

A PROPOSAL FOR KNOWLEDGE FORMALIZATION IN PRODUCT DEVELOPMENT PROCESSES

Klein, Patrick (1); Lützenberger, Johannes (2); Thoben, Klaus-Dieter (1)

1: University of Bremen, Germany; 2: BIBA - Bremer Institut für Produktion und Logistik GmbH, Germany

Abstract

With respect to time-to-market and product development Knowledge Based Engineering (KBE) solutions, which enable an automation of design tasks, can contribute to a significant reduction of development time. But to develop KBE solutions, not only engineering but also informatics competencies are required, which leads often to case-based solutions and in addition to costly extra effort, even in those cases where solutions only have to be adapted to a new context. This way, KBE becomes too expensive for development teams, because they are only useful in a specific context and expensive to be managed.

Enabling a broader usage context for KBE solutions, e.g. for different domains or in different projects, increases the efficiency of KBE. This paper aims to support designers to set-up and maintain KBE solutions, which address diverse product development tasks at once. Against this background, this paper identifies a need for a framework-independent solution to formalize design knowledge and derives corresponding objectives to be fulfilled by this solution. Further, an extension of an existing modelling language to become a new framework-independent standard for KBE projects is proposed.

Keywords: Knowledge Based Engineering, KBE, Knowledge management, Collaborative design, Knowledge Formalization

Contact:

Patrick Klein
University of Bremen
BIK Institut für integrierte Produktentwicklung
Germany
klp@biba.uni-bremen.de

Please cite this paper as:

Surnames, Initials: *Title of paper*. In: Proceedings of the 20th International Conference on Engineering Design (ICED15), Vol. nn: Title of Volume, Milan, Italy, 27.-30.07.2015

1 INTRODUCTION

In order to stay competitive on the global market companies in countries with high labour costs need to optimize the product development processes. This optimization can be performed by a reduction of time-to-market and costs respectively, which can be achieved by a re-organisation of the development process itself or by accelerating the individual development steps. The latter, is addressed by software, such as Knowledge Based Engineering (KBE) solutions, which enable an automation of repetitive design tasks. Since nearly 80% of the design tasks ((Skarka, 2007)) appear to be repetitive, a significant reduction of development time (up to several weeks) has been achieved in nearly all documented KBE projects (Vermeulen, 2007).

Unfortunately KBE solutions are developed to be used in one specific development context, such as generation of body wing designs of airplanes or the automated calculation of ship structural design (Yang et al., 2012), because of their complex set-up process. Nowadays initiating a KBE project in a product development team means an involvement of stakeholders. While computer scientists usually perform the software implementation, domain specialists can only perform the acquisition and capturing of knowledge (e.g. for the identification of formulas to calculate weld seam thickness). While big companies (e.g. in aircraft industry) usually form interdisciplinary teams of companywide specialists (Ammar-Khodja et al., 2008), small and medium companies often stick to non KBE supported development, third party support or ad-hoc teams. In any case, bringing together competences from different domains (engineering and informatics) to develop and deploy KBE solutions leads to effort and demands time. Thus, the focus usually lies on setting up a specific solution for a concrete task and not on a more generic application (which would demand for even more experts from different domains and a longer KBE development process).

Contemporary solutions are highly limited to their own task related boundaries. The identified and applied engineering knowledge can only be used inside the respective model. Even adaptations or enhancements often can't be done by the end users, since this would mean to go through the code itself.

A product developer without sophisticated programming skills will not be able to use, set-up or maintain such a solution, properly. In addition, the solutions often lack transparency and documentation. The designers (or engineers) have to trust in the output of KBE calculations and generated designs and thus often avoid using KBE solutions, if a development task slightly changes.

In this way KBE solutions (& projects) become too expensive for the development teams, because they are only useful in one context and expensive to be managed. Enabling a broader usage context for KBE solutions, e.g. for different domains or in different projects, unleashes the potential of KBE to reduce time-to-market, even for small and medium enterprises without specific KBE teams.

In consequence this paper aims to support designers to set-up and maintain KBE solutions, which address diverse product development tasks at once. Against this background, this paper identifies a need for a framework-independent solution to formalize design knowledge and derives corresponding objectives to be fulfilled by this solution. Further, this paper proposes an extension of an existing modelling language to become a new framework-independent standard for KBE projects.

2 NEED FOR FRAMEWORK INDEPENDENT FORMALIZATION OF KNOWLEDGE

The above-mentioned challenges in terms of setting up KBE solutions and gaining access to the underlying rationale have been picked up by industrial software systems. Commercial vendors of CAX tools (e.g. CATIA, SolidWorks, Inventor etc.) provide add-ons to automate the design directly from within the CAD (or CAE tool) in order to automate the product-design process (e.g. (IBM, 2013)). They enable the user to incorporate dependencies using formulas or rules (if Length A < x then Length B=Y). This became possible, because of fully parameterized CAD models. Along with the possibility to use construction tables or standard parts catalogues to be integrated into a CAD model, simple KBE solutions are supported without further need of IT experts. Based on a parameterized CAD model, they provide functions like formulas (to create dependencies between parameters), rules and user defined features (IBM, 2013). But in such CAD add-ons the KBE intelligence (e.g. a design rule) directly remains inside a CAD-model and is directly stored within the CAD file.

Also standalone KBE frameworks, such as the Adaptive Modelling Language framework (AML) (TechnoSoft Inc, 2003), designed to support large KBE projects, are enforcing an encapsulation of knowledge due to their proprietary formats and modelling languages.

As identified in a comparative analysis of different frameworks to support KBE (a comparison between CATIA Knowledgeware, AML, Rulestream and Siemens NX) (Colombo et al., 2014), one of the main challenges is founded 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 after it is codified in rules or in source code. For instance in a scenario of an automated bookshelf design, the rule that the width of a shelf can be a function of width of a rack may appear as part of a cryptic notation (If $LR > LS$ then $LR = LS * nS \dots$) (Klein et al., 2014).

Even if it would be possible to break up the file encapsulation, which is given by the proprietary structure (of e.g. CAD models, or AML projects), 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.

Depending on domain and profession of involved experts, skills in programming as well as knowledge modelling 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 ...). Other developers 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 ...), whilst other are focused on costs, etc. Hence, the capability to provide framework independent access to the codified knowledge for different stakeholders is an important constraint.

In consequence, the development of a methodological approach as well as a standard for modelling and representation of codified knowledge can be perceived as an important objective to ensure sustainability and user-acceptance on the one hand and enable an exchange and reuse of engineering knowledge in automated design on the other hand.

2.1 Objectives of a new approach to support Knowledge Formalization

A framework independent access to the codified engineering knowledge should aim to address different objectives as given by the field of product development and in particular KBE. In order to achieve a representation and formalization of knowledge in this context several objectives have been identified:

- **Capabilities to represent the KBE related elements:** By developing disjunctive implementations for one KBE scenario it turned out that there are distinct similarities and to some extent an overlapping in syntax and structure between those elements, which are provided by the different KBE frameworks (Klein et al., 2014). As a basic precondition for framework-independent solution to formalize design knowledge, the solution should provide modelling capabilities to represent those KBE expressions (e.g. product structure, input variables, equations & rules, etc.).
- **Reflecting the KBE life cycle in a structured way:** 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 (Verhagen et al., 2012). In addition, most solutions are still suited to a single case and neither grounded in the workflow organisation nor based on structural frameworks (Verhagen and Curran, 2010). A new modelling language easy to use and suited to the process of setting up and deploying KBE solutions shall be capable to address this.
- **Support collaboration among different domains:** Due to missing abstractions - e.g. neglected design aspects for a desired solution – KBE solutions are often too limited to their original context and thus the reuse of knowledge in different contexts will be hindered or impossible (Verhagen et al., 2012). For this purpose, an easy incorporation of knowledge from different domains and business units should be supported in order to allow product development teams to be more flexible. (If the KBE intelligence directly remains inside encapsulated models or proprietary languages, this is of course not possible.)
- **Reflecting the process of formalization and codification:** Verhagen, et al. criticize that many KBE-solutions store and represent codified knowledge decoupled from its original context (Verhagen et al., 2012). An appropriate documentation is missing and formulas or equations remain unexplained (Kulon et al., 2006). The cause is often founded in an unstructured

knowledge acquisition process. Without a documentation of the problem in terms of objectives, constraints, etc. the traceability of the design process becomes impossible. The envisaged approach should enable formalisation of knowledge, since this enables a transfer of the ownership of the technical knowledge from the individual experts to the organization.

- **Enabling reuse of knowledge in different contexts:** Along with the insufficiency of a structured knowledge codification, a lack of knowledge reuse has been identified. Knowledge is often codified 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 (Verhagen et al., 2012). The envisaged approach 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.
- **Capture design process related background knowledge:** In contemporary KBE projects, the process related context (e.g. the relation to a specific development phase) is usually not annotated in the implemented 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. For this purpose the new approach should incorporate/reflect the design/development process.
- **Support visualization of concepts and their relations:** Engineers “like” visualizations and graphics and visual support can be one key advantage to overcome communication barriers between different domains. Thus, the envisaged approach should provide capabilities for visualization of codified knowledge.

Next to the identified functional objectives, the researchers followed the idea to rely the approach on well-established standards in research and industry. This way, the usability increases and potential users are attracted by lower effort required to get started. Further, a standardisation body may facilitate the sustainability of the approach in terms of evolution, exploitation and dissemination.

3 FORMALIZATION OF DESIGN KNOWLEDGE

With respect to the identified objectives, an analysis of existing established standards for representation of codified knowledge has been conducted as a pre-study (Colombo et al., 2014). Since the comparative analysis exceeds the limits of a publication, Table 1 just provides an overview of the examined standards within the context of the identified objectives.

Table 1. Comparative analysis of standards for representation of codified knowledge

Requirement Language	Product Structure	Rules and Constraints	Suitable for Domain Expert	Design Process	Visual representation
MathML	X	✓	✓	X	X
xOWL	✓	✓	X	✓	X
Rule Interchange Format of W3C	X	partially	X	✓	X
IDEF	IDEF0	X	✓	✓	✓
	IDEF3	X	✓	✓	✓
MML	✓	partially	partially	✓	✓
BPMN	X	partially	✓	✓	✓
SysML	✓	✓	✓	✓	✓
UML	X	partially	✓	✓	✓

In the performed analysis, SysML turned out as the most appropriate standard to be extended for the purpose of modelling and exchanging product design knowledge. As outlined below, SysML is supported by the OMG and has its roots in UML. It also has become an established standard for systems engineering (Bone and Cloutier, 2010). One of the key advantages of this "standard family" lies in its inherent capability to be extendable. The needed meta-model layer is provided by default in the specification of UML itself. Hence, an adaption to domain specific needs can be performed

without contradicting the specification. In other words, because SysML is an extension of UML the presented approach of extending SysML also allows customizing the UML/SysML ecosystem to the needs of KBE.

In the following, the idea of an extension of the established SysML standard to provide a new modelling language in context of KBE will be reflected by a new name as well: **KbeML**.

3.1 Definition of KbeML as an extension of UML and SysML

To visualize the relationship between the UML and SysML languages and the envisaged KbeML, a Venn diagram is provided in Figure 1. The sets of language constructs that comprise UML, SysML and KbeML are shown there as boxes marked “UML”, “SysML,” and “KbeML” respectively. The intersection of the two boxes, shown by the region marked “UML reused by SysML & KbeML” indicates the UML modelling constructs, that SysML reuses, called the UML4SysML subset (ref to (OMG, 2012)).

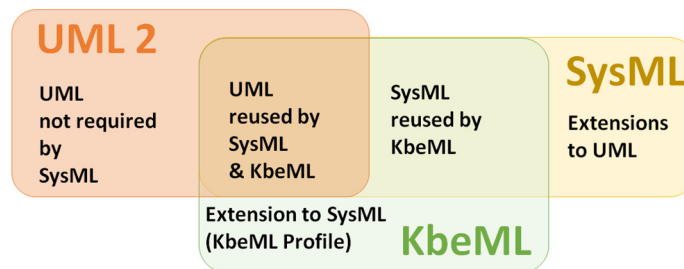


Figure 1. Relation between UML, SysML and KbeML

The region marked “Extensions to SysML (KbeML Profile)” in Figure 1 indicates the new modelling constructs defined for KbeML that have no counterparts in SysML or UML.

SysML reuses a subset of UML 2 and provides additional extensions to satisfy the requirements of the language. SysML is defined as an extension of the OMG UML 2 Superstructure specification. It is intended to be supported by two evolving interoperability standards including the OMG XMI 2.1 model interchange standard for UML 2 modelling tools and the ISO 10303 STEP AP233 data interchange standard for systems engineering tools (ref to (OMG, 2012)).

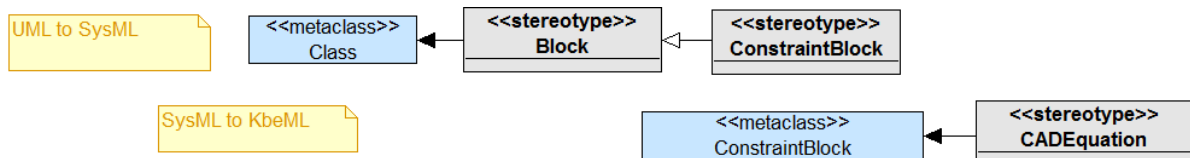


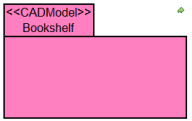
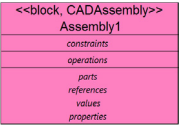
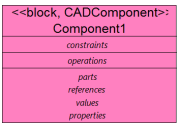
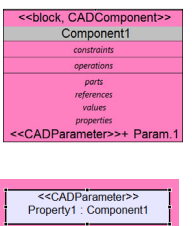
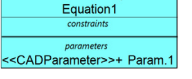
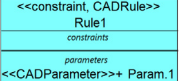
Figure 2. From UML to SysML to a KBE extension

The extension of SysML has been realized by establishing so-called stereotypes. Stereotypes are representing pre-defined information types. As outlined in Figure 2, the UML 2 meta-classes are connected to stereotypes with the extension connector (graphical representation is a filled triangle). New stereotypes can be based upon existing ones via generalization.

3.2 Stereotypes proposed for the SysML extension to define KbeML

The provided behavioural elements of SysML allow describing design processes without the need to be adapted. Thus, this research focusses on the definition of structural elements for the envisaged KbeML. These new elements are summarized in table2. The approach suggests to define all elements with prefix “CAD” for a better human readability. This may be misleading, the proposed specification is aiming to support the whole spectrum of KBE (such as other CAx tools).

Table 2. New KbeML Elements defined as an extension of SysML Metaclasses

Vizualization	Description	Formal Definition
	<p><i>CADModel</i> provides a mechanism for representing a spatial model as being typically defined by model files in CAD. A <i>CADModel</i> is associated to the package element of SysML. In SysML/UML packages partition the model elements into logical groupings that minimize circular dependencies among them (OMG, 2012). A <i>CADModel</i> may contain the elements <i>CADComponents</i>, <i>CADAssemblies</i>, etc. and in addition a block diagram for visualising the product structure by using those elements.</p>	<pre> classDiagram class Package class CADModel["<<stereotype>> CADModel"] Package -- > CADModel </pre>
	<p><i>CADAssembly</i> provides a mechanism for representing a product assembly as typically defined by components in CAD. The <i>CADAssembly</i>, which is defined as a stereotype of a block intends to represent the structural perspective of a product geometry. According to the proposed approach a <i>CADAssembly</i> can consist of <i>CADComponents</i> and/or <i>CADAssemblies</i>.</p>	<pre> classDiagram class Block class CADAssembly["<<stereotype>> CADAssembly"] Block -- > CADAssembly </pre>
	<p><i>CADComponent</i> provides a mechanism for representing a component as being typically defined by its shape and geometry in CAD. The <i>CADComponent</i>, which is defined as a stereotype of a block intends to represent the structural perspective of a product geometry. <i>CADComponents</i> can be represented in Block Definition Diagrams, e.g. to visualize the structural dependency to <i>CADAssemblies</i> or SysML blocks.</p>	<pre> classDiagram class Block class CADComponent["<<stereotype>> CADComponent"] Block -- > CADComponent </pre>
	<p>A <i>CADParameter</i> represents always a geometry related parameter, which means that this type of parameter can be processed by a CAD application. A <i>CADParameter</i> is instantiated by a <i>CADComponent</i>, <i>CADAssembly</i>, <i>CADEquation</i>, or <i>CADRule</i> (always as stereotype of property), but can also be used by other elements for non-CAD related modelling or calculations. It is visualized as a property in a component or as a box in a parametric diagram.</p>	<pre> classDiagram class Property class CADParameter["<<stereotype>> CADParameter"] Property -- > CADParameter </pre>
	<p><i>CADEquation</i> provides a mechanism for representing dependencies between CAD related parameters. A <i>CADEquation</i>, which is based upon a constraint block can be used to specify a network of constraints that represent mathematical expressions such as {height=2*width}, which constrain the geometrical properties of a CAD object.</p>	<pre> classDiagram class ConstraintBlock class CADEquation["<<stereotype>> CADEquation"] ConstraintBlock -- > CADEquation </pre>
	<p><i>CADRule</i> provides a mechanism for representing more complex dependencies between CAD related parameters such as geometrical dependencies based upon conditions (e.g. height > 10). Formally, a <i>CADRule</i> is a set of instructions, which conditionally executes a group of statements depending on the values of some <i>CADParameters</i>.</p>	<pre> classDiagram class ConstraintBlock class CADRule["<<stereotype>> CADRule"] ConstraintBlock -- > CADRule </pre>

4 DEMO CASE - AUTOMATED DESIGN OF A BOOKSHELF

In the previous section, the proposed extension to SysML has been presented. In order to evaluate the applicability of the specified KbeML elements for typical development tasks, an automated bookshelf design has been chosen as a demo case. In the demo case the automated design of a bookshelf is driven by a set of geometry related input parameters, such as thickness of the shelves and an geometry related output parameter (shelf length) and in addition based on two equations to formalize the deflection of a shelf under load (ref. to Figure 3).

Next to the applicability of the proposed modelling language, the demo case aims to indicate the capacity of KbeML to re-use mechanical knowledge (about deflection) a new development context. The formalized engineering knowledge is encapsulated in (constraint-) blocks, which can be assigned to other components in a new context, simply by a parameter mapping.

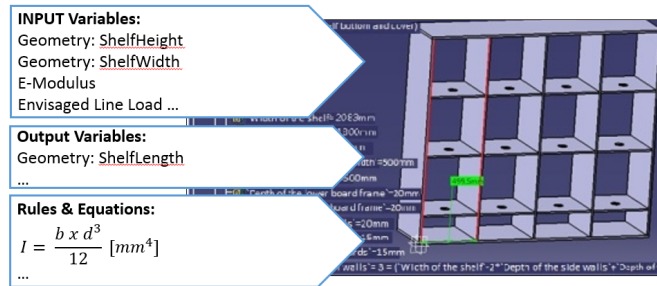


Figure 3. Scenario - Automated design of a bookshelf

4.1 KbeML - Formalization of product structure

In the following, an extract of a framework independent KBE model of an automated design of the bookshelf is provided to indicate the practicability of the KbeML approach. Next to the representation of the product structure, the codified rule to calculate the maximum length of the shelves is modelled using the KbeML elements.

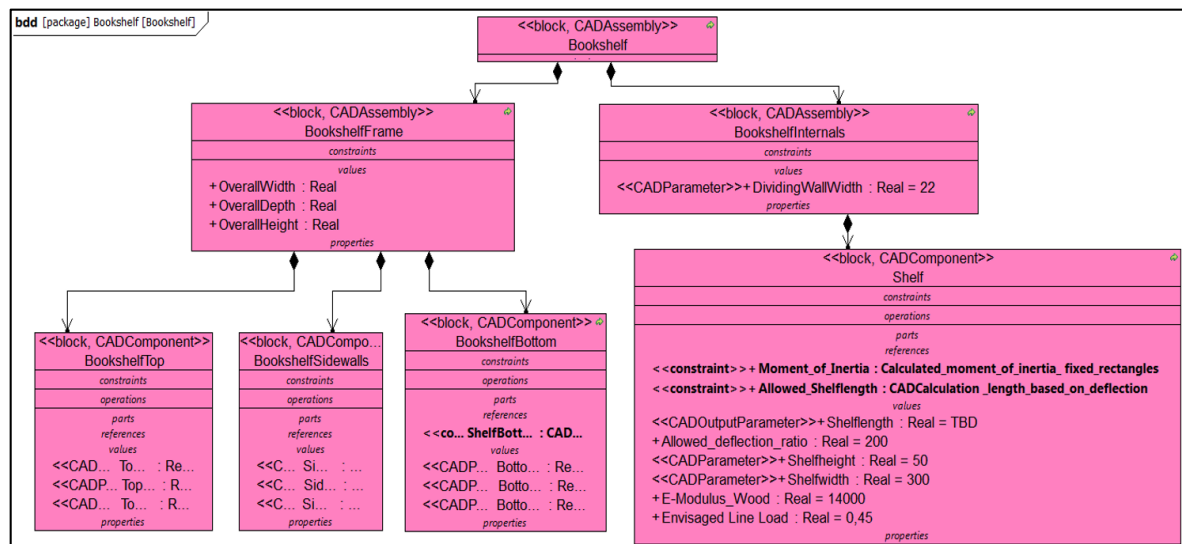


Figure 4. Representation of bookshelf model with KbeML

As outlined in Figure 4 the bookshelf product is represented by a stereotype CADAssembly. Further, the product consists of two sub-assemblies called *BookshelfFrame* and *BookshelfInternals*, which are also defined as CADAssembly. The sub-assembly *BookshelfFrame* consists of the components *BookshelfTop*, *BookshelfBottom* and *BookshelfSidewalls*, all of type CADComponent. All mentioned composites of the bookshelf are based on the SysML element block. A composite association (black diamond) is used to indicate that the sub-assemblies belong to an assembly etc.

By usage of properties for CADComponent and CADAssemblies, the belonging parameters are defined; e.g. the component *Shelf* is described by the CADparameters: *ShelfWidth*, *ShelfHeight* and *ShelfLength*. While the first both parameters are described by concrete values the *ShelfLength* is marked with TBD (abbreviation for To Be Defined), which indicates that this parameter has to be calculated.

4.2 KbeML Formalization of Equations and Rules related to KBE.

In order to achieve a calculation of a specific parameter the concrete parts are linked to an equation formalized in a general manner. For this purpose in a first step a block diagram of the CADComponent Shelf has been modelled (refer to Figure 5). Within this block diagram the specific CADComponent as

well as the CADEquations are represented in parallel. The concrete component is represented in magenta, while the description of the equations is represented in cyan, in order to support a visual distinction between "KBE knowledge" and a concrete part. Hence, the purpose of this diagram is solely to indicate, which generic equations are used for the concrete component. The relation itself is defined by a reference property. (In Figure 5 you will find in the CADComponent a line defining the constraint: *Allowed_Shelflength*: *CADCalculation_length_based_on_deflection*.)

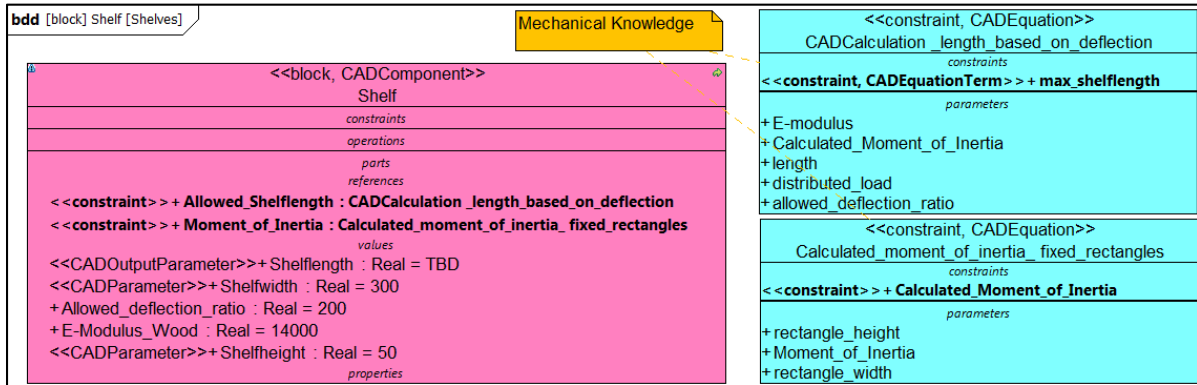


Figure 5. Block diagram of CADComponent Shelf

In addition, a link between the concrete parameters to the generic parameters of the equation is required to represent the KBE knowledge. For this purpose SysML parametric diagrams can be used allowing to map properties of a constraint block to properties of a block. In context of KbeML this means to map the concrete parameters of a CADComponent, such as *BookshelfBottom* to the generic parameter of the instance (called *Allowed_Shelflength*) of the generic KBE equation (*CADCalculation_length_based_on_deflection*). This way, Shelf parameters (CADParameters and additional Information such as E-Modulus of wood) will be used as input parameters to calculate the missing parameter *Shelflength*. Associations outline the mapping in Figure 6.

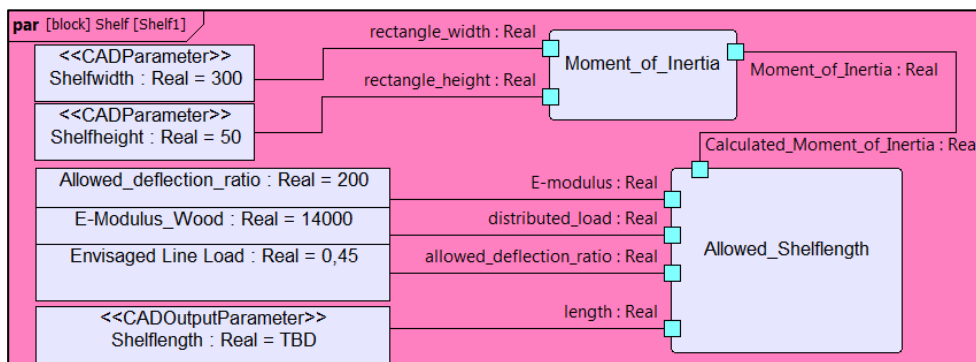


Figure 6. Parametric diagram of CADComponent Shelf

The equation *Allowed_Shelflength* is composed by an equation, which uses the *Moment_of_Inertia* (MoI) as an input parameter. The EquationTerm itself is defined as a value and composed by the following elements: CADParameters (e.g. *deflection_ratio*) and one CADOutputParameter (e.g. *length*) (ref. to Figure 7). In addition, CADOperators (e.g. ***) and numbers (e.g. 384) can be used (ref. to Figure 7). A CADOutputParameter is defined as a specific CADParameter and represents a parameter of a CAD system, which depends on other parameters.

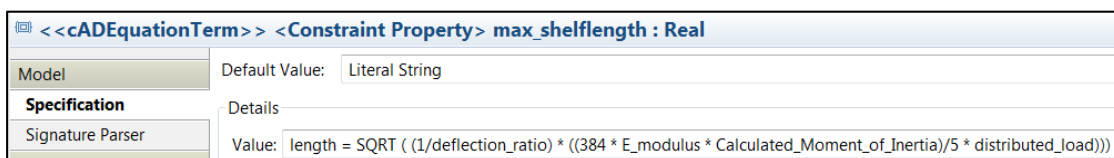


Figure 7. Formula of Deflection represented in KbeML CADEquationTerm

In principle the equation for calculating a deflection is valid in several development contexts. For the calculation of deflection for a steel beam the same rules and equations can be used as for the shelves (assuming fixed support on both edges). Hence, basic mechanical knowledge can be represented in SysML/KbeML libraries to be re-used in different KBE projects later on.

In case of the shelf the cross-section is typically a rectangle, but if one decides for a bookshelf with trapezia shaped shelves the MoI can be simply picked from a MoI library and replaced in the Parametric diagram of CADComponent Shelf (Figure 6) without changing the CADEquationTerm (Figure 7).

According to proposed KbeML the Bookshelf deflection is now completely defined. At this stage several options appear to the end-user, indicating the degree of freedom of a knowledge representation codified in a neutral format:

- Calculation of shelf length, exploiting the capabilities of commercial modelling environments such as EnterpriseArchitect, which allow to create executable ConstraintBlocks (as well as CADEquations). Thus, an integrated solver is able to “calculate” the CADOutputParameters.
- Using a mapping software to link third party solver providing the value of the calculated length: In principle solvers can range from complex (e.g. Open Modelica) to simple applications (e.g. Excel).

5 EVALUATION IN CONTEXT OF THE DEMO CASE (EXTRACT)

Even if the above-mentioned demo case of the bookshelf is not challenging for designers in a sense that it could not be solved simply by creation of design variants (for different loads). A major difference to framework-based KBE (and variants) appears - the usage of "generic" equations.

While this notation appears to be a bit cumbersome at a first glance, a separation of equations and rules from components and assemblies provides major advantages in terms of reusing knowledge. For instance, the equation for the moment of inertia is only based upon an transversal section of an element, the CADEquation can be re-used for automated design of different components (e.g. a beam in a steel frame), simply by mapping each time CADComponent parameters to CADEquation parameters.

However, while using the proposed KbeML and capturing the design and engineering knowledge for demo cases such as the automated bookshelf design, several limitations and improvement points have been identified so far. The definition of CADEquation terms may serve as just one example. Currently there is no controlling instance incorporated that only existing CADEquation parameters are used in the value (which defines the CADEquationTerm). While those limitations are not directly contradicting the approach (maybe OCL Rules can be used to provide a more sophisticated modelling environment)(Alt, 2012), one major challenge appeared while representing rules, which control the product tree itself. This is for instance the case if the existence design variants will be managed by a rule (e.g. exchange of the CADComponent *SideWalls* with *SideWallsWithDoorMountings* if a bookshelf with doors shall be a possible variant). Since SysML itself lacks of a standardized possibilities to represent and manage design variants, a literature analysis has been conducted to identify different "workarounds" as being proposed by other researchers (e.g. Maga and Jazdi, 2011). An adaption of KbeML is under preparation, proposing (amongst others) additional KbeML Elements in alignment to table 2, namely CADVariant, CADVariationPoint.

However, even though the example is too simple to fully represent the potential complexity of all practical KBE solutions, it proves that it is possible to codify KBE related design knowledge independently from a KBE framework and allow a re-use in a new context.

6 CONCLUSION AND OUTLOOK

The cumbersome setup and implementation phase to deploy a KBE project can be perceived as a bottleneck for an efficient usage and broad acceptance of KBE in product development. In addition, the existing solutions often lack of transparency and documentation. Furthermore, adaptations cannot be performed by end users. In this sense, the effort for setting up KBE solutions and managing the rationale behind is contradicting the applicability of KBE solutions targeting usually only specific design tasks and lacking a possibility to be reused in different contexts.

Against this background, the development of a methodological approach as well as a standard for modelling and representation of codified knowledge is proposed to ensure sustainability and user-acceptance on the one hand and enable an exchange and reuse of engineering knowledge in automated design on the other hand. In reference to this identified gap, the conducted research focussed upon the identification of objectives to be fulfilled by such a new standard.

A comparative analysis of different standards & modelling languages for management of formalized knowledge turned out, that SysML provides an appropriate basis for a new modelling language to fulfil the identified needs in context of KBE. The established Standard has been enhanced and adapted using the UML/SysML extensions (according to UML2 Superstructure specification (OMG, 2012)). The proposed extension to capture and exchange formalized knowledge in context of KBE can be perceived as the main result of the presented findings. The elements being specified in section 3 are approaching to be a preliminary work for a contribution to a standardisation directly addressing the needs of KBE. If stakeholders show interest, the drafted KbeML extension will perform a starting point corresponding working groups of the OMG.

An automated design of a bookshelf has been picked up for an initial validation of the proposed KbeML modelling language. The representation of the design related knowledge using KbeML indicates that it is possible to use a generic codification for at least this example. Hence, the scenario proves that it is possible to codify KBE related design knowledge independently from a proprietary framework.

Since SysML/KbeML are based upon an open format (XMi standard), exchange of the captured knowledge between different frameworks shall become possible (Huang et al., 2007). As a next step, the research focuses upon knowledge exchange between a KbeML model and a CAx application. For this purpose a mapping software is currently under development. This application allows to handover CADparameters and CADEquations to a commercial CAD application in order to control the design directly from within the model.

REFERENCES

- Alt, O. (2012) *Modellbasierte Systementwicklung mit SysML*. München: Carl Hanser Verlag.
- Ammar-Khodja, S., Perry, N., Bernard, A., (2008) Processing Knowledge to Support Knowledge-based Engineering Systems Specification. *Concurr. Eng.* 16, pp. 89–101.
- Bone, M., Cloutier, R., (2010) The Current State of Model Based Systems Engineering: Results from the OMGTM SysML Request for Information 2009. *Proceedings of the 8th Conference on Systems Engineering Research*. Hoboken, New Jersey.
- Colombo, G., Pugliese, D., Klein, P., Lützenberger, J. (2014) A study for neutral format to exchange and reuse engineering knowledge in KBE applications. *Engineering, Technology and Innovation (ICE)*, 2014 International ICE Conference, Bergamo, 2014, pp. 1–10.
- Huang, E., Ramamurthy, R., McGinnis, L.F. (2007) System and Simulation Modeling Using SysML, in: *Proceedings of the 39th Conference on Winter Simulation: 40 Years! The Best Is Yet to Come, WSC '07*. IEEE Press, Piscataway, NJ, USA, pp. 796–803.
- IBM (2013) Dassault Systemes: Product Synthesis Solutions [online], http://www.3ds.com/products-services/catia/portfolio/catia-v5/all-products/domain/Product_Synthesis/ (accessed 29.07.14).
- Klein, P., Pugliese, D., Lützenberger, J., Colombo, G., Thoben, K.-D. (2014) Exchange of Knowledge in Customized Product Development Processes. *24th CIRP Design Conference, Procedia CIRP*, Vol. 21, pp. 99–104.
- Kulon, J., Broomhead, P., Mynors, D. (2006) Applying knowledge-based engineering to traditional manufacturing design. *Int. J. Adv. Manuf. Technol.* Vol. 30, No. 9-10, pp. 945–951.
- Maga, C.R., Jazdi, N. (2011). Survey, Approach and Examples of Modeling Variants in Industrial Automation. *J. Control Eng. Appl. Inform.* Vol. 13, pp. 54–61.
- OMG (2012) *OMG Systems Modeling Language (SysML)*
- Skarka, W. (2007) Application of MOKA methodology in generative model creation using CATIA. *Eng. Appl. Artif. Intell.* Vol. 20, pp. 677–690.
- TechnoSoft Inc. (2003). *Adaptive Modeling Language. A Technical Perspective.* [online], www.technosoft.com/docs/AML-Technical-Perspective.pdf (accessed 31.03.15)
- Verhagen, W.J.C., Bermell-Garcia, P., van Dijk, R.E.C., Curran, R. (2012) A critical review of Knowledge-Based Engineering: An identification of research challenges. *Adv. Eng. Inform.* Vol. 26, pp. 5–15.

- Verhagen, W.J.C., Curran, R. (2010) Knowledge-Based Engineering Review: Conceptual Foundations and Research Issues. In: New World Situation: New Directions in Concurrent Engineering. London: Springer Verlag, pp. 267–276.
- Vermeulen, B. (2007) Knowledge based method for solving complexity in design problems (Thesis). Delft University of Technology, Delft.
- Yang, H., Chen, J., Lu, Q., Ma, N. (2012) Application of knowledge-based engineering for ship optimisation design. Ships and Offshore Structures. Vol. 9, No.1, pp. 64-73.

ACKNOWLEDGEMENT

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.

