INTERNATIONAL DESIGN CONFERENCE - DESIGN 2008
Dubrovnik - Croatia, May 19 - 22, 2008.

DESIGN 2008

TOWARDS REPRESENTING, EVOLVING AND NETWORKING ENGINEERING KNOWLEDGE FOR COMPUTATIONAL DESIGN SYNTHESIS

F. Hoisl, K. Shea and B. Helms

Keywords: computational synthesis, formal knowledge representation, engineering design grammars, computationally-based innovation

1. Introduction

Innovation is a knowledge-intensive process where new knowledge is created and evolved throughout. A central part of the innovation process is concept generation that involves the synthesis of new product concepts. Several methods for formal, computational design synthesis have been successfully developed during the last decades [Antonsson & Cagan 2001, Chakrabarti 2002]. However, most approaches are based on static knowledge that is captured up-front while the method and tool is under development and are often limited to a narrow, engineering viewpoint of a synthesis task. To enable new computational synthesis methods that support the innovation process, this is insufficient due to the temporal nature and connectivity of distributed knowledge throughout different domains (engineering, marketing, production, etc.), company divisions and within today's virtual enterprises. New and evolving knowledge must be related and integrated efficiently within synthesis methods, leading to new requirements for highly dynamic knowledge creation and evolution as well as integration of interrelated knowledge. The aim of this paper is to motivate and describe new requirements for representing engineering knowledge in formal synthesis methods to capture more fully both product and process knowledge as well as facilitate their integration into a new framework for networking, reusing and evolving distributed knowledge. First, the paper reviews current research in formal approaches to supporting engineering knowledge needs in synthesis and innovation as well as the relation to the wider areas of knowledge based engineering (KBE) and knowledge management (KM). Next, a model for computational design synthesis based on engineering design grammars is presented including a discussion of the types of knowledge involved. Extensions to this model are then proposed to integrate important aspects of the knowledge lifecycle, namely networking, exchanging and evolving formalized multi-domain knowledge for computational synthesis. The paper concludes with a discussion of future perspectives.

2. Background

This section reviews research in the areas of knowledge-based engineering (KBE) and engineering design grammars, which both contain the integrating topic of engineering knowledge representation. Further it discusses key ideas stemming from KM to support engineering design and product development that have potential to be extended to computational synthesis.

2.1 Formal engineering knowledge representation and use

Much research has been done in knowledge representation in engineering; a summary of approaches is given in Szykman et al. [2001] and Fenves et al. [2005]. Such knowledge representations generally

rely on the fact that human knowledge, and in particular expert domain knowledge is modular, for example consisting of perceptual chunks or production rules. While cognitive scientists have shown justification for representing knowledge modularly, it is also recognized that "knowledge chunks" are highly interdependent when used in problem solving [Newell 1990]. To support product development, Szykman et al. [2001] propose a generic product knowledge representation with a focus on incorporating function models, design rationale and relationships between knowledge. The same research group later developed the Core Product Model (CPM), at the information model level, that is capable of capturing all product information within the product lifecycle to provide a base product model that is not specific to any application, software or product development process [Fenves et al. 2005]. Further, the MOKA project (Methodology and tools Oriented to Knowledge based Applications), that preceded CPM but shares conceptual ideas, aimed at developing a general framework for structuring and representing engineering knowledge [Klein 2000]. A two pronged representation was developed: an informal model that used a structured, natural language representation of engineering knowledge and a formal model using the Unified Modeling Language (UML). In the informal knowledge representation, which is purely text based in Illustration, Constraints, Activities, Rules, and Entities (ICARE) forms, relationships between entities are modeled, e.g. indicating relations between an assembly and its parts. The informal knowledge must then be transformed into formal knowledge through the assistance of a knowledge engineer.

2.2 Knowledge based engineering (KBE)

The term KBE has been used in two different ways in both academic research and industry. The original use stems from expert systems where symbolic systems were created to encode domain and problem-solving knowledge that could automatically be reasoned about using an inference engine to solve synthesis, diagnosis, analysis and planning tasks [Dym & Levitt 1991]. The more recent use of the term is less restrictive and can refer to any process-related or reference knowledge used within CAx tools [Fenves et al. 2005]. From an industry perspective current CAx tools claiming KBE capabilities use this more relaxed definition including, for example, automating parametric modeling and modification of geometry. Knowledge exists in limited forms as contained in product models, templates, parametric models, scripts, constraints and to a less commonly in design rules and case libraries. However, this level of knowledge representation is not standardized, is tool dependent and generally not suitable for exchange or common processing and therefore limited for distributed use.

Current increased use of KBE and parametric modeling in CAD tools gives initial indication that it is possible for engineering knowledge to be formalized by engineers while they work in a familiar software environment, such as a CAD tool. Examples illustrate the benefits of automating routine design tasks, e.g. the BAe Airbus wing rib design KBE tool, which generates ribs automatically in any location on an airplane wing [Cooper et al. 2001]. The capture of knowledge, however, is limited to knowledge that can be associated with geometry and the problem-solving knowledge is related to a routine design task of a single part or component, often in the detail design stages and only from the perspective of one domain. These limitations can also be found for mainstream commercial software that is starting to provide capabilities for implementing design rules and rule bases, e.g. CATIA Knowledgeware or NX Knowledge Fusion.

In routine design tasks typically both the knowledge sources and problem solving strategies are known. While supporting routine design, problem-solving is based on retrieving and directly using well-organized knowledge. However, in order to support non-routine design, knowledge sources or problem solving strategies may not be known and one must be able to combine knowledge in new ways [Brown 1996].

2.3 Computational synthesis and engineering design grammars

Synthesis can be thought of as creating form, or product structure, to fulfill desired behavior and function [Starling & Shea 2005]. It is the creative step itself, the conception and postulation of possibly new solutions to solve a problem [Antonsson & Cagan 2001]. Previous work in formal methods for computationally-based design synthesis was aimed at aiding designers in developing better products faster through rapid generation of spaces of optimized, simulation-driven designs.

Generation can be both fully automated and semi-automated. New design alternatives produced as a result of the methods are targeted at both providing novel solutions, outside a designer's own experience and sparking innovation. Methods often support lateral thinking and expanding the range of perceived design and performance limits. Examples provided in Antonsson & Cagan [2001] and Chakrabarti [2002] use different forms of formal knowledge representation and methods covering applications to structures, consumer products, automotive styling, microelectromechanical systems (MEMS), digital VLSI (very large-scale integration), and chemical processes. Examples of the authors' work are mechanical gear systems [Starling & Shea 2005], structural systems [Antonsson & Cagan 2001] and MEMS [Bolognini et al. 2006], all of which are based on the combination of design grammars, automated simulation and multi-criteria search.

Engineering design grammars are a production, or rule-based, system and overlap with KBE. A uniform characterization of production systems, including shape and graph grammars, can be found in Gips & Stiny [1980]. The examples that can be found in this area, however, are generally restricted to the perspective of a single domain. In addition, the systems are often time intensive to implement and many engineering design grammars are pre-compiled before use, which means that the embedded knowledge remains static. The designer can only influence the design generation through changing the initial design and the search model used to find good designs within the language. Two approaches to overcome this and enable knowledge evolution are: (1) supporting emergence and learning of emergent rules and (2) grammar interpreters, which enable designers to describe new rules that can be compiled with the current rule-base. One approach to creating and compiling engineering graph grammars, called "Design Compiler 43", focuses on providing a domain independent representation for conceptual design [Alber & Rudolph 2003]. In a graph-based representation approach, Campbell et al. [2007] provide in GraphSynth a GUI-based graph grammar rule specification and common engine for recognizing and applying rules to designs. In Bolognini et al. [2006] a modular system is developed to enable straight-forward customization of the primitives, or vocabulary, used within synthesis. However the general graph transformation rules used for synthesis remain static. Finally, little work has been done in supporting recognition and learning of emergent grammar rules.

2.4 Knowledge Management

The overall goal of knowledge management is to enable an organization to best exploit its intellectual capital, both past and present [McMahon et al. 2004]. Approaches to KM include organization, management and technology views. The assumptions of many of the original approaches in KM research, called first-generation KM, were that valuable knowledge existed that must be codified so that it can be accessed in the future and re-used [McElroy 2002]. This has been termed the commodity view of knowledge that considers knowledge as an object to be collected and managed. Since it is commonly believed that knowledge is defined in working practice through activities and collaborations among workers, a new generation of KM has emerged that takes a more holistic perspective to include people and processes and focus on new knowledge production and integration [McElroy 2002]. This is termed the community view of knowledge that, by contrast, assumes it is impossible to define knowledge universally, or think of everything up-front, to compile a massive knowledge system that can then be deployed in a company. Rather, knowledge personalization is imperative. Research in KM in computer science shares the community view of KM where technology support should assist workers to constantly create and share new knowledge as they work [Fischer & Ostwald 2001].

Knowledge management in product development, specifically engineering design, creates many added complexities, e.g. the integration of diverse disciplines, high level of technical content in addition to business and organizational content and the globally distributed nature of product development today. A review of research in this area is provided by McMahon et al. [2004]. Culley & McMahon [2006] note that creating and sharing knowledge is essential to promoting innovation in engineering design and identify conceptual design as the key, yet most difficult area where embedding knowledge in design systems for design automation could have the most benefit.

2.5 Summary

To summarize, knowledge representation in engineering design links formal, computational synthesis methods, KM and KBE research. Review of literature illustrates that capturing knowledge across all activities in product development leads to a very diverse set of knowledge types and taxonomies covering both product and process foci. CAD tools now incorporate parametric modeling and limited capabilities for KBE where designers can encode their own design rules to automate small-scale routine design tasks. The limitations are, however, that they are difficult to update, extend to representations not based on geometric models alone, network knowledge among heterogeneous CAx tools and evolve formalized knowledge across domains. Engineering grammars supporting synthesis tasks in innovation processes are mostly limited to pre-compiled sets of grammar rules or vocabularies of components as well as limited to a single knowledge domain, e.g. engineering, styling or production. The commodity view of knowledge where past knowledge is captured, stored and managed is insufficient for supporting synthesis tasks within innovation processes. While it can be argued that many methods incorporate only fundamental knowledge, with rapid advances in new technologies even fundamental knowledge grows. Recent changes in KM towards the community view of knowledge should be considered now within the context of computational synthesis methods and tools. KM research in computer science indicates the advantages of formalizing new knowledge while working, where formal knowledge is then owned and evolved by communities. Such research also shows, although in less technical domains, that it is possible to support both personalization and codification of knowledge simultaneously [Fischer & Ostwald 2001].

3. Computational design synthesis and formal knowledge representations

Drawing from several sources, synthesis is defined here as the combination of fundamental components, or building blocks, to produce a unified and often complex system that efficiently exhibits at least the required behavior, in a conventional or novel way, which is especially important for innovation. Figure 1 illustrates a model for formal engineering design synthesis extended from previous research by the authors that was based on a bottom-up viewpoint of design synthesis. The model combines the use of engineering design grammars, a function-behavior-structure (FBS) representation, integrated simulation, design performance models and multi-criteria search algorithms.

Engineering design grammars, the core of the model, combine product knowledge representations with problem-solving knowledge represented as grammar rules. The use of grammars to assist design is conceptually simple. In the same way as a natural language is based on symbolic representations (alphabet and words) and rules (grammar), it is also possible to develop a language of designs via design (or engineering design) grammars. Starting with a legal construct, repeated application of different grammar rules generates new designs. The combinatorial expansion of all valid sequences of grammar rules applied to a starting symbol is termed the design language.

Using the model shown in Figure 1, first, requirements (Q) for the synthesis task are defined and mapped into a quantitative performance model (P) that can include design objectives and constraints, similar to a multi-criteria optimization model. Starting with an initial design, either an empty model or a current solution, control of synthesis is driven by an inference engine, as used in KBE systems, by human designers, or by search algorithms. While an inference engine provides deterministic problem solving and single solutions, search provides the potential for generating a set of optimized solutions. Engineering design grammars are used to define transformation rules (R) and vocabularies (V), that form the basis of a knowledge library, to generate designs described through both function models (F) and structure models (S) in parallel, which are interconnected by the behavior (B) of the design. The FBS model [Umeda et al. 1990] provides a knowledge representation scheme for flexible functional modeling separated from, but interconnected with, structure and behavior representations. The behavior model is simulated and the resulting performance models (P) are evaluated. In combination with further performance models, in addition to simulation-based evaluation, quantitative performance of a current solution is determined. Using multi-criteria search combined with engineering grammars enables the generation of optimal designs within the design language that meet defined requirements

and optimize performance objectives, which can describe behavior, efficiency, cost, aesthetics and manufacturability [Starling & Shea 2005].

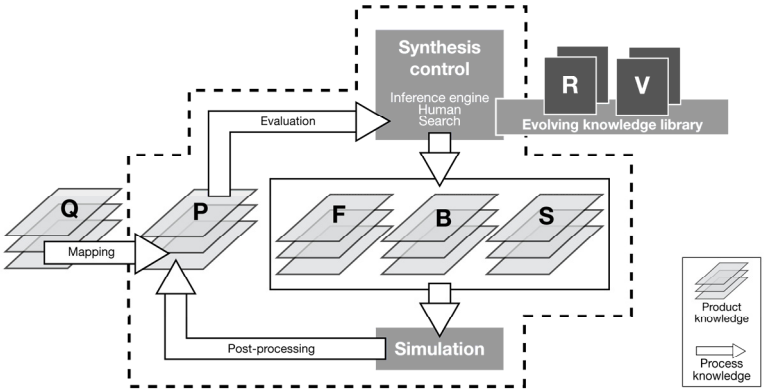

Figure 1. Model for formal engineering design synthesis

To create new formal engineering knowledge representations for design synthesis with a focus on supporting dynamic knowledge and relationships among knowledge, extended requirements are now defined. They include supporting multiple levels of abstraction for use in a wider range of tasks, explicitly representing relationships, among domains and knowledge sources, supporting multi-domain knowledge, supporting rapid modification and evolution of knowledge, and enabling knowledge to be networked, exchanged, re-used and evolved. The representation should be easily created, updated and evolved by engineers while they work, applicable to a variety of design tasks and capable of incorporating dynamic influences from related domains of production and market requirements. Finally, formal product knowledge representations should be compatible for integration with CAx tools where possible, i.e. in terms of current capabilities for geometric modeling and product structures (assemblies and subassemblies). The outcome will lead to standardized, flexible, formal knowledge representations for product and process knowledge, with a focus on design grammar rules and their relationships to other knowledge chunks, such that they can be, in the long term, processed in a common way by distributed CAx tools.

The formal knowledge needed for the synthesis model shown can be divided into product and process knowledge. Product knowledge includes function (F), structure (S) and behavior (B) models of an artifact along with product specifications, requirements, and constraints, which are mainly represented within Q, P, and V. Process knowledge includes design rules (R), strategies, automated model mappings, analysis scripts, problem-solving algorithms, e.g. constraint solvers, generative algorithms and search methods. Within the scope of the synthesis model presented and engineering grammar formalisms, product knowledge is focused around definition of the vocabulary of the design language while process knowledge facilitates computational processing of product knowledge through both design rules (R) and the many model mappings and problem-solving algorithms included. Product knowledge in Figure 1 is depicted by several layers to indicate the inclusion of several levels of abstraction. Constraints, a central and often driving component of computational synthesis methods, are modeled in both the design vocabulary (V) and design rules (R). The elements inside the boundary shown in Figure 1 are internal to the computational synthesis method and are "driven" by the external knowledge contained in Q, R and V, which are updated dynamically. For example, adding knowledge of a new component to the vocabulary requires definition of how the component should model itself in terms of FBS models as well as definition or extension of the grammar rule set to make use of the new component in synthesis. Finally, relationships between all knowledge chunks, product knowledge to product knowledge, product knowledge to process knowledge, and process knowledge to process knowledge must be supported in the formal representation. Such relationships can be determined and specified manually or automatically detected using agent-based and data mining approaches.

4. Networking and evolving formal engineering knowledge

Throughout the entire innovation process, the knowledge lifecycle is subject to several dynamic influences through which knowledge is evolved over time (Figure 2). Knowledge is first captured and formalized, then networked for re-use by multiple people and, over time evolved. Knowledge evolution is influenced by exchange of formalized knowledge with other domains, e.g. production, service, and marketing, so that inter-relations are taken into consideration during the synthesis process. To network knowledge, a new high level IT framework is required that provides capabilities for networking, exchanging and co-evolution of formalized, distributed engineering knowledge throughout the innovation process.

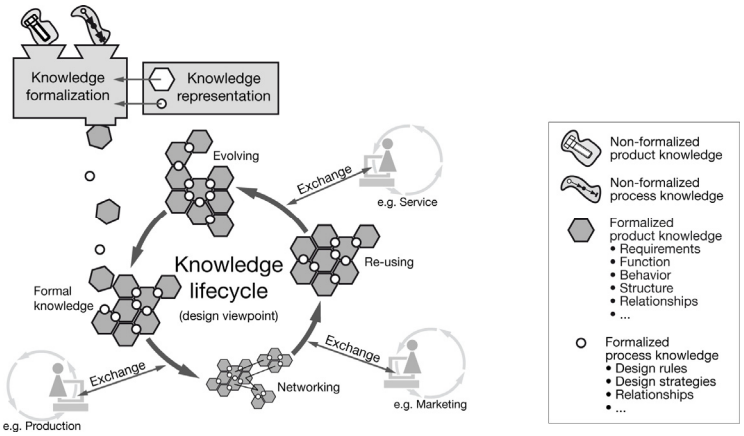

Figure 2. Networking, exchanging and evolving formalized knowledge throughout the knowledge lifecycle

For example, engineering grammars, are generally limited to encoding design rules for small subsystems or a single domain, e.g. gearboxes or automotive body styling, but not the vehicle as a whole. Engineering grammars can also be created to generate custom gearbox housing, car bodies, chassis, etc. Small grammars for each subsystem that encode the product and process knowledge of that subsystem can then be networked to rapidly generate larger scope conceptual designs. Grammars can also be developed across domains, e.g. a grammar that encodes the manufacturing viewpoint and multi-domain grammars added to describe not only all valid but all feasible designs from a manufacturing viewpoint. Related, distributed knowledge must be exchanged continuously with other domains throughout the lifecycle, e.g. a grammar encoding manufacturing capabilities of the vehicle body is related to the design rules governing automotive styling and space allocation tasks of components under the car hood. Introduction of new knowledge concerning engine types, e.g. inclusion of hybrid engines impacts the knowledge required for gearbox configuration. A networked, multi-domain approach to knowledge representation in engineering grammars offers potential to increase the scale of computational design synthesis and incorporate multiple viewpoints.

As a result of the knowledge lifecycle, the computationally defined design language, or design space, does not remain static any longer but evolves over time according to dynamic influences, e.g. new market requirements and production capabilities (Figure 3). The design language grows over time continuously integrating new knowledge that often results in a larger space of possible solutions. Further, given a set of new requirements, the design language can be used to determine if no solutions exist that meet the new requirements identifying the need to either increase capabilities and modify the modeled knowledge or refine the requirements.

A further example is that of mechanical wristwatches. A mechanical wristwatch consists of the gear system, power source, watch hands, case, and watch strap. In the authors' previous work [Starling & Shea 2005], the vocabulary and rules to generate basic watch gear systems were defined and

implemented upfront. However, new knowledge is constantly being generated, e.g. new components, new production technologies, new materials and changing market requirements. The synthesis system and therefore the underlying knowledge representation must dynamically evolve, e.g. include new components and rules for generating new watch functionalities to meet new requirements, e.g. a date display functionality, a chronograph functionality or use of alternative power sources, e.g. photovoltaic. This requires evolving the product knowledge, i.e. vocabulary of new components and the process knowledge, i.e. how to use new components in design synthesis, to expand the design language. Knowledge of new production knowledge can also enhance the design language for example, by changing production constraints on components.
A further benefit of formalizing and networking knowledge is to avoid informal knowledge exchange face-to-face, by phone or by email with a subsequent recreation of the knowledge formalization in the target software system. While this can be achieved initially through experts themselves formalizing knowledge, in the long term in order to achieve true benefits, automatic capture, formalization and networking of explicit engineering knowledge created by engineers while they work is required. Having engineering knowledge formalized also is a pre-cursor to digital knowledge exchange and manual re-use of up-to-date knowledge among heterogeneous CAx tools in new tasks and domains.

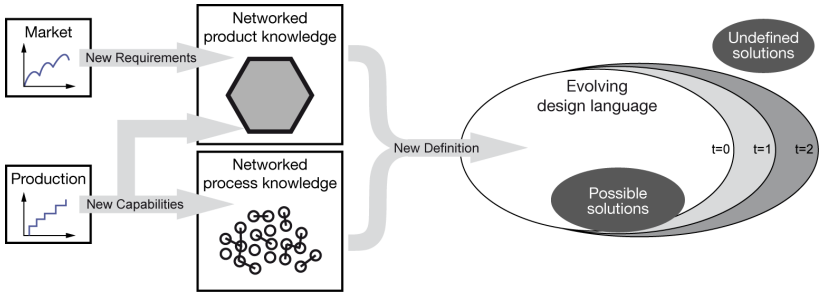

Figure 3. Evolving design language

5. Conclusion

Innovation processes are knowledge-intensive and dynamic due to their continuous creation and evolution of new knowledge. In order to move computational synthesis methods to truly support innovation processes requires consideration of the dynamic aspects of engineering knowledge that is generated, inter-related and used during the process. Appropriate formal engineering knowledge representations and a framework for integrating and re-using distributed knowledge in innovation processes are required to accelerate integration of new and evolving knowledge as well as the rapid derivation of computationally defined design languages.
A networked, multi-domain framework for design synthesis is speculated to offer potential to increase the scale of knowledge captured in the definition of grammar vocabularies, identifying relationships between vocabularies and rules as well as the ease of modifying and evolving engineering grammars throughout the innovation process. Further, it is aimed to support communication and cross-fertilization of knowledge across domains and departments as well as enable co-evolution of rule sets across domains. This would provide engineers with the power of collective knowledge of the whole rather than purely the knowledge they have formalized and encoded themselves. The approach would unlock the potential of many engineers formalizing their knowledge while working, networking and re-using knowledge as well as evolving formal engineering knowledge as a collective body of engineers in the extended enterprise.

Acknowledgement

This research is supported by the Deutsche Forschungsgemeinschaft through the SFB 768 “Zyklenmanagement von Innovationsprozessen”.

References

Alber, R., Rudolph, S., "'43' – A Generic Approach for Engineering Design Grammars", Papers from the 2003 AAAI Symposium, Stanford, USA, Technical Report SS-03-02, AAAI Press, pp. 11-17.

Antonsson, E.K., Cagan, J. (Eds.), "Formal Engineering Design Synthesis", Cambridge University Press, 2001.

Bolognini, F., Shea, K., Vale, C. W., Seshia, A. A., "A Multicriteria System-Based Method for Simulation-Driven Design Synthesis", Proc. ASME IDETC/CIE, Design Automation Conference, DETC2006-99354, 2006.

Brown, D.C., "Routineness Revisited", Mechanical Design: Theory and Methodology, Waldron, M., Waldron, K. (Eds.), New York: Springer 1996, pp. 195-208.

Campbell, M.I., Nair, S., Patel, J., "A Unified Approach to Solving Graph Based Design Problems", Proceedings of the ASME IDETC/CIE Conferences, DETC2007/DTM34523, 2007.

Chakrabarti, A. (Ed.), "Engineering Design Synthesis", Springer-Verlag London, 2002.

Cooper, S., Fan, I., Li, G., "Achieving Competitive Advantage Through Knowledge Based Engineering – A Best Practice Guide", White Paper for the Department of Trade and Industry, Cranfield, UK, 2006.

Culley, S., McMahon, C., "The Relationship between Engineering Design and In-formation and Knowledge", Proceedings of IDETC/CIE 2006: ASME 2006 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference, DETC2006-99739, Philadelphia, USA, 2006.

Dym, C.L., Levitt, R.E., "Knowledge Based Systems in Engineering", McGraw-Hill Inc, New York, 1991.

Fenves, S.J., Sriram, R.D., Subrahmanian, E., Rachuri, S., "Product Information Exchange: Practices and Standards", ASME Journal of Computing and Information Science in Engineering, Vol. 5, 3, 2005, pp. 238-246.

Fischer, G., Ostwald, J., "Knowledge Management: Problems, Promises, Realities and Challenges", IEEE Intelligent Systems, 16 (2001) 1, pp. 60-72.

Gips, J., Stiny, G. "Production systems and grammars: a uniform characterization," Environment and Planning B: Planning and Design, 7, 1980, pp. 399-408.

Klein, R., "Knowledge Modeling in Design – The MOKA Framework", Artificial Intelligence in Design '00, Gero, J.S. (Ed.), Kluwer, 2000, S. 77-102.

McElroy, M.W., "The New Knowledge Management: Complexity, Learning, and Sustainable Innovation", Butterworth-Heinemann, Oxford, 2002.

McMahon, C., Lowe, A., Culley, S., "Knowledge Management in Engineering Design: personalization and codification", Journal of Engineering Design, 15 (2004) 4, pp. 307-325.

Newell, A., "Unified theories of cognition", MA: Harvard University Press, Cambridge, 1990.

Starling, A.C., Shea, K., "A Parallel Grammar for Simulation-Driven Mechanical Design Synthesis" Proceedings of the Design Automation Conference, DETC05: ASME Design Engineering Technical Conferences, DETC 2005-85414, Long Beach, CA, USA, September, 2005.

Szykman, S., Sriram, R.D., Regli, W.C., "The Role of Knowledge in Next-Generation Product Development Systems", ASME Journal of Computing and Information Science in Engineering, Vol.1, Issue 1, 2001, S. 3-11.

Umeda, Y., Takeda, H., Tomiyama, T., Yoshikawa, H., "Function, behavior, and structure", AIENG '90 Applications of AI in Engineering, pp. 177-193, Computational Mechanics Publications and Springer-Verlag Southernpton and Berlin, 1990.

Hoisl Frank, Dipl.-Ing.
Scientific assistant
Technische Universität München (TUM), Institute of Product Development
Boltzmannstr. 15, D-85748 Garching b. München, Germany
Tel.: +49 (0) 89 289 15137
Fax.: +49 (0) 89 289 15144
Email: frank.hoisl@pe.mw.tum.de
URL: http://www.pe.mw.tum.de