# Revisiting the Divide and Conquer Strategy to Deal with Complexity in Product Design

Magdalena K. Chmarra, Andrés A. Álvarez Cabrera, Thom van Beek, Valentina D'Amelio, Mustafa S. Erden, and Tetsuo Tomiyama, *Member, IEEE and ASME*

***Abstract*—Development of modern products is complex and exceeds the comprehension of a single engineer, who cannot understand every detail of a product. Some of the current approaches to deal with the complex product design are based on the "divide-and-conquer" (D&C) approach. Those approaches in their present form, however, reveal shortcomings in many multidisciplinary problems. This paper gives an overview on current D&C approaches to deal with complex products. It also presents the general philosophy of the work carried out in the Intelligent Mechanical Systems (IMS) group toward developing tools and methods for improvement of complex product development.**

## I. Introduction

MANY modern products are very complex due to the involved technologies, the sheer size of the product, and the existing multi-disciplinarity, inter-disciplinarity, or cross-disciplinarity An example of such complex products is a mechatronics product, which incorporates knowledge of different engineering fields (e.g., mechanics, electronics, computer science).

Architecture of modern products is very complex, since the large amount of components and details counts up to $10^7$. Moreover, there are interactions among those components and details. Fig. 1 shows a schematic representation of layers in a multidisciplinary complex product. The top layer is very abstract and consists of a description of functions, requirements, and modules of the product. The bottom layer is strongly mono-disciplinary and consists of technical details (e.g., components). The middle layer is multidisciplinary and consists of systems-level descriptions of behaviors and relations between different subsystems in different domains. Currently, engineers and designers have good tools to deal with the top and bottom layers in many engineering domains. There is, however, no good tool that can deal with the multi-disciplinarity of the product in the middle layer (Fig. 1). Table I lists examples of descriptions in those top, middle, and bottom levels.

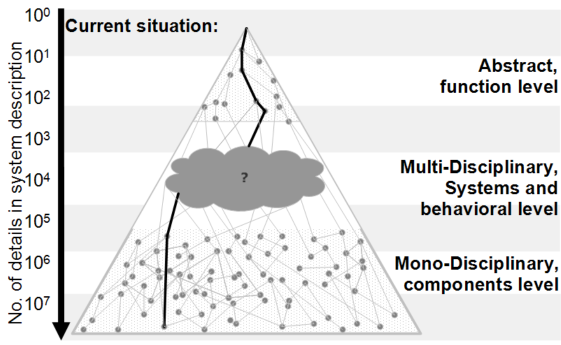


Fig. 1. Representation of product complexity (adapted from [15])

TABLE I
EXAMPLES OF DESCRIPTIONS

| | No. of details | Mechanical | Electronics/ Control | Software |
|---|---|---|---|---|
| <u>Top</u>: functional | $10^0$–$10^3$ | • Functional block diagram | | |
| <u>Middle</u>: systems, behavioral | $10^3$–$10^5$ | • Assembly drawing | • State transition diagram<br>• Transfer function diagram | • Algorithm |
| <u>Bottom</u>: details | $10^5$> | • CAD models<br>• FEM models | • Circuit diagram | • Program codes |

Development of complex products is difficult and exceeds the comprehension of a single engineer, who cannot understand every detail of a product, since he/she is educated and trained in a single discipline [1], [2]. Therefore, development process forces experts from different disciplines to work as a team. Unfortunately, this introduces new problems (e.g., miscommunication – a result of using different common languages among engineers with

The authors gratefully acknowledge the support of: the Dutch Innovation Oriented Research Program 'Integrated Product Creation and Realization (IOP-IPCR)' and the BSIK program of the Dutch Ministry of Economic Affairs, to the projects "Darwin," "Smart Synthesis Tools," and "Automatic Software Generation for Control of Mechatronics Systems"; the Embedded Systems Institute (Eindhoven, The Netherlands) to the projects "Darwin," "OCTOPUS," and "Smart Synthesis Tools"; and to Océ-Technologies B.V. to the "OCTOPUS" project.

Magdalena K. Chmarra is a researcher at the Faculty of Mechanical, Maritime, and Materials Engineering at Delft University of Technology (TUDelft), Mekelweg 2, 2628 CD, Delft, The Netherlands (corresponding author, phone: +31-(0)15-278-3514 e-mail: m.k.chmarra@tudelft.nl).

Andres A. Álvarez Cabrera is a Ph.D. researcher at the Faculty of Mechanical, Maritime, and Materials Engineering at Delft University of Technology (TUDelft), (e-mail: a.a.alvarezcabrera@tudelft.nl).

Thom van Beek is a Ph.D. researcher at the Faculty of Mechanical, Maritime, and Materials Engineering at Delft University of Technology (TUDelft) (e-mail: t.j.vanbeek@tudelft.nl).

Valentina D'Amelio is a Ph.D. researcher at the Faculty of Mechanical, Maritime, and Materials Engineering at Delft University of Technology (TUDelft) (e-mail: v.damelio@tudelft.nl).

Mustafa S. Erden is a post doctoral researcher at the Faculty of Mechanical, Maritime, and Materials Engineering at Delft University of Technology (TUDelft) (e-mail: m.s.erden@tudelft.nl).

Prof. Dr. Tetsuo Tomiyama is Professor at the Faculty of Mechanical, Maritime, and Materials Engineering at Delft University of Technology (TUDelft) (e-mail: t.tomiyama@tudelft.nl).

different backgrounds). Complexity in design increases the complexity of product development processes. For example, as a consequence of mutually connected subsystems, a small change in one domain (e.g., thermal) of a complex product can easily propagate to many other engineering domains (e.g., mechanics, electronics). This can trigger unmanageable system-wide changes and unexpected problems that will be found in later stages of product development.

"Reductionism" – the belief that complicated whole can be explained by considering it as a combination of simple parts – has been often used in science. Engineers divide problems into sub-problems of a manageable size. Those sub-problems are individually solved and their solutions are later integrated to form a solution for the whole problem. This "divide-and-conquer" (D&C) approach requests clear separation of sub-problems. This means that interactions among sub-problems should be clearly identified, specified, and minimized. If these conditions are not satisfied, which is often the case for multidisciplinary problems, D&C cannot be implemented. Therefore, it is desirable to develop new methodologies and tools that can enhance D&C to deal with complexity, particularly, at the systems level middle layer during multidisciplinary product development.

This paper gives an overview on current approaches to deal with complex products. It also presents the general philosophy of the work carried out in the Intelligent Mechanical Systems (IMS) group at Delft University of Technology towards developing tools and methods for improvement of complex product development.

## II. Current Approaches

Several different approaches to deal with the design of complex systems can be found in the literature. In general, they can be classified as technological approaches oriented towards product architecting, organizational, and managerial approaches that suggest methods to organize the design process, and computer-based tools that improve data sharing and management in design. In this section we describe some of the methods and tools pertaining to these categories and then perform a brief analysis around their advantages and disadvantages.

### *A. Technological approaches*

This category includes methodologies that search formalization of the architecting activity; that is, formalize how to derive functions and decompose them appropriately, or how to distinguish good and bad architectures according to some criteria.

Interfaces between the constituents of a system have been identified as a crucial element of a system's architecture, and in fact, the choice of interfaces is representative of the choice of components in a system. Design Structure Matrices (DSM) [3] contain relations between the elements that result from a decomposition process. The matrix representation allows application of clustering techniques to find groups of components and/or subprocesses that are related strongly, and to evaluate the grouping choice by choosing the alternatives that attain higher degree of clustering (e.g., less isolated components that relate to components in other groups). The well known $N^2$ method [4] makes part of this family of methods.

The Functions and Key drivers (FunKey) [5] method proposes relating system's functions to key drivers and requirements and coupling them in a matrix. Functions and key drivers correspond respectively to the headers for rows and columns. The coupling matrix elements contain the contribution of each function to the different key drivers as in a system budget. The method seeks mainly to provide an easy way of documenting a certain choice for an architecture (by the choice of functions) and its performance (by the budget distribution) providing the system architect with an overview of his choices.

Axiomatic Design (AD) [6] applies the decomposition rule of functional independence (the independence axiom) in a zigzag decomposition from the customer domain towards the solution space of the process domain. A design matrix is used to represent the relations between function requirements and design parameters in each step of decomposition allowing to verify and maintain the decoupling through the whole decomposition process. Then, the merit of a design choice (i.e., an architecture) is measured with the information axiom to choose the best design.

CAFCR [7] is a decomposition of the architecture into five views that capture the needs of the customer (in the views of "what?" and "how?"), the functions the product performs, and the design of the product from the conceptual and realization standpoints. CAFCR's author offers about forty sub-methods to help the designers define these views more clearly. The views are captured roughly in a sequence, while keeping track of stakeholder concerns named as "qualities" (e.g., safety, usability, performance) to maintain integrating goals through the views. Story telling and use case techniques are used to explore and analyze specific details of the problem. The method is completed with an explanation of how to perform iteration cycles for defining the elements described previously. This reasoning approach gives the means to integrate the views of the CAFCR method.

### *B. Organizational and management approaches*

One of the methods that comprises most of this philosophy is Concurrent Engineering (CE); also known as simultaneous engineering. Cooperative product development is the essence of CE, and is aimed at being able to produce complex products with short lead time and low cost while achieving quality [8]. The concept emphasizes the importance of the design of production and manufacturing being concurrently conducted along with the other product design activities. To do so, at design stages, various aspects of product's life cycle stages (such as manufacture, maintenance, service, and end of life) should be taken into consideration, resulting in "Design for X" (DfX) methodologies. Regarding complex product development, CE practices reduced the time-to-market and improved product quality at the cost of increased process complexity.

The concept of cooperative product development is based on organizational and management techniques to arrange the execution of design activities and the work groups. This arrangement seeks to obtain a parallel (or concurrent) development of the design activities, and is heavily supported by the use of computer tools to group activities and people, and to share design data.

### C. *Computer-based tools*

Good examples of methods that focus on the capabilities of computers to manage and transfer data are Product Data Management (PDM), Enterprise Resource Planning (ERP), and Product Life-cycle Management (PLM). From the perspective of improved data management, these tools (or methods, depending on the author) share many common points, and differ basically in the range of activities they cover. All of them span over more than just the activities design of the product (seen as the pure solution to the design problem).

As common points we find capabilities for data sharing, application integration, design release management, product structure management, and change management [9]-[12]. Such similarities come from the fact that these tools share origins from environments related to Computer Aided Design (CAD) software [13]. These systems largely decreased process complexity, rather than product complexity, by computerization of data management.

ERP's capabilities pivot around the improvement of production and manufacturing of products [9], [10]. PDM is often confused with ERP, but some authors state that PDM spans towards all the life-cycle of the product [10], while other authors [11] accept the shortcomings of current systems which lack some of the more advanced capabilities that would enable it to cover all the life-cycle (e.g., part classification, systems management, impact analysis). Based on this information, we can say that PDM and ERP stand as part of the product information-managing tools that serve as a base to PLM [13]. As a differentiating factor of PLM we can observe that it devotes an important amount of attention to the modeling of user-related aspects [12].

### D. *Recapitulation*

Technical, organizational and management approaches together with computer-based tools, although closely related, deal with different aspects of development of complex systems. The biggest disadvantage of all the approaches mentioned above is the fact that none of them alone can deal with all the aspects of complex system development. Moreover, so far there is no high-level, top-down modeling framework that is able to capture all the aspects of a product's lifecycle stages and to translate and/or connect them effortlessly [12].

Current approaches lack a formal representation of a product that clearly describes its function, behavior, and states. Instead, they mostly focus on the representation at one level and domain [11]. In this way, the majority of valuable information about products (e.g., geometric representation, function, requirements) is not easily available to the designer.

Another shortcoming of present approaches is their implementation. It takes often a long learning curve before the user is able to fully make use of them [10]. One of the reasons is the lack of user-friendly interfaces. Besides that, there are often problems related to communication of engineers with different backgrounds within a tool [10]. Although many tools make the local communication in single domains easy (e.g., for mechanical engineers), there is still a shortage of support of all the domains together.

All the tools described above can be used together as an aid during development process of a product. The user will find it very difficult to exchange data between all the tools, since they do not share a data language. In the literature, most of the authors agree that UML (serialized in XML) is a promising alternative [12]. Nevertheless, the question of the ideal language that can be used for all the tools and stages of complex systems development is still present.

## III. Approach Of the Intelligent Mechanical Systems Group

The previous section contains a review of tools and methods created to support development process of complex products and systems. In this section we present the general philosophy of the work carried out in the IMS group at Delft University of Technology towards developing tools and methods similar to those mentioned in section II. The application of that general philosophy is described using a number of research projects in which the IMS group works currently [14]-[18]. Since these projects are in an early stage, the case studies are not yet completely defined and, therefore, are not mentioned here.

### A. *General philosophy*

Our fundamental strategy is how to reduce complexity intrinsic to product development of such large-scale multidisciplinary products as modern mechatronics systems. The complexity results from the sheer size of the product (numbers of subsystems and components) as well as from the multi-disciplinarity of the product [2]. It can be the product complexity itself and the process complexity induced by the product complexity.

The D&C strategy seeks to subsequently break up a problem, or more precisely the source of a problem, into several parts until they individually become simple enough to be solved directly. Then, the individual solutions must be combined to arrive to a solution for the initial problem. This strategy has been used successfully for centuries to deal with all sorts of problems, from politics and war [19] to the more modern "D&C algorithm" in computer science [20].

When we try to apply it for the matter in hand, where complexity due to size and multi-disciplinarity are present, we can find that: (a) as we divide the original problem we may loose the sense of the problem as a whole; (b) there is no single, obvious, way to divide the design problem.

Statement (a) leads to problems to integrate the individual solutions, to overlook chances for improvement because the individual problem solutions are left for a single domain, and to miss interactions not taken in account at the moment of the division when a detail about this new level is not completely defined. From statement (b), we obtain decompositions that may seem "logical" or "obvious" to one individual but that may be hard to understand to another one, or that serve more to a few particular interests.

As mentioned in the Introduction, development of multidisciplinary products can introduce interdisciplinary problems. This is often caused by the difficulty to comprehend the whole development process of complex products by one engineer with a background in a single discipline (e.g., mechanical engineering). By definition, multidisciplinary product development includes inter-disciplinary problems. It is difficult to locate and recognize all problems that occur during the development of a complex product having, for example, only the mechanical engineering point of view. In other words, it is almost impossible to find out how to deal with multidisciplinary problems as long as that problem is viewing from a single discipline point of view. Providing a "bird's eye view" of the whole product information is one of the solutions to that problem.

A bird's eye view clearly presents information about both the exact location of the problem and the relations between disciplines in the product. Such information can be used to analyze the structure of knowledge and to find methods to tackle various problems. The bird's eye view alone does not solve interdisciplinary problems, but it offers useful information and advice to find a problem-solving strategy. In the complex product architecture, the bird's eye view provides engineers and designers with clear top-down understanding of the system. It can also be seen as an aid to coordinate their activities in a product development team which consists of multidisciplinary domain experts.

Seeing the problems mentioned in previous sections, the IMS group focuses efforts on two points to allow application of the D&C scheme in the development process of complex systems:

--Applying the correct decomposition: Try to formalize the division process of a design problem or a system so that the result answers the goals of different members of the design team. This also requests keeping in mind that formalization is one of the first steps towards automation.

--Keeping the coherence in the problem solution process: The keyword in this point is abstraction. It is agreed that a good way to deal with complexity of a system is to describe the system at a high level (low resolution) [21]. The main idea is to convey a concept of the whole problem to the groups in charge of the solutions of the individual problems. In this way it is possible to maintain a link between the individual solutions, allowing their coherent and complete integration into the final solution.

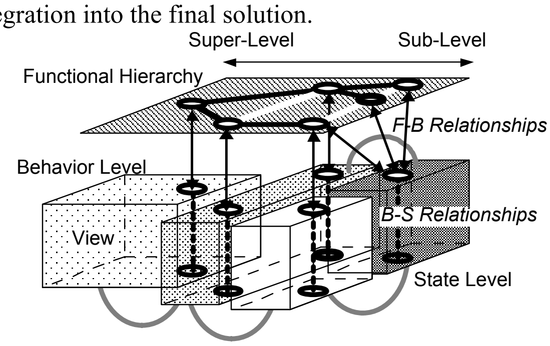


Fig. 2. Scheme of the Function-Behavior-State knowledge representation [22]

The following applications are partially based on the Function-Behavior-State (FBS) knowledge representation [22]. As shown in Fig. 2, the FBS scheme embodies a structure in which the functional description at different levels of abstraction is linked to a behavioral description. In turn, each behavior is described in terms of state transitions. Behaviors and their descriptions are grouped in several "aspects" that correspond to the different points of view of a system (e.g., mechanical, control, economic). The FBS model uses knowledge bases for function prototypes, physical features, and physical phenomena. Function prototypes are built from FBS model pieces known as physical features that relate a function to a structure of entities, physical phenomena, and their relations, providing a way to fulfill the function. After specifying the functions, the user proceeds to decompose them aided by the knowledge from function prototypes, supporting the philosophy of applying the right decomposition.

### *B. Providing a better overview*

Evolving the design into new products with more functionality is a demanding process. Because of the large amount of design knowledge, it is difficult to obtain good traceability of the relations between subsystems in complex multidisciplinary design processes. Creating a system architecture that fosters evolvability is a challenge for system architects. Support can be provided in this process by using both, system-wide models on different levels of abstraction, and computer based tools which allow the system architects to get a bird's eye view of the system obtaining advantages exposed in section III.*A* [15]. To increase design traceability we need models that connect high levels of abstraction to more detailed levels. Most models used now do not span different levels of abstractions [23]. For example, a mechanical 3D CAD concerns only shapes and assembly of components, and does not link to functional information; and the requirement specification sheets are models made for different levels of abstraction (i.e. for the product at different stages of development) that do not really combine into one traceable knowledge carrier covering the complete product development.

With respect to the objectives of providing a bird's eye view and improving traceability, the functional representation in FBS serves as a base to achieve the first one, while the underlying Behavior-State structure is used to verify the relations at the subsystems and component levels to reach the second one [15].

### C. *Identification of interferences*

Mechatronics machines are complex systems in which many engineering domains are integrated. To deal with complexity, the main function of the machine is decomposed into sub-functions that should be mapped to sub-modules and consequently to sub-projects. This decomposition task is essential to resolve the system complexity, but it can lead to a lack of system overview. Each designer is involved only in his own discipline and module and, as a consequence, may not oversee problems that come when the whole system is integrated. Phenomena or parameters that seemed irrelevant in one domain unexpectedly interfere when more domains are integrated.

The Design Interference Detector (DID) [17] aims at developing an automated analysis tool that detects unpredicted problems that will appear in the combination stage of D&C. This is done by reasoning about the behavior of the system based on qualitative physics (qualitative process theory) [24]. The DID is composed of a conceptual design model (based on FBS) described by the engineer, a qualitative reasoning system (QRS) to detect all possible behaviors, and a filter to distinguish unexpected problems. To reason out unexpected physical phenomena DID uses pattern matching techniques between the machine-model and the physical features which are stored in the system. In this way, the design team can envision destructive problems well in advance, without having to wait after a prototype-machine has been developed.

### D. *Adaptable product architecture*

Currently, there is no methodology that deals with optimization of a complex product architecture, which is influenced by the sheer size of the product, its multi-disciplinarity, and the conditions of the environment. The IMS group works on the development of a method that helps to design an adaptable product architecture. From the point of view of the product architecture, adaptability can be seen as questions of where (in which domain and at which level, e.g., mechanical), how (which strategy), and against what (e.g., aging, deterioration) the reorganization and reconfiguration of the mechanical structure, states, and functional behaviors takes place. In an adaptable architecture, there are always trade-offs that have to be made. Finding an optimal architecture of a product working under different conditions has to be investigated at various levels, subsystems, and components.

Our approach to obtain adaptable architectures is based on the FBS modeling, optimization studies, and reasoning about product behaviors by means of QRS. This will aim to provide guidelines to deal with statement (b) in section III.*A* from the point of view of adaptability.

### E. *Integration and automation*

Due to product complexity, the traditional, sequential methods for development of mechatronic products and their controllers carry a series of difficulties. Examples of these difficulties are: arise of unexpected forms of interaction besides the intended subsystem interfaces; malfunction of the "integrated" system due to unintended interactions; incomplete optimization of global performance of the system and restriction to solution alternatives originating form inappropriate decomposition of separate designs for each domain; and tardy design of the controller that transfers the unexpected problems to late phases of the design when changes are more expensive.

The IMS group introduces a proposal of a framework of prototype tools that aim to improve controller design for mechatronic systems, providing control software generation and allowing simulation of particular and, possibly, incomplete solutions (i.e., the outcome of the combination stage of D&C). Next, we emphasize the high level system description embedded in such framework and the model integration aspects [16].

The use of a high level model of the system based on a functional description (i.e., FBS) serves to represent the top-level conceptual hierarchy. This aims to act as a skeletal structure that links models of lower levels of abstraction, using functions as "wrappers" to the models. The basic hypothesis is that from the functional point of view it is possible to describe a system in as much detail as needed, focusing on the features of interest while maintaining coherence of the model as a whole. Another point to be explored is the function's potential to convey the design intention of systems, which could be used to guide or filter the results of a qualitative reasoning process towards the results of interest.

## IV. CONCLUSION

One of the competences of modern engineers is the capability to integrate different types of knowledge and apply it in practical (often complex) applications. This stands in a sharp contrast with the image of classical (traditional) engineers that are specialists in a narrow domain. Nevertheless, those two types of engineers are equally important, since the increasing complexity of modern products together with the increasing market demands request contemporary engineers and designers to be equipped with both "classical" and "modern" ways of thinking and solving problems.

To reduce and manage complexity, the D&C strategy is often used in engineering to divide problems into sub-problems that are manageable. D&C, however, cannot be used easily to deal with the design of complex systems. Therefore, several approaches have been developed to help engineers and designers to deal with the design of complex systems. Those approaches are oriented towards product architecting, organizational and management approaches, and computer-based tools. Although all those approaches are closely related, they deal with different aspects of development of complex systems and none of them alone can deal with all those aspects. Moreover, seamless exchange of data between different designing tools seems to be almost impossible to this date.

The IMS group works on the systems integration technology, which consists of a set of methods used for developing large-scale, complex, and multidisciplinary products, addressing both technical and managerial issues. Systems integration technology has four main elements:

--Systems architecture with established requirements and specifications: Systems architecture includes overall organization of the system in the functional, behavioral, and physical layers. This leads to a hierarchical systems decomposition which defines subsystems, their boundaries, and interactions among them. As the system decomposition goes into detail, it is possible to clarify the processes and technologies needed to build the system and its components.

--Project coordination: It begins with coordination of mono-disciplinary design and engineering processes. It also addresses allocation and management of resources such as personnel, budget, technology, and knowledge. Project coordination facilitates communications and negotiations to coordinate cross-disciplinary activities, to resolve conflicts and contradictions, and to control engineering processes and human factors.

--System integration in a narrow sense: The focus here is on the integration of individual elementary design results into a whole. Usually at this stage it is necessary to resolve conflicts and contradictions present between these individual results.

--Knowledge integration: Although knowledge integration is the core of systems integration technology, a significant part of it is still being investigated. Currently, knowledge integration can be learnt only through practice. There are, however, some technical clues for knowledge integration, i.e., knowledge structuring and abduction [25], [26]. Knowledge structuring identifies relationships between different domains, whereas abduction is considered to be an algorithm for integrating knowledge domains.

## References


[1] 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.

[2] T. Tomiyama, V. D'Amelio, J, Urbanic, and W. ElMaraghy, "Complexity of multidisciplinary design", *CIRP Annals*, Vol. 56 (1), 2007, pp.89-92.

[3] T. R. Browning, "Applying the design structure matrix to system decomposition and integration problems: a review and new directions," *IEEE Transactions on Engineering Management*, Vol. 48, 2001, pp. 292-306.

[4] INCOSE SHE Working Group, *Systems Engineering Handbook*, 2nd edition, 2000.

[5] 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.

[6] N. P. Suh, *The Principles of Design,* Oxford University Press, Oxford, 1990

[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] G. Sohlenius, "Concurrent Engineering," *CIRP Annals*, 41 (2) 1992, pp. 645-655

[9] V. Botta-Genoulaza, P. A. Milleta, and B. Grabot, "A survey on the recent research literature on ERP systems," in *Computers in Industry*, 56 (6), 2005, pp. 510-522

[10] D. T. Liu and X. W. Xu, "A review of web-based product data management systems," *Computers in Industry*, 44 (3), 2001, pp. 251-262

[11] T. Bilgic and D. Rock, "Product data management systems: state-of-the-art and the future," *Proceedings of DETC'97*, ASME September 14-17, 1997, Sacramento, California.

[12] G. Thimm, "Towards unified modelling of product life-cycles," *Computers in Industry,* 57 (4), 2006, pp. 331-341

[13] D. Martyn, "What is PLM?," *CAD digest* [Webpage]. Available at: http://www.caddigest.com/subjects/PLM/select/day_plm.htm

[14] M. S. Erden, H. Komoto, T. J. van Beek, V. D'amelio, E. Echavarria, and T. Tomiyama, "A review of function modeling: Approaches and applications,' in *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 22 (2), 2008, pp. 147-169.

[15] T. J. van Beek and T. Tomiyama, "Connecting views in mechatronic systems design, a function modeling approach," *IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications*, 2008, submitted for publication.

[16] A. A. Alvarez Cabrera, M. S. Erden, M. J. Foeken, and T. Tomiyama, "High level model integration for design of mechatronic systems," *IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications*, 2008, submitted for publication.

[17] V. D'Amelio and T. Tomiyama, "Design at the Cross-sections of Domains," *CIRP Design Conference on Design Synthesis 2008*

[18] M. K. Chmarra, L. Arts, T. Tomiyama, "Towards adaptable architecture", *ASME International Design Conference, Design Automation Conference*, DETC2008-49971 (accepted for publication).

[19] F. Josephus, "De Bello Judaico (The Wars of the Jews)," Book I (http://en.wikipedia.org/wiki/Divide_and_rule).

[20] D. R. Smith, "The design of divide and conquer algorithms," in *Science of Computer Programming*, Vol. 5, 1985, pp.37-58

[21] J. Rasmussen and M. Lind, "Coping with complexity," [revised reprint.] presented at the *European Conference on Human Decision and Manual Control, 1981*, Delft, The Netherlands. ISBN 87-550-0769-4 ISSN 0418-6435.

[22] Y. Umeda, M. Ishii, M. Yoshioka, Y. Shimomura, and T. Tomiyama: "Supporting Conceptual Design based on the Function-Behavior-State Modeler," *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 10(4), 1996, pp. 275-288.

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

[24] A. Barr, and P. R. Cohen, "The Handbook of Artificial Intelligence," Vol. 4. Chapter 21. Los Altos, CA: William Kaufmann, Inc. 1989.

[25] T. Tomiyama, "Knowledge Development: How to Use Design Knowledge", in U. Lindemann (ed.) *Human Behavior in Design – Individuals, Teams, Tools*, Springer, 2003, pp.261-271.

[26] T. Tomiyama, H. Takeda, M. Yoshioka, and Y. Shimomura, "Abduction for Creative Design," *Proceedings of the 2003 ASME Design Engineering Technical Conferences & Computers and Informaion in Engineering Conference*, September, 2003, DETC2003/DTM-48650.