INTERNATIONAL DESIGN CONFERENCE - DESIGN 2014
Dubrovnik - Croatia, May 19 - 22, 2014.

SEMANTIC SUPPORT FOR ENGINEERING DESIGN PROCESSES

T. Breitsprecher, M. Codescu, C. Jucovschi, M. Kohlhase, L. Schröder
and S. Wartzack

Keywords: design knowledge, ontology, semantics in product development

1. Introduction

The engineering design of mechanical products is a multi-stage process, and is formally described as such by a range of product development process models and methodologies. The *Systematic approach to Engineering Design* of Pahl et al. [2007], the *Münchener Vorgehensmodell* of Lindemann [2009], the *Systematic Approach to the Design of Technical Systems and Products* according to the guideline VDI2221 [VDI 1995] and the *Design Methodology for Mechatronic Systems*, also known as the V-model [VDI 2004] are just a few well-known examples. For some specific stages within such product development process models, computer-based approaches have been developed to support the design engineer, such as Computer-Aided Design (CAD), Computer-Aided Engineering (CAE) or Computer-Aided Manufacturing (CAM). These approaches are based on formal (i.e. machine-interpretable) representations of the outcomes of the relevant development stage, such as CAD-models, FEA-models (CAE), or CNC-code. However, other recognized development stages and their associated documents such as requirement lists, function models or the principle solution are left largely informal and in fact are often not laid down in any immediately machine-processable form (being, e.g., just hand drawn sketches). This circumstance leads to gaps regarding the semantic links between stages of the product development process and related documents and objects. These links embody questions such as "*Does the embodiment X fulfil the requirement Y?*" or, as a more specific example, "*Does this shaft-hub connection still correspond to the predetermined principle solution and function structure?*". Little or no machine-support is currently available for verifying or even just managing such consistency assertions in the development process.
This contribution is the result of a research collaboration between the School of Engineering and Science (Jacobs University Bremen), the Chair of Theoretical Computer Science and the Chair of Engineering Design (both Friedrich-Alexander Universität Erlangen-Nürnberg). In the current work, the authors propose a semantic approach where objects are linked across the stages of the product development process using a federated ontology, in which all objects are grounded via annotations with ontological concepts and assertions. The approach is embedded into a document-oriented design work flow, in which the federated ontology and semantic annotations in design documents are exploited to trace parts and requirements through the development process and across different applications. Requirements and ensuing design decisions are made explicit and, hence, available for further machine processing at all stages of the development. In the proof of concept for this approach, a sample requirement is traced through the development of a spring tester.
After a short description of a common product development process model and exemplary related documents, the state of the art in the application of semantic technologies in engineering design is discussed. Subsequently, the overall approach is presented, and layed out in detail how semantic

services within the product development process are enabled via the Multi-Application Semantic Alliance Framework (MASally). To illustrate the added value of this approach, the spring tester case study is described in detail.

1.1 The Systematic Approach to the Design of Technical Systems and Products

The Association of German Engineers (VDI) published the first sversion of the guideline VDI2221 in 1993 to replace the guideline VDI2222. The stages of the engineering design process according to the VDI2221 [VDI 1995] are recalled:

1. **Clarify and define tasks:** A concise formulation of the purpose of the product to be designed. The step results in the *requirement list*.
2. **Determine functions and their structure:** Based on the requirements, one determines sub-functions to be performed for specific tasks; these sub-functions are combined in the *function structure*.
3. **Search for solution principles and their combinations:** Effects (e.g. physical or chemical) that can fulfil a sub-function are determined in the form of working principles. These are combined to yield the overall *principle solution*.
4. **Divide into realizable modules:** Working from the principle solution, one identifies subsystems, groupings, and interfaces to guide the detailed design, thus determining the *module structure*.
5. **Develop layout of key modules:** For each module *preliminary designs* are created, assessed and released.
6. **Complete overall layout:** The preliminary layouts of the modules are completed by the addition of further details and by adding the layout of components not included in the previous step. The combination of all assemblies and components then leads to the *definitive layout*.
7. **Prepare production and operating instructions:** All documents that are necessary for the manufacturing and assembly of the product are created. This *product documentation* includes e.g. technical drawings, assembly or usage instructions.

In all these stages, several variants of a solution may be analysed and, where necessary, tested by means of virtual or physical prototypes, which are then evaluated. Note that these stages can be further subdivided, depending on the complexity of each task. Furthermore the stages do not necessarily follow a fixed procedure. Rather, they are carried out iteratively, with jumps forward and backward, thus achieving a step-by-step optimization.

The overall approach is embedded into a document-oriented work flow based on documents arising in the development process as indicated above; Table 1 gives an overview of the most important document types.

Table 1. Different types of document and their occurrence within product development

	Stage	Result	Data type and content
1	Clarify and define tasks	Requirement list	• Research results, client surveys, requirements, technical offerings, etc. • Open text, tables, PDF, pictures
2	Determine functions and their structure	Function structure	• Function diagrams, • Mind maps, graphs, sketches, ontologies, function trees
3	Search for solution principles and their combinations	Principle solution	• Data in context of idea search and assessment • Sketches, tables, mind maps
4	Divide into realizable modules	Module structure	• Rough structure of product, arrangement of components • CAD-file with product skeleton

5	Develop layout of key modules	Preliminary layouts	• Detailed CAD files, data from simulations (e.g. FEA), calculations for dimensioning • Proprietary and neutral CAD- or FEA-files, calculation documents
6	Complete overall layout	Definitive layout	• CAD-files of standard part catalogues, calculations for recalculation of final design • Proprietary and neutral CAD- or FEA-files, calculation documents
7	Prepare production and operating instructions	Product documentation	• Manufacturing drawings, bill of material, meta data for interfaces • CNC files, XML files

2. Semantics and ontologies in Engineering Design

Various approaches have been explored to integrate semantics into the engineering design process. One such approach is the so-called feature technology, which has been researched by several institutes. According to VDI2218 [VDI 2003], features are an aggregation of geometry items and semantics. Different types of features are defined (e.g. form features, semantic features, application features, compound features), depending strongly on the technical domain and the product life-cycle phase in which features are used. Features are to be expected to play a role in further semanticizing step **S6** (embodiment, Section 1.1) in future work.
Li et al. [2013] have developed an ontology-based annotation approach to support multiple evaluations of computer-aided designs, especially in later phases of the design process. The annotation data are contained within a consistent three-layered ontology framework that supports the integration of multiple specialist viewpoints by associating annotation content with anchors in a boundary representation model. The ontology also allows checking of data structures and other reasoning.
Several ontologies in the field of CAD have been developed, with the typical application scenario being interoperability and data interchange between CAD systems, rather than verification of qualitative properties of CAD assemblies. E.g., OntoSTEP [Barbau et al. 2012] enriches the semantics of CAD objects represented in the system-independent ISO-standard interchange format STEP. The heterogeneous approach in the present contribution allows integrating OntoSTEP (or any other ontology of features) into the federated engineering ontology and relating it to the ontology of features, without having to modify the verification methodology.

3. Semantic support of a document-oriented engineering design workflow

The documents generated for each development stage are initially related only in the mind of the engineer. These ties can be made explicit, so that computers can act on them, by annotating parts of these documents with concepts in ontologies. Such annotations (depicted with dashed arrows in Figure 1) can express simple facts e.g. that F_{hand} (from requirements list) is the hand force (from domain ontology) with which a user can interact with a product and constraints like $0N \leq F_{hand} \leq 200N$ for this force (see Section 5 for details). Depending on the complexity of the statement that needs to be made explicit, different logics can be used. This feature is enabled by federated ontologies (the cloud in Figure 1) – a method of combining heterogeneous ontologies by meaning-preserving interpretations [Rabe and Kohlhase 2013].

3.1 Semantic Annotations in Design Documents

Semantic annotations are also used to relate objects from different design stages e.g. the gear nut from the CAD model in the complete overall layout stage can be annotated as a refinement of the gear nut object from the principle solution. Most software products do not, by default, support the user in creating/updating semantic annotations. However, product dependent extensions may enable users to create/update such annotations.
AktiveMedia [Chakravarthy et al. 2006] is used for annotating images like the principle solution (see Figure 2). As other semantic authoring solutions for word processing documents that fit the needs of

this research could not be found, *sTEX* was used – a semantic extension of the *TEX/LATEX* format. To annotate CAD documents, the ability of the CAD environment (Autodesk Inventor) was used to associate custom information to CAD objects and hence no specialized solution had to be used.
As annotations are assigned to parts of documents (e.g. some character range or assembly part), change management services can approximate how changes made to the document a ect the semantics of these annotations. This allows services to perform impact analysis and management of change.

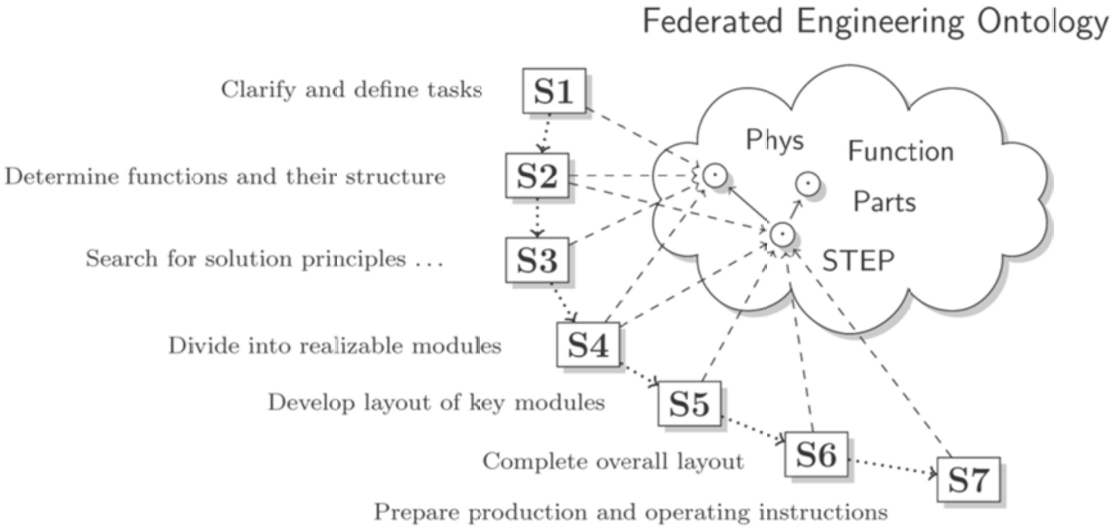

Federated Engineering Ontology
Clarify and define tasks
S1
Determine functions and their structure
S2
Search for solution principles ...
S3
Divide into realizable modules
S4
Develop layout of key modules
S5
Complete overall layout
S6
S7
Prepare production and operating instructions
Phys
Function
Parts
STEP

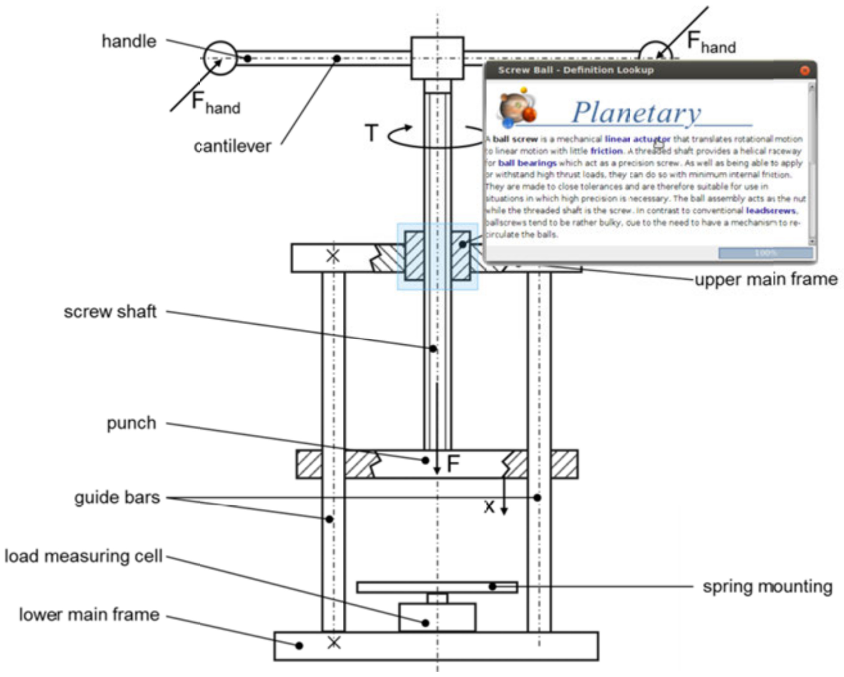
Figure 1. An ontology-supported document-oriented design process, in accordance to Breitsprecher et al. [2013]

3.2 Semantic services via the MASally System

The Multi-Application Semantic Alliance Framework (MASally) is a semantic middleware that allows embedding semantic interactions into (semantically preloaded) documents. The aim of the system is to support the ever more complex work flows of knowledge workers with tasks that so far only other humans have been able to perform without forcing them to leave their accustomed tool chain.
The MASally system is realized as:

- a set of semiformal knowledge management web services (comprised together with their knowledge sources under the heading Planetary on the right of Figure 3);
- a central interaction manager (Sally, the semantic ally) that coordinates the provisioning and choreographing of semantic services with the user actions in the various applications of her workflow;
- and per application involved (see CAD system and document viewer for **S4/S5** in Figure 3)
  - a thin API handler Alex that invades the application and relates its internal data model to the abstract, application-independent, content-oriented document model in Sally;
  - an application-independent display manager Theo, which super-imposes interaction and notification windows from Sally over the application window, creating the impression the semantic services they contain come from the application itself.

This software and information architecture is engineered to share semantic technologies across as many applications as possible, minimizing the application-specific parts. The latter are encapsulated in the Alexes, which only have to relate user events to Sally, highlight fragments of semantic objects, handle the storage of semantic annotations in the documents, and export semantically relevant object properties to Sally. In particular, the Theos are completely system-independent. In the author's experience developing an Alex for an open-API application is a matter of less than a month for an experienced programmer; see [David et al. 2012] for details on the MASally architecture.



A **ball screw** is a mechanical **linear actuator** that translates rotational motion to linear motion with little **friction**. A threaded shaft provides a helical raceway for **ball bearings** which act as a precision screw. As well as being able to apply or withstand high thrust loads, they can do so with minimum internal friction. They are made to close tolerances and are therefore suitable for use in situations in which high precision is necessary. The ball assembly acts as the nut while the threaded shaft is the screw. In contrast to conventional **leadscrews**, ballscrews tend to be rather bulky, due to the need to have a mechanism to re-circulate the balls.

Figure 2. Principle Solution of the Spring Tester with Definition Lookup

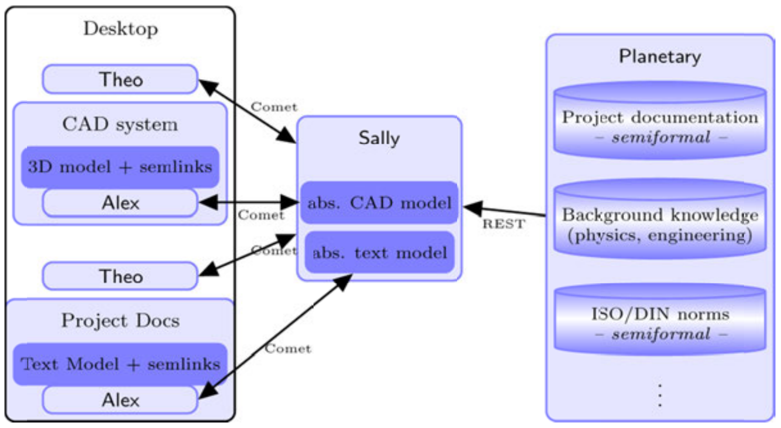


Figure 3. The MASally Architecture, in accordance to Breitsprecher et al. [2013]

To fortify the intuition about semantic services, the following situation is considered: The design engineer is working on the principle solution from Figure 2 – a sketch realized as a vector image, displayed in an (in this case browser-based) image viewer. The user clicked on a detail of the sketch and received a (Theo-provided) menu that:

1. identifies the object as “ScrBall4” (the image is extended with an image map, which allows linking the region “ScrBall4” with the concept of a “screw-ball” in the ontology); further information about the object can be obtained by clicking on this menu item;
2. gives access to the design refinement relation between the project documents: here, the object “ScrBall4” is construed as a design refinement of the requirement “ScrBall2” from the project requirements and has been further refined into object “ScrBall6” in the CAD assembly and the plans generated from that;
3. gives access to the project configuration that identifies the other documents in the current design;
4. allows direct interaction with the ontology (e.g. by definition lookup; see Figure 2, here triggered from the CAD system for variety);
5. gives shortcuts for navigation to the other screw balls in the current project.

Generally, the MASally system supports generic help system functionalities (definition lookup, exploration of the concept space, or semantic navigation: lookup of concrete CAD objects from explanations) and allows focus-preserving task switching (see [Kohlhase A. et al. 2013] for a discussion). All that is needed for this are annotations of the VDI2221 relations, ontology links and of course the ontology itself, which is discussed next.

4. The Federated Engineering Ontology (FEO)

The design of the ontology that acts as the central representation of the background knowledge and the common ground of all actors in the design process is discussed. It serves as a synchronization point for semantic services, as a store for the properties of and relations between domain objects, and as a repository of help texts for the MASally system. The backbone of the federated ontology is provided by *flexiformal documents* consisting of concept definitions and statements of properties of the objects described using these objects. The statements are given in natural language and are interspersed with formulas. Furthermore, the federated ontology contains formal ontologies that enable verification of properties between different stages of design.

For further information on the design or content of the FEO, the reader can refer to [Breitsprecher et al. 2013]. However, it is noted that a document is called flexiformal, if it contains material at different levels of formality [Kohlhase M. 2013], ranging from fully informal – and thus foreign to machine support – text via text annotated with explicit semantic relations – i.e. open to semantic services – to fully formal – i.e. expressed in a logical system, which supports machine inference and thus verification of constraints – content. As the FEO has to cover quite disparate aspects of the respective engineering domain at different levels of formality, it is unrealistic to expect a homogeneous ontology in a single representation regime. Instead, the heterogeneous OMDoc/MMT framework [Kohlhase M. 2006], [Rabe and Kohlhase 2013] that allows representing and interrelating ontology modules via meaning-preserving interpretations (i.e. theory morphisms) is utilized.

As an example of formal ontologies, as detailed in [Breitsprecher et al. 2013], OWL ontologies are built for stating qualitative properties (e.g. abstract geometrical properties, but also function and behaviour or parts could be included) and for representing a CAD model as an assembly of parts built using features, according to its history of construction. A further OWL ontology of rules regarding geometrical properties of objects is built by repeated applications of features and thus enables verification of these properties for actual designs. The formal ontologies are related to the backbone flexiformal ontology by theory morphisms. A similar approach can be pursued to obtain tool support for checking that other steps of the design process are correct, e.g. that the principle solution fulfils the functions specified in the function structure.

5. Case study

The case study is based on a spring tester that can measure the spring force of cylindrical compression springs at a given deflection. This example was previously used in practical design assignments for engineering students. Students were given a principle solution (see Figure 2) for the spring tester (stage **S3** of the design process in Figure 1) along with a requirements specification and a function structure (stages **S1** and **S2**), and were asked to design an embodiment, i.e. to complete stages **S4** to **S7** of the design process.

The requirement specification (**S1**) states the goal of creating a spring tester to determine the spring rate of cylindrical compression springs according to a specific norm. The device has to be hand-driven, where it is assumed that a normal person can act with a hand force of approximately 200 N per hand. During the measurement, the spring is to be compressed by 5mm. The resulting force is detected via a suitable load cell. In order to avoid distorting the force measurement, the spring tester must not exceed a critical elastic deformation of half of the load cell's accuracy class.

In this case study, documents from stages **S1-S3** were annotated, which were produced by faculty members, and two sets of documents from stages **S4-S7** produced by students – the intension is to study supporting design alternatives. As these documents come from an ongoing educational process, the annotation was necessarily post-mortem; experiments with integrating the described methods in a live development process are the subject of future research. In particular, a focus is put on the following services: *i) definition lookup* for document elements (see Figure 2), *ii) topical navigation* among documents in different development stages along the refinement relations (see Figure 4), and *iii) propagation of change impacts* by highlighting document elements in other documents that would need to be revised (see Figure 6).

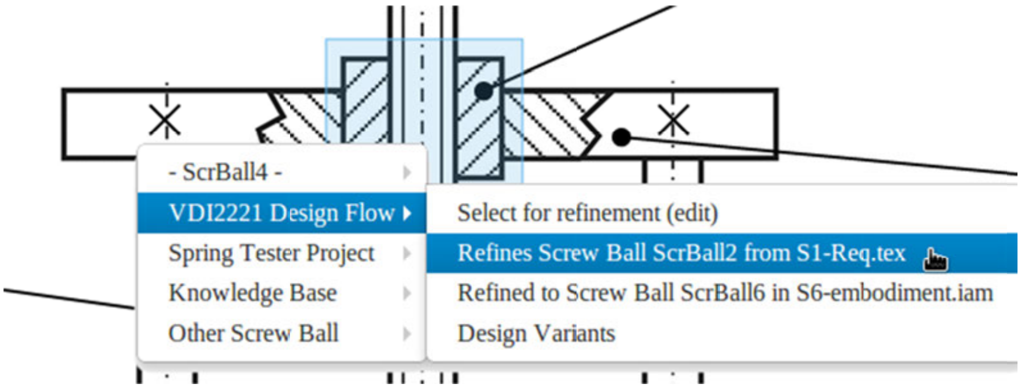

- ScrBall4 -
VDI2221 Design Flow
Spring Tester Project
Knowledge Base
Other Screw Ball
Select for refinement (edit)
Refines Screw Ball ScrBall2 from S1-Req.tex
Refined to Screw Ball ScrBall6 in S6-embodiment.iam
Design Variants

Figure 4. Navigating the Refinement Relation

Note that these services can only work if explicit semantic annotations in the documents exist. For instance, for definition lookup, documents with concepts in the ontology had to be annotated. For text documents, that meant using the *sTEX* macros that link text fragments with ontological concepts. From these a pdf/html document can be generated where a piece of text, e.g. "screw nut", is annotated with a suitable ontology term such as screw-nut (internally represented as a URI). The use of *sTEX* serves to provide a proof-of-concept; current work focuses on the integration of semantic annotation functionalities into more widely used document preparation systems. It is also noted that many of these annotations could be semi-automatically detected using the NNexus [Ginev 2013] framework in the future; the integration of this framework with the current approach is ongoing.
Annotating relations among different document was done using the MASally frames. The same document parts that were annotated for definition lookup were used and enriched with additional relations such as "X refines Y", meaning that a document element Y (usually from a previous development stage) was refined at a later stage of development by document element X (see Figure 4).
For impact analysis, the documents were enriched with more domain specific annotations. Consider the requirement:

"Max hand force $F_{hand} = 200$ N" (*)

from **S1**. In the following design steps the user annotates all artifacts in S2-S7 that is influenced by this requirement with a refinement link to (*). For the function structure (**S2**; see Figure 5) these include the sub-functions "Induce F_{hand}" (**) or "Amplify F_{hand}" (***). In the principle solution **S3** the handle of the spring tester ("Induce F_{hand}") and the cantilever between the handle and the spindle ("Amplify F_{hand}") are annotated as refinements for (**) and (***).
S4 concerns the modular structure of the design, it identifies the main design-affecting requirements and creates a first CAD-file which represents the main dimensions by limiting reference elements (points, lines, planes). The exemplary requirement influences, for example, the length of the cantilever, because the hand force has to be transformed into a torque (according to the law of the lever) to deflect a spring via the screw spindle. This (canti-)lever length can be represented within the CAD-file by a reference line.
Step **S5** is quite extensive because preliminary designs have to be created and assessed. As shown in Table 1 these assessments include the dimensioning calculation of key modules. These calculations can either be done manually on paper or digitally via text/table files. In this case study created MathCAD® files were created and the formulas within were annotated. Here, F_{hand} was used to calculate dimensions of key modules. An example is the cantilever diameter, which must be sufficient to withstand the bending moment that is caused at the fixing point to the spindle.
At the end of stage **S6**, the CAD-model of the spring tester, consisting of parts, subassemblies and the main assembly, was finalized and annotated. Annotations were assigned both to parts and assemblies,

depending on the purpose of the annotation. The part model *cantilever.prt* was annotated and linked with the requirement $F_{hand} = 200N$ and thus the cantilever diameter in the CAD-model was linked with the initial requirement.

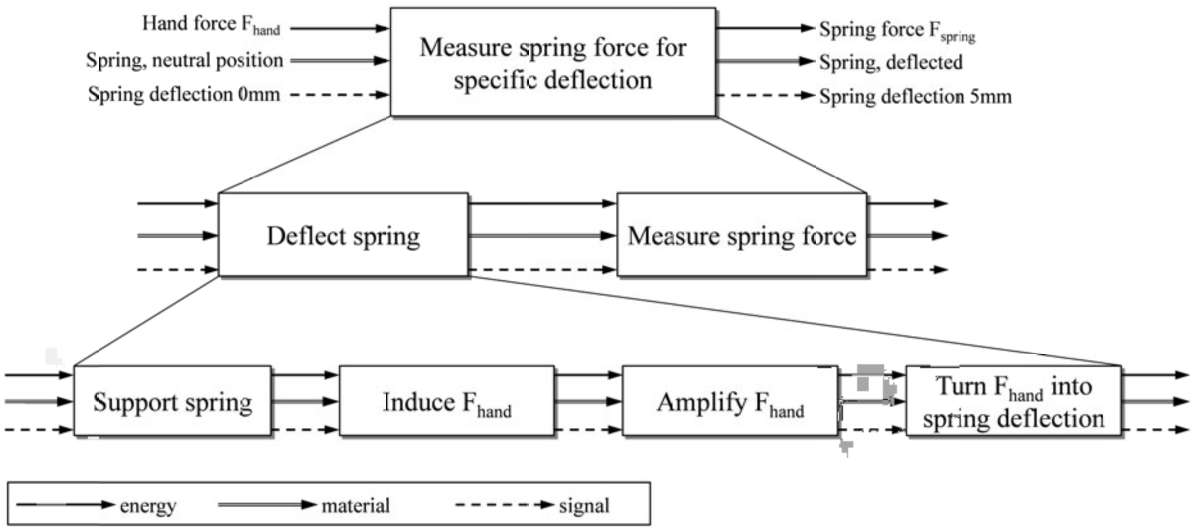
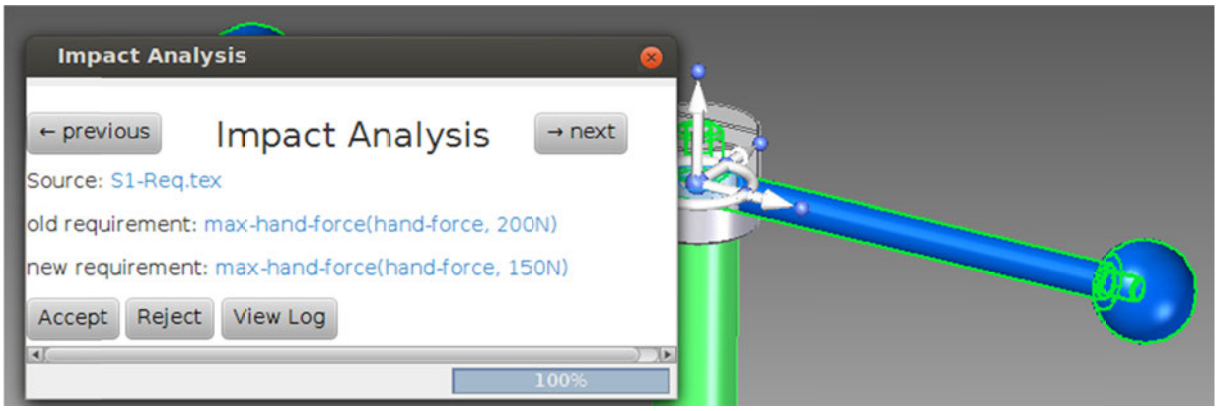


Figure 5. Function structure of the spring tester

The utility of an impact management service is at hand: assume the spring tester is to be changed so that it can be used by people with a decreased hand force of $F'_{hand} = 150N$ (e.g. for a different market). The change impact service shown in Figure 6 highlights the handle of the spring tester – as it is (annotated to be) a refinement of the changed requirement in **S1** – clicking it results in a popup that details the root changes and their influences. The "next/previous (conflict)" buttons are another instance of a semantic interaction (navigation to the next affected part in this document, later even across documents) which supports the change management work flow of the engineer.
Even in the final step **S7** (product documentation) semantic annotations are helpful. For instance, a service that justifies the spindle pitch for the trapezoid spindle embedded in the manufacturing drawing helps avoid questions in the manufacturing process, since a change of the spindle pitch affects the torque that is to be provided via hand force.


Impact Analysis

Impact Analysis

Source: S1-Req.tex

old requirement: max-hand-force(hand-force, 200N)

new requirement: max-hand-force(hand-force, 150N)

Figure 6. Impact propagation/resolution for changes to the hand force requirement

6. Summary and outlook

While the final stages of the engineering design process have well-established information-technological support in the shape of modern CAD/CAE/CAM systems, tool support for earlier stages of the process (e.g. requirements, function structure, principle solution) is less well-developed, regarding the use of semantic services. Above, a framework for pervasive semantic support in engineering design processes is described, as part of a program to unify the underlying tool chain and enhance tool support in all stages of the design process. The framework is based on a flexiformal background ontology that combines informal and semiformal parts serving informational purposes with fully formalized qualitative engineering knowledge and support for the annotation of design documents (e.g. specifications, principle sketches, embodiments, documentation) with formal qualitative constraints. In the current work, a focus was put on a document-oriented work flow that relies on the background ontology for tracking the identity of parts through the design process and across different applications, which are accessed in a unified manner within the MASally framework. In complementary work it was shown how the approach can be augmented to enable *automated* requirements tracing and verification of the CAD model against aspects of the principle solution (see [Breitsprecher et al. 2013] for details).

The approach was exemplified on the development process of a spring tester, illustrating MASally support for semantic navigation between the various design documents and for tracing and testing requirements. The flexiformal nature of the federated engineering ontology governing the annotation of the design documents makes the integration of formal and informal approaches feasible.

The federated engineering ontology is under continuous development. It will be further integrated with established domain ontologies including geometric ontologies (whose development is an active research field in its own right, see, e.g., the Shapes workshop series [Kutz et al. 2013]), CAD feature ontologies (e.g. [Brunetti and Grimm 2005], [Abdul-Ghafour et al. 2007]), ontologies of function (e.g. [Colombo et al. 2007]), and repositories of standard parts, using modularity mechanisms enabled by modern logical frameworks such as Distributed Ontology, Modeling and Specification Language DOL [Mossakowski et al. 2013a], [Mossakowski et al. 2013b]. Moreover, its base of formalized engineering knowledge will be broadened; the associated knowledge acquisition process is an important aspect of further investigation. In future works the efficiency of the FEO will be proven, for example by different groups of engineering design master students, whereas a control group will be able to use the presented approach including the FEO. In proportion to the degree of formalization of the underlying engineering knowledge, the potential for automated verification of later stages in the design process against requirements formulated in earlier stages increases, as illustrated in [Breitsprecher et al. 2013].

One major impediment to employing the semantically enhanced work flow described here in industrial applications is the fact that currently the only annotation system for the documents is a variant of TEX/LATEX, which is not commonly used by engineers. The choice of *sTEX* for this purpose is based purely on availability, and currently the authors are working on Alexes for various word processors (primarily MS Word and OO Writer).

Acknowledgement

The authors thank Oliver Kutz and Till Mossakowski for fruitful discussions. The work presented in this paper was supported by the German Research Foundation (DFG) under grant KO-2484/12-1 / SCHR-1118/7-1 (FormalCAD).

References

Abdul-Ghafour, S., Ghodous, P., Shariat, B., Perna, E., "A common design-features ontology for product data semantics interoperability", Proceedings of the IEEE/WIC/ACM Int. Conf. on Web Intelligence, Washington DC, USA, 2007, pp. 443-446.

Barbau, R., Krima, S., Sudarsan, R., "OntoSTEP: Enriching product model data using ontologies", CAD, Vol. 44, 2012, pp. 575–590.

Breitsprecher, T., Codescu, M., Jucovschi, C., Kohlhase, M., Schröder, L., Wartzack, S., "Towards ontological support for principle solutions in mechanical engineering", <http://arxiv.org/abs/1312.2359>, CoRR, 2013.

Brunetti, G., Grimm, S., "Feature ontologies for the explicit representation of shape semantics", Journal of Computational Applied Technology, Vol. 23, 2005, pp. 192-202.

Chakravarthy, A., Ciravegna, F., Lanfranchi, V., "Aktivemedia: Cross-media document annotation and enrichment", Proc. of 1st Intern. Workshop on Semantic Web Annotations for Multimedia (SWAMM-06), Edinburgh, Scotland, 2006.

Colombo, G., Mosca, A., Sartori, F., "Towards the design of intelligent cad systems: An ontological approach", Advanced Engineering Informatics, Vol. 21, No. 2, 2007, pp. 153-168.

David, C., Jucovschi, C., Kohlhase, A., Kohlhase, M., "Semantic Alliance: A framework for semantic allies", Intelligent Computer Mathematics, Jeuring, J., Campbell, J. A., Carette, J., Dos Reis, G., Sojka, P., Wenzel, M., Sorge, V. (Ed.), No. 7362, 2012, pp. 49-64.

Ginev, D., "Nexus Glasses: a drop-in showcase for wikification", Proceedings of MathUI, OpenMath, PLMMS and ThEdu Workshops and Work in Progress at the Conference on Intelligent Computer Mathematics, Lange, C., Aspinall, D., Carette, J., Davenport, J., Kohlhase, A., Kohlhase, M., Libbrecht, P., Quaresma, P., Rabe, F., Sojka, P., Whiteside, I., Windsteiger, W. (Ed.), Aachen, Vol. 1010, 2013.

Kohlhase, A., Kohlhase, M., Jucovschi, C., Toader, A., "Full semantic transparency: Overcoming boundaries of applications", Kotzé, P., Marsden, G., Lindgaard, G., Wesson, J., Winckler, M. (Ed.), Human-Computer Interaction, Vol. 8119, Springer, 2013, pp. 406-423.

Kohlhase, M., "OMDoc - An open markup format for mathematical documents", No 4180, Springer, 2006.

Kohlhase, M., "The exiformalist manifesto", 14th International Workshop on Symbolic and Numeric Algorithms for Scientific Computing, Voronkov, A., Negru, V., Ida, T., Jebelean, T., Petcu, D., Watt, S. M., Zaharie, D. (Ed.), 2013, pp. 30-36, in press.

Kutz, O., Bhatt, M., Borgo, S., Santos, P., "The Shape of Things", SHAPES 2013, volume 1007 of CEUR Workshop Proceedings, 2013.

Li, C. L., McMahon, C., Newnes, L., "Supporting multiple engineering viewpoints in computer-aided design using ontology-based annotations", Proceedings of 19th International Conference on Engineering Design, Lindemann, U., Venkataraman, S., Kim, Y. S., Ion, W., Malqvist, Y. (Ed.), Seoul, Korea, 2013.

Lindemann, U., "Methodische Entwicklung technischer Produkte", Springer, Berlin, 2009.

Mossakowski, T., Kutz, O., Codescu, M., Lange, C., "The distributed ontology, modeling and specification language", Workshop on Modular Ontologies WoMo, Vol. 1081, 2013a.

Mossakowski, T., Lange, C., Kutz, O., "Three semantics for the core of the distributed ontology language", International Joint Conference on Articial Intelligence, IJCAI 2013. IJCAI/ AAAI, 2013b.

Pahl, G., Beitz, W., Feldhusen, J., Grote, K.-H., "Engineering Design", Springer, Berlin, 2007.

Rabe, F., Kohlhase, M., "A scalable module system", Information & Computation, Vol. 1, No. 230, 2013, pp. 1-54.

VDI, "VDI guideline 2206 Design methodology for mechatronic systems", Verein Deutscher Ingenieure (Ed.), Beuth, Berlin, 2004.

VDI, "VDI guideline 2218 – Information technology in product development – Feature Technology", Verein Deutscher Ingenieure (Ed.), Beuth, Berlin, 2003.

VDI, "VDI guideline 2221 - Systematic approach to the development and design of technical systems and products", Verein Deutscher Ingenieure (Ed.), Beuth, Berlin, 1995.

Dipl.-Ing. Thilo Breitsprecher, Chair of Engineering Design
University of Erlangen-Nürnberg
Martensstraße 9, 91085 Erlangen, Germany
Telephone: +49 9131 85-27286
Telefax: +49 9131 85-27988
Email: breitsprecher@mfk.fau.de