Connecting Views in Mechatronic Systems Design, a Function Modeling Approach

Thom J. van Beek, and Tetsuo Tomiyama, *Member, IEEE and ASME*

***Abstract*—This paper discusses problems in complex mechatronic system design observed in an industrial setting. In the process of adding functionality to mechatronic systems it is important to control complexity. It is observed that an important aspect in controlling complexity is keeping a system wide overview and understanding among system architects and engineers. A function modeling approach is proposed to connect different design aspect views (e.g. functional view, workflow view, requirements view) on different levels of abstraction to support the architects in the conceptual design process. The overview will improve system understanding.**

I. Introduction

Introducing additional features into complex mechatronic systems is a difficult task. Therefore active complexity management in engineering design is essential but it has not yet been satisfactorily addressed in literature and practice [1].

This research considers a medical system (e.g. MRI machine) as a case of complex mechatronic, embedded systems. The complexity is caused by the strong multi-disciplinary nature (e.g. mechanics, electronics, computer science, and fundamental physics). Managing and coordinating this multi disciplinary product development process is extremely difficult [2] and exceeds the comprehension of a single engineer who cannot understand every detail ([1], [3], [4]).

Lindemann and Maurer [5] 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.

To control complexity a shared model between architects and engineers is needed, the Functions and Key drivers (FUNKEY) [6] method proposes relating system's functions to key drivers and requirements and coupling them in a matrix. The method seeks mainly to provide an easy way of documenting a certain choice for an architecture and its performance providing the system architect with an overview of his choices.

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.

Thom J. van Beek is with the Delft university of technology, Delft, 2628 CD the Netherlands (corresponding author, phone: +31 (0)6 48254856; e-mail: t.j.vanbeek@tudelft.nl).

Tetsuo Tomiyama is with the Delft university of technology, Delft, 2628 CD, the Netherlands (e-mail: t.tomiyama@tudelft.nl).

Importance of a system architecture overview is also recognized by the CAFCAR [7] model. It proposes a decomposition of the architecture into five views that capture the needs of the customer, the functions the product performs, and the design of the product from the conceptual and realization standpoints.

This paper first discusses complexity overview problems based on observations done in the conceptual design process of a medical embedded system in industry. It describes our approach to solving complexity overview problems using Function Modeling (FM) to create a system wide model that relates system level considerations to components level. A computer tool is proposed that will improve system transparency. It will allow the system architects to navigate and trace design decisions through different views by means of implementing FM. In section II this paper will discuss observations done in industry on different kinds of complexity. Section III will discuss how FM could address the problems found in industry. Section IV will focus on one aspect of the complexity problem and deduct industrial needs for a computer based design support tool. The application of the FBS modeler [9], which is considered to be a candidate tool to fulfill the need, will be discussed. Finally the proposed tool is discussed as future work.

II. Industrial Case Study

This research investigates and analyzes problems in the process of adding new features to existing complex system architectures.

The observations are done in the conceptual phase of the design process of an added functionality to the overall system. Interviews with a system application, marketing, and engineering expert were conducted. The project leader was interviewed to give his view on the total process. Besides interviews requirements documentation and interface specifications were used as source.

A. Observations

We found 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 configuration decomposition that supports the newly added features and does not compromise other features.
- Increased time to market resulting from unexpected problems.

1) Design Traceability

Typical methods for communication used in the conceptual design phase are workshops and meetings. These sessions result in a high level description of how the system should be used (workflow or scenarios), what the new system should do (functions) and how well it should do its task (requirements). Typically these descriptions are only captured in documents and spreadsheets.

The transitions from one level of abstraction to another often are iterative processes both ways. Because of the large amount of design information content, good traceability of the relations between design aspects in different levels of abstraction is difficult to realize in complex multi-disciplinary design processes.

2) Design Understanding

Both the size of the information embedded in the designed product and the information gathered in the design process is growing. The size of the problems has grown beyond the limits of one person's comprehension ([1],[3],[4]). In our research it was estimated by architects that maybe 0.5 % of all employees have a total system overview. Not understanding the overall system is a source of uncertainty and errors in the design.

3) System decomposition

System architects decompose the system into smaller sub systems. Where two sub systems meet, an interface should be defined. Creating an ideal interface description for one sub system often conflicts with the ideal interface for another sub system.

The systems are highly customizable and therefore configurations exist as a sub set of all available sub systems. It was observed that navigating through the product configuration space is very difficult without models and tools that support the architects.

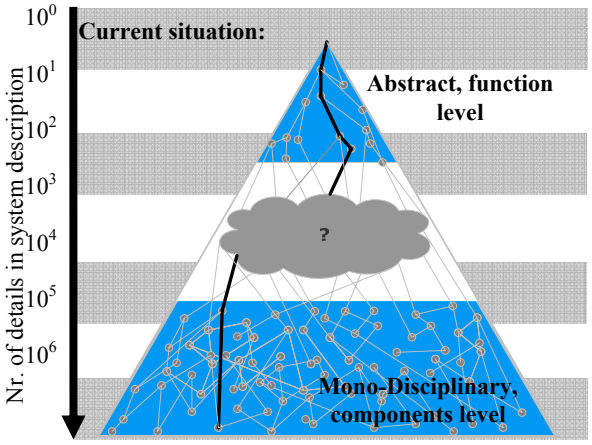

Fig. 1. Schematic representation of system complexity. The cloud represents the unclear view on system properties on systems level during the conceptual design process and makes it impossible to trace the thick line.

B. What is Missing? Bird's-eye view

To increase design traceability we need models of complex systems that connect high levels of abstraction to low levels of abstraction. Most models used now, do not span different levels of abstractions [10]. For example, a mechanical 3D CAD concerns only the geometry of components, and does not link to functional information. The link between these aspects is missing. They are not considered in parallel and connected, but sequential and only linked in the mind of the designers.

When for example changes are executed in the workflow models of the system, the designer has to determine manually where he has to change the requirement and function models. There is a need for better traceability of design requirements and system decomposition choices [11].

To increase system understanding a map (shared model) (Figure 1) is needed that communicates the system composition and outline to the architects. A modern high tech product typically has details that reach $O(10^7)$. For example, an aircraft has unique components of this order. Complex mechatronic machines (e.g. mobile phones, medical systems, printers, 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 dedicated tools. However, the middle layer is systems level multi-disciplinary. The current industrial situation lacks a good way to deal with this level.

C. Proposed Solution based on FM

The solution for the problems described should fulfill the following key features necessary for a Bird's-eye view:

- Flexible in level of abstraction. It should connect high level of abstraction functions to low level detailed component descriptions.
- It should be capable of capturing and storing design information in different design dimensions.
- It should support the architects in navigating through the system in an intuitive way. The view should be similar to the model an architect has in his mind to make the model useful for communication purposes.

Using a model in the early phases is significant for managing the increasing complexity of the design processes. This is acknowledged by America and Wijgerden [12] who make use of extensive requirements modeling in a real industrial application. Bonnema and Van Houten [10] 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 problems.

What is needed is a model that connects different design

aspects, e.g. function and workflow, at different levels of abstraction. In this paper a method and a tool are proposed based on the FM technique of Function-Behavior-State (FBS) modeling [13]. An FBS approach is considered because it already integrates design concepts at different levels. The FBS model creates system overview from the early, abstract levels of needs and functions through the concepts of objective system behavior all the way to low level detailed component descriptions. FBS also can potentially be connected to the workflow system view by exploring the state transition relation (behavior) of the FBS model.

III. Function Modeling

A. Introduction to FM

This chapter will first give a short summary of a Function Modeling (FM) based on the author's previous work [14].

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 with regard to behaviors and structure. The representation framework serves as a general and common communication frame on the one hand, and accommodates automated reasoning systems on the other.

FM is not just about modeling system functions. Modeling relationships between functions, behavior 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 expertise 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. [15], '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" ([15], [16]).

B. 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 [17], Chandrasekaran and Josephson [18], Umeda et al. [9], Umeda and Tomiyama [13], Yoshioka et al. [19], Gero [20], and Keuneke [21] are attempts to build functional ontologies. For this research the concepts of the Function-Behaviour-State (FBS) model of Umeda and Tomiyama [22] will be used.

The FBS method deals with three main concepts, namely; function, behavior 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 [13] delineate two phases for the design process. In the first phase the user describes functions independent of any physical behavior or system structure. In the second phase the designer enters the objective layer by embodying the functions into behaviors and structural models. Umeda and Tomiyama mention that manipulation of the behavioral 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 behavioral 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-behavior relationship. The authors define the function as 'a description of behavior recognized by a human through abstraction in order to utilize it' [13]. They argue that it is difficult to disassociate function from the behavior; therefore, they represent function as a tuple in which both the human intention (function as to do something) and physical semantics (behavior) are represented. In this way they come up with a representation through which the subjective selection of some behavior as a function is formalized.

C. Functional Decomposition

Umeda and Tomiyama [13] 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 behaviors or any other objective category. In Umeda et. al

[22], 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. Yoshioka et al. [19] demonstrate that functional models provide a structure for the design process and ease the handling of large amounts of data.

D. Need for FM

Considering the case where the system architects need to add a new feature to an existing system, the functional view is the most natural view to start. Adding a new feature means that we want the system 'to do something' new. '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 behavior is needed to implement the functions. This function-behavior relation is a subjective one-to-many mapping between functions and the detailed system components, or state. Because the relations between the functions, behaviors 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. FBS does this by considering the connection between the functional and structural levels.

E. Extending FM

Despite the promises of FM it is not widely used in industry to solve the mentioned 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 modeler [9], 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 modeling tools.

1) 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 of Pahl and Beitz [23]. 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.

2) 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 (Figure 2) and are not related to other views like the workflow, but they do have to be met and they are necessary to include into the system overview model because they contain important design rationale and information.

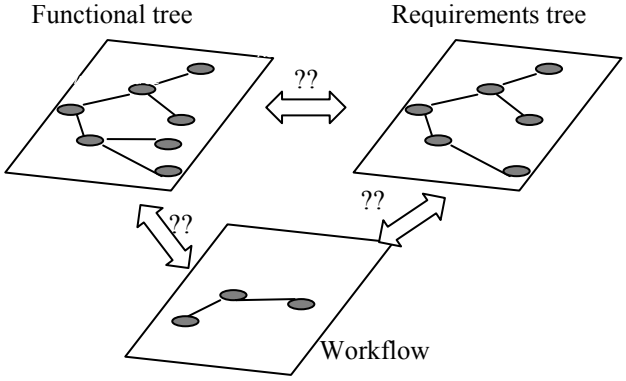

Fig. 2. Illustration of possible interesting relations between design entities like functions, requirements and workflow.

These kinds of requirements should be traceable. This indicates that only considering functions in this model is not enough. Discovering which entities have to be involved in the models is an important part of this research.

3) Systems decomposition

A third problem is that the existing FBS method does not have a facility to consider system of systems decompositions. FBS does support functional and behavioral decomposition, but doesn't support decomposing systems into smaller systems. In a modeling activity as described in this paper it would be convenient to manage and create models of sub systems both individually as well as in relation to the surrounding systems. This research will investigate this drawback of the FBS method and tools.

There are also practical drawbacks of using FM in industry. Most organizations use the term function in their product development processes. The term is used freely and not as part of a 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 labeled as functions for example.

Some issues that need to be added to FM and FBS in particular in order to address the problems found in section II are summarized here:

- Create a better usable systems overview.
- Support system architects in the systems decomposition task.
- Create a platform that allows system architects to trace relations between entities in the system.
- Detect interaction between different sub-systems as discussed by d'Amelio and Tomiyama [2].
- FM should support interface management throughout the design process.
- Handle non traditional functions like "facilitate cable management".

IV. Tool Requirements

The objective of this research is to develop computer based support for system architects in maintaining overview and understanding of mechatronic systems in an optimal way and to support them in defining interfaces. Based on the industrial and FM analysis presented in the above sections, desired tool characteristics are now given.

A. Managing design information

The foregoing sections have revealed basically three areas where the system architects can be assisted. The first is to assist the architects in structuring, storing and connecting design information. Because the size of the problem has grown bigger than a human can grasp he needs to be helped in managing design knowledge. This support is already present for detailed components description, but not yet connected to system level functional knowledge. This connection is necessary to capture design rationale.

Having a good ontology (FBS) capable of capturing and presenting necessary modeling entities is essential in managing the design information. In figure 3 the repository and the pyramid are illustrations of the needed ontology for a computer based design support system. The repository serves as the database for the model and the pyramid is the visualization created based on this data.

B. System overview

The second area where the system architect can be assisted is in keeping overview of the system of systems, as illustrated in Figures 1 and 3 by means of a pyramid with nodes and relations between nodes. Navigating through this design data requires maps. These maps, just like traffic roadmaps, have boundaries, legends, nodes and connections with certain other design view maps. The support should allow the user to look at very detailed level (zoom in) as

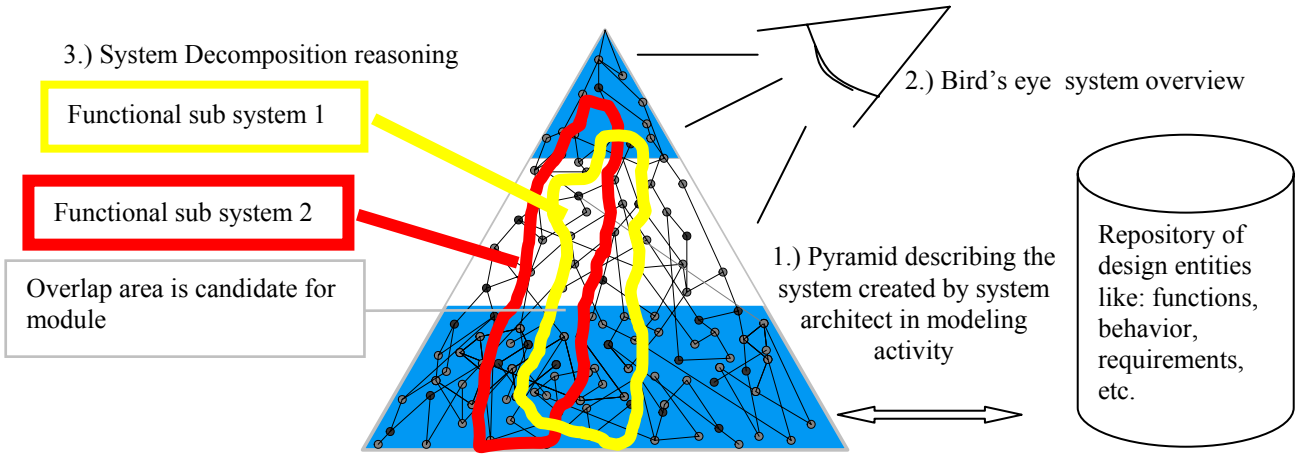

Fig. 3. Objectives of this research in supporting system architects.

well as at a very holistic, zoomed out level. The map shall be used in the process of choosing what will be the best way to add functionality to the system. The map should allow the user to determine, draw border around clusters, and dividing the system up into sub systems.

The FBS model discussed in chapter 3 in essence is a map between high level functions and low level states or components. Therefore combining this model with other engineering views (e.g. requirements, workflow) is chosen for creating the system overview. Determining which other views should be supported is future work of this research.

C. Computer based tooling

The third area supports the architect in reasoning about the system. Computers are capable of dealing with large amounts of data and can find patterns in data structures like for instance the qualitative process based abduction system discussed in [19] which is a computer reasoning system to propose candidate design solutions from a series of qualitative state transitions. A similar pattern searching is present in the process of system decomposition. The architect tries to find the best decomposition of modules so that modules can be exchanged with other systems. Choosing modules in an optimal way cuts back on development costs and will reduce overall system complexity. When a systems overview is available and sub systems with different functionalities are mapped onto this system view, a computer can assist the architect in reasoning about creating clusters in the design. This reasoning can be done based on behavioral knowledge stored in the model. This activity would be based on pattern recognition in the large design database constructed at the modeling phase. In Figure 3 this is illustrated by the overlapping area of two functional sub systems. Developing and determining evaluation criteria for this tool is future work of this research

V. Concluding Remarks

Modern mechatronic product development suffers from increasing complexity due to technology advances. To control the complexity of the architecture of these systems, methods and tools are needed to support system architects in defining and decomposing their architecture.

Traditionally FM was considered only for defining and reasoning about functions. In this paper we propose an application of FM to support architecting of complex mechatronic products. To do so an extension of FBS modeling was proposed.

A computer based tool that captures, stores, and communicates different system views is proposed. We identified requirements for this tool. The tool should give support to the architects on creating bird's-eye system overview, system decomposition tasks, design traceability, system interactions detection, managing interfaces and handling non traditional functions.

Future work includes the development of such a methodology and computer based tool.

References

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

[2] d'Amelio, V. and T. Tomiyama, Predicting the unpredictable problems in mechatronic design, in International conference on engineering design, ICED'07. 2007, Ecole centrale Paris: Paris.

[3] T. Tomiyama, and B. R. Meijer, "Directions of next generation product development," in H.A. ElMaraghy and W.H. ElMaraghy (eds.) *Advances in Design*, Springer, London, 2005, pp.27-35.

[4] T. Tomiyama, V. D'Amelio, J, Urbanic, and W. ElMaraghy, "Complexity of multi-disciplinary design", in *CIRP Annals-Manufacturing Technology*, Vol. 56, No. 1, 2007, pp.89-92.

[5] 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.

[6] G. M. Bonnema, "FunKey architecting - An integrated approach to system architecting using functions, key drivers and system budgets," Ph.D. thesis,, University of Twente, Enschede, The Netherlands, 2008.

[7] G. J. Muller, "CAFCR: A multi-view method for embedded systems architecting," Ph.D. thesis, Delft University of Technology, Delft, The Netherlands, 2004.

[8] 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.

[9] 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.

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

[11] 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.

[12] 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.

[13] 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.

[14] M.S. Erden, H. Komoto, T.J. van Beek, V. D'Amelio, E. Echavarria and T. Tomiyama (2008). A review of function modeling: Approaches and applications. AI EDAM, 22, pp 147-169.

[15] Kitamura, Y. and R. Mizoguchi, Ontology-based systematization of functional knowledge. 2004, Taylor & Francis. p. 327-351.

[16] 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.

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

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

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

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

[21] 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.

[22] 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.

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