INTERNATIONAL DESIGN CONFERENCE - DESIGN 2012
Dubrovnik - Croatia, May 21 - 24, 2012.

SUPPORTING TRACEABILITY OF DESIGN RATIONALE IN AN AUTOMATED ENGINEER-TO-ORDER BUSINESS MODEL

F. Elgh and M. Poorkiany

Keywords: engineer-to-order, design rationale, knowledge management, knowledge modelling, design automation

1. Introduction

The ability to efficiently and quickly design and manufacture highly customized product can provide a competitive advantage for companies acting on a market with shifting customer demands. A business model based on highly customized product requires advanced application systems for automating the work of generating product variants based on different customer specification. The establishment of a system for automated design and production preparation is a significant investment in time and money and is expected to give revenues over many years. To maintain a design automation system's usefulness over time, frequent updating of design rules and execution control will normally become a necessity. Reuse of the system encapsulated generic product family descriptions when developing a new product family is also perceived to significantly increase the efficiency in system development. The scope and the purpose of this research originate from industrial problems and needs which have been identified within research projects carried out in near collaboration with industrial partners. New concepts, perceived as prescriptive models, are in this work introduced, evaluated, and refined which is in accordance with the design modelling approach [Duffy and Andreasen 1995]. The focus of this paper is a case study carried out at a company with long experience of systems for automated variant design. The main objective is to provide a system foundation for modelling and management of product knowledge in design automation systems to support reuse, expansion and maintenance.

1.1 Design rational and traceability

Design rational is the set of reasons behind the decisions made during the design of an artefact (e.g. a product or an application system). The access to design rational can support development of new artefacts, modification of existing artefacts (design changes) or the reuse of an existing solution in a new context. The realisation of a design rationale system includes methods and tools to capture, structure, manage and share information across organisations, processes, systems and products. The requirements concerning the scope and the granularity of design rational to be captured depend on future needs. These can be difficult to foreseen, however, a limitation has to be set as is not feasible to capture everything during the design process. Two different approaches to represent design rational are Argumentation-based and Template-based [Tang et al. 2007]. Argumentation-based representation uses nodes and links whilst Template-based representation makes use of predefined standard templates. The selection of approach will affect the scope, the granularity and the structure of the captured design rationale; however, the key factor for successful implementation of a design rational recording tool is simplicity [Bracewell et al. 2009]. The development of a design automation system is preferably a part of, or integrated with, the development of the actual product. Four sub-processes can

be identified within such a development process resulting in four different outputs: the product design, the design space, the system adapted definition of the design space, and the system implementation. Traceability, defined as *"...the ability to describe and follow the life of a conceptual or physical artefact."* [Moham and Ramesh 2007], across these sub-processes is essential. The artefact of concern in this study is mainly the design automation system. The design automation system encapsulates product knowledge that has been expanded and transformed into different levels of completeness and generalisation throughout the four sub-processes. Traceability, both forward and backward, across different knowledge levels would support the work of pursuing affected objects when changes occur in the premises of a design or the work of using an existing solution in a new context; i.e. knowledge traceability, defined as *"...the ability to follow the life of a knowledge component from its origins to its use."* [Moham and Ramesh 2007], is required.

2. Development process and documentation issues

The development process for companies utilizing systems for highly customized product variants differs from a traditional product development process as it is aimed at describing a product space by rules and digital models, starting with marketing research and ending with an application program describing the design space of a product family. One approach is to develop individual instances of a planned product that are verified, including, by example, structural analysis, functional tests, CAD modelling and building prototypes. Based on these instances, a design space of the product family is defined and described by rules and associated 3D solid models. The rules are documented and structured as expressions, tables and figures. The rules are required input for the design programmers who prepares the 3D models with information (e.g. geometries, datum features and named surfaces) to be used when creating programs for the product design. Design programming also includes: the adaptation of the product family description for the system used for application development, the actual coding in that system and the verification of the final application. The result is what can be called a Design Automation Model (Figure 1) which includes two sub-models, a Rule-model and a CAD-model, defined by a number of different model elements (building-blocks). The output when executed includes variant specific: 3D models configured for CAM preparation and CMM preparation; quotation drawings, assembly and manufacturing drawings; customer data (e.g. drawings and 3D model).

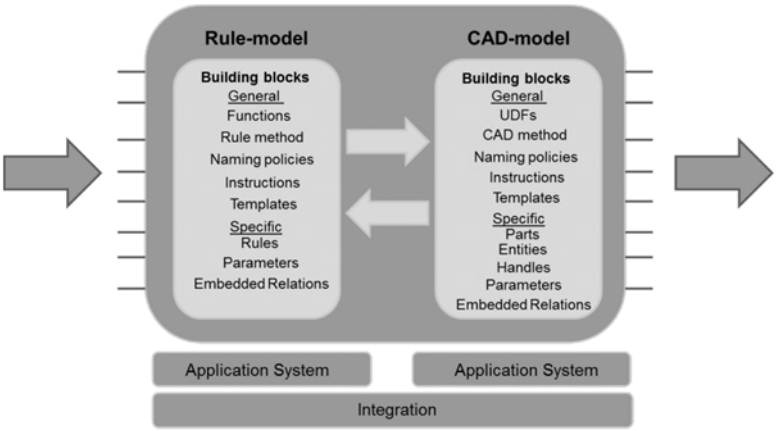

Figure 1. Building-blocks of a design automation model

When studying the documentation of these systems it can be concluded that it is mainly directed towards describing the final results of the different activities, i.e. answering "What?" questions. To reuse a rule in a new context (another product family) requires more information, for example scope, range, simplifications and underlying assumptions. Such information might be enough if the rule is to be used as it is, but if the rule has to be modified and adapted to specific circumstances even more information is required to support the adaptation while ensuring the validity of the rule. However, there is a challenge due to the reasons that documentation is perceived as important in the long term perspective but commonly viewed as a non-value adding activity within a specific project.

3. Framework for modelling and management of product knowledge

To maintain a design automation system's usefulness over time, frequent updating of design rules and execution control will normally become a necessity. Reuse of the system encapsulated generic product family descriptions, for example design rules, when developing a new product family is also perceived to significantly increase the efficiency in system development. Consequently, there is a need of principles and methods to support capturing and structuring of associated knowledge. In this chapter, a conceptual framework for modelling and management of product knowledge in an engineer-to-order business model is defined.

3.1 Tasks, domains and general enablers

Tasks to be supported can be related to two domains concerning either the family specific building blocks, i.e. rules and CAD-models, or the general building blocks, i.e. functions, UDFs and templates. Three different tasks have been identified as essential to support and these are reuse, expansion and maintenance. Reuse is the use of existing building-blocks in a new context (e.g. a new product family or system foundation). Expansion implies increasing the design space or functionality (e.g. scale the parameters' ranges or extend the topology). Maintenance concerns modifying existing family specific building blocks or general building blocks according to new circumstances (e.g. changes in manufacturing constraints, material properties, manufacturing processes, legislations, standards etc.). In addition to the domains and the tasks identified above, three general enablers for successful task execution are the structuring, the validation and the adaptation of model elements. Structuring is required for the purpose of enabling searching in pursuit of finding candidate building blocks for reuse, expansion or maintenance. Validation is required to ensure the applicability of candidate building blocks. Adaptation is necessary when changes are required to make the selected building blocks applicable in a new context.

3.2 Model elements and knowledge

Initially, the focus will be on the domain of family specific building blocks. Looking at these objects and especially how rules relate to the concept of knowledge, where knowledge is seen as an intentionally defined element that systematically transforms input to output, leads to the conclusion that rules are a kind of knowledge. Commonly, rules implement computations, actions, consequences and relations but they do not encapsulate the argumentation for their existence or the reason behind their design. The definitions of rules are based upon insights, decisions or facts derived from prerequisites, trial and error, experience, calculations, simulations, experiments, filed tests, literature etc. which constitutes another kind of knowledge that can provide a deeper understanding of the rules. A deeper understanding of a rule can be supported by the answers to questions such as: Why, When, Scope, Valid ranges of input/output, Origin, Supporting theories, Simplifications, Assumptions etc. The answers to these questions constitute knowledge about knowledge i.e. Meta-Knowledge and the rules, the UDFs and the parametric CAD-models are different types of Knowledge Objects. In the product development process, knowledge are processed in different steps and appears in different states. To support reuse, expansion and maintenance of different Knowledge Objects (building blocks), it is required that the focus in the product development process is not limited to the definition of Knowledge Objects exclusively, but also includes the definition and collection of associated Meta-Knowledge. Potential and relevant Meta-Knowledge can appear in different objects (e.g. documents, models and items) stored in different locations (repositories). These objects, labelled Meta-Knowledge Carriers, are generated throughout the development process to support the definition of Knowledge Objects. Commonly, there is no mapping between the output from a sub-process and supporting documents, files and items (Meta-Knowledge Carriers) and no description that provides a selection, a context and a meaning to the content of the Meta-Knowledge Carriers in respect to the output from the sub-process. As there is no mapping, there will not exist any traceability between the sub processes: Product Development, Engineering Design and Design Programming. Traceability could be achieved by the introduction of Meta-Knowledge Containers. The Meta-Knowledge Containers would provide mapping of, and meaning to, individual Meta-Knowledge Carriers as depicted in Figure 2.

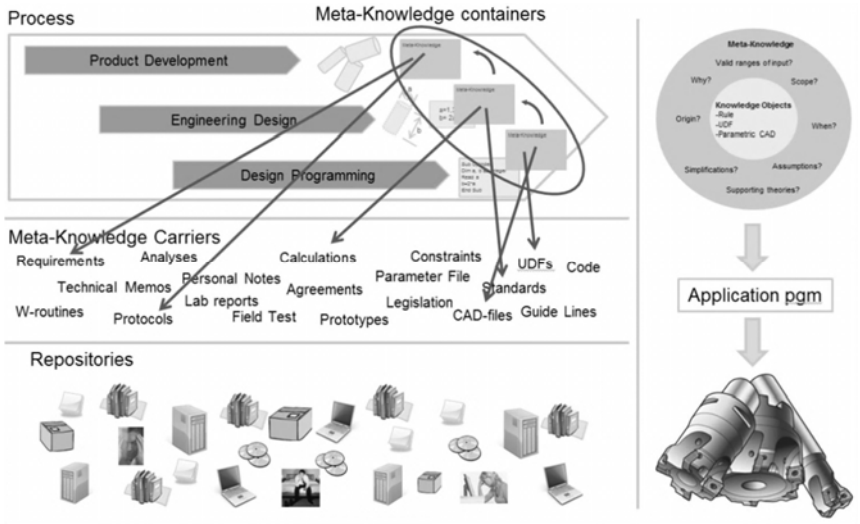

Figure 2. Principle solution with meta-knowledge containers

3.3 Descriptions

In this specific case, the concept of Meta-Knowledge Container was labelled Description. The concepts of Design Definition and Design Rational were also introduced. The main focus of the Design Definition is the construction and the function of a process output object whereas the main focus of the Design Rationale is the argumentation and supporting descriptions unfolding and justifying the object's design. Both the Design Definition and the Design Rationale provides essential meta-knowledge about the process output object and together they constitute the foundation for the Design Description. Three different Design Descriptions related to the three sub processes of the development process are introduced to enable documentation and traceability:

- Product Development, Product Instances Description (PID)
- Engineering Design, Product Family Description (PFD)
- Design Programming, Product Automation Description (PAD)

A fourth Design Description is the Design Module Description (DMD) for general building blocks to be used across product families (e.g. a rule block or a UDF).

In general, the three Descriptions serve as collectors. Main function and properties are:

- Support capturing of Design Definition and Design Rationale
- Links to supporting documents, models and items
- Links to preceding Descriptions
- Written for a clearly defined purpose and potential users
- Based upon templates with predefined headings, keywords and fields
- Simple and visual
- Continuously updated
- Versioning control
- Authorization functions
- Has an owner

The intention with the templates is to facilitate the work of documenting and to support high quality documentation. The content of a Description include, by example, an explanation of the overall product, its building blocks at different levels (e.g. product, assemblies, parts, features and geometrical entities), relations between building blocks (e.g. functional structure and assembly sequence), parameters (input, internal and output), and rules describing the design space. This will constitute the Design Definition of a Description. By adding information and links concerning aspects such as calculations, analyses, field test, underlying principles for design, assumptions, constraints, context, valid ranges of parameters and aspects for validity of rules, together with statements regarding what to consider when changing, ideas not yet implemented and workarounds, the Design Rational of a Description is completed. Means for information representation include tree models, text, illustrations,

pictures, tables, formulas, links and meta-data. An outline of the overall solution aimed at supporting reuse, expansion and maintenance is depicted in figure 3. Process output objects (e.g. Knowledge Objects), PID, PFD, PAD, DMD, Knowledge Carriers (e.g. project documents, models and items), meta-data and links are stored in a database managed by a Database Management System. System functionality includes means to enter, structure, map, store, retrieve, search and visualize information, together with versioning and authorization control and of essential importance is the underlying information model. The actual code is to be divided into collections of statements or objects linked to PAD statements. An individual PAD statement is linked to a PFD statement as well as to applicable Meta-Knowledge Containers. The idea is to work with complete Descriptions to enable completeness, overview and context to support understanding of individual statements but also provide accurate meta-knowledge with high granularity by the subdivision into statement that can be linked.

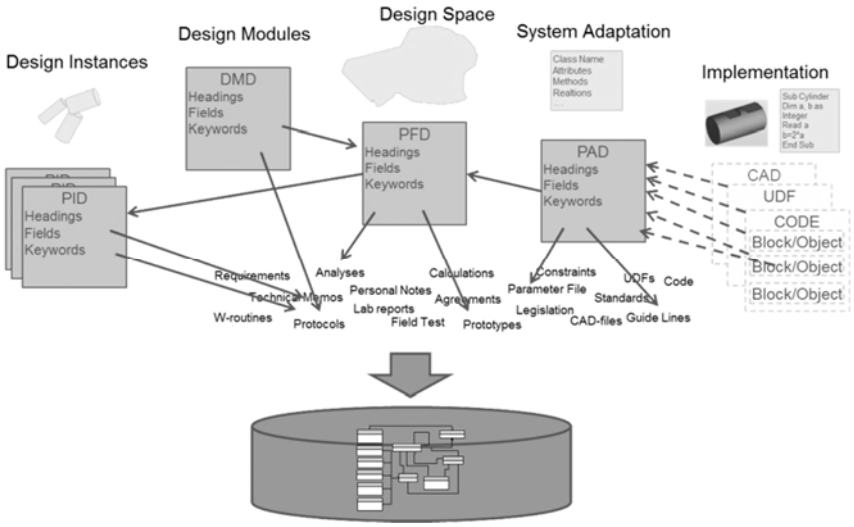

Figure 3. Overall principle solution

4. Review of candidate principles and applications

The framework has been developed in collaboration with industry and is based upon the working practice, encountered problems, identified needs and required functionality of a supporting tool. In this chapter, candidate principles and applications for a system realization, based upon the outlined framework, are surveyed.

4.1 General principles for knowledge modelling

The first candidate principle for knowledge modelling applicable in the domain of design automation systems is the Systems Modelling Language (SysML). SysML is a general purpose modelling language for systems engineering applications. It supports the specification, analysis, design, verification and validation of a broad range of systems and systems of systems. These systems can include hardware, software, information, processes and facilities [Friedenthal et al. 2008]. The language provides graphical representations with a semantic foundation for modelling system requirements, behaviour, structure and parametrics, which is used to integrate with other engineering analysis models. MOKA [Stokes 2001] is a methodology to support the deployment of knowledge based engineering applications. To provide means for developing and maintaining KBE applications and reducing costs, risk and lead time are the main purposes of MOKA. MOKA defines different meta classes to structure the product model. Different meta views are also pre-defined for engineering knowledge modelling of structure, function and behaviour, as well as the relation between them. CommonKADS is a methodology to document knowledge engineering and management. It acts as a baseline for system development and research projects. CommonKADS originates from the need to build industry-quality knowledge systems on a large scale, in a structured, controllable and repeatable way [Schreiber et al. 2001]. CommonKADS has a predefined set of models (organisation, task, agent,

knowledge, communication and design), each of them focusing on a limited aspect, that together provids a comprehensive view. The finale candidate is the Product Variant Master (PVM), an operational tool to model and visualize a product range. In general, a family in PVM can be modelled as [Hvam et al. 2008]: Part-of structure which shows the components included in the product and Kind-of structure that shows the variants available.

4.2 Applications for knowledge modelling

A supporting tool could either be realised by the development of a special purpose application or by the use of an available application with functionality suitable for the purpose. One available application is PCPACK [Epistemics 2008]. PCPACK uses a number of tools that provide user-friendly graphical interfaces to structure knowledge. For example categories, sub-assemblies and sub-components can be shown in PCPACK. Another use of this software is on defining and presenting relationships and properties of pieces of knowledge. Ten tools are defined to make the knowledge modelling more easy and flexible; five acquisition and modelling tools and five specialized tools. In order to re-use knowledge, PCPACK uses XML, which is fully compatible with modern web technologies such as the semantic web that provides a formal, machine-readable content. Design Rationale Editor (DRed) [Bracewell et al. 2009] allows designers to record their design rationale at the time of its generation and deliberation. The design rationale is displayed in a document as a graph of nodes linked with directed arcs. The user creates the nodes by choosing from a predefined set of element types. The functionality is based on four main applications for: diagnosing a problem (problem understanding), designing a solution (solution synthesis), completing a standard checklist template, and communicating the final design and its rationale. Semantic MediaWiki (SMW) [Semantic MediaWiki 2011] is a free open-source extension to MediaWiki that enables querying data within the wiki's pages. The purpose of SMW is to allow users to improve the structure and organization of the knowledge in a wiki by adding simple, machine-processable information to wiki articles. With this additional information, searching, browsing, and sharing the wiki's knowledge can be improved, both within the wiki's pages and from external computer programs.

4.3 Comparison and foundation for system realisation

When comparing the four mentioned principles, SysML seems as a simple way to show the rationale, requirements, constraints and rules by using the concept of the block diagrams, while CommonKADS looks more like a dominant method to manage the knowledge. In CommonKADS, all the information from design to delivery is shown in a simple way. Storing the experience, geometry and data that are related to a product and show them within different classes and views are outstanding for MOKA. When it comes to reducing costs, risks and lead time in a project, providing a way of developing and maintaining KBE makes MOKA more specific. Product variant master (PVM) gives a general overview of the product according to sub or super parts with the relations between different components which all can be seen on a big piece of paper. Regarding the three specific applications, PCPACK has an integrated suite of ten knowledge tools designed to support the acquisition and use of knowledge. Analysing knowledge from text documents and structuring knowledge using various knowledge models makes PCPACK a powerful system. DRed is a simple and unobtrusive software tool that allows engineering designers to record their rationale as the design proceeds. It allows the issues addressed, options considered, plus associated pro and con arguments (arguments for or against an answer), to be captured in the form of a directed graph of dependencies. Improving data structure by using categories and searching specific information according to user's queries are the advantages of SMW and the editing of a published page can be done in the easiest way by SMW.
Based on the required system functionality and the comparison above, the foundation for a system realisation can be outlined. The system has to be able to provide a general view of a product family with all relations and constraints. This is supported by all four general principles, however, additional element has to be added to support structuring of design rationale and relations to other domains and supporting documents. Of great importance is also the functionality and the mechanisms enabling querying or aggregation of information within and across all documentation together with support of versioning and authorization control, this is all supported by SMW.

5. Case study

Information about the case company was gathered by meetings, demonstrations of applications, reviews of documents and in-depth interviews. The result from the case study includes a description of the company's means of providing special products at the same cost as for standard products. A pilot system for documentation and the management of product related knowledge at the company is introduced focusing on system realisation, knowledge representation and system evaluation by company's representatives.

5.1 Business model and means for custom engineered products

The company develops and manufactures products for the mechanical industry. The product catalogues with standard products contain ten thousands of articles. Each individual product structure is not complex but a large number of variants exist and the catalogues contain only the most frequent variants. It is of vital importance for the company to, beside the standard products, provide special products based on different customer demands. These custom engineered products represent an essential part of the delivered products. A request for quotation of a custom engineered product is guaranteed to be replied within 24 hours, including design drawings and a final price. All the necessary documents and manufacturing programs are automatically generated when the bid is accepted. The automated activities include: process planning (workflow in production), design with CAD (3D-models and drawings), production preparation with CAM (tool paths to CNC machines), steer information to production cells, and measuring preparation (creation of programs to CMM machines). The automation of these different activities has resulted in a stream-lined process for quotation and order preparation, Figure 4.

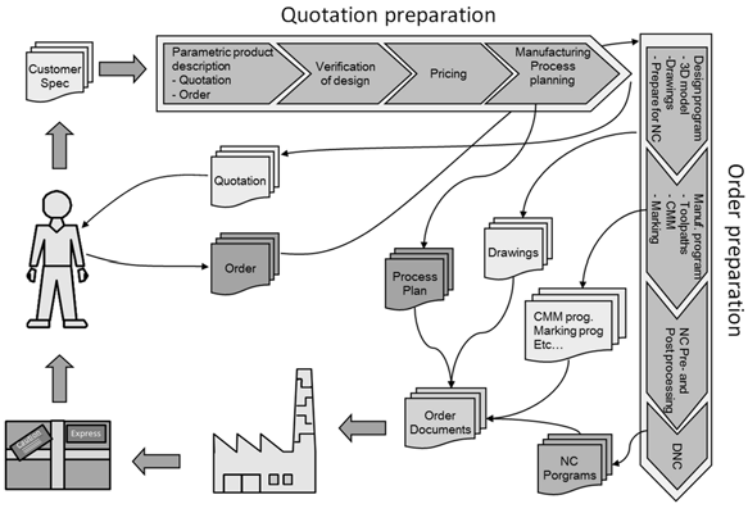

Figure 4. Automated process for quotation and order preparation

5.2 Pilot system realisation and knowledge representation

A system, labelled Design descriptions repository, founded on the presented framework for modelling and management of product knowledge together with the functionality provided by SMW has been developed. A recently develop product family was selected for setting up a PFD. When setting up the PFD, the concept of classes has been used and the product family is explained according to these classes. By example in figure 5, the PFD is described by linking to seven articles. For each article a wiki page is created. The documentation of knowledge relevant for the class is placed within that page. An article can contain supporting documents such as Excel, Word and etc. These can be added to a page by uploading the specific files and then create links to them. Current documentation at the company of product families focuses solely on the design definition and in order to set up a PFD, design rationale should be recorded as well. During several meetings and discussions with the designers, the rationale behind every rule and the knowledge applied were discussed and documented. The design rationale was recorded and then entered for storage in SMW. The information and

knowledge was described by using text, figures, tables, rules, schemas and tree structures. A page describing a component contains both the design definition and the design rationale to form a complete description. The text describes different parameters that are used to design the component. It also includes the principle for designing, the function of the component in the product, the rules and their validity for the product family. It is important to prevent multiple records of the same information and knowledge. For example in the documentation of the test product family, some information, tables or values are general for a range of parts and have previously been stored for each of those parts separately. In order to prevent duplication, documentation can be done in two categories; one as a general category, contained general information which is valid for a range of parts, the second one is a specific category for the knowledge which is valid just for the specific part.

Figure 5. Main page of the Design Descriptions Repository

The representation of knowledge, incorporating both the design definition and the design rationale, in the Design descriptions repository is based on the information model depicted in figure 6, which also act as a template. Of central importance is the Rationale class that connects to all other classes (except PFD), individually or in combinations. The Rationale class also enables specification of relations to Rationale classes in the PID and DMD domain and relations to supporting documentation.

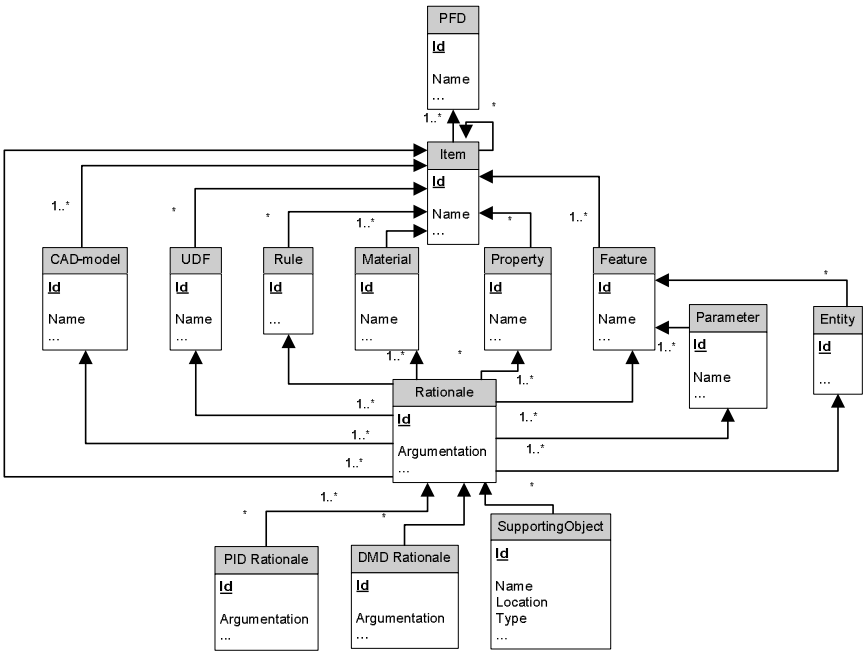

Figure 6. Information model (and template parts) for PFD with relations to PID, DMD and supporting objects (e.g. documents)

5.3 Pilot system evaluation

A final qualitative evaluation was conducted using a questionnaire with open-ended questions, table 1. In summary, both the design automation manager and the project leader believe that PFD will be used at the company. The system developer thinks that PFD is essential for traceability, knowledge reuse and a streamlined parametric and rule-based process. The engineer designer says that PFD is more efficient than the current company documentation. The project leader and the design automation manager consider PFD as good method to provide input to design programming while the system developer thinks it should be mandatory to have a PFD as basis for the design programming. The project leader is not sure how effective the semantic part of SMW will be when properties and categories are manually entered. The system developer has a different opinion, and suggests more work on SMW. The engineer designer sees the advantages with the functionality provided by the search engine and the project leader states that if documentation can be done more efficient, there will be a saving of costs and this is also express by design automation manager. In a further investigation, the design automation manager wants to focus on the way of documenting and storing information in PFD while the project leader prefers to see how the designers can manage PFD in an efficient way. The engineer designer asks for an evaluation other software alternatives for system realisation to support a selection of the most suitable solution according to the company requirements.

Table 1. Pilot system evaluation

Query	Design Automation Manager	Project Leader of Documentation Project	System Developer	Engineering Designer
1. What is your opinion about PFD?	PFD is great. That's a really good definition of our documentation at the company.	Great! I like the concept and will work for a PFD solution at the company.	The PFD is essential for traceability, knowledge reuse and a streamlined parametric and rule-based process	The current documentation at the company describes our product families. PFD is perhaps a better description for that in the future. Also if we can implement the rationale part of this in an easy way it could be great.
2. What is your opinion regarding PFD as an input for design programming?	PFD is excellent. If it's also divided in DMD, it's easy to reuse and gets the needed knowledge for a family.	I think it will be great.	It should be mandatory to have a PFD as basis for the design program.	If the PFD is created during PDP in close cooperation with design automation it generates a good input for the design programing.
3. How do you think about making structured documentation based on categories?	It can be a way. It's like a template (for me).	If it would be possible to have it more in forms then it would be better. I mean that the properties and categories should have some kind of automatic handling.	Properties is one way of making the information structured. A structured information model is essential to leverage reuse of knowledge. By having a structured information model it is possible to classify and search the information in a relevant way.	You should always strive to make the documentation in a structured way. It could be difficult to strictly document based on properties and categories. During documenting a product family I think you have to make sure that there is a good overview what is included in that particular family.
4. How do you think properties and categories can provide access to the stored knowledge?	It makes the documentation searchable.	Implementing in the right way would be helpful for information finding. But see question 3.	See answer for question 3.	When designing a new product you can look for what is already made in a certain area. Can you reuse it?
5. What is your opinion about making documentation in the pre-defined templates?	I like that.	Great, I like it a lot and we will use templates in the future. In which way depends on the system solution.	This is part of an information model. The template represents the class of the information and the document based on a template is an instance of that class. I see template as a contract or interface for consumers of the information (and by consumers I mean humans or other systems).	With pre-defined templates you are helped during making documentation, easy to see what is expected from you. For new designers it should be great, and also the format for the documentation will be more streamlined.
6. How can the results of the project be used at the company?	We will implement PFD and DMD. Probably not the other, at least not now. We have to see what our RBD project comes up with.	PFD will definitely be used. SMW is under consideration but mostly without the semantic part.	The important result of the work is to emphasize on the need to classify information. I haven't seen enough of SMW to have a well-founded opinion about it, but as I see the future of PFD's and design rule documentation I would say that SMW does not suffice. In the long term the PFD recording application should support the creation and versioning of runnable design rules and that is not possible with any known system today. But the technology is available for creating such a system and I think that should be the long term goal.	It can be a source of information and insight during the future work within rule based design at the company.
7. What are the drawbacks of the project results?	I don't see any drawbacks. Although I would like to have more focus on DMD as a part of the whole Description parts. For me it feels that we can document Design Modules separately and by inheriting of various PFDs. That's a way of reusing knowledge.	Maybe should have been considered having look at SharePoint.	-	If there had been more time perhaps looked into other software and evaluated those as comparison with the ones chosen in the project Also look into more product families.
8. What is needed for further investigation?	We will investigate a way (system?) to document and store documentation about PFD.	How can the designer handle the PFD in the most efficient way?	See answer for question 6.	To look into more software alternatives and decide most suitable for our requirements.

The focus of this work is on structuring of knowledge, including design rationale and a support for traceability, with the objective to enable reuse, expansion and maintenance of generic product family objects embedded in design automation systems. The major advantages are: the inclusion of design rational (with a high level of granularity), traceability to other domains and supporting documents, and the possibility to query or aggregate the stored knowledge. History and authorization control are also supported by the pilot system. At the company, standardisation of structuring product family knowledge is reached together with a reduction in the effort of publish it on the intranet.

6. Conclusion

The main objective of this work was to develop, implement and evaluate a system foundation for modelling and management of product knowledge supporting reuse, expansion and maintenance of design automation system embedded generic product family objects. One of the central parts of the framework is the Meta-Knowledge Containers, labelled Descriptions for the specific application in this paper. Descriptions are to be created for identified sub-process and delivered together with the main output and should contain both the definition of the output as well as the rationale behind its design. Traceability is supported by linking rationale between Descriptions and to Meta-Knowledge Carriers. A pilot system focusing on the Product family Description (PFD) has been develop in collaboration with a company. Company representatives confirm the applicability and usefulness of the proposed approach in general but also stressed the need for further investigations of its applicability at the company and of supporting tools for efficiently feeding a system with information and knowledge of sufficient amount and of right level of quality. To fully validate the presented explorative work and its feasibility requires studies on a large scale system. Another question, which has not been in the scoop of this work, is if the representation can support the knowledge capturing that is part of the process of transforming PID to PFD. This is two directions for future work.

Acknowledgement

This work was conducted within a Swedish Knowledge Foundation granted project and financial support is gratefully acknowledged. The author would also like to express his gratitude to the company for providing information and knowledge as well as for collaboration and helpful discussions.

References

Bracewell, R., Wallace, K., Moss, M., Knott D., "Capturing design rationale", Computer Aided Design, 41(3), 2009, pp.173-186

Duffy, A., Andreasen, M.M., "Enhancing the evolution of design science". In International Conference on Engineering Design 1995, Vol. 1, Heurista, Zürich, Schweiz, 1995, pp. 29-35

Epistemics, "PCPACK", http://www.epistemics.co.uk/Notes/55-0-0.htm (Acc. 9 December 2011), 2008

Friedenthal, S., Moore, A., Steiner, A., "A practical guide to SysML: the Systems Modeling Language", Morgan Kaufmann, San Francisco, US, 2008

Hvam, L., Mortensen, N.H., Riis, J., "Product customization", Springer Verlag, Berlin, Germany, 2008

Moham,K., Ramesh, B., "Traceability-based knowledge integration in group decision and negotiation activities", Decision support systems, 43, 2007, pp. 968-989

Schreiber, G., Akkermans, H., Anjewierden, A., Hoog, R., Shadbolt, N., Velde, W., "Knowledge engineering and management: The CommonKADS methodology", The MIT Press, Cambridge, US, 2000

Semantic MediaWiki, "Introduction to Semantic MediaWiki", http://www.semantic-mediawiki.org (Acc. 9 December 2011), 2011

Stokes, M., "Managing engineering knowledge – MOKA", Prof Eng Publications ltd, London, UK, 2001

Tan, A., Jin, Y., Han, J., "A rational-based architecture model for design traceability and reasoning", The journal of systems and software, 80, 2007, pp. 918-934

Fredrik Elgh PhD
Accociate Professor Product Development
Jönköping University, School of Engineering, P.O.Box 1026, SE-551 11Jönköping, Sweden
Telephone: +46 36 101672
Email: fredrik.elgh@jth.hj.se
URL: http://www.hj.se/jth