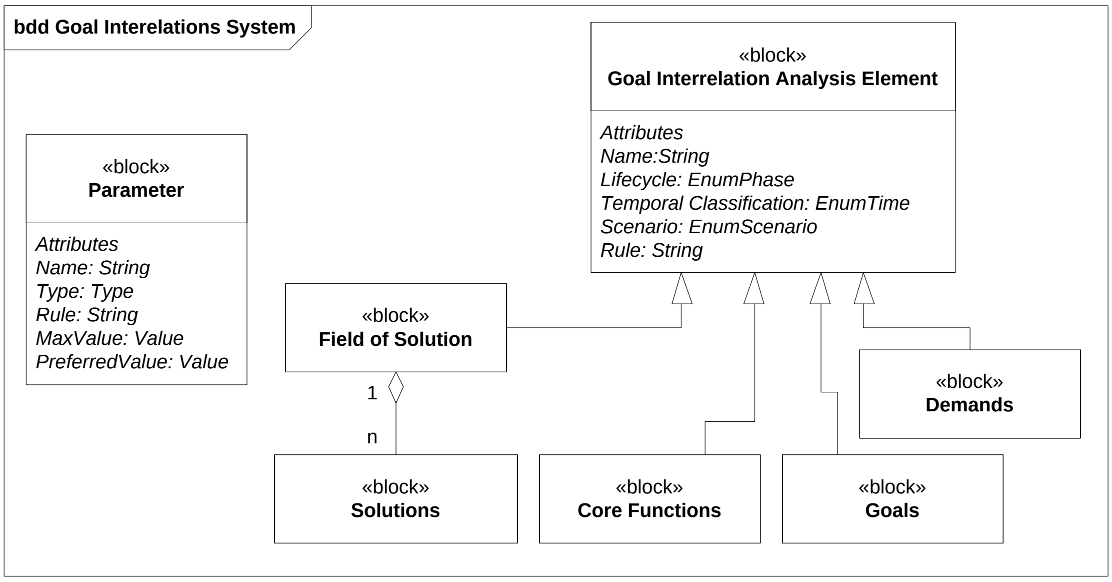


Figure 5-3 Block-Definition diagram of a GIA-system

Next, the internal-block diagram in Figure 5-4 shows the internal structure of the elements within the approach. It shows how the elements are interrelated among each other and how the information is distributed indicated by the arrows. Consequently, the interrelations between the elements as well as market pull and technology push can be described. As it can be seen in the figure, the market pull starts at the demands, is further connected to goals, which are leading to the core function. On the other side, the technology push is represented by the solution assertion which is also connected to the core function. The solution assertion describes in general the system of solutions collected in a field of solutions. The field of solution is linked to parameters through a specific type which describes the parameter assignment to every solutions collected in the field of solution. Thus, all solutions in the field are completely interchangeable, since they are connected to the same parameters in an equal way. Lastly the solutions are connected to the parameters defining the value for the linkage. However, it often occurs that solutions of one field do not influence the same parameters. Nevertheless it is necessary to collect and define all connections to parameters in the field of solutions in order to enable the interchangeability and manageability of the solutions. Consequently, the values of parameters which are not influenced by the according solution are just not set. Concluding, the diagram presents a complete definition of the interrelation between the elements of the approach showing how the information is distributed among them and naming the connections.

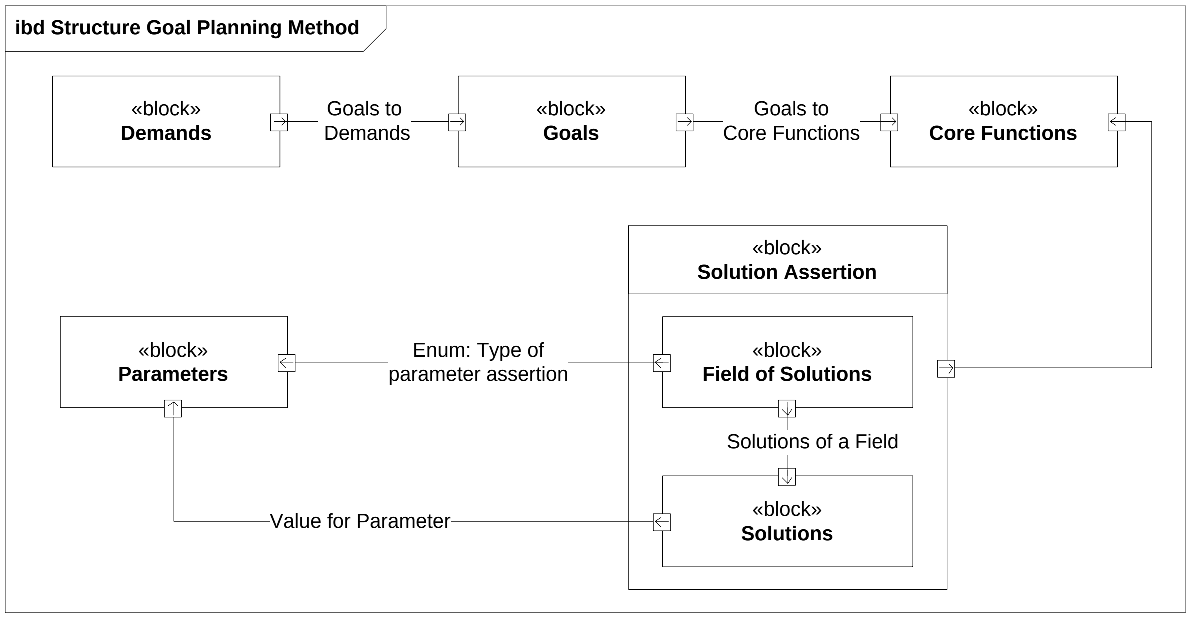


Figure 5-4 Internal-Block diagram of a GIA-System

Recapitulating, the presented diagrams are defining the basic structure for the implementation of the approach. Hence the elements are defined with their describing attributes, their dependencies as well as the interrelation among each other. Together with the requirements in chapter 3.2 a complete definition of all necessary elements for the analysis of goal interrelation of a PPS is given.

5.3 Goal Interrelation Analysis Software

In this section a general framework for the implementation of a software tool is presented, which supports the modeling of a PPS in order to analyze goal interrelations. Thus the main use cases of the program are defined, showing the core features of the tool as well as the interactions to the user. Furthermore, the various parts of the software tool are identified within a block-definition diagram and the interaction between is lastly shown in an internal-block diagram. Combining these definitions with the requirements listed in chapter 0 gives a complete framework for an implementation of the requested software tool.

Firstly, the use-case diagram which describes the main features and interaction between the user and the software tool is shown in Figure 5-5. As it can be seen, the user has three interaction points with the program: He has to enter all information of the system, the demands, core functions, fields of solutions, solutions, parameter and their interrelation. This use case can be further subdivided into two cases because the user needs to create new elements as well as updating existing elements. Additionally the user sees the information shown by the graphical visualization, being the organized information in the system, the identified goal interrelation, and the computed conflicts. Lastly the user has to interact with the program by selecting relevant elements which are needed to be considered for the GIA. Besides the use cases which are interacting directly with the user, the software has to consider further ones. Thus the software has to organize the acquired information and to transfer it to a fixed scheme of elements and their interrelations. Based on this, the tool has to identify goal interrelations by checking connections between goals through core functions, solutions and parameters and also compute the values of the solutions in order to identify conflicts between solutions and consequently goals. With the identified interrelations and conflicts another use case is to compute an optimized system, which tries to identify an optimal system constellation. Finally the software needs to deal with the use case to store the results of organizing the information, identifying goal interrelations, detecting conflicts, as well as creating an optimized solution for the system constellation. Summarizing the use case diagram provides a basic overview on the relevant cases for the implementation of a software tool which supports a GIA.

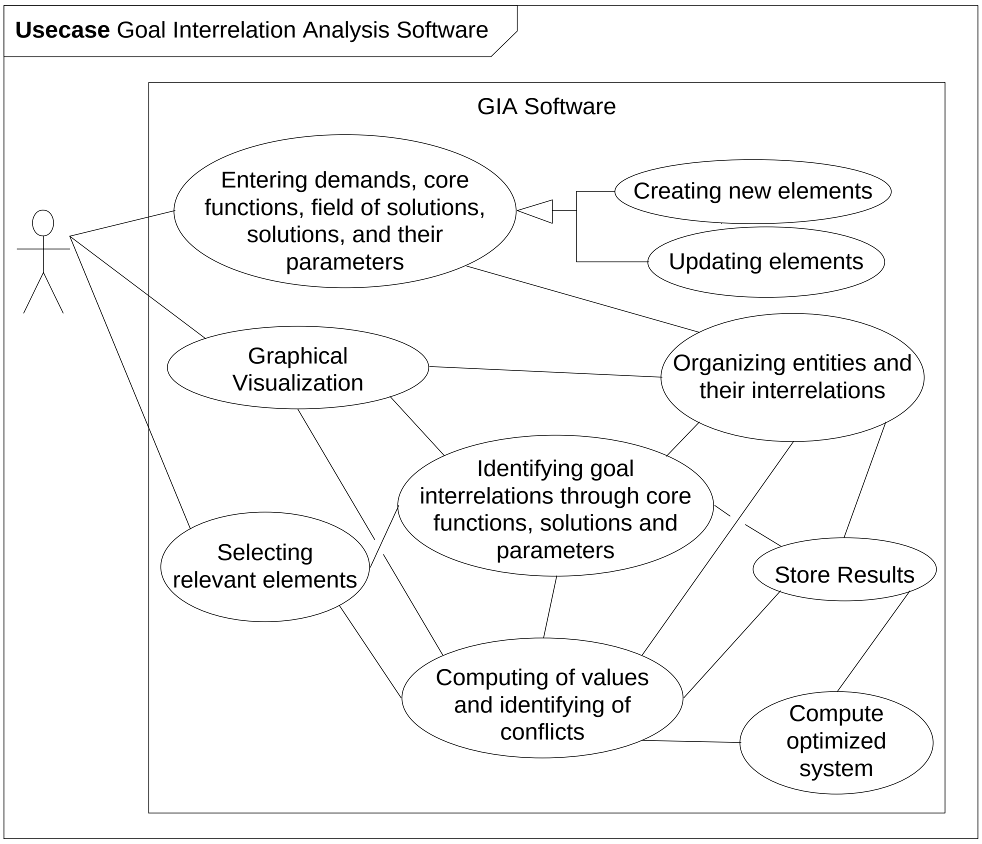

Usecase Goal Interrelation Analysis Software
GIA Software
Entering demands, core functions, field of solutions, solutions, and their parameters
Creating new elements
Updating elements
Graphical Visualization
Organizing entities and their interrelations
Identifying goal interrelations through core functions, solutions and parameters
Selecting relevant elements
Store Results
Computing of values and identifying of conflicts
Compute optimized system

Figure 5-5 Use-Case diagram for the GIA-software

Following, Figure 5-6 shows an block-definition diagram which defines the main subsystems of the GIA-software tool according to the identified use cases and requirements. Consequently the program can be subdivided into 5 parts: the data acquisition, the data organization, the data processing, the export/output, and the visualization part. The data acquisition part has to provide a system which enables the user to enter the information about the PPS. Based on these acquired information the data organizing part has to deal with the transformation of these information into a manageable form, for example an OWL-scheme or a relational database. This data furthermore has to be accessed by the data processing part which derives information from it, like the identification of goal interrelation, computation of conflicts, and the computation of an optimized system. The export/output part of the software has to deal with the storage of the received information as well as the transformation of it in order to be used in other programs. Finally the visualization unit provides an graphical feedback to the user, of the acquired information, and the identified interrelation, conflicts and optimization potential. Thus all main parts of the software tool to analyzed goal interrelations are clearly identified serving as a source for a subsequent implementation.

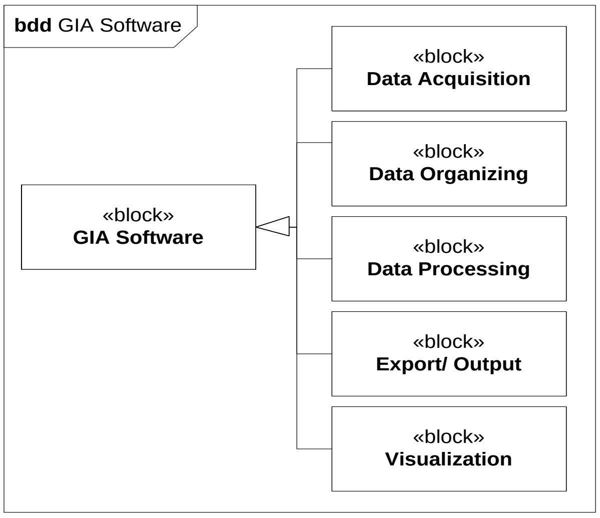


Figure 5-6 Block-Definition diagram for the GIA-software tool

Finally the internal-block diagram of the GIA-Software, which supports the clarification of the internal flow of information, is shown in Figure 5-7. Consequently the data is entered into the system through the data acquisition part and further organized by the data organization part. This organized information is further sent to the visualization unit, the processing unit and the export/output unit. The data processing part is computing the information and derives results from it, which are sent to the visualization unit as well as to the export output unit. The visualization part receives, as already mentioned its information from the data organization and the data processing unit in order to provide a graphical representation of it. Lastly the data output/export part of the software transform and stores the information which therefore exits the system. Concluding with the diagram an holistic representation of the interconnection between all units of the GIA-software is giving supporting a further implementation.

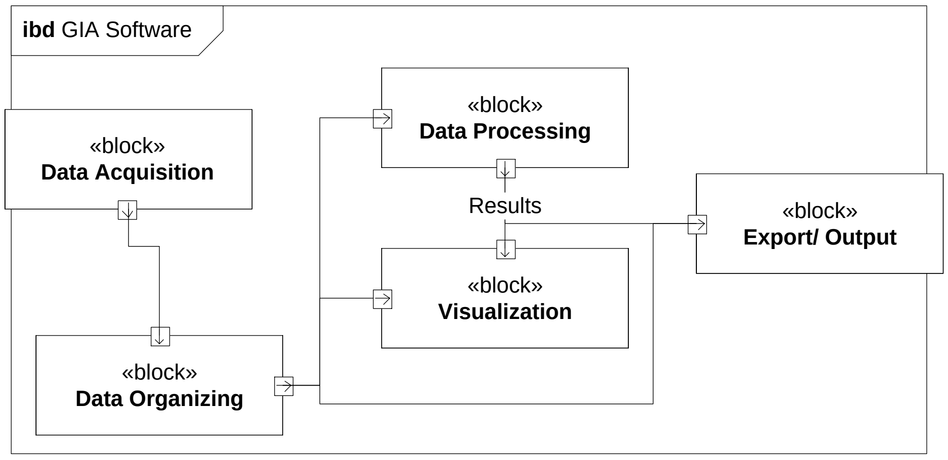


Figure 5-7 Internal-Block diagram for the GIA-software

Recapitulation the whole chapter a complete framework for the implementation of a software tool for an enhanced analysis of goal interrelation is given. Therefore, an enhancement for the approach for GIA presented by (HEPPERLE ET AL. 2011a) and (FÖRG 2010) is shown, adding extra information about the temporal aspects, the belonging scenarios as well as a rule-based knowledge description to it. Based on this approach a formal definition of the approach is given with the description of all basic elements and their attributes as well as their interrelations. Finally the software implementation is regarded by identifying the use cases with the information about the user interaction, the main structure as well as the internal flow of data. Consequently this chapter describes all relevant information which provides together with the listed requirements in chapter 3 a complete framework for the implementation of a software tool which supports the modeling of a PPS in order to analyze goal interrelations.

6 Conceptual Implementation of Computer-Supported Goal Interrelation Analysis

This chapter presents a conceptual implementation of a computer-supported goal interrelation analysis. Therefore an example of a complex PPS is introduced. It considers several lifecycle phases, scenarios and product generation of the Formulas Student team *TUfast*. This example is transformed to a graph-based representation based on the framework for an enhanced GIA shown in chapter 5.1. Besides, a conceptual implementation of a software-tool is shown. It is especially adapted to support a GIA regarding the framework for the implementation of the content of a GIA and for the software described in chapter 5.2 and 5.3. Its functionality is pointed out through the application of the example within the data acquisition, as well as the visualization part of the software.

6.1 Example for Lifecycle Oriented Product Planning

The application of a strategic product planning with goal interrelation analysis will be shown with an example of a PPS for racecar development. The therefore presented example is based on the TUfast project team, introduced in chapter 4.1.2. The PPS, which is presented in this section, is a more complex and more accurate version of the previously introduced example. To achieve this, it has been developed together with the head of engineering of the *TUfast* team 2011, Martin Lacher. The overall objective is the development of a racecar that complies with the Formula Student regulations. It is an excellent example for the GIA, since a racecar is a complex product with a lot of interrelations, there is a new product generation every year, and 2011 has been a split between the electric and the classic combustion engine department. However, the presented PPS has its focus on the project planning, but the deduced consequences can also be transferred to more technical problems, which is a strength of this approach. Thus, the example is described by firstly presenting the recent developments at *TUfast* and transferring these to an enhanced graphical representation of the PPS.

6.1.1 Strategic Product Planning of *TUfast*

TUfast developed eight racecars since their foundation in 2002 (TUFAST E.V. 2011a). Consequently the team is every year confronted with the *Strategic Product Planning* (SPP), considering all relevant goals and possible solutions within. This allows *TUfast* to be an excellent example for a goal interrelation analysis in the strategic product planning. Every season, the main goal for *TUfast* is to obtain a maximum competitive advantage. Thus they are not specializing on one discipline but are trying to developed a car with is balanced between the various disciplines. To obtain this the team has to consider goals from various phases of the product lifecycle among which are especially the project planning, the production and the utilization.

The example regards a period between 2007 and today, whereas the developed racecars and their main innovations can be seen schematically in Figure 6-1. So the nb08 was a racecar with a 600ccm four cylinder motor with a monocoque carbon chassis. A year later they introduced a new hybrid chassis made from carbon and steel to enable easier maintenance of the motor and gearbox which was improved with the nb10 in production and construction and successfully saved a lot of weight. In the following season the e-technology section was introduced with the development of an electric engine system.

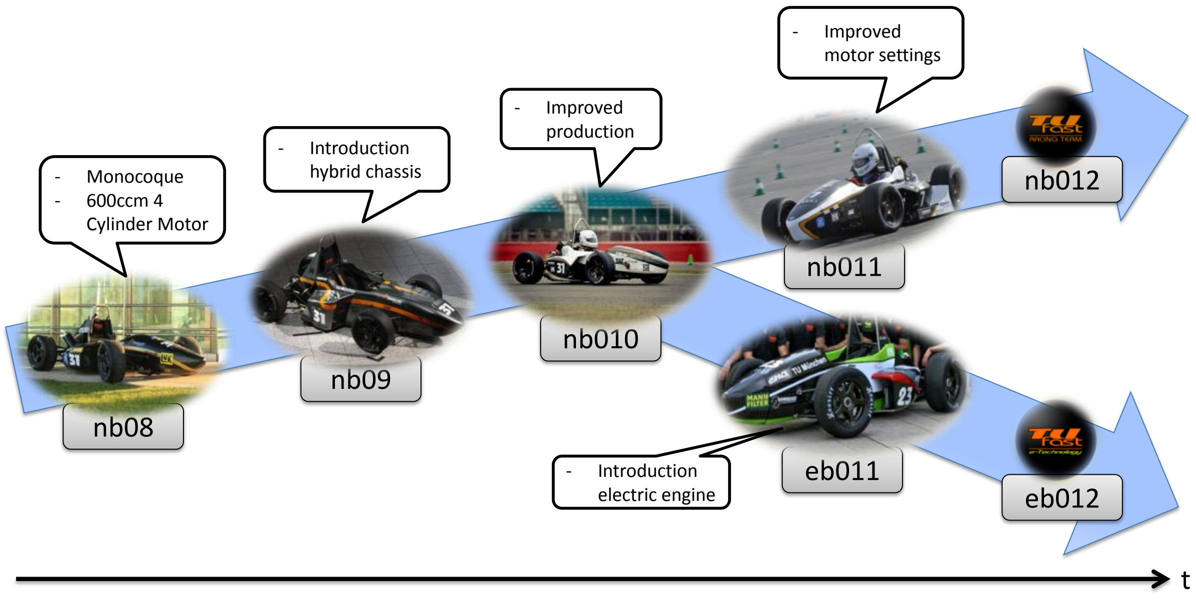

- Monocoque
- 600ccm 4 Cylinder Motor
- Introduction hybrid chassis
- Improved production
- Improved motor settings
nb08
nb09
nb010
nb011
nb012
- Introduction electric engine
eb011
eb012
t

Figure 6-1 Racecars developed by TUfast and their main innovations

However, *TUfast* is every year confronted with the decision to select strategies and solution for all fields of product planning. These are for example the budget acquisition, the human resources, as well as the motor and chassis development and production. The example is generally based on project planning task, but the methods are equally applicable with a focus towards technical demands and solutions.

The budget is usually acquired with donations of from several companies from related fields of interest. Sponsorships can occur in various ways, for example monetary, giving advisory, donating components or materials, or providing production facilities. These sponsorships are lasting one year, so every season the sponsors have to be recruited again. *TUfast* has some constant sponsors since several years, but the concrete composition of them changes every year and depends also on free human resources to identify and to build up possible sponsors.

In addition, the team members are part of project planning for every season. Since *TUfast* is a voluntary project, there is a high fluctuation of the participants, which needs to be organized. Thus members have to be assigned to the different jobs and divisions within *TUfast*. This depends on the amount of volunteers, their competences, and the overall strategy of the season, for example if the development of the motor or the chassis needs to be strengthened or if there is an equal distribution of the human resources. With the resulting resources of person-days new projects in e.g. development, production or administration can be planned accordingly.

One of the divisions is responsible for the chassis of the racecar. During the recent years, three basic types of reasonable chassis have been identified: a tube frame, a carbon-monocoque chassis, or a hybrid chassis. Thereby the hybrid chassis is a combination of a carbon monocoque front with a tube-frame back. All of these solutions for the chassis have their own advantages and disadvantages like the needed production quality and the costs. However, for more adequate planning it is said that using the solution from last season saves around one quarter in overall time and person-days.

Another division is handling the motor development. There, additionally restricted because of the Formula Student rules, it can be chosen among three strategies for the combustion motor and two for the electric motor: Using a motor with four, two or one cylinder on the one hand or two electric motors or one with a gearbox for the other. The advantages and disadvantages of these motors are for example the overall quality, and the requested maintenance time. Equally to the development of the chassis, using the same motor from the last season saves overall time and person days.

Another task is to select the right strategy for the production of these parts. Commonly parts can be produced at the *TUfast* garage, at the faculty garage, at a sponsor or completely externally. These solutions differ among others especially in costs, produced quality, required person days from team members and overall time due to longer transportation and queuing times.

It can be seen that the selection between the mentioned strategies and solutions for the goals of *TUfast* in one field alone is already a difficult task. Further with considering all interrelations between the various fields it is an even more complex task. To get an complete overview on the situation, a graphical representation is presented in the next section.

6.1.2 Graphical Representation of *TUfast* Goal Interrelation Analysis

In order to consider the SPP of *TUfast* during the recent years as a simplified example, the information has to be converted into a manageable format. This is done by creating a graphical representation of the goal interrelation planning system. The graph-based representation is based on the one presented in chapter 2.2.4, and enhanced according to the requirements (see chapter 3) and the framework (see chapter 5). Based on this approach the SPP of *TUfast* is shown and further discussed.

6.1.2.1 Enhanced Graphical Representation for Goal Interrelation Analysis

A graphical representation is a good way to deal with complex systems like the SPP process of *TUfast*. The approach for a graphical representation for a goal interrelations analysis, presented in chapter 2.2, considers demands, goals, core functions, solutions, field of solutions and parameters connected through simple graphs. In general this is a good basis for the visualization of the system. However, since interrelations between these elements can have very different properties and elements themselves have often complex behavior especially adapted techniques are needed to represent these circumstances. This is done by using techniques derived from SysML to define meta-elements and specific graphs in order to enhance the existing approach for their representation. Thus a ‘Solution’ can have several parameters, to which they are connected by one of the relations ‘LeadsTo’, ‘Requires’, and ‘Affects’, which will be discussed in the next section. The ‘Field of Solution’ is a collection of solution, why it needs to define a set of parameters in order to make the collected solutions fully exchangeable. Finally every ‘Parameter’ has to be described by its type, its maximum value and its preferred value. Further, to consider more complex situations, all three elements can be en- and disabled by a specific rule-based description of more complex circumstances. Altogether, with these properties, the three elements can be described with all necessary attributes as it has been requested in chapter 3.

The other point which needs to be regarded more closely is the assignment of parameters to solutions or FoS. Figure 6-2 gives a graphical representation of the various types of the relation between a parameter and solutions or a FOS. Thus a solution is connected to a parameter by the relation ‘LeadsTo’ when a specific value is set by a not-addable solution. For example a good production facility leads to a good value of the parameter ‘quality of the production’, while two medium facilities are not. Instead they are still leading to a medium value. The relation ‘Affects’ describes the situation when a solution affects the amount of a parameter and several solutions can be summarized. This case is for example when the solution of a specific motor costs an amount of the parameter ‘Budget’ and the solution of a specific chassis as well; both amounts are summarized for the complete price. Finally the relation ‘Requires’ describes the situation when a solution needs a specific value of a parameter to be enabled. For example the solution to use a specific technology requires a high value of the parameter ‘quality of production’. Using these definitions the strategic product planning system of *TUfast* during the recent years can be described in the next section in order to perform a complete goal interrelation analysis.

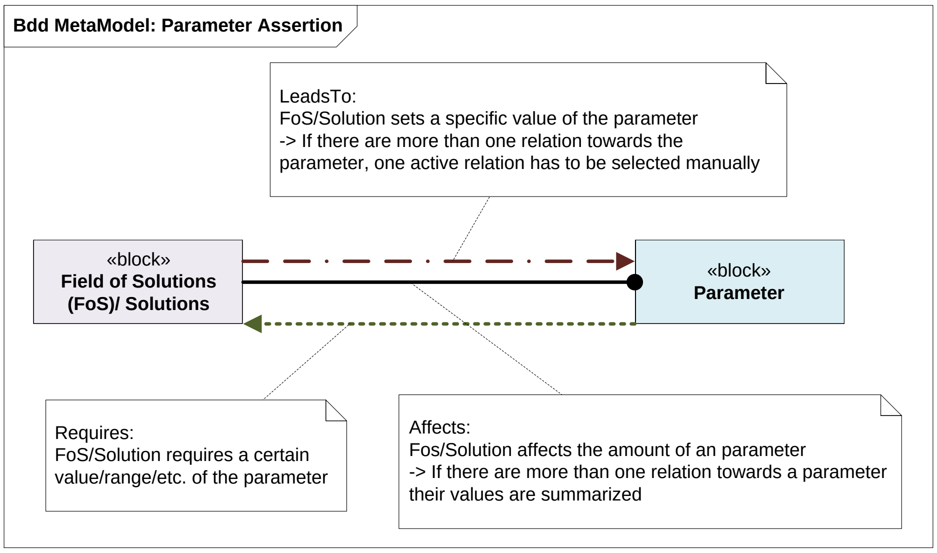

Bdd MetaModel: Parameter Assertion
LeadsTo:
FoS/Solution sets a specific value of the parameter
-> If there are more than one relation towards the parameter, one active relation has to be selected manually
«block»
Field of Solutions (FoS)/ Solutions
«block»
Parameter
Requires:
FoS/Solution requires a certain value/range/etc. of the parameter
Affects:
Fos/Solution affects the amount of an parameter
-> If there are more than one relation towards a parameter their values are summarized

Figure 6-2 Definition of the relations types between parameters and solutions or field of solutions

6.1.2.2 *TUfast* Graphical Goal Interrelation Analysis

In this section the graphical representation of an exemplary *TUfast* strategic product planning process is presented. The applied visualization is based on the approach presented by (FÖRG 2010; HEPPERLE ET AL. 2011a) and enhanced in the previous section. With this graphical representation the development of the *TUfast* racecar can be systematically analyzed to identify the main demands, goals, core functions, solutions, the main strategies and the temporal development. Furthermore, based on these results solutions are grouped into field of solutions which makes the different solutions comparable and parameters of the solutions are defined globally in order to identify dependencies between different fields. Since generating a graphical representation is an iterative process, where the results from the first cycle are used as an input for a second cycle in order to improve results, this section shows the result of the entire process. Thus a general overview of the temporal development and the strategies is given, following by a more detailed explanation of the various lifecycle phases. All values used in the example are just estimated and not real values. Since the objective of this example is to show the complexity of a strategic product planning, enabling a goal interrelation analysis and to finally identify possible conflicts between goals, it is not necessary to have an exact representation of the values of a system, but the general interrelations are important.

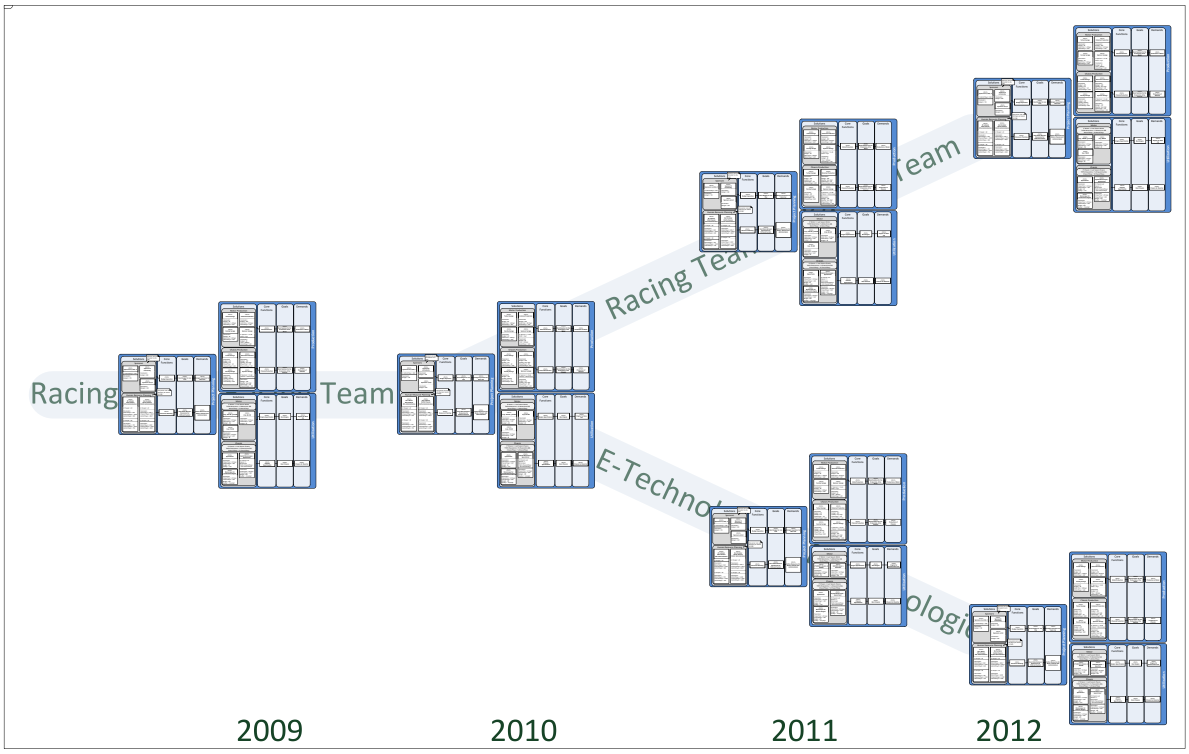


Figure 6-3 Overview of the TUfast strategic product planning

The PPSs for the development of racecars can be seen in Figure 6-3. The figure shows schematically that it is necessary to perform a new planning process for every season by having one PPS, for every single car. These PPS include the lifecycle phases of the graph based approach for the goal interrelations analysis. In addition, the consideration of a second scenario is presented. Hence, until 2010 there was just one racecar to design by the *TUfast* Racing Team, which uses a combustion motor. With the founding of the e-technology division of *TUfast* an additional product has to be planned. This results in an overall of six PPSs which can be seen in the figure.

Figure 6-3 gives also a first impression of the complexity of the product planning process and the connected goal interrelation analysis within *TUfast*. During the analysis, all interrelations between goals and solutions have to be considered. Thus not only the solution of a single goal, but also of every lifecycle phase, of the preceding and succeeding products, and finally the different scenarios need to be taken into account. Furthermore, to get an overview on the goal interrelation analysis for each product, the single lifecycle phases are presented in the following. Due to the fact that demands and goals for the racecar are changing only slightly every season, only important details of the PPS are shown in this section. Therefore, for the combustion racecar the phases project planning, production and utilization are presented while for the electric racecar the phase utilization is shown, being representative for the entire example.

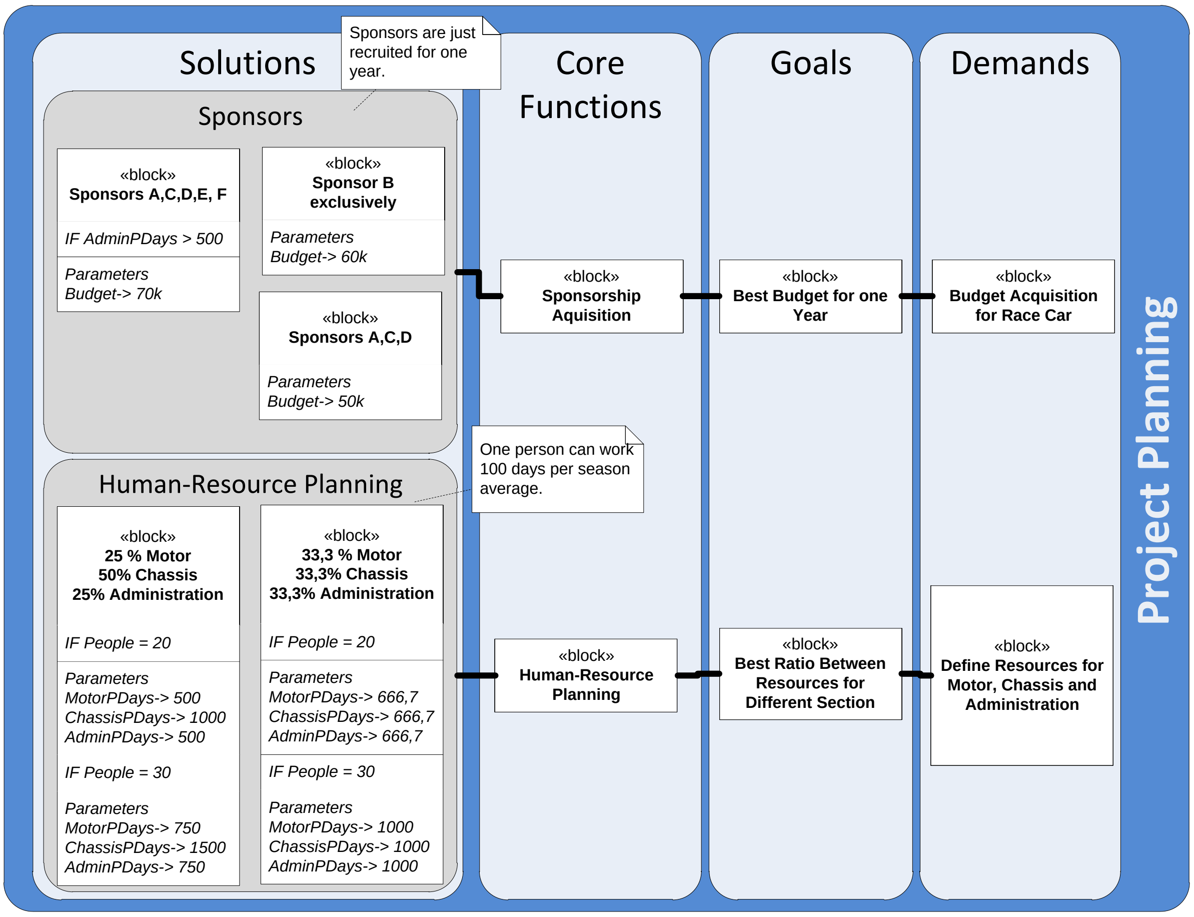

Solutions
Sponsors are just recruited for one year.
Sponsors
«block» Sponsors A,C,D,E, F
IF AdminPDays > 500
Parameters Budget-> 70k
«block» Sponsor B exclusively
Parameters Budget-> 60k
«block» Sponsors A,C,D
Parameters Budget-> 50k
Human-Resource Planning
One person can work 100 days per season average.
«block» 25 % Motor 50% Chassis 25% Administration
IF People = 20
Parameters MotorPDays-> 500 ChassisPDays-> 1000 AdminPDays-> 500
IF People = 30
Parameters MotorPDays-> 750 ChassisPDays-> 1500 AdminPDays-> 750
«block» 33,3 % Motor 33,3% Chassis 33,3% Administration
IF People = 20
Parameters MotorPDays-> 666,7 ChassisPDays-> 666,7 AdminPDays-> 666,7
IF People = 30
Parameters MotorPDays-> 1000 ChassisPDays-> 1000 AdminPDays-> 1000
Core Functions
«block» Sponsorship Aquisition
«block» Human-Resource Planning
Goals
«block» Best Budget for one Year
«block» Best Ratio Between Resources for Different Section
Demands
«block» Budget Acquisition for Race Car
«block» Define Resources for Motor, Chassis and Administration
Project Planning

Figure 6-4 Project planning phase of a TUfast racecar

Firstly the project planning is regarded, which can be seen in Figure 6-4. The project planning is not a typical lifecycle phase but is needed in this example to consider some superordinate administrative goals which cannot be assigned to a specific phase. In the upper part, the goal 'Best Budget for one Year' which is connected through a core function to the field of solutions 'Sponsors'. For 'Sponsors' there are several different solutions offered, which vary in the amount of donated budget or in the requirements for their acquisition. Thus 'Sponsors A,C,D,E,F' can just be chosen, if there are enough members working in the administration, represented by the rule 'IF AdminPdays > 500', saying that the team needs to have at least 500 person days for that solution. The lower part of the figure shows the goal 'Best Ratio Between Resources for Different Sections' which can be satisfied by different solutions for the human-resource planning. These solutions are depending on the overall strategy, for example having a strong chassis division as well as on the amount of *TUfast's* volunteer members for the season.

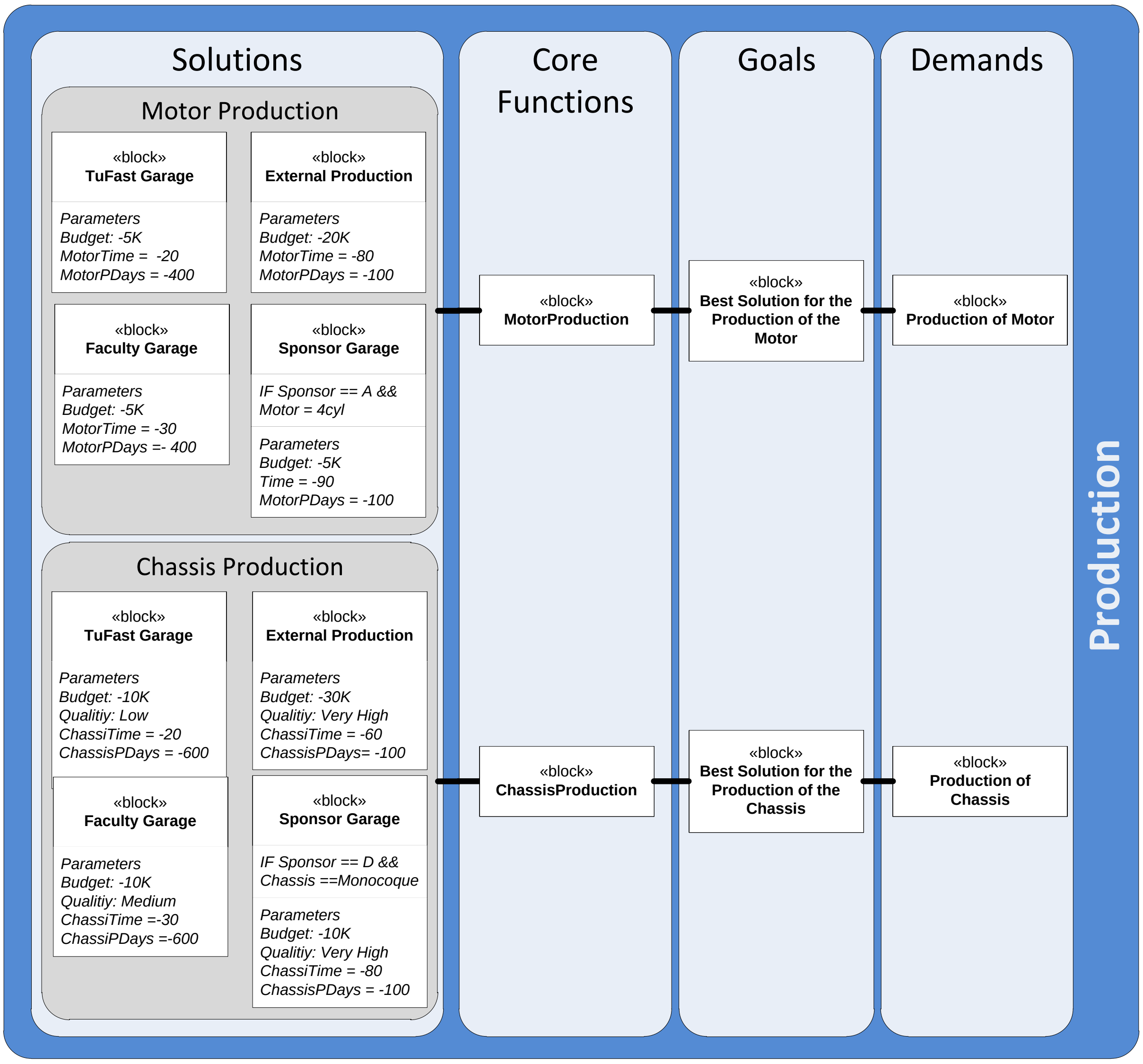


Figure 6-5 Production phase of a TUfast-Racecar

Next, the production phase is regarded which is graphically represented in Figure 6-5. On the upper half, the goal ‘Best Solution for the Production of the Motor’ can be fulfilled by choosing one from the four solutions for the motor production. The solutions differ from each other through the costs, the overall time and the person-days needed from team members. Additionally the ‘Sponsor Garage’ requires the sponsor to be a specific one which can handle a specific motor. Also the goal ‘Best Solution for the Production of the Chassis’ can be fulfilled by several solutions which differ by the overall time, the required person days and additionally the produced quality.

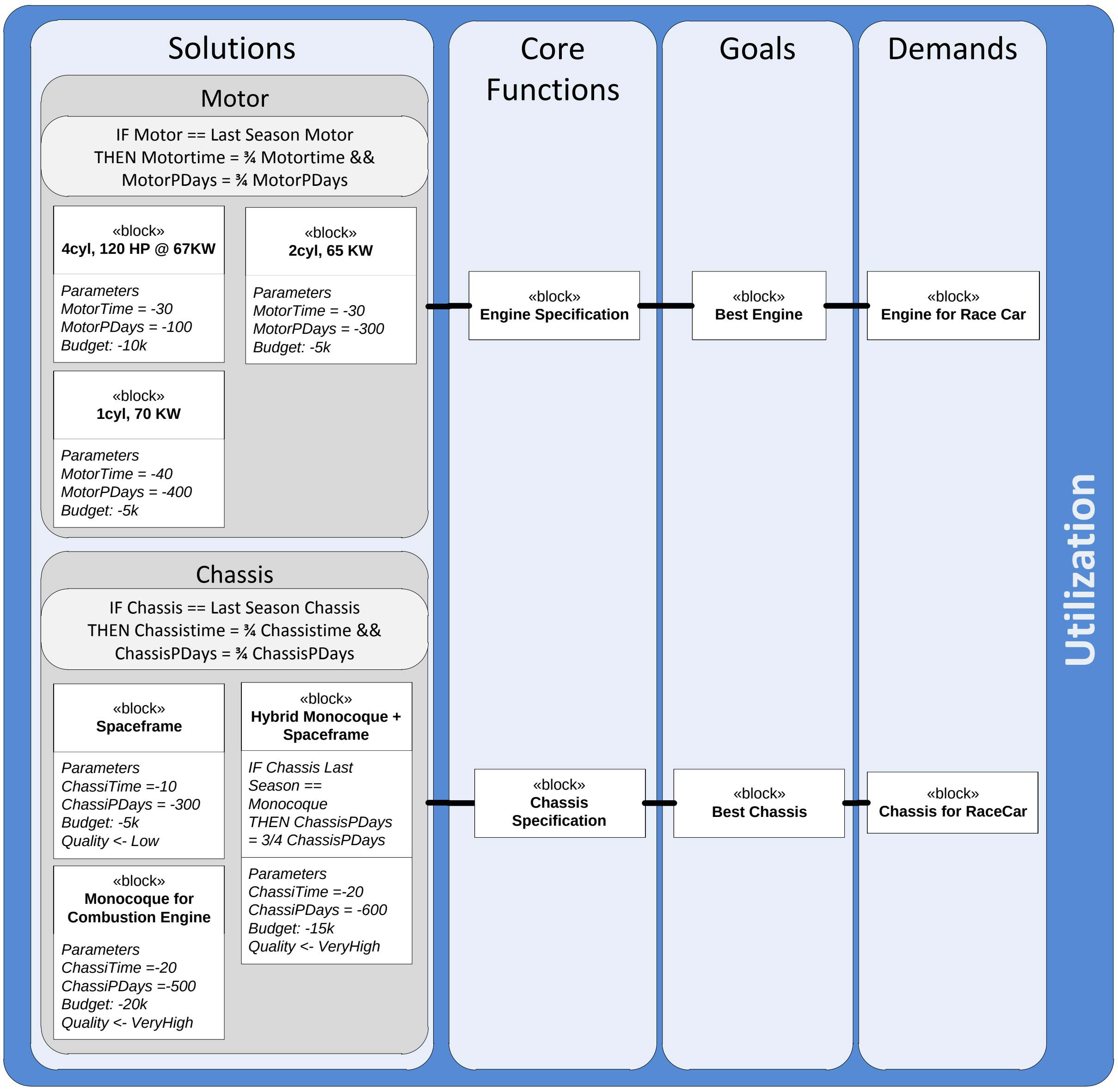

Solutions
Core Functions
Goals
Demands
Motor
IF Motor == Last Season Motor THEN Motortime = ¾ Motortime && MotorPDays = ¾ MotorPDays
«block» 4cyl, 120 HP @ 67KW
Parameters MotorTime = -30 MotorPDays = -100 Budget: -10k
«block» 2cyl, 65 KW
Parameters MotorTime = -30 MotorPDays = -300 Budget: -5k
«block» 1cyl, 70 KW
Parameters MotorTime = -40 MotorPDays = -400 Budget: -5k
Chassis
IF Chassis == Last Season Chassis THEN Chassistime = ¾ Chassistime && ChassisPDays = ¾ ChassisPDays
«block» Spaceframe
Parameters ChassiTime =-10 ChassiPDays = -300 Budget: -5k Quality <- Low
«block» Hybrid Monocoque + Spaceframe
IF Chassis Last Season == Monocoque THEN ChassisPDays = 3/4 ChassisPDays
Parameters ChassiTime =-20 ChassiPDays = -600 Budget: -15k Quality <- VeryHigh
«block» Monocoque for Combustion Engine
Parameters ChassiTime =-20 ChassiPDays =-500 Budget: -20k Quality <- VeryHigh
«block» Engine Specification
«block» Best Engine
«block» Engine for Race Car
«block» Chassis Specification
«block» Best Chassis
«block» Chassis for RaceCar
Utilization

Figure 6-6 Utilization phase of TUfast-Racecar with combustion engine

The phase 'Utilization' of the product lifecycle for combustion engines is shown in. In it, the goal 'Best Engine' can be met through specifying the combustion motor. The three solutions differ in price, overall time and person-days, due to higher expenses of maintenance during the utilization. Further, a rule for field of solutions 'Motor' specifies that using the motor from last season reduces the overall time and the required person days to one quarter, because of the gained experience. The other goal 'Best Chassis' can be fulfilled by choosing one of the three chassis. The solutions vary in the overall time, the person days, the required budget and additionally in the required quality for their production. The solutions for the chassis have the rule-based advantage when the chassis from the last season is chosen like the motor as well. Further the solution ‘Hybrid Monocoque + Spaceframe’ can have improved parameters, when a specific solution was used in the last season.

Finally the phase ‘Utilization’ in the case of an electric engine is shown in Figure 6-7. The solutions for the goal ‘Best Engine’ have the same parameters and rules for the field of solution like the ones for the combustion engine. The solutions for the goal ‘Best Chassis’ are described by the same parameters and rules. A fact to be pointed out is the rule for the hybrid monocoque and space frame. It says that if any monocoque was used in the last season, the hybrid has advantages in the amount of required person-days. Thus, it respects the circumstances that when introducing the electric racecar, the front monocoque from the combustion racecar could be used with just adapting the space frame in the back to handle the electric engine.

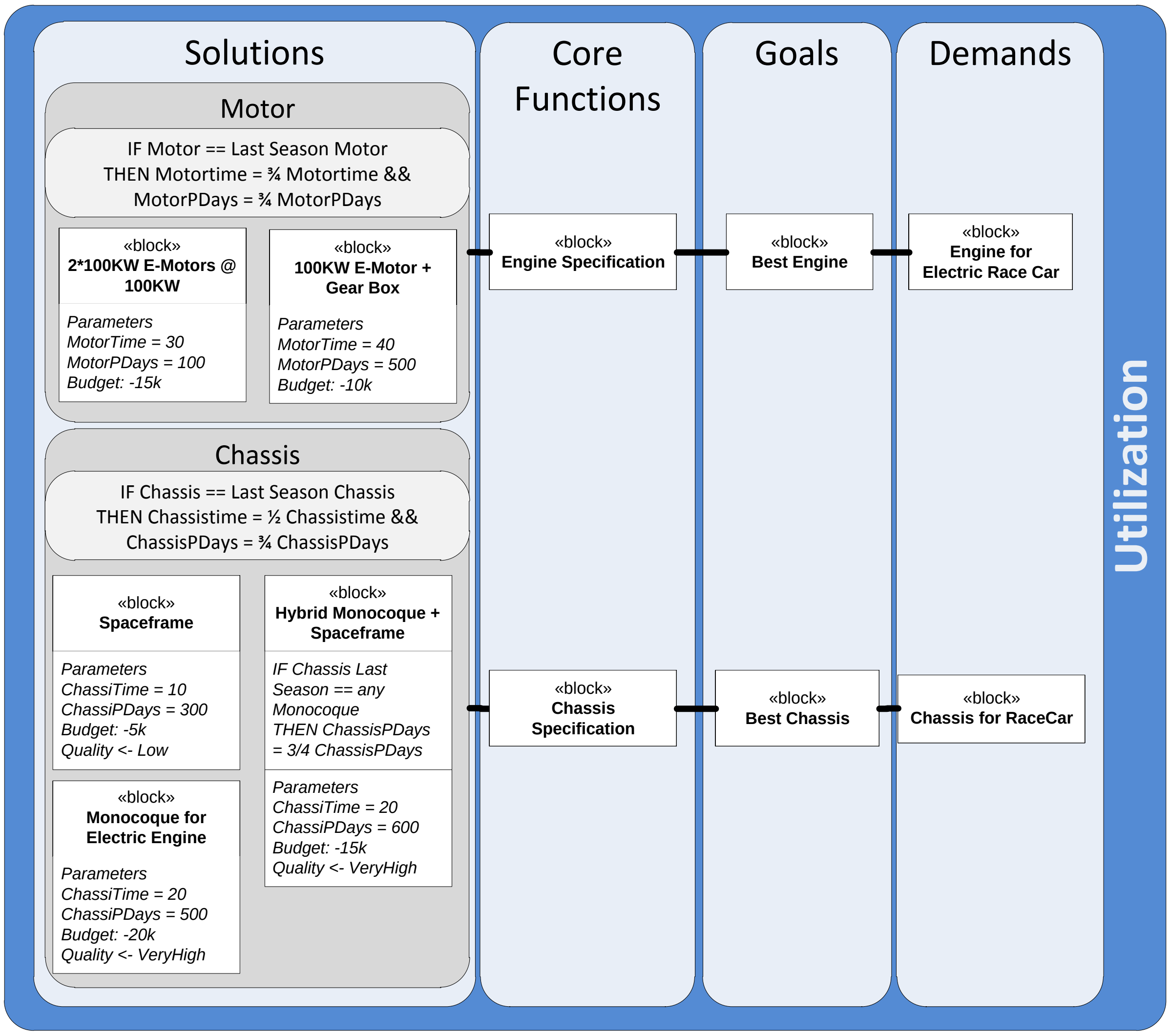

Solutions
Core Functions
Goals
Demands
Utilization
Motor
IF Motor == Last Season Motor THEN Motortime = ¾ Motortime && MotorPDays = ¾ MotorPDays
«block» 2*100KW E-Motors @ 100KW
Parameters MotorTime = 30 MotorPDays = 100 Budget: -15k
«block» 100KW E-Motor + Gear Box
Parameters MotorTime = 40 MotorPDays = 500 Budget: -10k
Chassis
IF Chassis == Last Season Chassis THEN Chassistime = ½ Chassistime && ChassisPDays = ¾ ChassisPDays
«block» Spaceframe
Parameters ChassiTime = 10 ChassiPDays = 300 Budget: -5k Quality <- Low
«block» Hybrid Monocoque + Spaceframe
IF Chassis Last Season == any Monocoque THEN ChassisPDays = 3/4 ChassisPDays
Parameters ChassiTime = 20 ChassiPDays = 600 Budget: -15k Quality <- VeryHigh
«block» Monocoque for Electric Engine
Parameters ChassiTime = 20 ChassiPDays = 500 Budget: -20k Quality <- VeryHigh
«block» Engine Specification
«block» Best Engine
«block» Engine for Electric Race Car
«block» Chassis Specification
«block» Best Chassis
«block» Chassis for RaceCar

Figure 6-7 Utilization phase of TUfast-Racecar with electric engine

Summarizing all these blocks, an example for simplified strategic product planning of a racecar is given upon which a goal interrelation analysis has to be performed. Although the process is already simplified, the complexity is difficult to manage. The solutions are connected to global parameters in the same lifecycle phase, other phases and even other product generation or scenarios, resulting in a lot of relation being necessary to consider. Analyzing the example stochastically there can be $3 * 2 * 4 * 4 * 3 * 3 = 864$ variants for a combustion racecar and $3 * 2 * 4 * 4 * 2 * 3 = 576$ variants for an electric racecar. Consequently seen over the whole example $864^4 * 576^2 =$ 184.884.258.895.036.416 variants are theoretically possible, without considering rule-based dependencies. A first step towards managing this complexity is understanding the interrelation which will be further discussed in the following section.

6.1.2.3 Further Characteristics of the Example

In order to receive a deeper understanding of the presented example, some further characteristics have to be explained. Thus the presented *Fields of Solutions* (FoS) are regarded disconnected from their goals in order to understand the interchangeability of the solution and their influences on the parameters. Following, a global definition of the parameters is given so that finally the interrelations between FoS and parameters can be presented graphically.

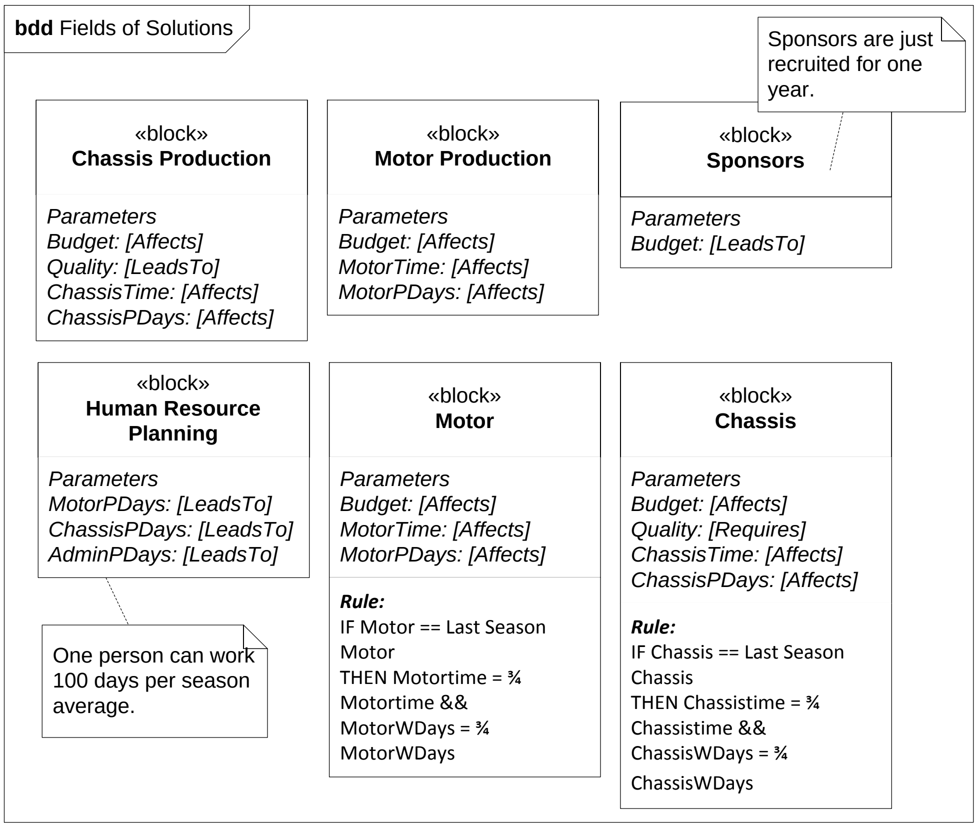
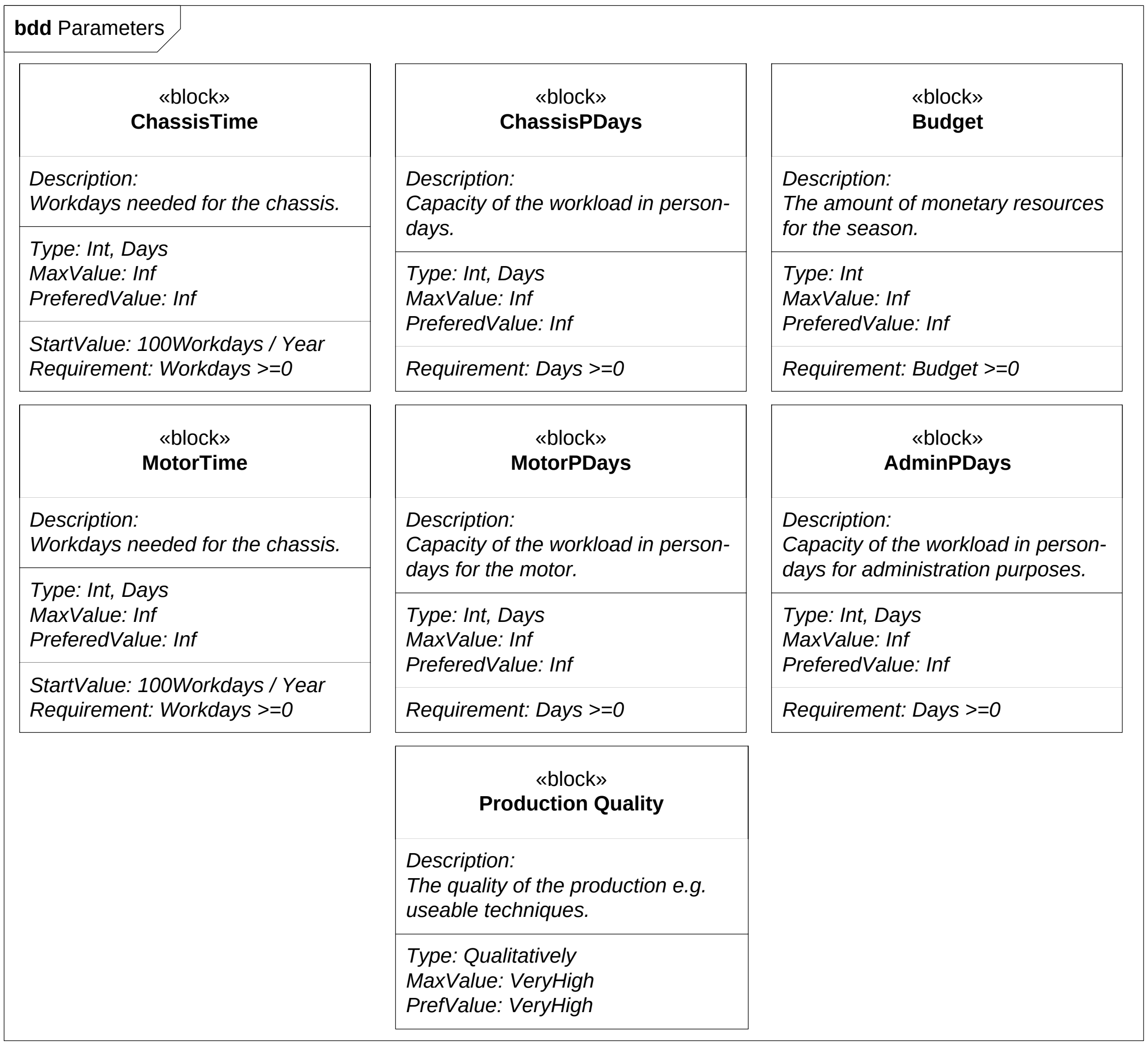


Figure 6-8 Field of Solutions of the TUfast product-planning system.

FoS are collection of fully interchangeable solutions, which are connected to the same core functions and therefore satisfying the same goals. The six FoS used in the example can be seen in Figure 6-8 . The FoS are described according to the enhanced graphical representation introduced in chapter 6.1.2.1. Thus, it can be seen what kind of parameter the solutions the according field are connected to, as well as how they are influencing them. Moreover the rules for the solutions from the field 'Motor' and 'Chassis' are given. Additionally further information is attached to the field 'Human Resource Planning' and 'Sponsors' in order to give more detailed information about the context.


bdd Parameters

«block»
ChassisTime

Description:
Workdays needed for the chassis.

Type: Int, Days
MaxValue: Inf
PreferedValue: Inf

StartValue: 100Workdays / Year
Requirement: Workdays >=0

«block»
ChassisPDays

Description:
Capacity of the workload in person-days.

Type: Int, Days
MaxValue: Inf
PreferedValue: Inf

Requirement: Days >=0

«block»
Budget

Description:
The amount of monetary resources for the season.

Type: Int
MaxValue: Inf
PreferedValue: Inf

Requirement: Budget >=0

«block»
MotorTime

Description:
Workdays needed for the chassis.

Type: Int, Days
MaxValue: Inf
PreferedValue: Inf

StartValue: 100Workdays / Year
Requirement: Workdays >=0

«block»
MotorPDays

Description:
Capacity of the workload in person-days for the motor.

Type: Int, Days
MaxValue: Inf
PreferedValue: Inf

Requirement: Days >=0

«block»
AdminPDays

Description:
Capacity of the workload in person-days for administration purposes.

Type: Int, Days
MaxValue: Inf
PreferedValue: Inf

Requirement: Days >=0

«block»
Production Quality

Description:
The quality of the production e.g. useable techniques.

Type: Qualitatively
MaxValue: VeryHigh
PrefValue: VeryHigh

Figure 6-9 Parameter-Definition of the TUfast product-planning system

The solutions used in the example are influencing altogether seven parameters. All these parameters can be seen with their global definition in Figure 6-9. In the definition a more detailed description of every parameter is given as well as their type, their maximum values, and their preferred value. The preferred value tells thereby, whether high or low values should be desired. Additionally the parameters 'Chassis Time' and 'Motor Time' set an initial value to 100 workdays each season. Finally all parameters apart from the ‘Production Quality’ have the general requirement to have a non-negative value. Summarizing all these information a holistic understanding of the used parameters is given.

Based on the presented information on parameters and FoS a graphical representation on their relation is shown in Figure 6-10. The graphs are shaped according to the definition given in chapter 6.1.2.1 and their type. Since several fields of solutions are connected to the same parameter, there is a lot of potential for possible conflict. For example the ‘Sponsors’ are leading to a specific value of the budget which can be affected by four other solutions from the ‘Chassis’, ‘Chassis Production’, ‘Motor’, and ‘Motor Production’. Additionally it is to mention that the shown graphs represent only direct parameter assignments. If an parameter is connected to a solution by a rule, like the ‘hybrid monocoque + space frame’ is connected to the normal ‘monocoque’, it is not mentioned in the image. Nevertheless it can be seen that a lot of interrelations have to be considered, which makes it difficult for humans managing the system’s complexity. Consequently a computer-supported technique for the goal interrelation analysis is needed.

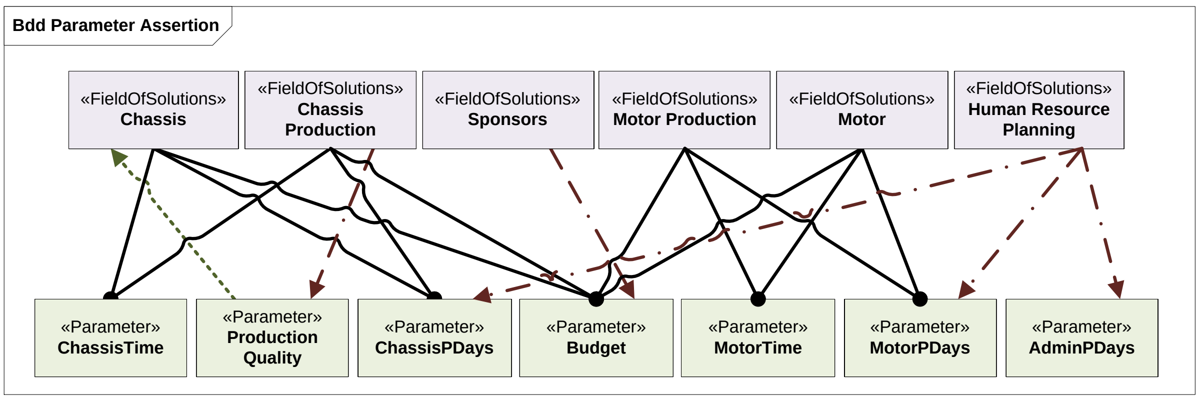

Bdd Parameter Assertion
«FieldOfSolutions» Chassis
«FieldOfSolutions» Chassis Production
«FieldOfSolutions» Sponsors
«FieldOfSolutions» Motor Production
«FieldOfSolutions» Motor
«FieldOfSolutions» Human Resource Planning
«Parameter» ChassisTime
«Parameter» Production Quality
«Parameter» ChassisPDays
«Parameter» Budget
«Parameter» MotorTime
«Parameter» MotorPDays
«Parameter» AdminPDays

Figure 6-10 Field of solutions and the connected parameters of the TUfast product-planning system.

6.2 Example for a Goal Interrelation Analysis Supporting Software Tool

This section describes a conceptual implementation for a software tool supporting the goal interrelation analysis. Thus, a solution for an implementation of a software tool is presented, based on the requirements in chapter 3 and the presented framework in chapter 5. The software tool uses a semantic web to store and organize the PPS. Furthermore it enhances the implementation with additional functions. In the following section an overview on the conceptual implementation is provided. Therefore, a general workflow for the software is described as well as the main parts of the software, the data acquisition and visualization. Additionally the previously presented example of the *TUfast* racecar development is applied within the implementation. However, the presented solution is not a working implementation, but just a concept which could serve as a basis for a later implementation.

6.2.1 General Workflow

In order to introduce the software concept, a general workflow for it is presented. Hence, the usage of semantic web within the software is described, followed by the main functions. Since the demonstrated software is just a theoretical model of an implementation of a computer-supported GIA, this solution provides only an idea of a final software tool.

As already mentioned in the introduction of this section, the software is based on a semantic web. This means not necessarily that it is an extension of *Protégé*, which was applied for the analysis in chapter 2.3.4.4. A semantic web is used in the form of OWL, to store and access the PPS. This can be supported through especially dedicated APIs, like the *Protégé*-API or OWL-API, which are providing the requested functions for the implementation of a PPS in semantic web. Consequently the various abstract types of elements as well as categories for their characterizations are represented by classes of a semantic web. Additionally the relations between elements can be model through object properties. Having defined these classes and properties, semantic attributes can be added to them enabling a consistency check or other logic computations. For example, member of the class ‘Goal’ can be limited to be not a member of the class ‘Demands’, etc. This leads to complete fundamental framework for an implementation of a PPS which stands out with its logic rule implementations through formal semantic attributes. Thus, the software has a strong basis for the organization and processing of all the required data, which is needed for all main features of it.

The main features as well as the general workflow of the conceptual software can be seen in the sysML activity diagram in Figure 6-11. Accordingly the user has to start the program and either load an existing ontology of a PPS or create a new one. The software loads the ontology or creates a new one with a basic set of classes and object properties for the implementation of a PPS. Having the general basis for further work ready, the software waits for a user input, which can be one of the five receiving signal actions. These actions are the acquisition of data, the visualization, the computation, the output/export, and exiting the program. For the acquisition of data, the user has to enter new data, edit existing one or import it from external sources. In a next step the software creates the according elements in OWL. Usually every element is represented by an individual, which is assigned to various classes, attributes, object-, and data properties in order to describe it completely. For the visualization, the user has to specify what he wants to get visualized, e.g. the whole system, one season, or all goals interrelated to one specific goal. Based on these settings, the software automatically generates a representation of the requested system. This representation can be further investigated by navigating through it and focusing on specific details. In order to perform a further computation, the user has to enter his request, which can be for example the detection of conflicts or an optimal system. This demand is used by the software to generate the requested data and to return in to the user. For an output or export of the software, the user has to define, the name and the requested file format, which will be usually an OWL file, and the software will save the PPS accordingly. Finally the user can exit the program. Since all the presented receiving actions are not in a continuous flow, they can also performed parallel, allowing the user to enter new data while visualizing them.

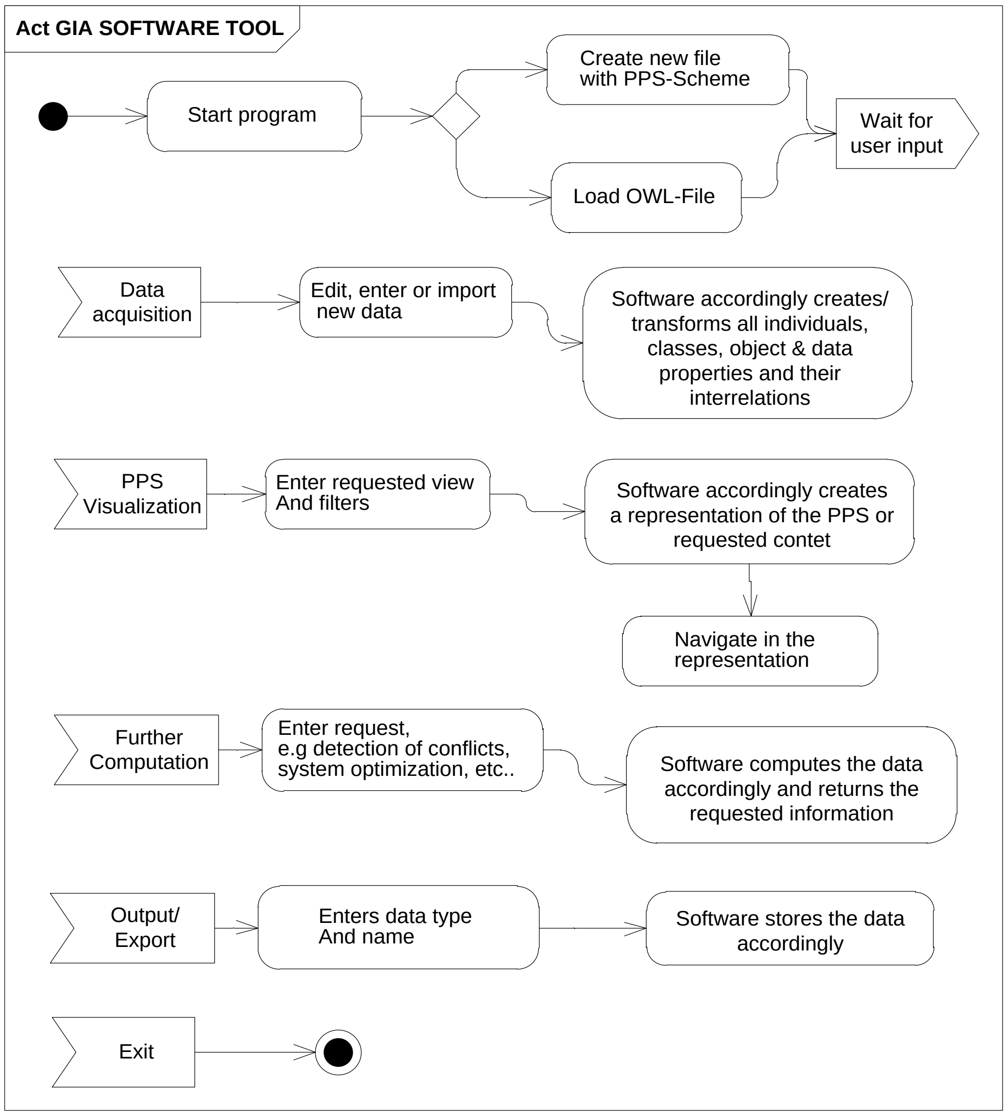

Act GIA SOFTWARE TOOL
Start program
Create new file with PPS-Scheme
Load OWL-File
Wait for user input
Data acquisition
Edit, enter or import new data
Software accordingly creates/ transforms all individuals, classes, object & data properties and their interrelations
PPS Visualization
Enter requested view And filters
Software accordingly creates a representation of the PPS or requested contet
Navigate in the representation
Further Computation
Enter request, e.g detection of conflicts, system optimization, etc..
Software computes the data accordingly and returns the requested information
Output/ Export
Enters data type And name
Software stores the data accordingly
Exit

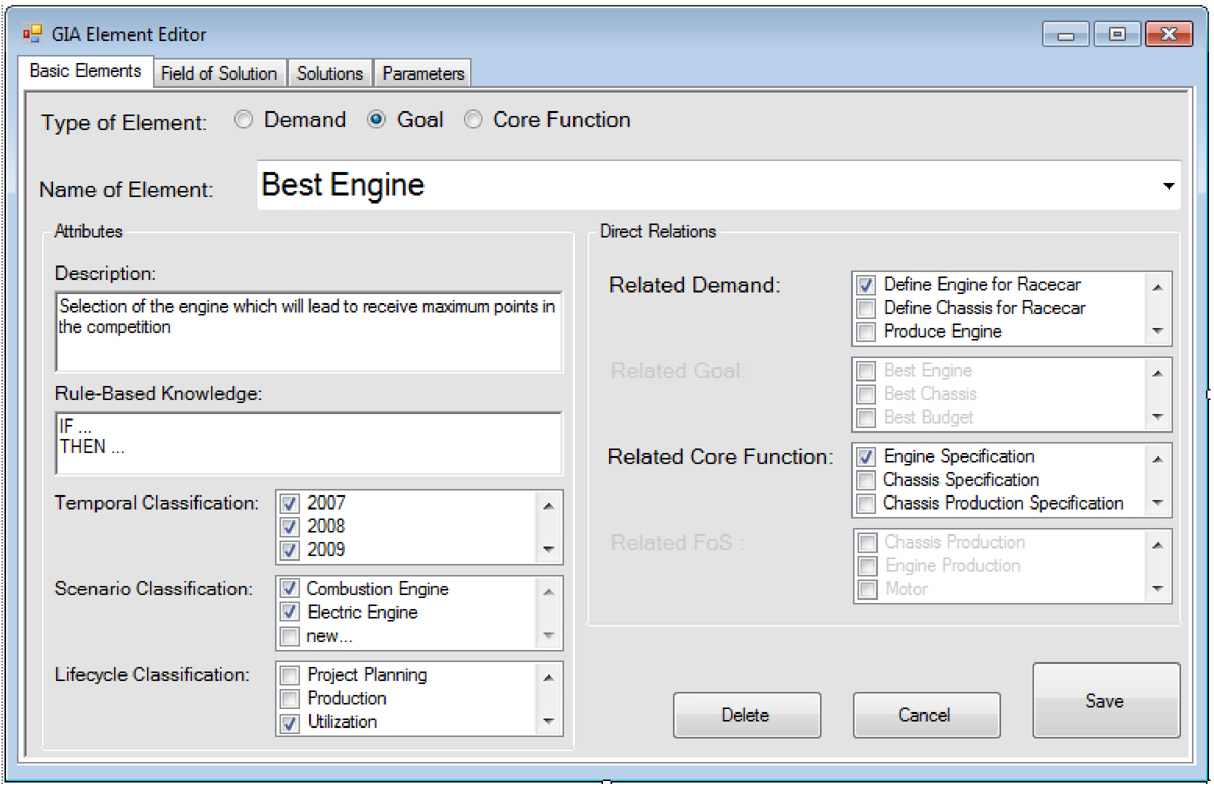
Figure 6-11 General workflow of the conceptual software for a GIA

Having demonstrated the general workflow with the fundamental usage of the semantic webs and the main features of the conceptual software the main interaction points for the user will be presented in the following. These are the data acquisition as well as the visualization.

6.2.2 Data Acquisition

The data acquisition part of the software deals with the collection of information as well as editing and updating the stored data. In general, the analysis of computer-supported modeling techniques has shown that a dedicated, clear and formal interface is important. It eases the acquisition of data for the user and helps to minimize errors. Consequently a formal and especially adjusted *Graphical User Interface* (GUI) for the collection of information about all elements within the PPS is presented. The GUI is created with *Microsoft Visual Studio*, which supports their drawing. The demonstrated order of interfaces is not necessarily the same for entering the information. It just provides an overview on the required interfaces.

The basic elements of a PPS are demands, goals, and core functions. Figure 6-12 shows the GUI for editing these. All of these elements use the same form with activated or deactivated fields according to the chosen type. The type of an element can directly be selected in the top of the form. In the next field the name of the element has to be entered for creating a new element or chosen from the existing ones to reedit them. On the lower left side the grouped box for entering the attributes is shown. In the box a more detailed description of the element, as well as rule-based knowledge can be directly entered. Additional a temporal, scenario, and lifecycle classification of the element can be performed in this field, by checking a box besides the relevant category or by creating new ones. On the right side the direct relations to other elements are regarded. Based on the selected element, the relevant types of elements are active and can be chosen from the list. Finally the element can be deleted, or the changes canceled or saved by clicking the according button. In the figure, the entered information about the goal 'Best Engine' is shown. In the attributes field, are more detailed description of the goal is given and the according selections for the classification are made. Since creating the best engine is a goal for every season and scenario of the racecar development all of them are selected. The chosen lifecycle phase in this case is the field 'utilization'. Furthermore the goal is directly related to the demand 'Define Engine for Racecar' and the core function 'Engine Specification'. Consequently with the form all relevant information of basic elements can be entered directly in an easy and error avoiding way.



Type of Element: Demand [✓ Goal] Core Function

Name of Element: Best Engine

Attributes

Description: Selection of the engine which will lead to receive maximum points in the competition

Rule-Based Knowledge: IF ... THEN ...

Temporal Classification: ✓ 2007, ✓ 2008, ✓ 2009

Scenario Classification: ✓ Combustion Engine, ✓ Electric Engine, new...

Lifecycle Classification: Project Planning, Production, ✓ Utilization

Direct Relations

Related Demand: ✓ Define Engine for Racecar, Define Chassis for Racecar, Produce Engine

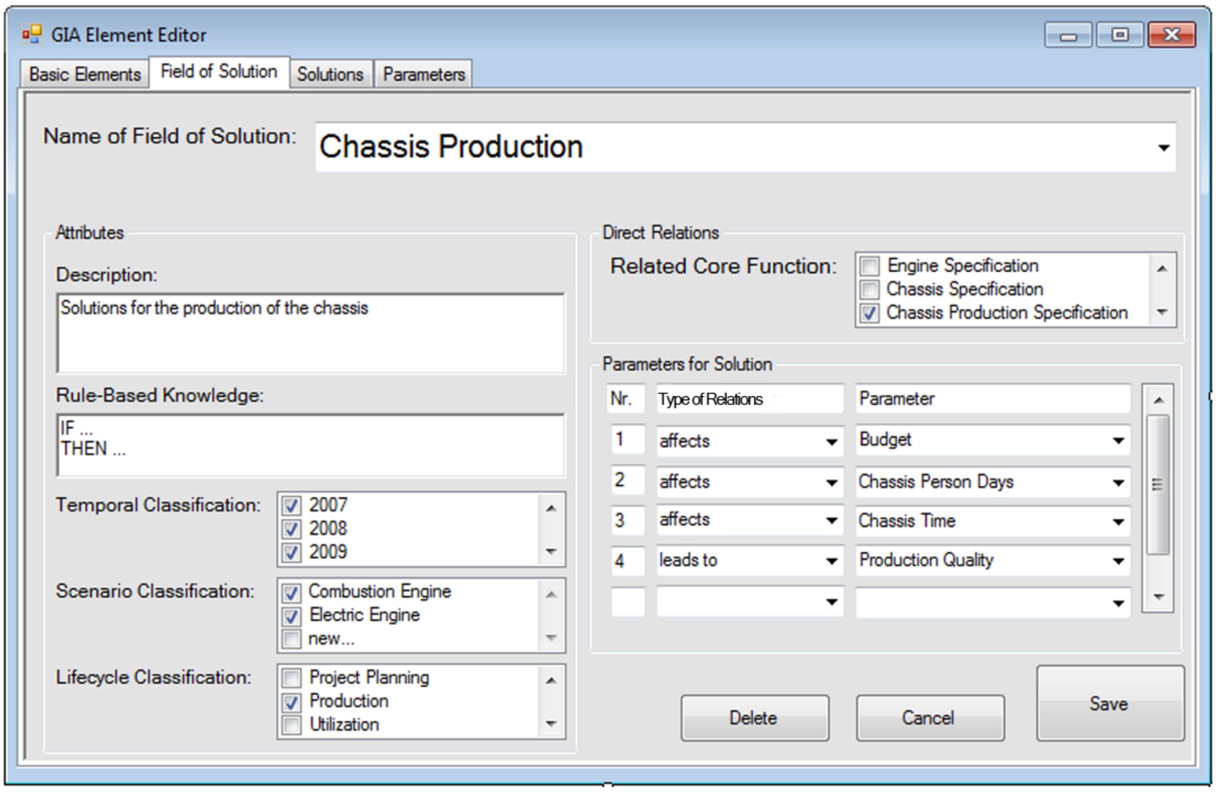
Related Goal: Best Engine, Best Chassis, Best Budget

Related Core Function: ✓ Engine Specification, Chassis Specification, Chassis Production Specification

Related FoS : Chassis Production, Engine Production, Motor

Figure 6-12 GUI for editing basic elements

In the following Figure 6-13, the GUI for the acquisition of fields of solutions can be seen. A field of solutions is a collection of solutions which, additionally to its basic information, needs to define the relations between the solutions and the parameters. Consequently they need an especially adapted form. Hence, in the top of the form, the name of the field of solutions can be specified or selected form an existing one to reedit it. On the left lower side, equally to the input form for the basic elements, the attributes of the field of solutions can be defined. By entering the attributes of the chassis production, it has to be known that all attributes are inherited by the solution elements but still can be changed when editing the solution. Furthermore, on the right side, the directly related elements, which are in the case of fields of solutions core functions, can be selected. Below this field, the parameters for the solutions have to be defined. This can be done by selecting the type of relation and the according parameter from a list. Finally the form offers the same buttons as the other forms on the lower right side, to delete the element, to cancel or save it. In the figure, the settings for the field of solution 'Chassis Productions' are shown. The attributes are describing that the field collects the solutions for the production of the chassis. In addition, no rule-based knowledge is defined, and a classification to all season, all scenarios, and the lifecycle phase 'Production' is done. On the right side, the related core function is the 'Chassis Production Specification'. Finally the parameters of the solutions are defined. Consequently all solutions of the field of solutions 'chassis production' need to affect the budget, the chassis person days, and the chassis time as well as leading to a specific production quality. Summarizing, in this GUI a clear and consequent acquisition of all important information about the elements field of solutions is offered.



Name of Field of Solution: Chassis Production

Attributes

Description: Solutions for the production of the chassis

Rule-Based Knowledge: IF ... THEN ...

Temporal Classification: ✓ 2007, ✓ 2008, ✓ 2009

Scenario Classification: ✓ Combustion Engine, ✓ Electric Engine, ☐ new...

Lifecycle Classification: ☐ Project Planning, ✓ Production, ☐ Utilization

Direct Relations

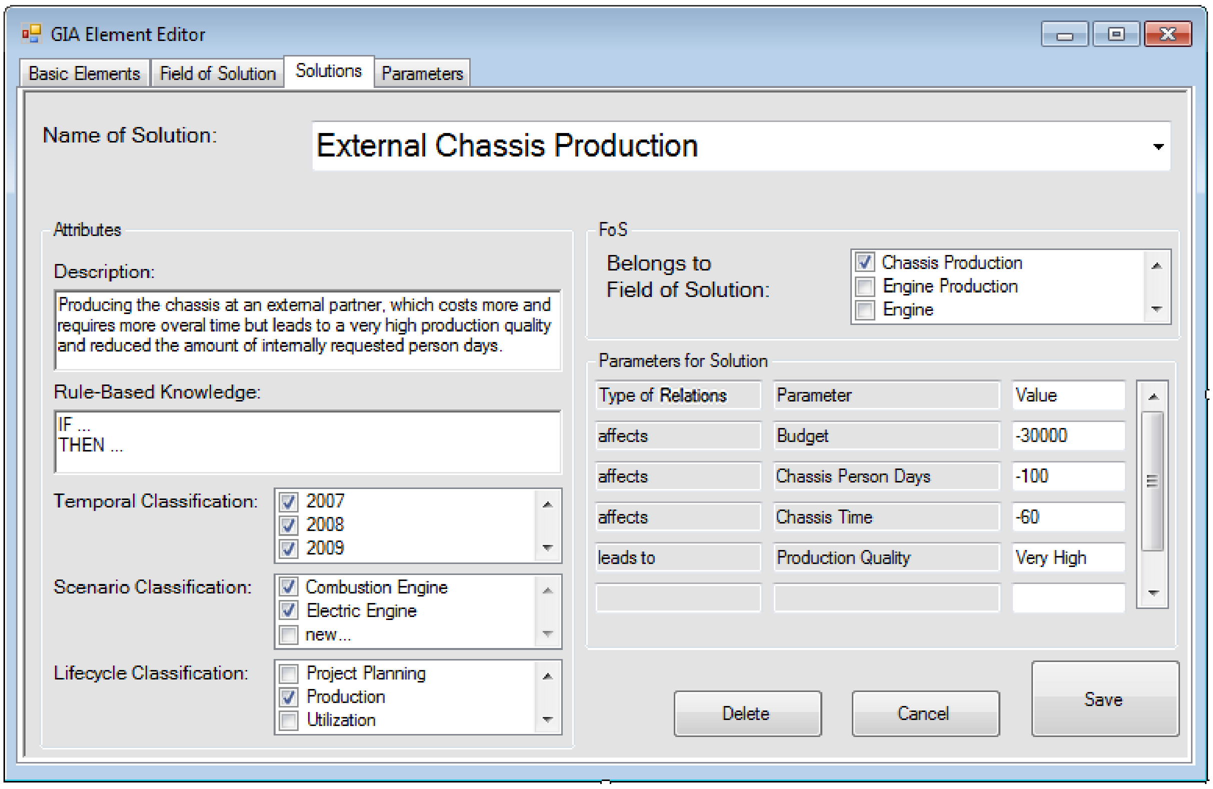
Related Core Function: ☐ Engine Specification, ☐ Chassis Specification, ✓ Chassis Production Specification

Parameters for Solution

Nr.	Type of Relations	Parameter
1	affects	Budget
2	affects	Chassis Person Days
3	affects	Chassis Time
4	leads to	Production Quality

Figure 6-13 GUI for editing field of solutions

Having created the field of solutions, the solution elements within can be defined. The supporting GUI can be seen in Figure 6-14. In the top of the form, a new solution is created by entering its name or selected from a list of existing ones. On the lower right side, equally to the other form the attributes of the solutions can be defined. Thus, a closer description, a rule-based knowledge and the classification of the temporal aspect, scenario, and lifecycle can be entered. On the right side one belonging field of solution for the solution can be selected, which is an elementary step since it leads to the related parameters as well as to a pre-defined set of attributed. Consequently, based on the selected field of solutions, the group box shows a set of parameters with a specified type of relation. In these fields a specific value for the selected solution can be entered. Lastly the form finishes with the three buttons, to delete, cancel the changes or save the element. The example in the figure shows the solution ’External Chassis Production’. It is more detailed described as the production of the chassis at an external partner, which costs more and requires more overall time, but leads to a very high production quality and reduces the amount of internally requested person days. The solution is assigned to the field of solution ‘Chassis Production’ and consequently has the same classifications for the temporal aspect, the scenarios, and the lifecycle phase. Finally the values for the relations to the parameters are set. Hence, the budget is affected by -30000, the chassis person days by -100, the chassis time by -60, and the solution leads to a very high production quality. As it can be seen, the presented form offers a good solution for acquiring all necessary information about the various solutions of the PPS.



Name of Solution: External Chassis Production

Description: Producing the chassis at an external partner, which costs more and requires more overal time but leads to a very high production quality and reduced the amount of internally requested person days.

Rule-Based Knowledge: IF ... THEN ...

Temporal Classification: [✓ 2007] [✓ 2008] [✓ 2009]

Scenario Classification: [✓ Combustion Engine] [✓ Electric Engine] new...

Lifecycle Classification: Project Planning [✓ Production] Utilization

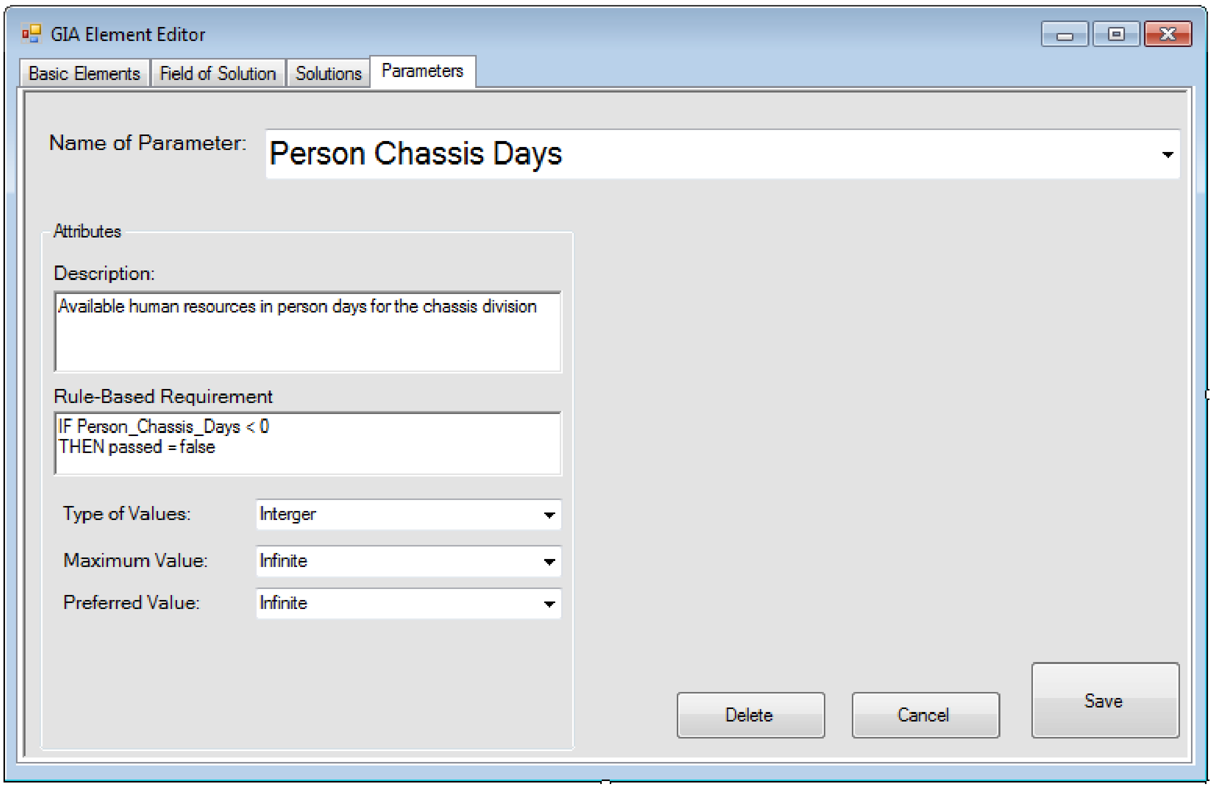
Belongs to Field of Solution: [✓ Chassis Production] Engine Production Engine

Parameters for Solution

Type of Relations	Parameter	Value
affects	Budget	-30000
affects	Chassis Person Days	-100
affects	Chassis Time	-60
leads to	Production Quality	Very High

Figure 6-14 GUI for editing solutions

Lastly, the form for the definition of parameters is presented in Figure 6-15. The form is less filled compared to the other forms, since parameters are globally defined and therefore not classified, as well as no relations are specified within the field. In the top of the form the name of the parameter has to be defined for creating a new one or selected among the existing ones for editing it. The attributes of the parameter are accordingly defined on the lower right side. These are a more detailed description of the parameter, a rule-based requirement to describe general knowledge of the parameter, as well as the type, the maximum, and the preferred value of it. The type of the element describes the classification of the values of the elements, which can be for example integer or string. The maximum value describes the highest possible value of the parameter and the preferred value describes whether a low or a high value is to be preferred In the presented figure the parameter 'Chassis Person Days' can be seen as an example. It is described as the available human resources in person days for the chassis division. The rule-based requirement checks, if the amount of person days for the chassis division is lower than zero. If it is so, the rule consequently indicates the error. Furthermore the values of the parameter are defined to be of the type integer and have the maximum and preferred value set to 'infinite'. Concluding it may be seen that the presented form offers all required possibilities for an acquisition of all relevant information for the definition of global parameters in a PPS.



Name of Parameter: Person Chassis Days

Description: Available human resources in person days for the chassis division

Rule-Based Requirement
```
IF Person_Chassis_Days < 0
THEN passed = false
```

Type of Values:	Interger
Maximum Value:	Infinite
Preferred Value:	Infinite

Figure 6-15 GUI for editing parameters

Recapitulating a GUI for the acquisition of data for the modeling of a PPS in order to analyze goal interrelations is presented. It offers all necessary input possibilities to enter all relevant data for the complete modeling of a PPS. Thus, the elements goals, demands, core functions, fields of solutions are defined by their names and a set of attributes allowing a more detailed description and classification. For the basic elements, goals, demands, and core functions, the direct relations between the relevant elements can be further selected. For the field of solutions and the according solution the relations to the parameters can be entered with the type of relation, the parameter and a solution specific value. Finally with the definition of the global parameters and their values, a complete model of the PPS can be acquired. This acquired information is now needed to be organized in the formal specification of the OWL–Implementation in order to allow further processing and visualization.

6.2.3 Visualization

The final section of this presentation of the conceptual implementation of a software tool, regards the visualization part of it. The main purpose of the visualization is to give a direct feedback to the user of the stored information in the system, as well as results of further processed information. Therefore a general visualization of the PPS is presented, followed by a representation for identified goal interrelations. The presented visualization is providing a complete overview for handling the acquired information about the PPS and with the visualization of goal interrelations a first an elementary step for a complete analysis of these.

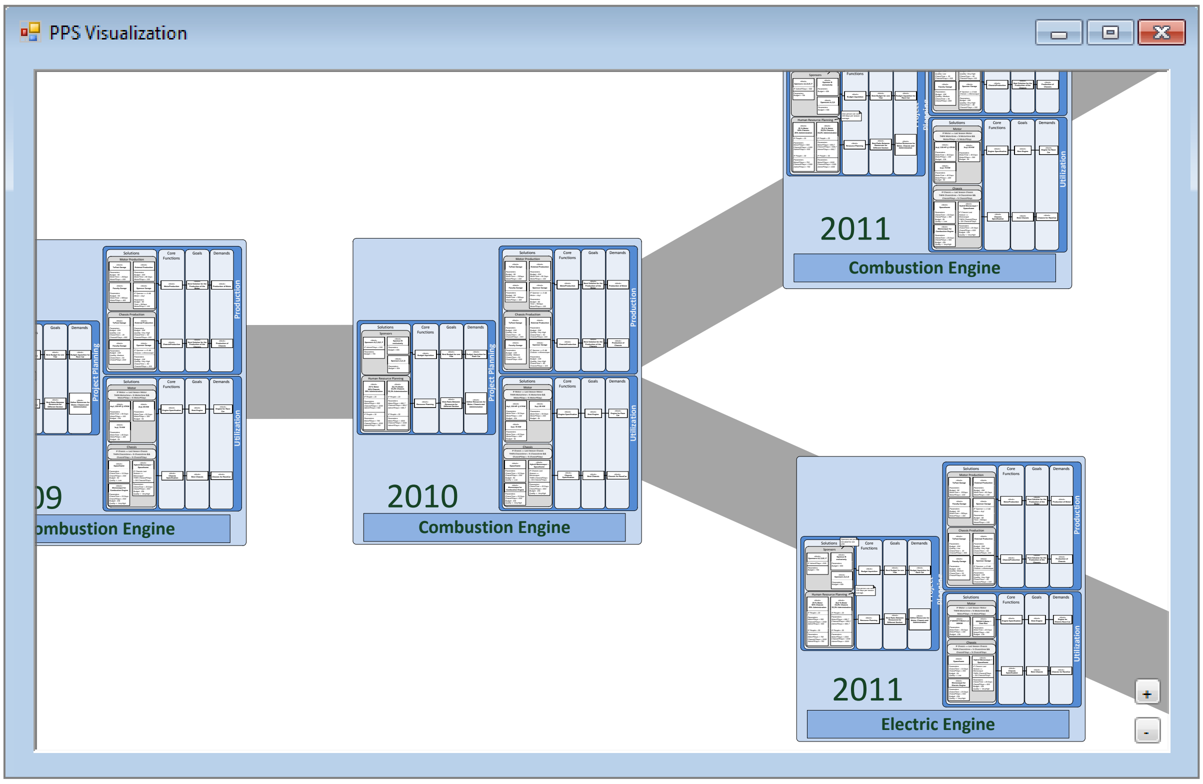


Figure 6-16 GUI for the visualization of a PPS

A fundamental task of the visualization is to provide a general overview of the acquired PPS. Therefore a graphical representation which shows an organized view of the various elements is presented in Figure 6-16. The visualization unit automatically arranges the elements in order to generate a graphical representation of the system. The representation is supporting the understanding of the system and is based on the graph representation used for the manual description of the system, presented in chapter 6.1.2.2. Consequently the elements are grouped according to the temporal and scenario classification. These are furthermore subdivided into the several lifecycle phases, where the belonging elements are presented. These elements are sorted and connected in order to represent the chain of relations from demand to goal to core function to field of solution. The field of solution finally includes also graphically its solutions with their various attributes. The whole system can be moved around with the mouse or zoomed in and out with the buttons on the lower right side, in order to receive a complete overview or a closely detailed look on the system. The example in the figure shows a system representation at a large scale where an overview on the various scenarios and product generation is given. Thus the first product generation from 2010 is shown on the right side, with its three lifecycle phases. It can be seen that that two scenarios are splitting after the first product generation, whereas the line from the scenario ‘combustion engine’ can be further seen with the product generation 2011 and 2012. In order to get a closer look at the various product generations, the user has to zoom in and is able to see all details of the various product generations with all their elements and interrelations. Summarizing with the presented organized graphical visualization of the PPS the user receives an easily understandable and complete overview on the acquired information.

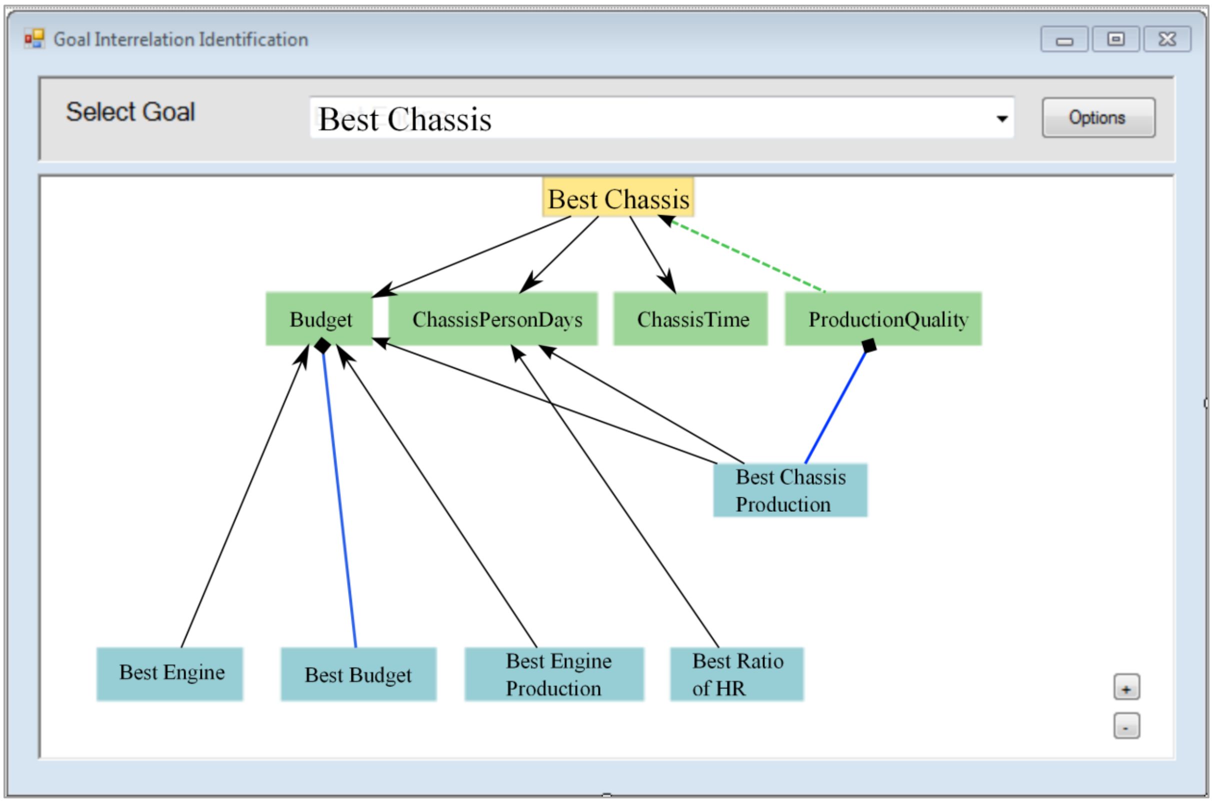


Figure 6-17 GUI for the identification of goal interrelations

Another provided visualization is especially dedicated towards the representation of goal interrelations. Therefore a GUI designed for identification of goal interrelations is shown in Figure 6-17. In general, the GUI offers a simple way to receive further information through which parameters a goal is related to other goals. Therefore the requested goal has to be selected from the list of goals in the top part of the window. The 'options' button on the right side of the form opens an menu, where further filters for the visualization can be selected. These filters can, for example, limit the results to a specific scenario or a certain product generation or change the categories of shown elements. Based on this information the graphical representation of the interrelations is shown is the lower field of the form. It begins from the selected goal in the top, showing the relations to the parameters and other goals which are related to the same parameter. In order to provide a better overview the goals are colored blue whereas the parameters are colored green. Furthermore the type of relation towards the parameters is shown graphically. Thus the relation 'affects' is represented by a black arrow towards the parameter, the relation 'requires' is an arrow with a green dotted line towards the goal, and the relation 'requires' is a blue line with a diamond ending at the parameter. Finally the goals are arranged according to the number of interrelations, supporting the identification of strongly interrelated goals. The example in the figure regards the goal 'Best Chassis'. For a good graphical representation it is selected to filter the results to just one product generation, one scenario, and to limit the shown elements to goals and parameters. Consequently its solutions are affecting the parameters 'Budget', 'ChassisPersonDays', and 'ChassisTime' as well as requiring the parameter 'ProductionQuality'. It is further shown that the strongest interrelated goal is 'Best Chassis Production' since it is arranged higher than the other related goals. The 'BestChassis' Production affects the 'Budget' and the 'ChassisPersonDays' as well as leads to a 'ProductionQuality'. Therefore it is interrelated through three parameters to the selected goal. The other goals are arranged at the lower side of the figure and are each interrelated through one parameter to the selected goal, with the according type of interrelations. Concluding, it can be seen that this visualization supports the identification of goal interrelations. With the application of various filters the interrelations can be presented accordingly to the user requests.

Recapitulation the entire chapter, a conceptual implementation of a computer supported goal interrelation software is presented. Therefore an example, based on the Formula Student team *TUfast* is created and presented in a graph based PPS which is enhanced according to the framework in chapter 5.1. Furthermore a concept for a software implementation is presented which applies the example. The software is based on the semantic web implementation presented in chapter 4.2.4 and the framework presented in chapter 5. It provides an especially adjusted GUI for data acquisition of the PPS, as well as a visualization unit. The visualization unit supports the user in receiving a complete overview on the system and in the identification of goal interrelation. Consequently the concept provides a solution for an implementation of the requirements defined in chapter 3 as well as the framework in chapter 5 and can serve as a basis for further solutions.

7 Conclusions and Outlook

In the following, the achieved results within this thesis are summarized and critically reflected. Furthermore an outlook is provided, identifying future research potentials and improvements in this field of research.

7.1 Conclusions

In general, this thesis provides an approach for an implementation of a computer-supported goal interrelation analysis. This approach is elaborated by regarding the state of the art of connected research fields, providing a broad background for the aim of this thesis. Based on this knowledge potentials, an iterative process is performed leading to a catalogue of requirements, an analysis of established modeling techniques, and a framework for a computer-supported analysis of goal interrelations. Finally a conceptual implementation is provided, considering the results from the requirements, the analysis and the framework.

Thus, a fundamental basis for a computer-supported goal interrelation analysis is provided by a catalogue of requirements and a framework. They are developed in an iterative process, supported by an analysis of multiple computer-supported modeling techniques. The catalogue of requirements lists requirements for a general approach to analyze goal interrelation, for the computational implementation of it, as well as requirements directly addressing the implementation of a software tool. Within the framework, the established approach is enhanced in order to consider temporal aspects, scenarios during the development, and rule-based knowledge content. Additionally the structure of the content of a goal analysis is defined as well as the structure of the related software. These were improved and detailed during multiple iteration loops and simultaneously tested with the four computer-supported modeling techniques multiple-domain matrix, graph grammars, relational database and semantic webs. Moreover through analyzing these techniques it could be evaluated, that relational databases as well as semantic webs, can be considered as the preferred techniques for a later implementation. In general, it can be said that relational databases stand out through the maturity of the technique, while semantic networks are offering more potentials for future improvements. Consequently a complete preliminary concept for an implementation of a computer-supported goal interrelation analysis has been provided through the requirements and the framework.

The achieved preliminary concept is further applied for a conceptual implementation of a computer supported goal interrelation analysis. Therefore an example of a lifecycle oriented product planning system, based on the Formula Student Team *TUfast* is presented. Firstly the example introduces the application of the enhanced graph-based representation of a product planning system. This representation is based on the framework for enhancing the established approach for goal interrelation analysis. Besides a conceptual computer program is shown. It considers the according preliminary concept and describes potential solutions for it. Thus, the major missing aspects of current solutions, which were identified in the analysis, are treated. These missing aspects can mainly be found in the data acquisition and the visualization of the system, and therefore especially dedicated solutions are presented.

Summarizing this thesis provides a framework as well as a conceptual implementation for computer-supported goal interrelations based on modeling techniques. Applying it is a first steps in order to make the product planning process more transparent, leading to a reduced amount of changes and connected to it to less costs. Thus, it can make the innovative process more efficient, which leads to an improved competitive capability of a company.

7.2 Outlook

In the context of this thesis, a framework as well as general requirements have been developed, which can be used for an implementation of a computer-supported goal interrelations analysis within product planning systems. Furthermore a concept for a software tool, based on these requirements, this framework, and an analysis of established modeling techniques is presented. Consequently, this thesis can serve as an initial point for several further research projects. These can be especially found in the implementation of software, the application of the approach at further product planning processes, improvement of the underlying definitions, as well as the specification of additional functions.

Based on the result of this thesis, a next step could be a functional implementation of a software tool for a computer supported goal interrelation analysis. As it was goal of this thesis to analyze an application of software methods to support the approach of Förg (2011) and Hepperle et al. (2011a), all the necessary information for its implementation is available. Therefore the tool should consider the defined requirements and be built according to the framework. Furthermore, with the conceptual implementation the first rough concept of a software tool is given, which can be enhanced to a technical concept, considering all programming issues. Finally the implementation with a higher programming language would lead to applicable software tool for computer-supported goal interrelation analysis.

Having a functional implementation of a software tool, its application has to be tested within a real product planning process. Since, especially the graphical user interface, the visualization system, or the rule-based description of knowledge have never been applied in this context, it has to be further investigated. This can be done within a research project at an enterprise, where the tool is applied at the strategic product planning process. Thus, the usability of the software for large and complex products can be checked and further improvement of it derived.

Furthermore the provided requirements and framework are giving a complete definition for a basic implementation of a computer-supported goal interrelation analysis, but need to be more detailed specified. Being based on general research as well as interviews with experts from academic research, there are still some stakeholders which need to be additionally considered. Especially future applicants or the integratability in existing processes have to be investigated. Therefore, for example requested file formats for in- and export, security issues or further attributes for the elements can be considered. This can lead to a more applicable definition of the yet existing one.

Finally further solutions for requested functions of the software tool have to be defined. Firstly, based on the identified goal interrelations a conflict detection has to be specified. Therefore general rules describing all possible situation between the parameters and the assigned solution have to be defined and a resulting indication whether a conflict occurs or not. Based on these results an further optimization system has to be specified which enables computing an optimal solution for the product planning system, considering all possible elements and values in it. This allows an automatically specification of a product and therefore helps to minimize mistakes and time usage of the innovation management.

In general, the implementation of the presented concept of a computer-supported goal interrelation analysis seems a useful task in order to improve the strategic product planning process. Identifying interrelations of goals is a first step towards managing the complexity of innovating products. Furthermore the extensibility to an solution for computational optimized product configurations can lead to an integrated computer aided strategic product planning. Having the benefits from a computer-supported product planning, processes can get faster and more accurate and consequently lead to an significant competitive advantages in the global market.

8 List of Abbreviations

API	Application Programming Interface
BDD	Block Definition Diagram
CAD	Computer Aided Design
CAE	Computer Aided Engineering
CAM	Computer Aided Manufacturing
CRM	Customer Relationship Management
DBMS	Database Management System
DMM	Domain Mapping Matrix
DSM	Design Structure Matrix
ERP	Enterprise Resource Planning
FoS	Field of Solution
GIA	Goal Interrelation Analysis
HoQ	House of Quality
IBD	Internal Block Diagram
MBSE	Model Based Systems Engineering
MDM	Multiple Domain Matrix
OWL	Web Ontology Language
PDM	Product Data Management
PLM	Product Lifecycle Management
PPS	Product Planning System
QFD	Quality Function Deployment
SCM	Supply Chain Management
SPARQL	Simple Protocol and RDF Query Language
SPP	Strategic Product Planning
SQL	Structured Query Language
SysML	Systems Modeling Language
UML	Unified Modeling Language
URI	Uniform Resource Identifier
VR	Virtual Reality
WWW	World Wide Web
XML	Extensible Markup Language

9 List of Figures

10 Bibliography

AFUAH 2003
Afuah, A.: Innovation management. Strategies, implementation and profits. 2nd
New York: Oxford University Press 2003. ISBN: 0195142306.

AKAO 1990
Akao, Y.: Quality function deployment. Integrating customer requirements into product design
Cambridge, Mass: Productivity Press 1990. ISBN: 1563273136.

ARNOLD ET AL. 2005
Arnold, V.; Dettmering, H.; Engel, T.; Karcher, A.: Product lifecycle management beherrschen. Ein anwenderhandbuch für den mittelstand
Berlin: Springer 2005. ISBN: 3540229973.

AUBER 2003
Auber, D.: Tulip-a huge graph visualization framework. In *Graph Drawing Software*.

BLOMER ET AL. 2011
Blomer, J.; Geiß, R.; Jakumeit, E.: The GrGen.NET User Manual
Karlsruhe 2011. Available online at http://www.info.uni-karlsruhe.de/software/grgen/GrGenNET-Manual.pdf, checked on 13/10/2011.

BRAUN 2005
Braun, T. E.: Methodische Unterstützung der strategischen Produktplanung in einem mittelständisch geprägten Umfeld. 1st ed.
München: Dr. Hut 2005. ISBN: 3-89963-274-5.

BROWNING 2001
Browning, T.: Applying the design structure matrix to system decomposition and integration problems: a review and new directions. In *Engineering Management, IEEE Transactions on* 48 (3), pp. 292-306.

BROY 1998
Broy, M.: Systemstrukturen und Theoretische Informatik. 2., überarb
Berlin: Springer 1998. ISBN: 9783540643920.

BUEDE 2009
Buede, D. M.: The engineering design of systems. Models and methods. 2nd ed.
Hoboken, N.J: John Wiley & Son 2009. ISBN: 0470164026.

BULLINGER ET AL. 2003
Bullinger, H.-J.; Hab, G.; Kiss-Preußinger, E.: Automobilentwicklung in Deutschland - wie sicher in die Zukunft? Chancen, Potenziale und Handlungsempfehlungen für 30 Prozent mehr Effizienz ; [Studie]
Stuttgart: Fraunhofer IRB-Verlag 2003. ISBN: 9783816763888.

BUNGARTZ 2009

Bungartz, H.-J.: Modellbildung und Simulation. Eine anwendungsorientierte Einführung
Berlin: Springer 2009. ISBN: 9783540798095.

CLARK & FUJIMOTO 2005

Clark, K. B.; Fujimoto, T.: Product development performance. Strategy, organization, and management in the world auto industry
Boston, Mass: Harvard Business School Press 2005. ISBN: 0875842453.

CLARKSON ET AL. 2004

Clarkson, P. et al.: Predicting change propagation in complex design. In *Journal of Mechanical Design* 126, p. 788.

COOPER & EDGETT 2005

Cooper, R. G.; Edgett, S. J.: Lean, rapid, and profitable. New product development. 1. publ.
Ancaster, Ontario: Stage-Gate Internat. 2005. ISBN: 1439224609.

DÖRNER 2010

Dörner, D.: Die Logik des Misslingens. Strategisches Denken in komplexen Situationen. 9th ed.
Reinbek bei Hamburg: Rowohlt 2010. ISBN: 9783499615788.

DOW & TAYLOR 2009

Dow, W.; Taylor, B.: Project Management Communications Bible
Hoboken: John Wiley & Son 2009. ISBN: 9780470399255.

EBEN ET AL. 2008

Eben, K. et al.: Modeling structural Change over Time10th. In: Mathias Kreimeyer, Udo Lindemann, M. Dabilovic (Eds.): Proceedings of the 10th International DSM Conference. 10th Internation Design Structure Matrix Conference. Stockholm, Sweden, 11-12 November.

EHRLENSPIEL ET AL. 2003

Ehrlenspiel, K.; Kiewert, A.; Lindemann, U.: Kostengünstig entwickeln und konstruieren. Kostenmanagement bei der integrierten Produktentwicklung. 4., bearb
Berlin [u.a.]: Springer 2003. ISBN: 3-540-44214-6.

EHRLENSPIEL ET AL. 2007

Ehrlenspiel, K. et al.: Cost-efficient design. Edited by Mahendra S. Hundal
Berlin: Springer 2007.

FELDHUSEN & GEBHARDT 2008

Feldhusen, J.; Gebhardt, B.: Product Lifecycle Management für die Praxis. Ein Leitfaden zur modularen Einführung, Umsetzung und Anwendung
Berlin, Heidelberg: Springer 2008. ISBN: 3540340084.

FÖRG 2010

Förg, A.: Elektromobilität. lebenszyklusgerechte Plannung zukünftiger Kraftfahrzeuge. Semesterarbeit. With assistance of Clemens Hepperle
Insitute of Product Development, Technische Universität München 2010.

FRIEDENTHAL ET AL. 2008
Friedenthal, S.; Steiner, R.; Moore, A.: A practical guide to SysML. The systems modeling language
San Francisco (Calif.), Oxford: Morgan Kaufmann; Elsevier Science 2008. ISBN: 9780123743794.

FRIEDRICH 2010
Friedrich, S.: Einsatzmöglichkeiten einer Design-Structure-Matrix im Rahmen des Strategischen Projektmanagements
München: GRIN 2010. ISBN: 9783640618637.

GAUSEMEIER 2001
Gausemeier, J.: Produktinnovation. Strategische Planung und Entwicklung der Produkte von morgen
München: Hanser 2001. ISBN: 9783446216310.

GAUSEMEIER & REDENIUS 2005
Gausemeier, J. & Redenius, A.: Entwicklung mechatronischer Systeme. In Bernd Schäppi: Handbuch Produktentwicklung. München, Wien: Hanser 2005. ISBN: 3-446-22838-1.

GAUSEMEIER ET AL. 2004
Gausemeier, J.; Lindemann, U.; Schuh, G.: Planung der Produkte und Fertigungssysteme für die Märkte von morgen. Ein praktischer Leitfaden für mittelständische Unternehmen des Maschinen- und Anlagenbaus
Frankfurt am Main, Hannover: VDMA Verl; Technische Informationsbibliothek u. Universitätsbibliothek 2004. Available online at http://edok01.tib.uni-hannover.de/edoks/e01fb05/50495301X.pdf, checked on 26/09/2011.

GAUSEMEIER ET AL. 2005
Gausemeier, J. et al.: Prototyping und Produktmodellierung. In Bernd Schäppi: Handbuch Produktentwicklung. München, Wien: Hanser 2005. ISBN: 3-446-22838-1.

GAUSEMEIER ET AL. 2009
Gausemeier, J.; Plass, C.; Wenzelmann, C.: Zukunftsorientierte Unternehmensgestaltung. Strategien Geschäftsprozesse und IT-Systeme für die Produktion von morgen
München: Hanser 2009. ISBN: 978-3-446-41055-8.

GEORGANTZAS & ACAR 1995
Georgantzas, N. C.; Acar, W.: Scenario-driven planning. Learning to manage strategic uncertainty
Westport, Conn: Quorum Books 1995. ISBN: 0899308252.

GRÖNING 2010
Gröning, V.: Markenpolitik- Verfahrensvergleich. Means-end-Analyse, Netzwerkanalyse und Repertory Grid; Methoden zur Messung des Markenwertes
München: GRIN 2010. ISBN: 9783640753284.

GROSS & YELLEN 2006
Gross, J. L.; Yellen, J.: Graph theory and its applications
Boca Raton: Chapman & Hall/CRC 2006 (Discrete mathematics and its applications). ISBN: 978-1-58488-505-4.

HABERER 2007
Haberer, Karl (Ed.): The semantic web. Proceedings. ISWC <6, 2007, Pusan>
Berlin, Heidelberg, New York, NY: Springer 2007. ISBN: 3540339930.

HAUSCHILDT & SALOMO 2011
Hauschildt, J.; Salomo, S.: Innovationsmanagement. 5., überarb., erg. und aktualisierte Aufl. München: Vahlen 2011 (Vahlens Handbücher der Wirtschafts- und Sozialwissenschaften). ISBN: 9783800636556.

HELMS & SHEA 2009
Helms, B. & Shea, K.: Object-Oriented Concepts for Computational design synthesis. In: Mathias Kreimeyer, J. Maier, G. Fadel, Udo Lindemann (Eds.): Proceedings of the 11th International DSM Conference, DSM'09. Greenville, South Carolina, USA.

HELMS & SHEA 2010
Helms, B. & Shea, K.: Object-Oriented Conceps for Computational Design Synthesis. In: D. Maranjanovic´, M. Štorga, N. Pavković, N. Bojčetić (Eds.): 11th International Design Conference DESIGN 2010. 11th International Design Conference DESIGN 2010. Dubrovnik, Croatia.

HELMS ET AL. 2009
Helms, B. et al.: Graph Grammars-A Formal Method for Dynamic Structure Transformation. In: Mathias Kreimeyer, J. Maier, G. Fadel, Udo Lindemann (Eds.): Proceedings of the 11th International DSM Conference, DSM'09. Greenville, South Carolina, USA.

HEPPERLE ET AL. 2009
Hepperle, C. et al.: An Integrated Product Lifecycle Model and Interrelations In-Between the Lifecycle Phases. In: Chris McMahon, Deba Dutta, George Huang (Eds.): Product lifecycle management. Proceedings of the PLM09 conference held at the University of Bath, UK, 6-8 July 2009. Geneve: Inderscience Enterprises. ISBN: 0-907776-49-3.

HEPPERLE ET AL. 2011A
Hepperle, C. et al.: Consideration of Goal Interrelations in Lifecycle Oriented Product Planning. In: ICED 2011. Internation Conference on Engineering Design. Copenhagen.

HEPPERLE ET AL. 2011B
Hepperle, C. et al.: Temporal Aspects in Lifecycle-Oriented Planning of Product-Service-Syste,. In: Amaresh Chakrabarti (Ed.): Research into design. Supporting sustainable product development. 17th International Conference on Design into Research - ICoRD 2011. Bangalore. Singapore: Research publishing, pp. 338–346. ISBN: 978-981-08-7721-7.

HERRMANN 2010
Herrmann, C.: Ganzheitliches Life Cycle Management. Nachhaltigkeit und Lebenszyklusorientierung in Unternehmen
Berlin, Heidelberg: Springer 2010. ISBN: 978-3-642-01420-8.

HERRMANN & HUBER 2009
Herrmann, A.; Huber, F.: Produktmanagement. Grundlagen - Methoden -Beispiele
Wiesbaden: Betriebswirtschaftlicher Verlag Gabler 2009. ISBN: 9783409125505.

HOECK 2005
Hoeck, H.: Produktlebenszyklusorientierte Planung und Kontrolle industrieller Dienstleistungen im Maschinenbau
Aachen: Shaker 2005. ISBN: 3-8322-4097-7.

INSTITUT AIFB 2011
Institut AIFB: semanticweb.org
Karlsruhe 2011. Available online at http://semanticweb.org/wiki/Main_Page, checked on 16/09/2011.

JAINTA 2009
Jainta, I.: Quality Function Deployment (QFD). Der "Königsweg" zur Erzeugung von Qualität?
München: GRIN 2009. ISBN: 9783640326648.

JAKUMEIT ET AL. 2010
Jakumeit, E. et al.: GrGen.NET. The Expressive, Convenient and Fast Graph Rewrite System. In *International Journal on Software Tools for Technology Transfer (STTT)* 12 (3), pp. 263-271. Available online at http://pp.info.uni-karlsruhe.de/uploads/publikationen/jakumeit10sttt.pdf, checked on 26/09/2011.

KASHYAP ET AL. 2008
Kashyap, V.; Bussler, C.; Moran, M.: The Semantic Web. Semantics for data and services on the Web
Berlin: Springer 2008. ISBN: 9783540764519.

KASTENS & KLEINE BÜNING 2005
Kastens, U.; Kleine Büning, H.: Modellierung. Grundlagen und formale Methoden
München u.a.: Hanser 2005. ISBN: 3446404600.

KLEINSCHMIDT 2005
Kleinschmidt, P.: Relationale Datenbanksysteme. Eine praktische Einführung. 3rd ed.
Berlin: Springer 2005. ISBN: 3540224963.

KNUBLAUCH ET AL. 2004
Knublauch, H. et al.: The Protégé OWL plugin: An open development environment for semantic web applications. In *The Semantic Web-ISWC 2004*, pp. 229-243.

KOCH & MEERKAMM 2001

Koch, M. & Meerkamm, H.: Ein Konstruktionsassistenzsystem für komplexe Produkte auf Basis eines hybriden Produktmodells. In: Informationsverarbeitung in der Produktentwicklung 2001. Effiziente 3D-Produktmodellierung - Fortschritte und Fallstricke. CAx-Tagung; Informationsverarbeitung in der Produktentwicklung 2001. Düsseldorf: VDI-Verlag. ISBN: 3180916141.

KÖNIG ET AL. 2008

König, C. et al.: Multiple-Domain Matrix as a Framework for Systematic Process Analysis. In: Mathias Kreimeyer, Udo Lindemann, M. Dabilovic (Eds.): Proceedings of the 10th International DSM Conference. 10th Internation Design Structure Matrix Conference. Stockholm, Sweden, 11-12 November, pp. 231–233.

KŘEMEN & KOSTOV 2011

Křemen, P.; Kostov, B.: OWL2Query 2011. Available online at http://krizik.felk.cvut.cz/km/owl2query/index.html, checked on 22/10/2011.

KUSS 2000

Kuss, A.: Käuferverhalten. Eine marketingorientierte Einführung.
Stuttgart: Lucius & Lucius 2000. ISBN: 9783825216047.

LAMBERTZ ET AL. 1996

Lambertz, M.; Geckeler, H.; Weyh, H.: Total Innovation Management. In 7 Schritten zum Erfolg
Düsseldorf: Econ 1996. ISBN: 343019606X.

LANGER & LINDEMANN 2009A

Langer, S. & Lindemann, U.: Managing Cycles in Development processes. Analysis and classification of external context factors. In: M. Norell Bergendahl, M. Grimheden, L. Leifer, P. Skogstad, Udo Lindemann (Eds.): Proceedings of ICED'09. Glasgow: Design Society. ISBN: 9781904670056.

LANGER & LINDEMANN 2009B

Langer, S. & Lindemann, U.: Managing cycles in development processes - Analysis and classification of external factors. In: Margareta Bergendahl, Martin Grimheden, Larry Leifer (Eds.): Proceedings of the 17th International Conference on Engineering Design (ICED'09): Design Society.

LIMBERG 2008

Limberg, T.: Examining Innovation Management from a Fair Process Perspective
Wiesbaden: Betriebswirtschaftlicher Verlag Dr. Th. Gabler / GWV Fachverlage, Wiesbaden 2008. ISBN: 9783834910707.

LINDEMANN 2005

Lindemann, U.: Methodische Entwicklung technischer Produkte. Methoden flexibel und situationsgerecht anwenden
Berlin u.a: Springer 2005. ISBN: 3-540-14041-7.

LINDEMANN 2011

Lindemann, U.: DSMweb.org. DSM Tools 2011. Available online at http://129.187.108.94/dsmweb/en/dsm-tools.html, checked on 14/09/2011.

LINDEMANN & KIEWERT 2005

Lindemann, U. & Kiewert, A.: Kostenmanagement im Entwicklungsprozess. marktgerechte Kosten durch Target Cositng. In Bernd Schäppi: Handbuch Produktentwicklung. München, Wien: Hanser 2005. ISBN: 3-446-22838-1

LINDEMANN & MAURER 2007

Lindemann, U. & Maurer, M.: Facing multi-domain complexity in product development. In *The Future of Product Development*, pp. 351-361, Berlin: Springer 2007.

LINDEMANN ET AL. 2009

Lindemann, U.; Maurer, M.; Braun, T.: Structural complexity management. An approach for the field of product design
Berlin: Springer 2009. ISBN: 978-3-540-87888-9.

MADU 2006

Madu, C. N.: House of Quality (Qfd) in a Minute. Quality Function Deployment
Fairfield: Chi 2006. ISBN: 9780967602363.

MAURER 2011

Maurer, M.: Loomeo 2.3. Anwendungen am Beispiel
Hannover 2011. Available online at http://donar.messe.de/exhibitor/hannovermesse/2011/T99497/anwendungsbeispiele-ger-114921.pdf, checked on 6/10/2011.

MICROSOFT CORPORATION 2011

Microsoft Corporation: Access 2011. Database Software and Application
Available online at http://office.microsoft.com/en-us/access/, checked on 6/10/2011..

MIßLER-BEHR 1993

Mißler-Behr, M.: Methoden der Szenarioanalyse
Wiesbaden: Deutscher Universitäts-Verlag 1993. ISBN: 3824401738.

MOTIK ET AL. 2009

Motik, B. et al.: OWL 2 web ontology language: Structural specification and functional-style syntax. In *W3C Recommendation* 27.

MÜNZER 2011

Münzer, C.: Integration of Model- and Rule-Based Knowledge Representation. Diplomarbeit. With assistance of Clemens Hepperle
Insitute of Product Development, Technische Universität München 2010.

PARRY 2005

Parry, M. E.: Strategic marketing management. A means-end approach
London: McGraw-Hill 2005. ISBN: 9780071450935.

PAVLIDIS 1972
Pavlidis, t.: Linear and Context-Free Gaph Gramars,. Journal of the Association for Computing Machinery.
New York: acm 1972 (11-22, 19).

PINE 2008
Pine, B. J.: Mass customization. The new frontier in business competition.
Boston, Mass: Harvard Business School Press 2008. ISBN: 9780875843728.

PRUD'HOMMEAUX & SEABORNE 2008
Prud'hommeaux, E.; Seaborne, A.: SPARQL Query Language for RDF 2008. Available online at http://www.w3.org/TR/rdf-sparql-query/, checked on 21/10/2011.

PUHL 1999
Puhl, H.: Komplexitätsmanagement. Ein Konzept zur ganzheitlichen Erfassung, Planung und Regelung der Komplexität in Unternehmensprozessen
Lehrstuhl für Fertigungstechnik und Betriebsorganisation, Universität Kaiserslautern 1999.

REA & KERZNER 1997
Rea, P. J.; Kerzner, H.: Strategic planning. A practical guide
New York: John Wiley & Son 1997. ISBN: 9780471291978.

SCHILLER 2009
Schiller, F.: Modellbildung und Simulation
Institute of Automation and Information Systems, Technische Universität München 2010.

SCHWANINGER 2005
Schwaninger, M.: Systemorientiertes Design. ganzheitliche Perspektive in Innovationsprozessen. In Bernd Schäppi: Handbuch Produktentwicklung. München, Wien: Hanser 2005. ISBN: 3-446-22838-1

SEIDEL 2005
Seidel, M.: Methodische Produktplanung. Grundlagen, Systematik und Anwendung im Produktentstehungsprozess
Karlsruhe: Universitätsverlag 2005. ISBN: 3-937300-51-1.

SENDLER 2009
Sendler, U.: Das PLM-Kompendium. Referenzbuch des Produktlebenszyklus Managements. 1st ed.
Berlin: Springer 2009. ISBN: 3540878971.

SHEA 2010
Shea, K.: Computer Aided Product Development. Script of the Lecture: Computer Aided Product Development
Institute of Product Development, Technische Universität München 2010.

STANFORD CENTER FOR BIOMEDICAL INFORMATICS RESEARCH 2011
Stanford Center for Biomedical Informatics Research: The Protégé Ontology Editor and Knowledge Acquistion System 2011. Available online at http://protege.stanford.edu, checked on 20/10/2011.

SUMATHI & ESAKKIRAJAN 2007
Sumathi, S.; Esakkirajan, S.: Fundamentals of relational database management systems
Berlin, London: Springer 2007. ISBN: 9783540483977.

TIETZE 2003
Tietze, O.: Strategische Positionierung in der Automobilbranche. Der Einsatz von virtueller Produktentwicklung und Wertschöpfungsnetzwerken. 1st ed.
Wiesbaden: Deutscher Universitäts-Verlag 2003. ISBN: 3-8244-7972-9.

TUFAST E.V. 2011A
TUfast e.V.: TUfast e-technology. Formula Student Electroc Racing Team TU München 2011. Available online at http://tufast-etechnology.de, checked on 26/09/2011.

TUFAST E.V. 2011B
TUfast e.V.: TUfast Racing Team. Formula Student Racing Team TU München 2011. Available online at http://tufast-racingteam.de, checked on 26/09/2011.

UGHANWA & BAKER 1989
Ughanwa, D. O.; Baker, M. J.: The role of design in international competitiveness
London: Routledge 1989. ISBN: 9780415000130.

VAJNA 2001
Vajna, S.: Die neue Rchtlinie VDI 2209. Praxiserprobte Hinweise zur §D-Produktmodellierung. In: Informationsverarbeitung in der Produktentwicklung 2001. Effiziente 3D-Produktmodellierung - Fortschritte und Fallstricke. CAx-Tagung; Informationsverarbeitung in der Produktentwicklung 2001. Düsseldorf: VDI-Verlag. ISBN: 3180916141.

VAJNA 2005
Vajna, S.: Informationsmanagement. Management der produkt- umd prozessbezogenen Informationen in der integrierten Produktenwicklung. In Bernd Schäppi: Handbuch Produktentwicklung. München, Wien: Hanser 2005. ISBN: 3-446-22838-1.

VDI-SOCIETY FOR AUTOMOTIVE AND TRAFFIC SYSTEMS TECHNOLOGY 2011A
VDI-Society for Automotive and Traffic Systems Technology: Formula Student Electric. International Design Competition. With assistance of Ludwig Vollrath, Kötke Rainer 2011. Available online at http://www.formulastudentelectric.de, checked on 26/09/2011.

VDI-SOCIETY FOR AUTOMOTIVE AND TRAFFIC SYSTEMS TECHNOLOGY 2011B
VDI-Society for Automotive and Traffic Systems Technology: Formula Student Germany. International Design Competition. With assistance of Ludwig Vollrath, Kötke Rainer 2011. Available online at http://www.formulastudent.de, checked on 26/09/2011.

WALTER 2010
Walter, U.: Systems Engineering
Institute of Astronautics, Technische Universität München 2010.

WARE 2004
Ware, C.: Information visualization. Perception for design
Amsterdam u.a.: Elsevier, Morgan Kaufmann 2004. ISBN: 1-55860-819-2.

WEBBER & WALLACE 2008
Webber, L.; Wallace, M.: Qualitatskontrolle Fur Dummies
Weinheim: Wiley-VCH 2008. ISBN: 9783527704293.

WEIGEND 2010
Weigend, M.: Objektorientierte Programmierung mit Python 3. Einstieg, Praxis, professionelle Anwendung
Heidelberg: mitp 2010. ISBN: 3826617509.

WEILKIENS 2006
Weilkiens, T.: Systems engineering mit SysML/UML. Modellierung, Analyse, Design. 1st ed.
Heidelberg: Dpunkt 2006. ISBN: 389864409x.

WEINBERG ET AL. 2010
Weinberg, P. N.; Groff, J. R.; Oppel, A. J.: SQL, the complete reference. 3rd
New York: McGraw-Hill 2010. ISBN: 9780071592550.

WILDEMANN ET AL. 2007
Wildemann, H.; Ann, C.; Günthner, W.; Lindemann, U.: Plagiatschutz. Handlungsspielräume der produzierenden Industrie gegen Produktpiraterie
München: TCW 2007 (Forschungsbericht). ISBN: 3-937236-63-5.

WILMS 2006
Wilms, Falko E. P. (Ed.): Szenariotechnik. Vom Umgang mit der Zukunft. 1st ed.
Bern, Stuttgart, Wien: Haupt 2006. ISBN: 3258069883.

YOUNG ET AL. 2010
Young, M. L.; Barrows, A.; Stockman, J. C.: Access 2010. all-in-one for dummies
Hoboken, NJ: John Wiley & Son 2010. ISBN: 0470532181.

ZWERENZ 2009
Zwerenz, C.: Geschftsprozesse im projektmanagement. Best practices der implementierung
Hamburg: Diplomica 2009. ISBN: 9783836682749.