Semantic Support for Engineering Design Processes

Thilo Breitsprecher Mihai Codescu
Constantin Jucovschi Michael Kohlhase Lutz Schröder
Sandro Wartzack

December 10, 2013

The engineering design process typically follows a series of standardized stages, beginning with a requirement specification and ending with the final design, the embodiment. While the embodiment stage nowadays has highly developed tool support in the shape of modern CAD systems, which represent the design as a formalized object with a fairly clear geometric and physical semantics, the other stages have only intermittent information-technological support. In fact, documents from the early stages of the process are often represented in general-purpose office applications or even as hand-drawn sketches. Here, we discuss an approach to enabling pervasive tool support in the engineering design process based on semantic technologies. In particular, we show how parts and requirements can be traced through a document-oriented semantic workflow using a combination of an invasive semantic middleware and a background ontology of engineering knowledge.

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 and Beitz [PBFG07], the *Münchener Vorgehensmodell* of Lindemann [Lin09], the *Systematic Approach to the Design of Technical Systems and Products* according to the guideline VDI2221 [VDI95] and the *Design Methodology for Mechatronic Systems*, also known as the V-model [VDI04] 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 fulfill 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, we propose a semantic approach where objects are linked across the stages of the product development process process using a *federated ontology*, in which all objects are grounded via annotations with ontological concepts and assertions. We embed this approach into a document-oriented design workflow, 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. We thus make requirements and ensuing design decisions explicit and, hence, available for further machine processing at all stages of the development. In our proof of concept for this approach, we trace a sample requirement through the development of a spring tester.

The material is organized as follows. After a short description of a common product development process model and exemplary related documents, we discuss the state of the art in the application of semantic technologies in engineering design. Subsequently, we present our overall approach, and lay out in detail how semantic services within the product development process process are enabled via our Multi-Application Semantic Alliance Framework (MASally). To illustrate the added value of this approach, we then describe the spring tester case study in detail.

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

The Association of German Engineers (VDI) published the first version of the guideline VDI2221 in 1993 to replace the guideline VDI2222. We recall the steps of the engineering design process according to the VDI2221 [VDI95]:

1. **Clarify and define tasks:** A concise formulation of the purpose of the product to be designed, laid down 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 fulfill 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 layouts* 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 instructions, and user manuals.

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.

We embed our overall approach into a document-oriented workflow based on documents arising in the development process as indicated above; Figure 1 gives an overview of the most important document types.

2 Semantics 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 [VDI99], features are an aggregation of geometry items and semantics. Different types of features are defined (eg. 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. We expect features to play a role in further semanticizing step **S6** (embodiment, Section 1.1) in future work.

Li et al [LMN13] 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.

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

Figure 1: Screenshot of the table with objects in design process

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 2) 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 $0\,N \leq F_{hand} \leq 200\,N$ 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 2) — a method of combining heterogeneous ontologies by meaning-preserving interpretations [RK13].

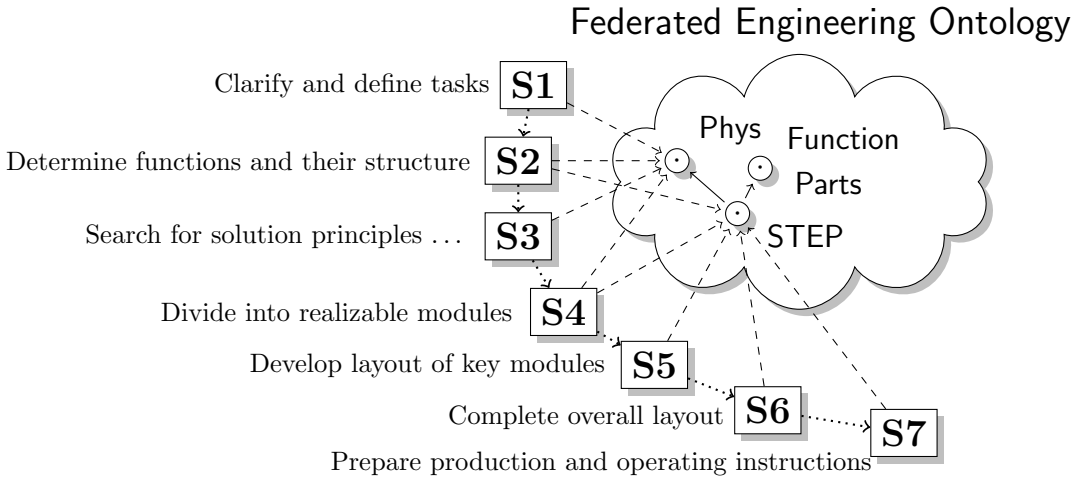

Figure 2: An Ontology-Supported Document-Oriented Design Process

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 extentions may enable users to create/update such annotations.

We used AktiveMedia ([CCL06]) for annotating images like the principle solution (see Figure 3). As we could not find a semantic authoring solutions for word processing documents that fit our needs, we used sTEX — a semantic extension of the TEX/LATEX format. To annotate CAD documents, we used the ability of our CAD environment (Autodesk Inventor) 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 affect the semantics of these annotations. This allows services to perform impact analysis and management of change.

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 workflows 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 4);

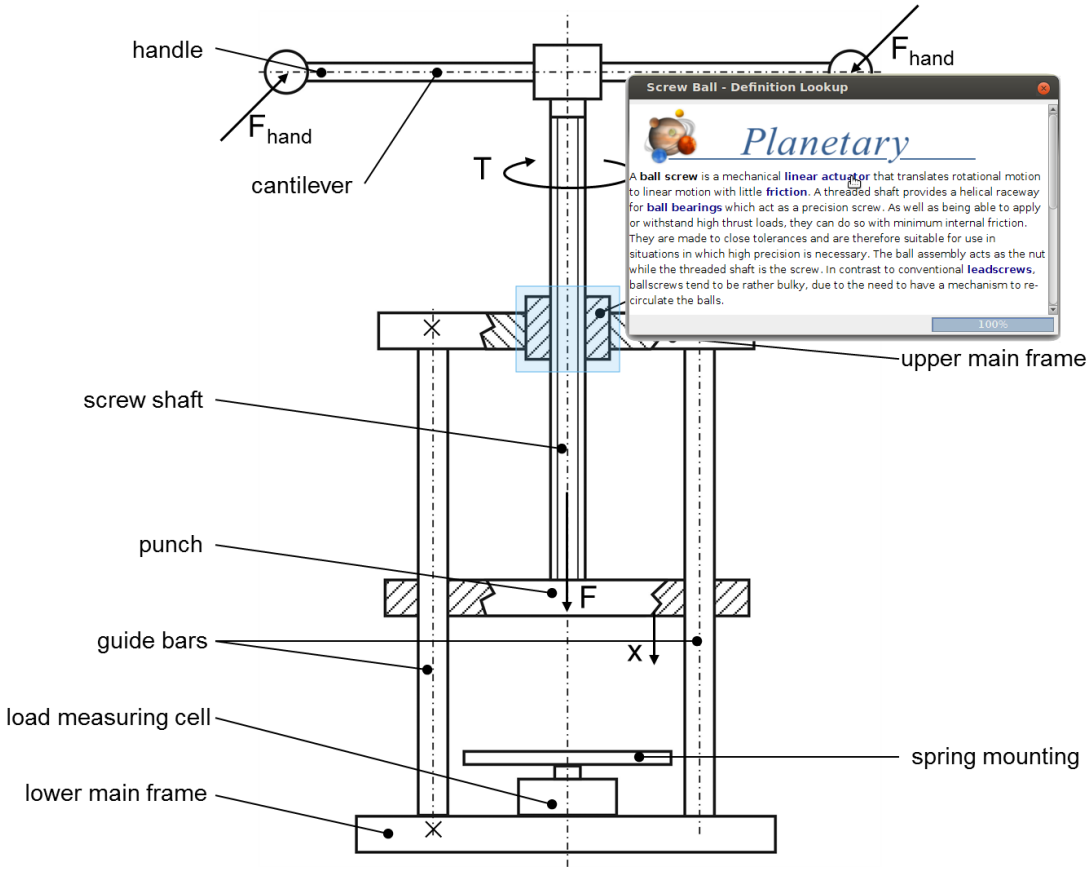

Figure 3: Principle Solution of the Spring Tester with Definition Lookup

- 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 (we show a CAD system and a document viewer for **S4**/**S5** in Figure 4)
  - 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 our experience developing an Alex for

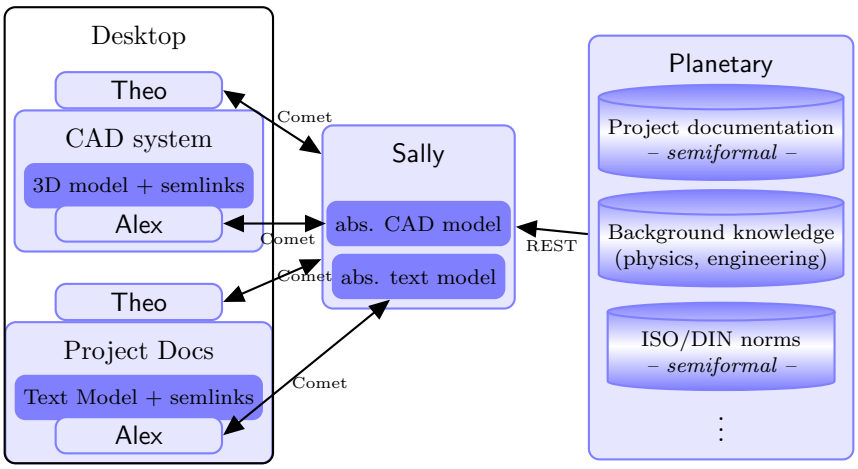

Figure 4: The MASally Architecture

an open-API application is a matter of less than a month for an experienced programmer; see [DJKK12] for details on the MASally architecture.

To fortify our intuition about semantic services, let us consider the following situation. The design engineer is working on the principle solution from Figure 3 – 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 3, 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 [KKJT13] for a discussion). All we need for this are annotations of the VDI2221 relations, ontology links and of course the ontology itself, which we will discuss next.

4 The Federated Engineering Ontology (FEO)

We now describe 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. 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, our federated ontology contains formal ontologies that enable verification of properties between different stages of design.

We will not go into the design or content of the FEO, and refer the reader to [BCJ+13]. We only note that we call a document flexiformal, if it contains material at different levels of formality [Koh13], 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, we utilize the heterogeneous OMDoc/MMT framework [Koh06, RK13] that allows representing and interrelating ontology modules via meaning-preserving interpretations (i.e. theory morphisms).

As an example of formal ontologies, as detailed in [BCJ+13], we build OWL ontologies for stating qualitative properties (e.g., abstract geometrical properties, but also function and behavior 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

Our case study is based on a spring tester (an intentionally simple example chosen for being presentable in limited space) that can measure the spring force of cylindrical compression springs at a given deflection. We have previously used this example in practical design assignments for engineering students. Students were given a principle solution (see Figure 3) for the spring tester (stage **S3** of the design process in Figure 2) 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 200N 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 our case study, we annotated documents from stages **S1**–**S3**, which were produced by faculty members, and two sets of documents from stages **S4**–**S7** produced by students – we want to study supporting design alternatives. As these documents come from an ongoing educational process, the annotation was necessarily post-mortem; experiments with integrating our methods in a live development process are the subject of future research. In particular, we looked at the following services: *i*) *definition lookup* for document elements (see Figure 3), *ii*) *topical navigation* among documents in different development stages along the refinement relations (see Figure 5), and *iii*) *propagation of change impacts* by highlighting document elements in other documents that would need to be revised (see Figure 7).

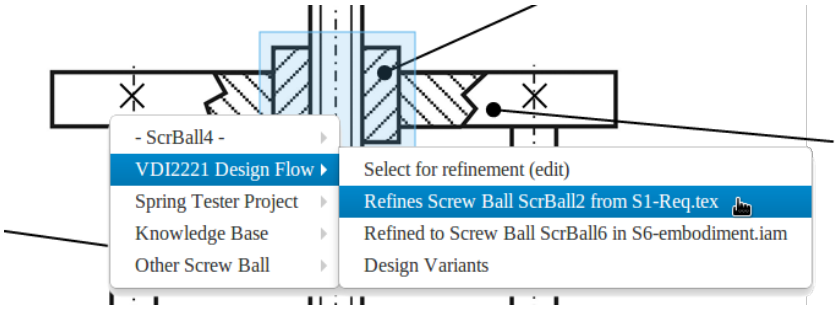

Figure 5: Navigating the Refinement Relation

Note that these services can only work if we have explicit semantic annotations in the documents. For instance, for definition lookup, we had to annotate documents with concepts in the ontology. For text documents, that meant using the sTEX macros that link text fragments with ontological concepts. From these we can generate a pdf/html document 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; we are currently working on the integration of semantic annotation functionalities into more widely used document preparation systems. We also note that many of these annotations could be semi-automatically detected using the NNexus [Gin13] 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. We used the same document parts that we annotated for definition lookup and enriched them 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 5).

For impact analysis, we enriched the documents with more domain specific annotations. Let us 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 6) 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 (***).

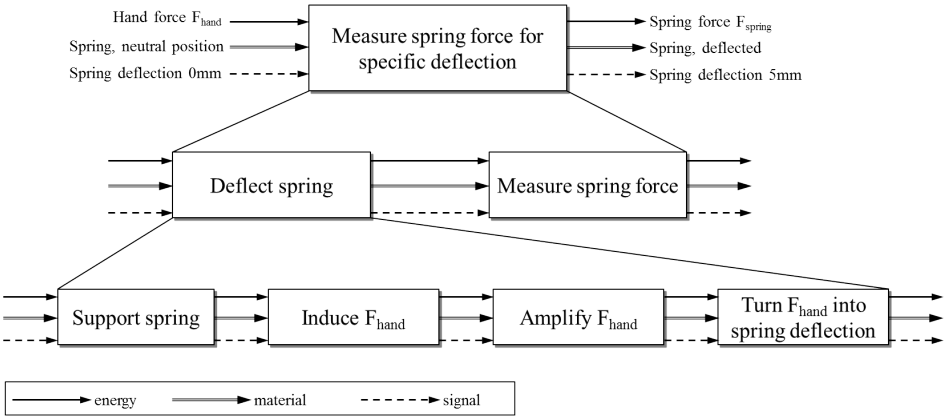

Figure 6: Function structure of spring tester

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). Our 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 Figure 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 our case study we created MathCAD® files and annotated the formulas, where 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. We annotated the part model *cantilever.prt* and linked it with the requirement $F_{hand} = 200\,N$ and thus linked the cantilever diameter in the CAD-model with the initial requirement.

Here we can see the utility of an impact management service: assume the spring tester is to be changed so that it can be used by people with a decreased hand force of $F'_{hand} = 150\,N$ (e.g. for a different market). The change impact service shown in Figure 7 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 workflow of the engineer.

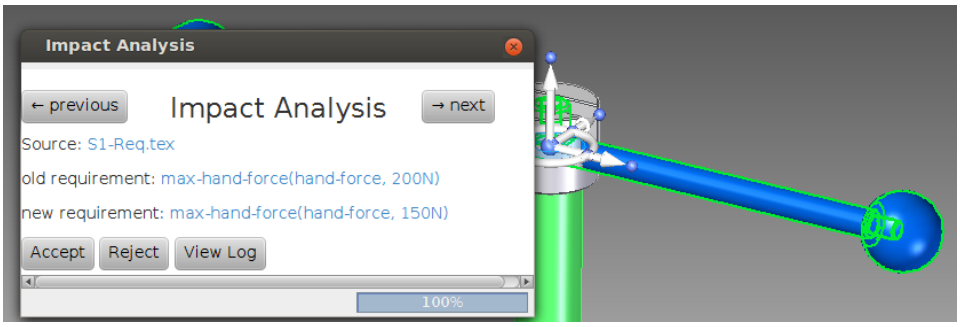

Figure 7: Impact Propagation/Resolution for Changes to the Hand Force Requirement

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.

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. Above, we have described a framework for pervasive semantic support in engineering design processes, as part of a program to unify the underlying tool chain and enhance tool support in all stages of the design process. We base our framework on a flexiformal background ontology that combines informal and semi-formal 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, we have concentrated on a document-oriented workflow 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. We have shown in complementary work how the approach can be augmented to enable *automated* requirements tracing and verification of the CAD model

against aspects of the principle solution (see [BCJ+13] for details).

We have exemplified our approach 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 [KBBS13]), CAD feature ontologies (e.g. [BG05, AGGSP07]), ontologies of function (e.g. [CMS07]), and repositories of standard parts, using modularity mechanisms enabled by modern logical frameworks such as Distributed Ontology, Modeling and Specification Language DOL [MKCL13, MLK13]. Moreover, its base of formalized engineering knowledge will be broadened; the associated knowledge acquisition process is an important aspect of further investigation. 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 [BCJ+13].

One major impediment to employing the semantically enhanced workflow 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 availablility, and we are currently working on Alexes for various word processors (primarily MS Word and OO Writer).

References

[AGGSP07] Samer Abdul-Ghafour, Parisa Ghodous, Behzad Shariat, and Eliane Perna. A common design-features ontology for product data semantics interoperability. In *WI '07: Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence*, pages 443–446, Washington, DC, USA, 2007. IEEE Computer Society.

[BCJ+13] Thilo Breitsprecher, Mihai Codescu, Constantin Jucovschi, Michael Kohlhase, Lutz Schröder, and Sandro Wartzack. Towards ontological support for principle solutions in mechanical engineering. *CoRR*, 2013. `http://arxiv.org/abs/0865290`.

[BG05] Gino Brunetti and Stephan Grimm. Feature ontologies for the explicit representation of shape semantics. *J. Comput. Appl. Technology*, 23:192–202, 2005.

[CCL06] Ajay Chakravarthy, Fabio Ciravegna, and Vitaveska Lanfranchi. Aktivemedia: Cross-media document annotation and enrichment. In *Semantic Web Annotation of Multimedia (SWAMM-06)*, 2006.

[CMS07] Gianluca Colombo, Alessandro Mosca, and Fabio Sartori. Towards the design of intelligent cad systems: An ontological approach. *Advanced Engineering Informatics*, 21(2):153–168, 2007.

[DJKK12] Catalin David, Constantin Jucovschi, Andrea Kohlhase, and Michael Kohlhase. `Semantic Alliance`: A framework for semantic allies. In Johan Jeuring, John A. Campbell, Jacques Carette, Gabriel Dos Reis, Petr Sojka, Makarius Wenzel, and Volker Sorge, editors, *Intelligent Computer Mathematics*, number 7362 in LNAI, pages 49–64. Springer Verlag, 2012.

[Gin13] Deyan Ginev. NNexus Glasses: a drop-in showcase for wikification. In Christoph Lange, David Aspinall, Jacques Carette, James Davenport, Andrea Kohlhase, Michael Kohlhase, Paul Libbrecht, Pedro Quaresma, Florian Rabe, Petr Sojka, Iain Whiteside, and Wolfgang Windsteiger, editors, *MathUI, OpenMath, PLMMS and ThEdu Workshops and Work in Progress at the Conference on Intelligent Computer Mathematics*, number 1010 in CEUR Workshop Proceedings, Aachen, 2013.

[KBBS13] Oliver Kutz, Mehul Bhatt, Stefano Borgo, and Paulo Santos, editors. *The Shape of Things, SHAPES 2013*, volume 1007 of *CEUR Workshop Proceedings*, 2013.

[KKJT13] Andrea Kohlhase, Michael Kohlhase, Constantin Jucovschi, and Alexandru Toader. Full semantic transparency: Overcoming boundaries of applications. In Paula Kotzé, Gary Marsden, Gitte Lindgaard, Janet Wesson, and Marco Winckler, editors, *Human-Computer Interaction – INTERACT 2013*, volume 8119 of *Lecture Notes in Computer Science*, pages 406–423. Springer, 2013.

[Koh06] Michael Kohlhase. OMDoc – *An open markup format for mathematical documents [Version 1.2]*. Number 4180 in LNAI. Springer Verlag, August 2006.

[Koh13] Michael Kohlhase. The flexiformalist manifesto. In Andrei Voronkov, Viorel Negru, Tetsuo Ida, Tudor Jebelean, Dana Petcu, and Stephen M. Watt ane Daniela Zaharie, editors, *14th International Workshop on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC 2012)*, pages 30–36, Timisoara, Romania, 2013. in press.

[Lin09] Udo Lindemann. *Methodische Entwicklung technischer Produkte, Methoden flexibel und situationsgerecht anwenden*. VDI Book. Springer Verlag, 2009.

[LMN13] Chun Lei Li, Chris McMahon, and Linda Newnes. Supporting multiple engineering viewpoints in computer-aided design using ontology-based annotations. In *Proceedings of the 19th International Conference on Engineering Design (ICED13)*. Design Society, 2013.

[MKCL13] Till Mossakowski, Oliver Kutz, Mihai Codescu, and Christoph Lange. The distributed ontology, modeling and specification language. In *Workshop on Modular Ontologies, WoMo 2013*, volume 1081 of *CEUR Workshop Proceedings*, 2013.

[MLK13] Till Mossakowski, Christoph Lange, and Oliver Kutz. Three semantics for the core of the distributed ontology language (extended abstract). In *International Joint Conference on Artificial Intelligence, IJCAI 2013*. IJCAI/AAAI, 2013.

[PBFG07] Gerhard Pahl, Wolfgang Beitz, Jörg Feldhusen, and Karl-Heinrich Grote. *Engineering Design*. Springer Verlag, 3rd edition, 2007.

[RK13] Florian Rabe and Michael Kohlhase. A scalable module system. *Information & Computation*, 0(230):1–54, 2013.

[VDI95] VDI-Gesellschaft Entwicklung Konstruktion Vertrieb. *Methodik zum Entwickeln und Konstruieren technischer Systeme und Produkte*, 1995. English title: Systematic approach to the development and design of technical systems and products.

[VDI99] VDI-Gesellschaft Entwicklung Konstruktion Vertrieb. *Informationsverarbeitung in der Produktentwicklung – Feature-Technologie – VDI 2218*, 1999. *(Information technology in product development – Feature Technology)*.

[VDI04] VDI-Gesellschaft Entwicklung Konstruktion Vertrieb. *Entwicklungsmethodik für mechatronische Systeme – VDI 2206*, 2004. English title: Systematic approach to the development and design of technical systems and products.