# Requirements for Complex Systems Modeling

Revision February 2008

T.J. van Beek[1], T. Tomiyama[1]

[1]Intelligent Mechanical Systems group, Department of Mechanical Engineering,

Delft University of Technology, Delft, the Netherlands

**Abstract**

Function modeling (FM) is the name given to the activity of developing models of devices/products/objects/processes based on their functionalities. Problems still exist in the design processes of complex systems which FM claims to address. Fundamental to these problems is the lack of system overview models. FM theories address issues including knowledge representation problems in product development, overall description and better understanding of complex systems. This paper investigates and analyzes why FM is not used in industry. It will indicate some weaknesses and shortcomings in FM theories and industry which need to be addressed in future research.

**Keywords**:

function modeling, system overview, complex systems

## 1 INTRODUCTION

Creating better design quality of high tech mechatronic products in less available design time is the aim of many companies. Accomplishing it is not a trivial task in a product development process with growing complexity. This research aims at improving the design process of complex, mechatronic systems. Complexity is present and causes problems in different aspects of the product development process. Therefore active complexity management in engineering design is essential but it has not yet been satisfactorily addressed in literature and practice [1].

The first cause of complexity in high tech mechatronic products is that they are characterized by a multi disciplinary nature. Conventional engineering disciplines such as mechanical, electronics, electrical and informatics are combined to create products capable of developing better and faster than conventional, mono disciplinary products. Managing and coordinating this multi disciplinary product development process is extremely difficult [2].

The size of products, product design processes and organizations is a second source of complexity in product development. Both the market pull by the user and the technology push drive products to incorporate more and more functionality and components. To create these products, the design processes and development organizations have also grown. Large industrial companies often have to manage their design process that is spread over a number of sites around the world with hundreds of product developers working on the same projects [3].

Lindemann and Maurer [1] recognize that controlling product complexity has become an important issue in product development and they state that although reducing complexity is purposeful, it is not favorable to reduce it at any cost. They introduce a methodology to control complexity consisting of a system definition phase that identifies the domains, multi-domain analysis phase where intra-domain networks are identified in the form of matrices and finally analysis of the selected networks using criteria based on graph theory.

Clarkson et al. [4] recognize the importance of dealing with changes in the design of complex systems like a helicopter. They developed a method for predicting and managing change in designs following new requirements based on design structure matrices.

This industrial research into difficulties experienced during product development, related to complexity, indicates problems in managing the above described complexity in the daily product development processes.

The first symptom observed is poor predictability of the consequences following certain design choices. Since no product development process starts from scratch, the starting point of each process is an existing product which will have added or changed functionality. Predicting the consequences of proposed changes in the product architecture has proven to become difficult or even impossible.

Problems in choosing a good system decomposition of the product into convenient modules or sub systems is the second symptom which indicates a complexity issue. System architects choose the interfaces for the high number of sub systems based on their expertise and experience but they do not have a good way of evaluating their choices.

As a third symptom we mention here the increasing number of unexpected and unforeseen problems during the test phase of the complex products. Similar to the first two, this third symptom also results in an increased time to market of the product.

This paper analyses the symptoms of complex product development difficulties identified in an industrial observation. An FM approach is proposed to improve design support for complex system architects in the conceptual stage of the design. The aim of this paper is to identify requirements for modelling complex systems with an FM approach.

In section 2 this paper discusses observations done in industry that indicate product development difficulties. Section 3 will discuss FM and how FM can be used to address the observed industrial problems. Section 4 concludes this paper and gives a brief description of the future work in this research.

## 2 OBSERVATIONS IN INDUSTRIAL PRACTICE

This paper investigated and analyzed problems in the process of adding new features to existing complex system architectures. Depending on the needs and nature of these new features it has to be decided by the system architects whether it will be implemented as a separate sub system, or as an integrated part of the existing main system. This decision process is a part of system decomposition tasks and interface management. The goal of this system decomposition process is to create an engineering decomposition of the product that fulfills the high level needs of the user.

In the introduction of this paper we discussed three symptoms that indicate problems in the system decomposition process, namely:

- Difficulty in predicting and evaluating consequences of proposed system changes.
- Difficulty in creating a total system decomposition that supports the newly added features and does not compromise other features.
- The increased time to market resulting from unexpected problems and more time needed then expected for testing and problem solving.

### 2.1 Experienced difficulties

**Design understanding and traceability**

The process of adding features to an existing product often starts at high level of system abstraction. Typically the market analysis shows that there is a need in the market for a certain feature and that it is a good business opportunity to address this need. The top level management then makes the decision to create a product that fulfils the need. Consulting different sections of their organization like research and development, marketing and engineering departments, the product development process is started.

In the first phase of the project the needs are made clear. In this phase the project team is relatively small and consists of members from different disciplines within the organization. Typical methods for communication used in this phase are workshops and meetings. These sessions result in a high level description of what the new system should do (functions) and how well it should do its task (requirements). Typically these descriptions are captured in specification documents and spreadsheets.

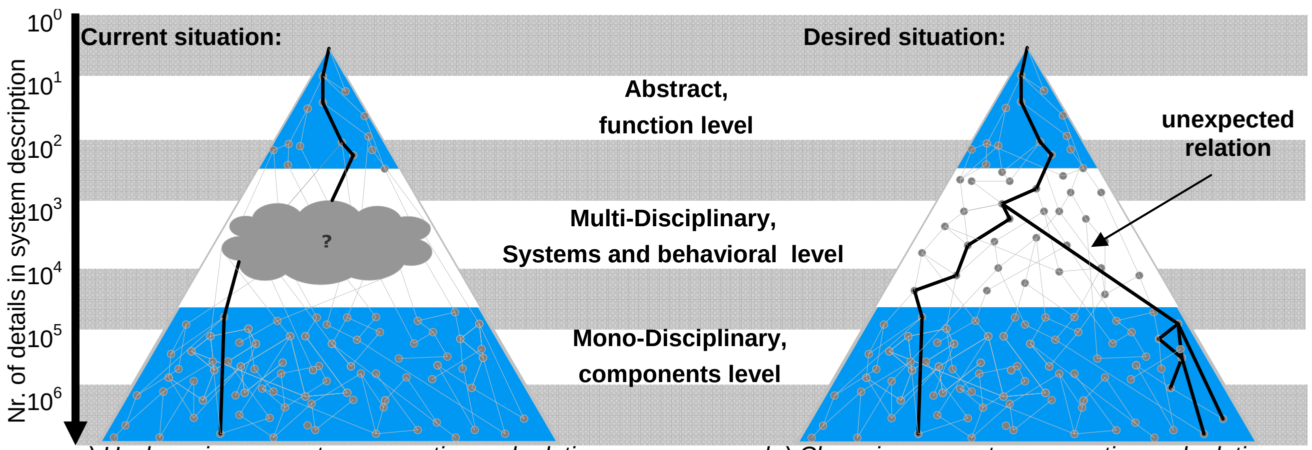
After this first phase the number of people involved increases and the work is divided among them. The product description transforms from abstract concepts at first to detailed component descriptions in the end (lines of code, 3D CAD models, Finite Element Analysis mesh and design documentation).

The transitions from one level of abstraction to another often are bidirectional, iterative processes. Because of the large amount of design knowledge, good traceability of the relations between design aspects in different levels of abstraction is difficult to realize in complex multi disciplinary design processes. There is a need for better traceability of design requirements and system decomposition choices [5]. Improved traceability will help evaluate the consequences of architectural choices when features are added to the system.

Both the size of the knowledge embedded in the designed product and the knowledge gathered in the design process is growing. The size of the problems has grown beyond the limits of one person's comprehension [6]. In our research it is estimated that maybe 0.5 % of all employees have a total system overview. And even then with a strong focus around their expertise. Not understanding the system you are working with is a source of uncertainty and errors in the design. The need for better system understanding was recognized by system architects as one of the most important issues in modern day engineering.

**System Decomposition**

System decomposition and interface management are two of the main activities of complex system architects. They need to decompose the system into smaller sub systems. The goal of this activity usually is to create sub systems which are relatively independent of other sub systems. With a proper decomposition it is possible to create independent sub systems that have their own development cycles. Where two sub systems meet an interface should be defined. Ideally these interfaces are independent of the sub systems implementation. Creating




*a.) Unclear view on system properties and relations*   *b.) Clear view on system properties and relations*

Figure 1 **Schematic representation of system complexity** an ideal interface description for one sub system often conflicts with the ideal interface for another sub system. It was observed that navigating through the product configuration space is very difficult without methods and tools that assist the architects.

### 2.2 Bird's-eye View

To increase design traceability we need models of complex systems that connect high levels of abstraction to detailed levels of abstraction. Most models used now, do not span different levels of abstractions. Most models used now, do not span different levels of abstractions [7]. For example, a mechanical 3D CAD concerns only shapes and assembly of components, and does not link to functional information. The requirement specification sheets for example are models made for different levels of abstractions, but in a discrete manner. A requirement spec. sheet is made for the product at different levels, but it is not really combined into one traceable knowledge carrier covering the complete product development. Changing this will help architects in better evaluation of their choices.

To increase system understanding a map (figure 1) is needed that communicates the system composition and outline to the architects. A modern high tech product typically has details that reach $O(10^6)$. For example, an aircraft has unique components of this order. Complex mechatronic machines (such as mobile phones, medical systems, hybrid car) are controlled by software that has number of lines in the same order of magnitude. At the top level there are abstract functional descriptions. At the bottom, component details of that order are needed, but at this level descriptions are very much mono-disciplinary and their complexity is high but manageable if engineers are provided with good tools. However, the middle layer is systems level multi-disciplinary. The current industrial situation lacks a good way to deal with this level.

In figure 1 the current situation is illustrated as having a cloudy, unclear area of relations between high level, system descriptors and low, concrete level descriptors. Starting at a node at the top side of the system description pyramid it is not unambiguously clear what relations connect it to nodes on the other side of the cloud. One view on the chain of relations is given in figure 1 a.

In figure 1 b, a more desirable situation is illustrated. In this situation the cloud has disappeared and the view on the system is clear and unambiguous. Now the view is cleared it is noticeable that the top level descriptor isn't just related to some mechanical parts in the left side of the pyramid, but also to some lines in the software code displayed in the right hand side of the figure. This is valuable input information for the system architect in the development process. Using such a map the architects and designers could navigate through the design. They would recognize the structure of the product, and sub system interfaces would become clearly visible. The architects would have a clearer and more complete view on the system and would make their decisions based on a better foundation.

Such a map of the system would greatly support system decomposition tasks. When new features are added to the system, the architects open the map and browse the map to find out where to position their added functions, requirements, modules or components. Inserting the new features and connecting them to the existing system will reveal to the architect what parts of the system are influenced by the new features. Once the correct position is chosen, the interfaces can be specified.

### 2.3 An FM Approach

The still mainly scientific field of FM claims to address several of the above mentioned problems. In the following section of this paper an introduction to FM and an approach to the sketched problem of creating a bird's-eye view using FM are given. The advantages and drawbacks of this approach are considered

## 3 FUNCTION MODELING

### 3.1 Introduction to FM

This chapter will first give a brief summary of Function Modeling (FM) based on the author's previous work [8].

FM is developing models of devices/ products/ objects/ processes based on their functionalities and the functionalities of their sub components. Such a high level representation scheme of objects provides many facilities. Some of these schemes include an overall system description to facilitate the communication and understanding between engineers of various disciplines and means to use the computer for reasoning purposes.

The basic concern of FM is how to represent knowledge about function. The representation framework serves as a general and common communication frame on one hand, and to accommodate automated reasoning systems on the other.

FM is not just about modeling system functions. Modeling relationships between functions, behaviour and structure makes FM an interesting candidate to assist system architects to arrive at good system decomposition into components and modules.

FM provides a framework for overall system description. By supporting decomposition of functionalities within one consistent model, FM bridges the gap between the high-level requirements and the low-level details. Such a common model provides a holistic view of the system above the domains of different expertises and makes it possible to go back and forth in the design process in order to check the satisfaction of high-level requirements by the lower level specifications.

A functional model shows how the general goal of a system is achieved by realization of sub goals via the sub functions in the system. Quoting Kitamura et al. [9], 'functional models represent a part of (but not all of) the designer's intentions, so called design rationale'. The framework which provides the viewpoints and the necessary vocabulary in order to represent functional knowledge is called a "functional ontology" ([9], [10]).

### 3.2 Functional Ontology

The functional concept ontology aims to develop the necessary framework and language to model the functionality of a system from the subjective viewpoint of the human (the designer, user, or developer). The work of De Kleer and Brown [11], Chandrasekaran and Josephson [12], Umeda et al. [13], Umeda and Tomiyama [14], Yoshioka et al. [15], Gero [16], and Keuneke [17] are attempts to build functional ontologies. For this research the concepts of the Function-Behaviour-State (FBS) model of Umeda and Tomiyama [18] will be used.

The FBS method deals with three main concepts, namely; function, behaviour and state. All three concepts are independent of engineering disciplines. Function for example is the top level concept that is closest to the user's need. Function is a concept applicable to both hardware and software and from a purely mechanical to an electronically controlled servo system. This discipline independency makes it possible to represent mechatronic systems in one model.

Umeda and Tomiyama [14] delineate two phases for the design process. In the first phase the user describes functions independent of any physical behaviour or system structure. In the second phase the designer enters the objective layer by embodying the functions into behaviours and structural models. Umeda and Tomiyama mention that manipulation of the behavioural structure is possible by making use of qualitative physics. On the other hand, the mental simulation of functions is noted to be still difficult to be done by computers. The FBS modeling is proposed as a new knowledge representation scheme to systematize functional decomposition in the subjective realm and then to develop a CAD system that helps the embodiment of the designed functions into a behavioural and structural system in the objective layer.

In their FBS model Umeda, Tomiyama and their colleagues develop a function representation, in which the subjective and objective layers are related to each other by function-behaviour relationship. The authors define the function as ‘a description of behaviour recognized by a human through abstraction in order to utilize it’. They argue that it is difficult to disassociate function from the behaviour; therefore, they represent function as a tuple in which both the human intention (function as to do something) and physical semantics (behaviour) are represented. In this way they come up with a representation through which the subjective selection of some behaviour as a function is formalized.

### 3.3 Functional Decomposition

Umeda and Tomiyama [14] consider one of the basic tasks in design to be a hierarchical decomposition of functions, which is followed by embodiment in order to arrive at substantial components at the objective level. They argue that, hierarchical decomposition is possible only in the subjective layer by making use of function, rather than the behaviours or any other objective category. In Umeda et. al [18], the authors argue, there is no objective method nor algorithm for functional decomposition. The process of functional decomposition includes both “top-down decomposition” and “bottom-up recognition” of some functions from lower level sub-functions.

What FM provides for the design process is basically a model based on the functionalities and sub-functionalities within the system. Making use of such a model in the early phases is significant for managing the increasing complexity of the design processes. This is acknowledged by America and Wijgerden [19] who make use of extensive requirements modeling in a real industrial application. Bonnema and Van Houten [7] investigate the use of models in conceptual design. They observe that models are used by designers to handle large amounts of data, for communication purposes and for analyzing of the problems. Yoshioka et al. [15] demonstrate that functional models provide a structure for the design process and ease the handling of large amounts of data.

### 3.4 FM Need

Considering the case where the system architects need to add a new feature to an existing system, the functional level is the most natural level to start. Adding a new feature means that we want the system ‘to do something’ new. As we have seen in the foregoing section ‘to do something’ is the short definition of function in FBS. In the newly added function description there is not yet a choice on how to implement the function. Often the added functions can be decomposed into sub functions. This decomposition process makes the architects change the abstraction level they are thinking about the system.

Once the system architects have determined which lowest level functions are to be added, they will start thinking about how to realize these functions. In other words what behaviour is needed to implement the functions. This function-behaviour relation is a subjective one-to-many mapping and is the in between step in going from a function, to detailed system components or state. Because the relations between the functions, behaviours and the state are captured in the FBS model there is traceability of the system objects. All low level components can be traced back to the top level function they originated from by following the relations.

All together the FBS model creates both a visual model that could help the architects in getting a better understanding of their system on different levels of abstraction, and a data object model that captures and stores design data in an continuous, overall system model knowledge base.

### 3.5 Extending FM

There are some reasons however why FM is not widely used in industry already to solve these kind of problems. Because the research area of FM is still relatively new, not a great deal of tools and methods are commercially available. The methods and tools that do exist, for example the FBS modeller [13], are mostly used in research labs and for dedicated case studies. This means that the methods and tools are not yet as commonly known and accepted in industrial practice as for instance 3D CAD modelling tools.

#### Ontology Problems

One fundamental issue in FM is the ontology problem. By the ontology problem we mean that it depends on the ontology used, in a certain method, how the FM method can describe certain functions. The ontology provides the frame in which the system is captured. If the frame is too narrow it might not allow for certain functions to be included into the model as desired. When the frame is too broad it will allow all functions to be included, but it will be difficult to create a manageable design object data model since all objects are allowed to be so different.

Take for example the well known systematic engineering design method from Pahl and Beitz [20], in this method FM is one of the activities in the conceptual design phase. In the ontology that Pahl and Beitz use for their FM they define function as the general input/output relationship of a system whose purpose is to perform a task. It represents a flow of energy, materials or signals. Functions are decomposed into sub-functions and usually have the “noun” and “verb” form. When we try to use this definition for design objects in where there is no energy, material or signal flow we run into trouble. Think for example about the head support beam of a music headphone. Although it has a distinct function in the users perspective to keep the device in the vicinity of the ears, it can not be characterized by  a flow of material, energy or signal. It is not a straightforward task to make a function structure of this device using the Pahl and Beitz definition. Although the ontology used in the FBS method could deal with this headphone example, it does have difficulties in other examples like ‘to facilitate cable management’ in a system.

**Missing Modeling Entities**

A second fundamental problem is that top level technical functions often do not directly map onto the user needs. There are intermediate stages in between. These could be additional boundary conditions and requirements for example that the organization poses on the product development due to strategic considerations. These additional requirements are not directly translatable into functions of the system and are not related to other stakeholders like for example the organizational view of the product development process. (figure 2), but they do have to be met and they are valuable to include into the system overview model because they contain important design rationale and knowledge.

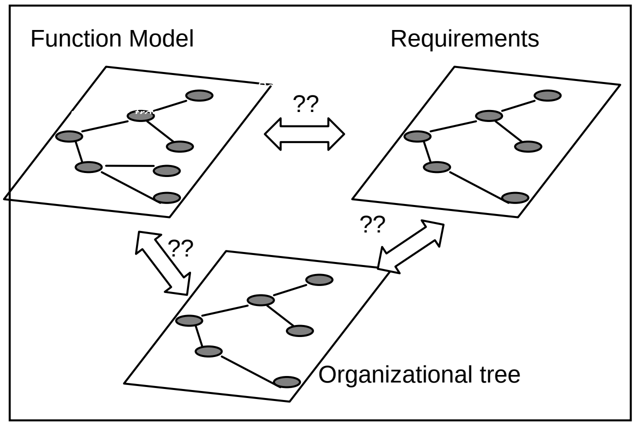



Figure 2 : Illustration of possible interesting relations between system design entities.

These kinds of requirements should be traceable. This indicates that only considering functions in this model might not be enough. It could prove very valuable to include a concept like requirements, which is closely related to function, in a complex systems overview model that we are developing. Discovering what entities have to be involved in the models is an important part of this research.

**Systems Decomposition**

A third problem is that the existing FBS method does not really have a facility to consider systems of systems decompositions. Complex mechatronic systems nowadays are systems that consist of many sub systems which are decomposed in a certain way. In a modelling activity as described in this paper it would be convenient to manage and create models of sub systems with regard to the other systems in its surrounding in a systematic, clear manner. This research will investigate this drawback of the FBS method and tools.

There are also some practical drawbacks of using FM in industry. Most organizations use the term function in their conventional complex product development processes. The term is used freely and not as a distinct part of some model. In practice we observed that talking about functions is not a problem in an industrial environment, but talking about the functions with both parties having the same definition of function is sometimes problematic. In some cases engineers use more the term requirements for concepts that we would have labelled as functions for example.

Some issues that need to be added to FM and FBS in particular are summarized here:

1. Create a better usable systems overview.
2. Support system architects in the systems decomposition task. Allow them to understand their systems better and to make a better evaluated design decision.
3. Create a platform that allows system architects to trace relations between entities in the system. This can be realized by highlighting relations between entities on different levels.
4. Detect interaction between different sub-systems. This can be both desired and undesired interactions like discussed by d'Amelio and Tomiyama [2].
5. Once the architects have made design decisions on system interfaces, they have to manage them. FM should support interface management throughout the design process
6. Handle non traditional functions like "facilitate cable management".

## 4 FUTURE WORK

This paper recognized and analyzed an industrial problem and proposes an approach to solving this problem using FM.

The next step in this research is to realize the clear system view described in chapter 2 and schematically visualized in figure 1. This bird's-eye view will be created for a real complex mechatronic case. The created view will be presented to the system architects and will be evaluated.

As mentioned in chapter 3 the FBS method will be used as a starting point for creating this system overview model, but it will not be just FM. Requirements as a design concept and design knowledge carrier will be used in addition to a FM model of the systems. How to adequately model the relations between the functions and requirements will be researched. In the coming research it will be investigated if function and requirements are all the concepts needed to create a clear system overview. Other possible concepts to be added to this list are: organizational structure, design time, employees' involvement, design decision responsibility structure and others.

A graphical user interface will be developed to present and communicate the systems view to the users, the system architects. Behind the user interface a design data model and knowledge base will be implemented to manage the large amount of data. What is needed for good user interaction and usability will be part of the coming research.

The created overview model will be evaluated in a system decomposition exercise where new systems are added to an existing main system. Measuring the performance of the system will be done by interviews , observations and design quality assessments by experienced architects.

## 5 ACKNOWLEDGEMENT

This work has been carried out as a part of the DARWIN project at Philips Medical Systems under the responsibilities of the Embedded Systems Institute. This project is partially supported by the Dutch Ministry of Economic Affairs under the BSIK program.

## 6 REFERENCES

[1] Lindemann, U. and M. Maurer. Facing Multi-Domain Complexity in Product Development. in The future of product development, Proceedings of the17th CIRP Design Conference. 2007. Berlin: Springer-Verlag.

[2] Tomiyama, T. and V. d'Amelio. *Towards Design Interference Detection to Deal with Complex Design*

*Problems*. in *The future of product development, Proceedings of the 17th CIRP Design Conference*. 2007. Berlin: Springer-Verlag.

[3] Szykman, S., et al., *A web-based system for design artifact modeling.* Design Studies, 2000. **21**(2): p. 145-165.

[4] Clarkson, P.J., C. Simons, and C. Eckert, *Predicting Change Propagation in Complex Design*. 2004. p. 788-797.

[5] Maletz, M., et al. *A Holistic Approach for Integrated Requirements Modeling in the Product Development Process*. in *The future of product development, Proceedings of the17th CIRP Design Conference*. 2007. Berlin: Springer-Verlag.

[6] Szykman, S., et al., *A web-based system for design artifact modeling*. 2000, Elsevier. p. 145-165.

[7] Bonnema, G.M., *Use of models in conceptual design.* Journal of Engineering Design, 2006. **17**(6): p. 549-562.

[8] Erden, M.S., et al., *A Review of Function Modeling: Approaches and Applications.* AIEDAM: Multi Modal Design, special issue. in press.

[9] Kitamura, Y., et al., *Deployment of an ontological framework of functional design knowledge*. 2004, Elsevier. p. 115-127.

[10] Kitamura, Y. and R. Mizoguchi, *Ontology-based description of functional design knowledge and its use in a functional way server*. 2003, Elsevier. p. 153-166.

[11] De Kleer, J. and J.S. Brown, *A Qualitative Physics based on Confluences*. 1984. p. 7-83.

[12] Chandrasekaran, B. and J.R. Josephson, *Function in Device Representation.* Engineering with Computers, 2000. **16**(3-4): p. 162-177.

[13] Umeda, Y., et al., *Supporting conceptual design based on the function-behavior-state modeler.* Artificial Intelligence for Engineering Design, Analysis and Manufacturing, 1996. **10**(4): p. 275-288.

[14] Umeda, Y. and T. Tomiyama, *FBS Modeling: Modeling scheme of function for conceptual design*, in *Proc. of the 9th Int. Workshop on Qualitative Reasoning*. 1995. p. 11-19.

[15] Yoshioka, M., et al., *Physical concept ontology for the knowledge intensive engineering framework.* Advanced Engineering Informatics, 2004. **18**(2): p. 95-113.

[16] Gero, J.S., *Design Prototypes: A Knowledge Representation Schema for Design.* AI Magazine, 1990. **11**(4): p. 26-36.

[17] Keuneke, A.M., *Device representation-the significance of functional knowledge.* IEEE Expert [see also IEEE Intelligent Systems and Their Applications], 1991. **6**(2): p. 22-25.

[18] Umeda, Y., et al., *Function, Behaviour and Structure, Application of Artificial Intelligence in Engineering V, Vol 1: Design, JS Gero*. 1990, Computational Mechanics Publications, Boston.

[19] America, P. and J. van Wijgerden, *Requirements Modeling for Families of Complex Systems*, in *IW-SAPF 3: Third international Workshop on Software Architecture for Product Families*. 2000, Springer: Las Palmas.

[20] Pahl, G. and W. Beitz, *Engineering Design: A Systematic Approach*. 1996, Berlin: Springer-Verlag.