Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 21 (2014) 99 – 104

www.elsevier.com/locate/procedia

24th CIRP Design Conference

Exchange Of Knowledge In Customized Product Development Processes

Patrick Klein[a]*, Dante Pugliese[b], Johannes Lützenberger[a], Giorgio Colombo[b], Klaus-Dieter Thoben[a]

[a]*BIBA – Bremer Institut für Produktion und Logistik GmbH, Hochschulring 20, 28359 Bremen, Germany*
[b]*Politecnico di Milano, Department of Mechanical Engineering, Via G. La Masa 1, 20156 Milano, Italy*

* Corresponding author. Tel.:+49-421-218-50115; fax: +49-421-218-50007; *E-mail address:* klp@biba.uni-bremen.de

Abstract

If Customized Product Development is perceived as developing products that fulfill the customers individual requirements and in parallel reflect production constraints, such as manufacturing capabilities, a direct demand can be derived for solutions to adapt a given design easy and fast to new requirements based upon the companies production knowledge - at best in an automated way. The latter is usually covered by Knowledge Based Engineering systems. KBE systems are capable to automate repetitive engineering tasks, such as the automated calculation of ship structural design.
However, while the efficiency of implemented KBE projects is non controversial, the development or modification of an existing KBE solution usually requires substantial investments due to knowledge acquisition, codification and software implementation. In addition most solutions are still case based and not grounded in structural frameworks. Knowledge is often written in a proprietary language; rules and algorithms are not compatible with other KBE-frameworks and are usually not on a level that is comprehensible for the engineers or domain experts. While this may not be crucial for long development cycles, it may become a hurdle in terms of Customized Product Development with its short cycles. In other words, future KBE must support an incorporation of knowledge from different domains and business units. Thus the objective of the paper is to explain the need for a change in collaborative knowledge sharing and re-use in context of KBE. Based upon, the constraints for a KBE related interchange format are drafted.
A three layered approach is proposed in order to adequately represent and exchange KBE knowledge. Each layer addresses different levels of abstraction: an upper layer describing just the core knowledge at a glance, a middle layer in order to codify the knowledge on abstract level, but with purpose of software development and a base layer covering the software code itself.
Utilizing an independent format for management of KBE knowledge, the users of CAx systems are able to exchange codified knowledge and gain the rationale behind. Hence the full paper attempts to deliver a substantial contribution for the development of systems, which are capable to easily adapt a given design to upcoming user-requirements, while facing the production challenges.

Keywords: Knowledge Based Engineering; Product Development; Knowledge Mangement; Knowledge Engineer; Customized Product Development

1. Introduction

In order to gain a competitive advantage companies intend to provide mass products, which are at most customized to user needs. This directly impacts the company’s product development; so called Customized Product Development aims at developing products that fulfil the customers individual requirements and in parallel reflect production constraints, such as manufacturing capabilities [1]. The large solution space, which is implied in customisation, is challenging companies as well as customers. It is often difficult for production companies to clearly communicate their capabilities in sufficient details without confusing customers and in parallel it is often difficult for customers to clearly articulate user-requirements without assessing a given design [2].

Facing this challenge a need for a solutions to adapt a given design easy and fast to acquired requirements based upon the companies production knowledge - at best in an automated way can be identified. The latter is usually one of

doi:10.1016/j.procir.2014.03.149

the technical domains covered by Knowledge Based Engineering systems.

Within a KBE system, design knowledge is represented in a formal manner and enables the system to automate specific design tasks mostly unique to the company's product development experience. This way KBE systems are capable to automate repetitive engineering tasks, such as the automated calculation of ship structural design [3].

In this context the meaning of "automate" even covers analysis tasks in terms of validation or quality checking, such as compliance to required safety parameters, or international and national standards.

In this sense, KBE can be seen as the process of gathering, managing, and using engineering knowledge to automate the design process (e.g. to deliver product variants) by software support [4].

Next to time savings a KBE solution can enable a broader variety of detailed design studies of a given master-concept by usage of a rule-based approach for an automated detailing and examination of design variants and in consequence extensively support the optimization of a given (mechanical) design against defined constraints and requirements (this incl. of course user requirements). KBE as an enabler for easy and fast examination of design variants can be seen as an enabler for Customized Product Development, because it enables differentiation in product variety, i.e. customization [5]. For this reason, the most fitting type of companies, where a KBE implementation enables its most profitable usage, is the Engineering To Order one.

In conjunction with virtual or rapid prototyping those design alternatives will enable a fast and easy assessment of mass producibility and customer acceptance (e.g. [6]) and this way provide controls to adequately integrate customers feedback into product development. As a result of that, a KBE solution is able to anticipate the engineering operations at the time of the order and in parallel it produces more accurate proposals.

2. Scope and Research foundation

By nearly all KBE implementations, a notable time reduction from several days to a few hours for the respective design tasks had been achieved, while in parallel a constant quality due to the repeatability can be ensured (e.g. [7], [8]).

However, while the efficiency of implemented KBE projects is non controversial, the development or modification of an existing KBE solution is often complex and time consuming. A structured development of a KBE solution means usually passing through different phases: starting from knowledge acquisition phase, over a knowledge formalization phase up to a knowledge representation phase [9].

In past, several projects dealt with the above mentioned structured development of KBE, such as MOKA [10], KOMPRESSA [11] or DEKLARE [12].

Even though the inherent approaches had been validated by successfully implemented KBE reference projects, they found no broad acceptance in Industry (section 2.1). Thus findings as well as current shortcomings of those projects had been one starting point for approaching a *Rule Interchange Format (RIF)* to become a standardized knowledge representation to be used by different KBE systems in order to make the reuse and sharing of companies' knowledge easier.

2.1. Identification of a need for a framework-independent management of knowledge

Even though KBE usually requires substantial investments and different sorts of expertise, many product developers seem to improvise a KBE project based upon an unstructured problem analysis [13]. In addition most solutions are still suited to a single case and neither grounded in the workflow organisation nor based on structural frameworks [14].

Knowledge is often written in a proprietary language; rules and algorithms are not compatible with other KBE-frameworks and are usually not on a level that is comprehensible for the engineers or domain experts. More, the products databases are often built for specific solutions and not for general usage.

Leading CAD applications provide add-on modules (e.g. ([15]) for KBE related features. In such modules the KBE intelligence (e.g. a design rule) directly remains inside a CAD-model and is directly stored within the CAD file. Based on a parameterized CAD model, they provide functions like formulas (to create dependencies between parameters), rules and user defined features, allowing to partly reuse a design procedure ([15]). Even if it would be possible to break up this encapsulation, which is given by the proprietary structure of such files, an utilisation of the already implemented design knowledge by other applications would fail, due to a lack of standardization of items, such as Namespaces, Relations or Operators & Rules.

In addition, Verhagen, et al. criticize that many KBE-solutions store and represent codified knowledge decoupled from its original context [16]. An adequate documentation is missing and formulas or equations remain unexplained [17], even though knowledge formalization is fundamental not only in order to provide the documentation to application developers, but also to document in detail all aspects of the automated design process for the user. The cause is often grounded in an unstructured knowledge acquisition process. Without a documentation of the problem in terms of objectives, constraints etc. the traceability of the design process of the implemented solution becomes impossible. Along with the insufficiency of a structured knowledge codification, a lack of knowledge reuse has been identified. Herewith formalisation of knowledge is getting more important, because it promotes further usage and sharing. The acquisition and formalization activities enable a transfer of the ownership of the technical knowledge from the individual experts to the organization. Due to missing knowledge - e.g. neglected alternatives for a desired solution – KBE solutions are too often limited to their origin context and thus the reuse of knowledge will be hindered or impossible [13].

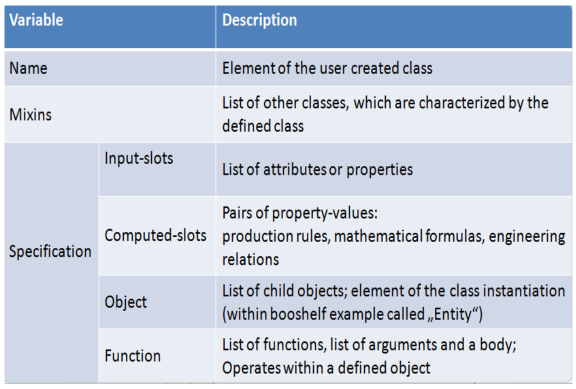
While this may not be crucial for long development cycles, it may become a hurdle in terms of Customized Product Development with its twofold objectives: a maximum of customisation and in parallel real efficiency in production. In other words, future KBE must support an easy incorporation of knowledge from different domains and business units in order to allow product development teams more flexibility. This is of course not possible, if the KBE intelligence directly remains inside encapsulated models or proprietary languages.

2.2. State of practice in context of knowledge codification – Industrial approaches

At present, KBE solutions can be found on many levels and in different industries. From simple templates in CAD software to extensive stand-alone software solutions with integration towards other CAx systems, there are many successful projects in terms of implementing codified knowledge on product design.

In order to identify the current state of practice in more detail a comparative analysis has been conducted based upon a common use case scenario. The scenario describes the development of a KBE solution to automate the design a bookshelf (e.g. no. of shelves as a function of the length). For this scenario a KBE solution has been developed several times by usage of four different frameworks: two KBE enhanced CAD systems (CATIA V5 and NX8), one O-O programming language framework (AML, with an embedded CAD engine) and in addition Rulestream, which can be seen as a mixed approach of object-orientation with a separated design process manager. The result of the developments in all examined frameworks is an automated design. In each solution the geometrical model is calculated as an output, according to a modification of a small set of input parameters (e.g. the overall length of the bookshelf).

Further the four implementations of the bookshelf example discovered a number of similarities like Input-Slots, Computed-Slots etc. between the used frameworks.



Variable		Description
Name		Element of the user created class
Mixins		List of other classes, which are characterized by the defined class
Specification	Input-slots	List of attributes or properties
	Computed-slots	Pairs of property-values: production rules, mathematical formulas, engineering relations
	Object	List of child objects; element of the class instantiation (within booshelf example called „Entity“)
	Function	List of functions, list of arguments and a body; Operates within a defined object

Fig. 1. Communalities in KBE expressions

Differently from a CAD based KBE application, in AML (which relies in its roots on LISP and considered as a pure O-O language) the product hierarchy is represented by defining elements such as classes, superclasses, objects and relationships of inheritance, aggregation and association, by usage of a specific operator [18]. With respect to the two CAD based KBE implementations there is no hierarchy in terms of classes and superclasses at first sight. But a “product tree” or a “product structure”, which are essential parts of contemporary CAD application, provide a substitute to such a rigorous object oriented paradigm. KBE forms, rules etc. always refer to elements of the CAD trees and in this way the underlying hierarchy is used for KBE. Similarly for what concerns the product representation, Rulestream adopts the object oriented approach, even if there is a difference in geometry management. In particular, AML is geometry generative, in the sense that the geometry is created at runtime using pure code and on the contrary, in Rulestream the geometry creation is handled by an external CAD system, driven by input parameters.

By developing disjunctive implementations for one scenario it turned out that there are distinct similarities and to some extend an overlapping in syntax and structure between the elements, which are provided by the different frameworks. Based on this a neutral and formal codification with respect to the identified meta-elements (defined-objects, computed-slots etc.) appears to be technically realizable. However, even if it may become possible to translate code snippets from one framework into another it may not sufficient to use the codified knowledge for different purposes (in other KBE solutions). As described in the following one cause lies in the comprehensibility of the codified knowledge itself.

Another considerable aspect is the representation of the design process. In all approaches, a separate user interface has to be created in order to give access to the global parameters that drive a design configuration. However, it is important to represent the logical and temporary order of the flow of the design. Thus, a representation that helps the designers in managing the company’s best practices appears to be fundamental.

2.3. Identification of constraints and boundaries for a framework-independent management of knowledge

As identified in the comparative analysis (bookshelf scenario) one of the main challenges is grounded in the comprehensibility and granularity of the domain knowledge itself. Domain-specific knowledge, which is used to solve complex design problems, is usually incomprehensible for domain experts, if it is codified in rules. To provide an example: in the bookshelf scenario a rule, that the width of a shelf can be a function of the width of a rack, may appear as part of a cryptic notation (If L_R>L_S then $L_R = l_S*n_S$ …).

Understanding the rationale of the acquired knowledge directly relates to considering the involved stakeholders. Depending on domain and profession of involved experts, skills in programming as well as knowledge modeling may differ. Some developers are only used to describe their procedural knowledge (e.g. *at first we make the overall shape, then we calculate this … , and final we measure parameter X to calculate that …*) others are focused on product structures (e.g. *we are responsible for optimizing this sub-part and have an interface to the rest of the product …*).

In addition the context (e.g. the relation to a specific development phase) is usually not annotated in the code. For example a function to calculate the X of Y may differ between the initial design phase and the detailed design phase. Thus it is crucial to know to which development phase the knowledge snippet relates.

The proposed *Rule Interchange Format* should improve the transparency in terms of understanding the rationale behind KBE solutions. It must be both human as well as computer readable and should be accessible for all stakeholders: domain experts, knowledge experts and software experts.

Especially in context of Customized Product Development the approach must have the potential to support product developers with codified knowledge in different contexts in order to prevent inappropriate design decisions and force a fastened and automated product development.

3. An initial approach to exchange formalized knowledge in Customized Product Development process

In order to address properly the given constraints and boundaries for such a framework-independent management of knowledge, a layered approach is proposed, which enables the exchange of formalized knowledge.

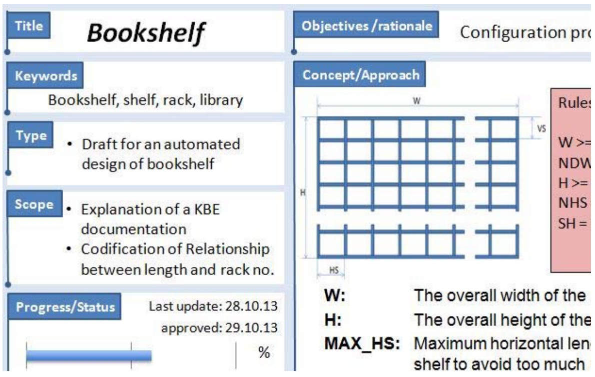
Currently, the concept foresees to have a three layered approach in order to adequately represent and exchange KBE knowledge. Each layer represents different levels of abstraction: an upper layer describing just the core knowledge at a glance, a middle layer in order to codify the knowledge on abstract level, but with purpose of software development and a base layer covering the software code itself. This way the base layer can remain in a proprietary format. Since the codified knowledge is abstracted on a middle layer it can be used for diverse contexts or different purposes by KBE developers. The domain expert's knowledge corresponds to the upper layer. Not only that the upper layer can be used for the knowledge acquisition process, but all layers directly correspond to the typical KBE lifecycle (refer e.g. to [19]). In the following sections the proposed layers are described in more detail.

3.1. K-Brief Layer – Upper Level

The K-Brief Layer represents the upper level of abstraction. It describes development and engineering knowledge in a human understandable manner and corresponds to the domain expert's knowledge. The Knowledge-Brief concept leans on an established concept originated by Toyota. It has been developed and proven in context of LEAN product development. In principle the idea is to provide a structured one page template (an A3 sheet) for capturing knowledge in meetings, discussions or similar events. The template should be used in a way to capture the knowledge in a visual and interactive way, but without loosing context- or meta-information.

In context of our approach the K-Brief templates will be used to represent the KBE related knowledge in a condensed manner (similar to best practices or design guidelines). This way not only the rationale behind existing KBE solutions is documented, but also a guidance to support the knowledge acquisition process is given by the template.

In practice it means initiating a KBE solution leads to the creation of a bunch of K-Briefs in reference to the identified engineering knowledge (To give just one example: a company internal calculation procedure to choose in-between welding seams for steel parts). The K-Brief aims not to codify knowledge in a formal manner. This will be provided by the middle layer.


Title Bookshelf

Objectives /rationale Configuration pr

Keywords
Bookshelf, shelf, rack, library

Type
- Draft for an automated design of bookshelf

Scope
- Explanation of a KBE documentation
- Codification of Relationship between length and rack no.

Progress/Status
Last update: 28.10.13
approved: 29.10.13
%

Concept/Approach

Rules
W >=
NDW
H >=
NHS
SH =

W:	The overall width of the
H:	The overall height of the
MAX_HS:	Maximum horizontal len shelf to avoid too much

Fig. 2. K-Brief for development of a bookshelf (extract)

3.2. K-Models Layer – Middle Level

The K-Models layer despites between declarative and procedural knowledge to define the presented knowledge more precisely. In order to formalize the acquired knowledge for KBE purposes the approach suggests to rely on two different standards for each type of knowledge: UML and IDEF0. The standardized Unified Modeling Language (UML) allows acquiring and visualizing concepts and their relations utilizing an object-oriented approach and is well established in research and industry [20].

While declarative knowledge can be represented in UML the standard does not provide sufficient support to represent procedural knowledge. Since the procedural knowledge refers to the product development process steps and not to the workflow to-be covered by the KBE software it is not convenient to make use of one or more behavioural diagrams as provided by UML. Here IDEF0 appears to provide appropriate complexity to capture and formalize the tackled procedural knowledge. The main advantages from adopting separate procedural knowledge formalization are the following: 1. A procedural knowledge formalization allows a focus on component relations, which differs from UML. For example, in an industrial mixer configuration procedure, the blade is the first object to be designed and the most important one, but in a hierarchical representation of the product, it is placed at a very low level [21]. A procedural representation such as IDEF gives the correct importance to the blade, rather than a product tree representation. 2. Since companies' stakeholders may have different roles (salesman, designer, technician...), there is a 1-to-many relationship between product structure and process representation. A salesman has a different view of the product from a designer or a production engineer, even if all of them are dealing with the same product.

Since the codified knowledge is abstracted on a middle layer, it can be managed and reused for diverse contexts or different purposes by KBE developers, while the code can be synthetic.

3.3. K-Code Layer – Base Level

The K-Code Layer represents the base level and corresponds to the code itself. Here the codified knowledge can remain in a proprietary format (e.g. LISP syntax as used AML [22]). If the code itself is based upon object oriented programming, code snippets can be directly linked to UML. This way knowledge snippets from one KBE project can be transferred into another if the same programming language (or framework) and an object oriented structure is in use.

4. Evaluation

An initial evaluation of the provided approach shall be given by referring to typical user requirements on the one hand and outlining already identified but not solved challenges on the other. In the following some considerations are provided:

- In a non CAD-based O-O approach complex architectures can be calculated in advance easier than in a CAD environment. In addition resulting CAD assemblies do not include parts (components) which are not calculated.
- A second consideration concerns the representation of rules, formulas and dependencies. Since they are formalized in an abstract layer and not embedded into proprietary framework, they can be managed and visualized in a logical way. A visual representation directly supports common understanding. This way it becomes possible to reuse the knowledge in different contexts or for different tasks respectively.
- In direct comparison to a CAD-based KBE framework, the access to set-up a KBE solution on top of the three layered approach is laborious and not yet enough intuitive. Users have to know about UML and IDEF and KBE knowledge snippets cannot be easily inserted by double clicking on CAD items.

A transition between knowledge acquisition and knowledge codification is by default difficult, because turning knowledge from an unstructured document to a structured format leads always to effort in categorizing the knowledge defining relations and procedures.

4.1. The bookshelf use case

In order to demonstrate the validity of the drafted approach, an example has been carried out taken from the research activities (in ref. to EC-project *LinkedDesign*).

In particular, the activities related to the knowledge acquisition and knowledge formalisation accomplished for creation of an automated configuration of a bookshelf are presented and discussed in the following.

Knowledge acquisition

In order to accomplish this task, it is necessary to take over the role of a KBE developer, who interviews the domain experts in bookshelf production (e.g. a carpenter).

While trying to use the K-Brief notation for acquiring the knowledge, the output could be similar to Figure 2. The parameters are highlighted in bookshelf sketch, while the structure is represented as a hierarchical tree of blocks, such as *frame, walls* and *shelves*. In addition properties and configuration rules are written.

In addition the design process has been synthesized by three steps: Input values; Create configuration; Generate CAD (view CAD model).

Knowledge formalization

Once collected the knowledge related to the bookshelf design (mainly configuration rules), the researchers used a standard language notation to describe the configuration process of the bookshelf and the hierarchical structure of the product, with related objects, relations, properties and methods.

For the configuration process, the Researchers used the IDEF0 language, whereas the description of the product hierarchical structure is made using UML.

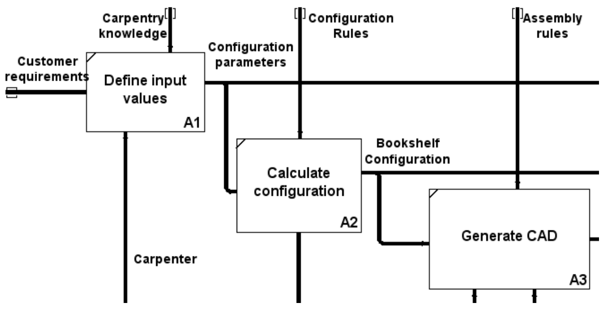


Fig. 3. Design process of the Bookshelf in IDEF0 (extract)

In the IDEF process (Fig. 3) tasks to be accomplished by the actors of the design process (carpenter, designer, etc.) are enriched by a description of the output documents (CAD models, BOM, etc.). The knowledge sources (Carpentry Knowledge, Configuration Rules, etc.) are on top of each task.

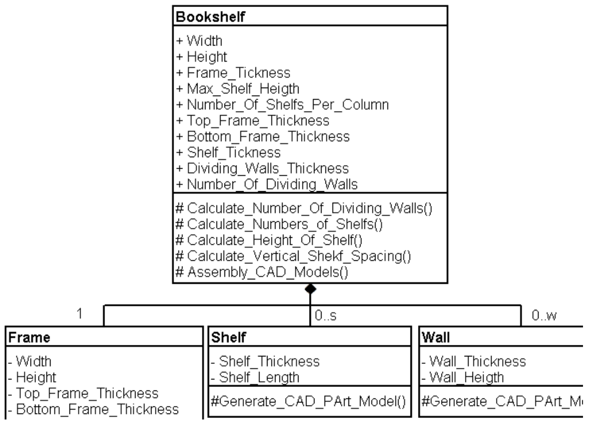


Fig. 4. Structure of the Bookshelf in UML (extract)

The corresponding UML model (Fig. 4) of the bookshelf structure shows the hierarchical levels of relationships among the parts (frame, shelves and walls), which are represented as objects and in addition their attributes (e.g. width or height). Further methods to calculate the dependent properties such as the quantity of the shelves or to assembly the CAD model are displayed for each object.

Using the formalizing languages as shown in the previous sections, it is possible to create a KBE application, independent of the select framework.

5. Conclusion

Even though customers increasingly participate in product development, many of them find it difficult to express needs or ideas that differ from those of manufacturers. In order to extract customer needs, developers create often several prototypes or variants to allow customers a concrete evaluation [23]. In this context Knowledge Based Engineering is providing the enabling technology to create fast and easy design variants and configurations respectively. However the effort to setup up and manage KBE properly is one of the major shortcomings.

The presented approach directly addresses current shortcomings by providing an approach based upon three layers. Utilizing an independent format for management of KBE knowledge, the users of CAx systems are able to exchange codified knowledge and gain the rationale behind. In order to reach the mentioned results, the explicit nature of IDEF and UML specifications has been taken into account. This way it is possible to formalize the product- and process-related aspects of a KBE implementation in unique files.

In particular, as next development of the project, the specifications of the *Rule Interchange Format* will be developed further. Since the design process is of course related to the product structure. The middle layer has to represent corresponding relationships between IDEF and UML representations. The proposed way is to transfer both representation into an XML representation, which includes a combination of properly organized tags and in addition linkages between both representations. This way the *RIF* will become a standardized representation of KBE knowledge, which can be visualized in form of interlinked UML and IDEF representations.

Hence the approach provides a substantial contribution for the development of systems, which are capable to easily adapt a given design to upcoming user-requirements, while facing the production challenges.

ACKNOWLEDGEMENTS

Part of this research has been funded under the EC 7th Framework Programme, in the context of the LinkedDesign project (http://www.linkeddesign.eu). The authors wish to acknowledge the Commission and all the LinkedDesign project partners for a fruitful collaboration.

References

[1] Shamsuzzoha A, Kyllönen S, und Helo P. Collaborative customized product development framework, In: Industrial Management & Data Systems, Bd. 109, Nr. 5, p. 718–735, 2009.

[2] K Sun K, Chen SL. Risk Reduction via Prototyping in Customized Product Development. In: Interdisciplinary Design: Proceedings of the 21st CIRP Design Conference.

[3] Yang H, Chen J, Lu Q, Ma N. Application of knowledge-based engineering for ship optimisation design. In: Ships Offshore Structure, Bd. 0, Nr. 0, p. 1–10, Nov. 2012.

[4] Prasad B. What Distinguishes KBE from Automation, 2005. http://legacy.coe.org/newsnet/Jun05/knowledge.cfm.

[5] Tseng MM, Jiao J, Su CJ. Virtual prototyping for customized product development. In: Integrated Manufacturing Systems, Bd. 9, Nr. 6, p. 334–343, 1998.

[6] Wu Q, Sun SQ, Dong ZX. A Computer-Aided Ergonomics Evaluation System for Customized Furniture Design, Advanced Materials Research, Bd. 102–104, p. 890–894, March 2010.

[7] Vermeulen B. Knowledge based method for solving complexity in design problems. Dissertation, Delft University of Technology, Delft, 2007.

[8] Milton N. Knowledge technologies Monza: Polimetrica, 2008.

[9] Colombo G, Pugliese D, Rizzi C. Developing DA Applications in SMEs Industrial Context. In: Computer-Aided Innovation (CAI), G. Cascini, Hrsg. Springer US, 2008, p. 69–82.

[10] Stokes M. Managing engineering knowledge: MOKA: methodology for knowledge based engineering applications, Bd. 3. Professional Engineering Publishing London, 2001.

[11] Lovett P, Bancroft C. Knowledge transfer for knowledge-based engineering. In: Proceedings of Technology Transfer and Innovation Conference, TTI, 2000.

[12] Fothergill P, Forster J, Lacunza JA, Plaza F, Arana I. DEKLARE, A Methodological Approach to Re-Design. In: In Opening Productive Partnerships, Advances in Design and Manufacturing (Conference on Integration in Manufacturing-IiM), 1995.

[13] Verhagen WJC, Bermell-Garcia P, van Dijk REC, Curran R. A critical review of Knowledge-Based Engineering: An identification of research challenges. In: Advanced Engineering Informatics, Bd. 26, Nr. 1, p. 5–15, Jan. 2012.

[14] Verhagen WJ, Curran R. Knowledge-based engineering review: conceptual foundations and research issues. In: New World Situation: New Directions in Concurrent Engineering, Springer, 2010, p. 267–276.

[15] IBM, Dassault Systemes: Product Synthesis Solutions, 2013. http://www.3ds.com/products-services/catia/portfolio/catia-v5/all-products/domain/Product_Synthesis/

[16] Verhagen WJ, Bermell-Garcia P, van Dijk RE, Curran R. A critical review of Knowledge-Based Engineering: An identification of research challenges. In: Advanced Engineering Informatics, Bd. 26, Nr. 1, p. 5–15, 2012.

[17] Kulon J, Broomhead P, Mynors D J. Applying knowledge-based engineering to traditional manufacturing design. In: International Journal of Advanced Manufacturing Technologies, Bd. 30, Nr. 9–10, p. 945–951, Okt. 2006.

[18] Rocca GL. Knowledge based engineering: Between AI and CAD. Review of a language based technology to support engineering design. In: Advanced Engineering Informatics, Bd. 26, Nr. 2, p. 159–179, Apr. 2012.

[19] Skarka W. Application of MOKA methodology in generative model creation using CATIA. In: Engineering Applications of Artificial Intelligence, Bd. 20, Nr. 5, p. 677–690, Aug. 2007.

[20] OMG. UML, 2011. http://www.omg.org/spec/UML/.

[21] Pugliese D, Colombo G, Spurio ML. About the integration between KBE and PLM. In: Advances in Life Cycle Engineering for Sustainable Manufacturing Businesses, Springer, 2007, p. 131–136.

[22] TechnoSoft Inc. Adaptive Modeling Language. A Technical Perspective. 2003.

[23] Terwiesch C, Loch CH. Collaborative Prototyping and the Pricing of Custom-Designed Products. In: Management Science, Bd. 50, Nr. 2, p. 145–158, Feb. 2004.