Knowledge Management for Systematic Engineering Design in CAD Systems

Michael Kohlhase
Computer Science
Jacobs University Bremen D-28759 Bremen
m.kohlhase@jacobs-university.de

Abstract: Systematic engineering design processes for solid objects and machine parts are highly knowledge-based endeavors whose geometric aspects are supported by CAD systems. The knowledge management aspect is currently not supported by machines. We present the FfCAD system that implements existing document-oriented processes from engineering design into a knowledge-based environment that integrates CAD/CAM files with requirements specifications. All documents are semantically annotated and interlinked via a structured background ontology, which allows to embed semantic added value services into CAD-based workflows.

1 Introduction

Much of our life is shaped by technical artifacts, ranging in terms of intrinsic complexity from ball point pens to autonomous robots. These artifacts are the result of *engineering design processes*[1] that determine their quality, safety, and suitability for their intended purposes and are governed by best practices, norms, and regulations. The systematic development of products is guided by descriptions of problems and their solutions on different levels of abstraction, e.g. the requirements list, the function structure, the principle solution, and eventually the embodiment design. The elements of these representations are linked by dependencies within and across different levels of abstraction. The present state of the art in computer-aided design and manufacture of industrial artifacts (CAD/CAM) does not support this cross-linking of dependencies. Consequently, e.g. non-embodied principle solutions are still often shared and stored in the form of hand-made sketches and oral explanations. In other words, large parts of the engineering process are not completely representable in current CAD/CAM systems, which are focused primarily on the embodiment level.

Best practices for designing technical artifacts are typically standardized by professional societies. In our exposition here, we will follow the German VDI 2221 [VDI95], which postulates that the design process proceeds in well-defined phases, in which an initial idea is refined step-by-step to a fully specified product documentation. We observe that the pro-

[1]As the English term "construction" standardly refers to the construction of buildings, we instead use the term "systematic engineering design" synonymous to the German term "Konstruktionslehre", which refers to the study of constructive aspects of design processes in mechanical engineering as described in [VDI95].

cess is similar to the software engineering process and that the stages in the design process result in specification documents, as they are e.g. found in the V-model (see Fig. 1).

In contrast to software engineering approaches like the V-model, however, VDI 2221 (and actual current practice in engineering design) does not provide a mechanism to ensure consistency between the design stages, or methods for verifying that products actually meet requirements specified in preceding phases of the development. In fact, the VDI 2221 process corresponds only to the left leg of the process depicted in Fig. 1, while the quality control process (the right leg in Fig. 1 and the main contribution of the V-model) is left unspecified. Even in software-engineering methods like the V-model, the quality control processes are only insufficiently supported by machines, leading to high levels of personnel costs and human errors.

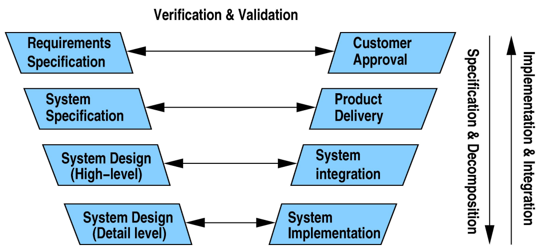

Verification & Validation
Requirements Specification
Customer Approval
System Specification
Product Delivery
System Design (High–level)
System integration
System Design (Detail level)
System Implementation
Specification & Decomposition
Implementation & Integration

Figure 1: The V-model of Software Engineering

In FM09 [KLSS09] we have proposed a document-oriented development process that includes documents as external knowledge sources, such as technical norms or requirements and design specifications, into the design process based on the VDI 2221 and software engineering practices. In this process, design documents are semantically annotated, making relations explicit that were only implicit before and using this information for extended machine support.

In this paper we describe a system that implements this proposal in a knowledge-based environment. In the next section we will review the VDI 2221-based processes and introduce our running example – a "blind flange pipe end fitting" – a particular pipe end that allows access to the piping system for revision and cleaning; we will refer to it as an "inspection fitting"[2] in the following. In Section 3 we present the FfCAD system that adds system support for the knowledge-based aspects of the systematic design process, and in Section 4 we show the system in action during the inspection fitting design process. Section 5 concludes the paper.

Acknowledgements The work presented in this paper was supported by the German Research Foundation (DFG) under grant KO-2484-12-1 (project FormalCAD). The direction of the project has been decisively shaped by discussions about formal methods for CAD objects with our FormalCAD partners, most prominently Lutz Schröder and Ewaryst Schulz and by discussions on flexiformal user assistance in the SiSsI project (KO-2484-12-1; most prominently Andrea Kohlhase and Constantin Jucovschi). We use the `Semantic Alliance` framework from the SiSsI project as the basis for our implementation. Alin Iacob has implemented the `Alex` for Autodesk Inventor (see Secction 3).

[2]This assembly is chosen for purely expository reasons, nothing in this paper hinges on this choice.

2 A Knowledge-Based Workflow for Engineering Design

We review the six stages of VDI 2221[3] and illustrate them with a simple example.

S1 **Purpose/Problem**: a concise formulation of the purpose of the product to be designed.
S2 **Requirements List**: a list of explicitly named properties of the envisioned product. It is developed in cooperation between designer and client and corresponds to the user specification document in the V-model.
S3 **Functional Structure**: A document that identifies the functional components of the envisioned product and puts them into relation with each other.
S4 **Solution in Principle**: a specification of the most salient aspects of the design, see [Lem08] for details.
S5 **Embodiment Design/"Gestalt"**: a CAD design that specifies the exact shape of the finished product.
S6 **Documentation**: accompanies all steps of the design process.

Note that most of these design steps result in informal text documents, with step 5 being the notable exception. In the envisioned document-oriented engineering design process we will concentrate on these documents, enhance them with semantic annotations and link them to background specifications to enable machine support: e.g. requirement tracing, management of change, or verification of physical properties. Before discussing this vision in more detail, let us set up an example.

2.1 Running Example: The Design Process of an Inspection Fitting

We cast our running example in the context of a small company that specializes in the production of small and fast luxury yachts. The design problems to be solved range from the yacht's shape down to pipe fittings and are carried out in a VDI 2221-based process (after all, the customers want to be sure that they are getting what they asked for and adhering to VDI 2221 enhances customer trust). To exemplify the process, we will take a closer look at one particular design task: a pipe end fitting that allows access to the piping system (e.g. the water cooling system for the yacht motors) for inspection and cleaning. We illustrate the VDI 2221-based engineering design process by presenting (abbreviated versions of) the design documents at the levels $\mathbf{S}i$ ($1 \leq i \leq 5$) introduced above.

S1 The Purpose of an Inspection Fitting The first and most important step in setting up a requirements list is the specification of the purpose of the product. The purpose describes the intended use of the product solution-neutrally. This is the highest level of abstraction within the design process. In the case of our inspection fitting, it takes the form of a very simple definition:

*The **inspection fitting** is a pipe fitting that allows access to a piping system*

[3]In fact, [VDI95] specifies additional stages for determining modular structures and developing their embodiments, which we will subsume in steps 3 and 5.

for inspection and cleaning.

S2 Requirements In the case of the inspection fitting, the requirements might include the following.

E1 The fitting must tolerate a pressure of 350 kPa (3.5 bar).
E2 The fitting must be resistant to corrosion from sea water at $95°C$.
E3 The inspection fitting must allow access to the piping system with only standard tools, even in the case of deposits in fitting.

Ideally, the list of requirements of a product should be unambiguous, clear and complete. However, this is rarely the case in a real life design process, e.g. due to implicit customer wishes, which in fact are often more important to the market-success of a product than the explicitly named requirements.

I1 The inspection opening must be reclosable without further materials.
I2 The inspection opening should not add much to the weight and outer size of the piping system.

S3 Functional Specification In the design process, the functional specification is done by setting up the function structure that breaks down the complex problem into manageable smaller sub-functions and represents the logical interrelationships of the given tasks. As in the previous design steps, the function structure is still solution neutral. The aim is to open up a preferably broad solution field, which will be narrowed by explicit, named criteria within further steps of the design process.

For the inspection fitting we have four functional parts:

F1 a *pipe fitting*: the part that connects to the pipe end,
F2 a *cap*: a part that closes off the hole,
F3 a *seal* in the connection between the fitting and the cap, and
F4 (optionally) fasteners that hold the cap to the fitting.

S4 The Principle Solution From the functional specification, we develop a *principle solution*, which can be a manual sketch (as our principle solution for the inspection opening in Fig. 2) or an abstract assembly in a CAD system. This solution abstracts from the physical traits of the eventual product and identifies the functional parts. For an inspection fitting the principle solution chosen here instantiates the pipe fitting to a flange fitting, that can be welded or soldered to the pipe, and the cap as a blind flange, that can be fastened to the flange. The principle solution does not specify the fasteners, but instantiates the seal to a gasket that can be reused to fulfill requirement **I1** (reclose without new materials: we can reuse the gasket).

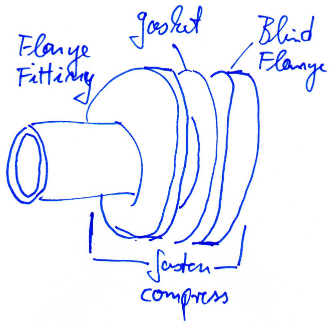


Figure 2: A Principle Solution

S5 The Embodiment of a Blind Flange Pipe Fitting Note that the principle solution is not a finished design yet, since it abstracts from most of the physical traits of a blind flange pipe end fitting, e.g. the dimensions, the fasteners, etc. In the embodiment, the ultimate three-dimensional shape and the materials of the product are derived, taking into account material properties, manufacturability constraints, and pricing factors (see the Figure on the right). This is the final stage of the design that will be used as the basis for manufacturing and documentation. The CAD document containing the embodiment may be accompanied by (or embedded into) documents such as design justifications, material technical reports or descriptions of surface finishes.

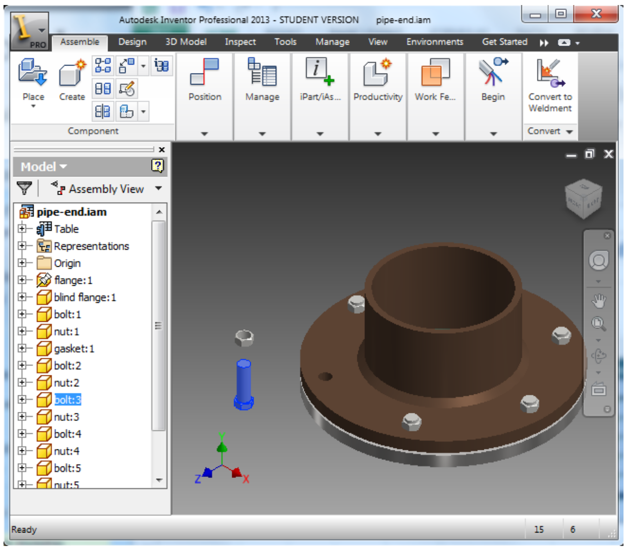


Figure 3: An Embodiment in Autodesk Inventor

2.2 A Knowledge-Based Design Process

We propose to reinforce the systematic engineering design process laid out above with technologies and practices from software engineering to obtain a knowledge-based process where designs are semantically enhanced and linked to semi-formal specifications. It is crucial to note that the various design documents necessarily have differing levels of rigor, ranging from informal and hard-to-quantify requirements like **I2** to mathematical proofs of security-relevant properties, e.g. in aerospace applications. Additionally, different product parts and aspects are subject to differing economic and security-related constraints, so that design quality control[4] must be supported at various levels of formality (going beyond strictly ontological annotation as e.g. in the EXPRESS language that forms part of the STEP *Standard for product data exchange*, ISO 10303 [ISO94, Pra01]). As a consequence, design documents need to be encoded in a document format that supports *flexible degrees of formality*. We use the OMDoc (Open Mathematical Documents [Koh06]) language that concentrates on structural aspects of the knowledge embedded in documents and provides a markup infrastructure to make it explicit by annotation. Crucially, the format supports a fine-grained mixture of formal and informal elements and thus supports the

[4]In our deliberations it is important to distinguish the concept of "quality of design" from the quality assurance in the production process. We are only concerned with the former in this paper: our methods are inspired by the software design process, where production quality problems can largely be ignored.

stepwise migration from informal user requirements to specifications expressed in formal logics supported by verification environments. The format itself is *semi-formal*; it focuses on explicitly structured documents where relevant concepts are annotated by references to *content dictionaries* that specify the meaning of the terms used in design documents. Semi-formal design documents already bring added value to the engineering process by enabling machine support for many common quality control tasks like *requirement tracing* (see Section 4.3) and *management of change* which are based on an explicitly given dependency relation. Fully formal development strands embedded in a semi-formal process additionally allow for the rigorous verification of critical properties in a design, thus providing a reliable link between various stages of the engineering design process.

2.3 Related Work

As the work in this paper concentrates knowledge-based methods for supporting the design of solid objects, we will restrict our survey of related work to those methods and disregard work on knowledge-based methods supporting engineering processes like quality assurance or error management, e.g [BANS11].

The potential usefulness of using knowledge-based and formalized approaches to deal with various widespread problems in engineering design has long been recognized. To begin with ontological approaches – logical approaches that deal with the nature and relationships of objects rather than with their exact geometric and physical properties – there is a substantial body of existing work on using ontology languages such as description logics and F-Logic for engineering tasks. [FKST10] presents a method for the automatic detection of design errors in a CAD document, in particular for classifying overlaps of parts, calculated by the standard tool *DMU analyser*, as being intentional (e.g. for gaskets or for bolts with intentionally omitted threads) or erroneous. Relevant knowledge about consistency conditions for designs is laid down as a TBox (terminological component) in an OWL background ontology, while a description of the actual CAD model is exported from the CAD system in the shape of an OWL ABox (assertional component); conformance checking of the design then amounts to checking the description of the model for consistency with the background knowledge.

F-logic has been used in the relatively coarse-grained ontological description of development processes in the automotive industry [MSS03], as well as for supporting the configuration of test cars [AEW08] and for specifying virtual 3D environments [DTBK09]. An overview of various approaches to *feature ontologies*, which serve to facilitate data exchange between CAD systems from different vendors, is given in [AGGSP08]. The OntoSTEP ontology aims at ontologizing the standard STEP interchange format for CAD data. Ontologies have moreover been applied in production automation; an overview is given in [MLD09]. An abstract methodology for modeling the *functionality* of technical artifacts in a description logic is presented in [CMS07].

More expressive logics, in particular first-order logic, have also been used for physical ontologies and ontologies of technical artifacts [BCGV09]. So far, such ontologies serve mainly the conceptual clarification of the formal notions involved in modeling such objects, and are typically not used in the actual analysis of concrete technical developments due to the lack of well-developed tool support. Finally, [DSS11] introduces domain-specific languages in the formalization of industrial standards, translates them into logic and uses that for verification (of applicability of the standard) in a combination of theorem provers and computer algebra systems.

In [KLSS09], we have developed an export function for the standard commercial CAD system SolidWorks, which generates a representation of the actual data structures used in the CAD system, thus encoding objects according to the way they are constructed; e.g., a length of pipe will be produced by drawing a circle in a plane, subtracting a slightly smaller concentric circle from it, and then extruding the resulting circular ring to obtain a three-dimensional object. The geometric semantics of these constructions is captured by a spatial background theory that specifies, e.g., the actual spatial extension of an extrusion. Given this semantics, one can then relate concrete constructions to abstract geometric shapes, such as a hollow cylinder; these relations can be proved formally with the help of a semi-automatic proof assistant, e.g. Isabelle/HOL.

All of the approaches surveyed in this section utilize ontologies (both in the sense of web ontologies or in the sense of formal descriptions of the objects and their relations) as formal counterparts of the CAD objects with the goal of verification of security properties. The approach presented in this paper is complementary: we use flexiformal ontologies for other semantic services, namely user assistance and requirement tracing. To the best of our knowledge this approach is novel, though ontology-based systems for other knowledge-based applications exist; most notably SACHS system [KK12] for spreadsheet systems developed in our group.

3 FfCAD: A Knowledge-Based System for Engineering Design

We chose the `Semantic Alliance` framework [DJKK12] as the basis for our implementation. This allows to superimpose semantic services over an existing (and possibly proprietary) application provided that it gives open-API access to user events. In our case this application is the CAD system Autodesk Inventor [AIn12], which exposes the internal object and event model via a C# interface. The main advantage of this choice is that users do not have to radically change their CAD-centered workflows. They only perceive the new, knowledge-based features as plugins to their CAD system that add user assistance and requirement tracing.

In a nutshell, the `Semantic Alliance` has three components in our setup:

1. a platform-independent semantic interaction manager `Sally` (as a "Semantic Ally" to Autodesk Inventor), that has access to contextualizable semantic services, and that partners with
2. an invasive, thin, application-specific API manager `Alex`, that essentially only manages user interface events in Autodesk Inventor, and has access to
3. an application-independent screen area manager `Theo` that can render the results

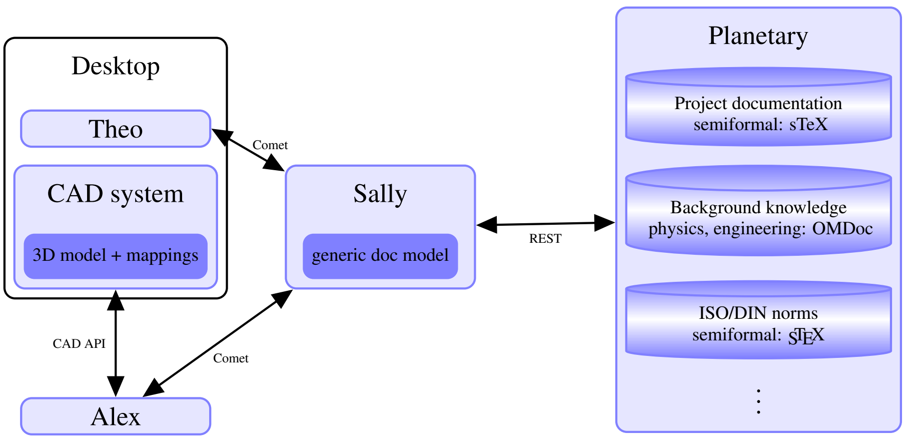

Desktop
Theo
CAD system
3D model + mappings
Comet
Sally
generic doc model
REST
Planetary
Project documentation semiformal: sTeX
Background knowledge physics, engineering: OMDoc
ISO/DIN norms semiformal: sTeX
CAD API
Comet
Alex

Figure 4: Architecture of the FfCAD System

generated by the services as pop-overs in Autodesk Inventor, so that they are perceived as Autodesk Inventor-native ones.

The `Semantic Alliance` framework presupposes that all "semantic objects" (in our case the parts and components of a CAD assembly) are linked to a structured background ontology via a user-specified semantic mapping. The central advantage for our purposes is that the `Sally` component provides an autonomous, application-independent process that provides the application intelligence. Only the `Alex` for Autodesk Inventor had to be developed from scratch (a programming task of less than a month) and `Sally` extended for the FfCAD-specific services.

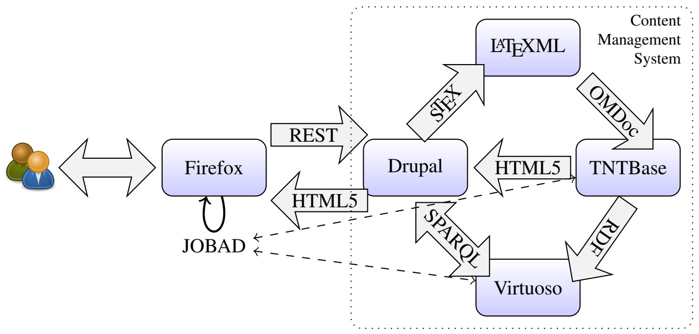


Figure 5: The Planetary System

For the structured background ontology and the specification documents we make use of the **Planetary System** (see Figure 5 for the architecture and [KCD+11] for a system description), which serves as a knowledge base and active document player for specification documents (see Figure 6). All documents are encoded in the OMDoc knowledge representation format and stored in the TNTBase system, which generates user-adaptive HTML5 presentations of the documents and delivers semantic services by aggregation. These can in turn be integrated into Autodesk Inventor via Theo and into the user documents by a special JavaScript framework "JOBAD" triggered by RDFa annotations in the presented documents that reflect the ontology relations.

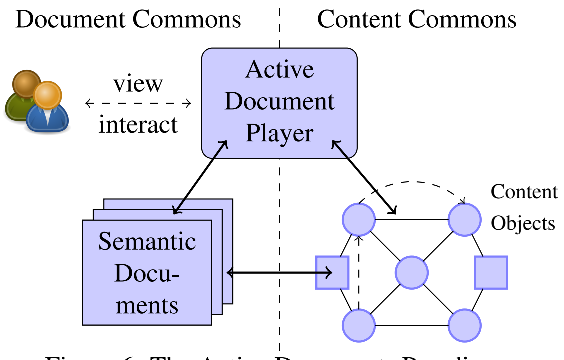


Figure 6: The Active Documents Paradigm

The structured background ontologies used in the Planetary system are flexiformal documents in mathematical style consisting of concept definitions, statements of properties of the objects described using these concepts, all organized in theories that are interlinked by theory morphisms (concept mappings that inter-relate theories in a way that conserves validity). Even though the statements themselves are given in natural language interspersed with formulae, they are content-structured, i.e. organized in an object-oriented manner that facilitates machine processing (the right hand side of Figure 6). Documents can now be annotated by a semantic mapping as well, e.g. linking symbols in formulae or technical terms in text to their definitions in the content commons.

As all technical terms in the ontology are linked to their definitions, we can compute the dependency relation between concepts (concept A depends on B if B is mentioned in A's definition). This relation can be used to compute concept graphs, which turn out to be good structures for exploring the prerequisites for understanding concepts.

4 Knowledge-Based Design via the FfCAD System

We present the user assistance and requirements tracing services provided by the FfCAD system using our running examples after we have introduced the structured ontology they are based on.

4.1 Structured Ontology

For our running example, we have implemented a structured ontology for engineering parts, including definitions of flanges, gaskets, nuts, bolts, etc. – see Figure 7 for the theory graph. The actual knowledge is authored in sTeX, a semantic variant of LaTeX that allows the author to add semantic annotations and can be transformed into OMDoc via the LaTeXML daemon.

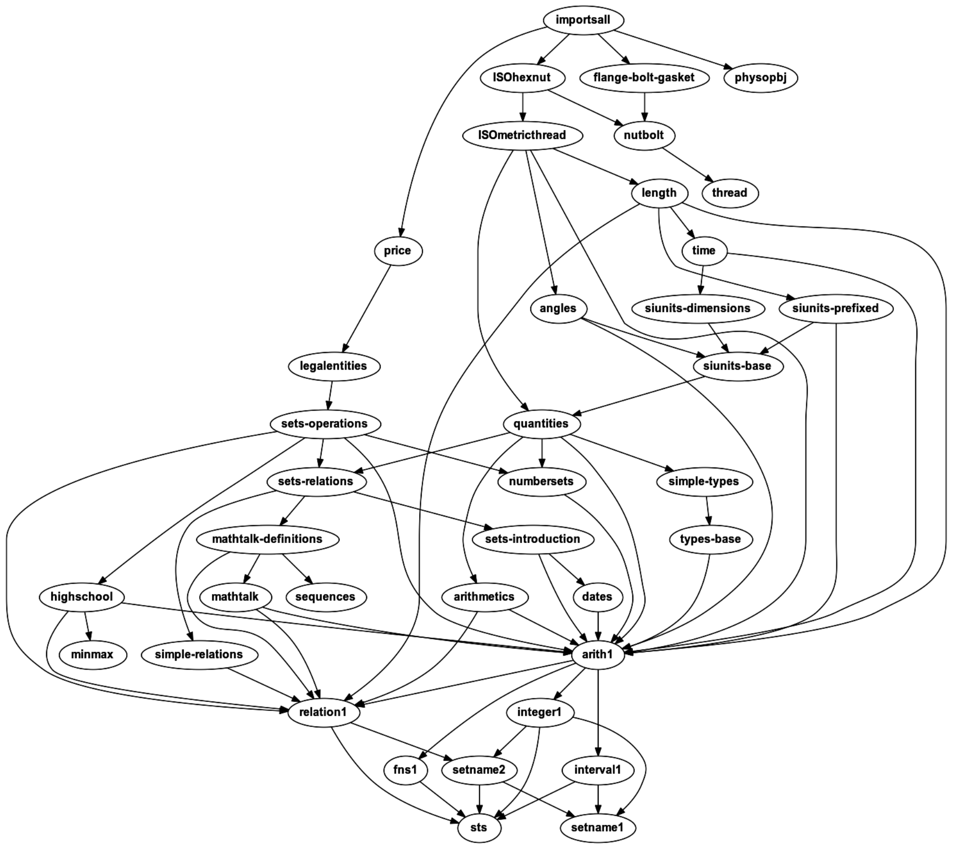


Figure 7: The Theory Graph of the Ontology

This ontology contains definitions such as the one in Figure 8 for the various parts in the inspection fitting. In the OMDoc sources, the definiendum is a concept `ISOhexnut` marked up as a function that maps (some small) numbers to solid objects (hexagonal nuts made of steel; their geometry is given in the table). Note that in the development of the ontology, we chose to only represent the object types, not the object instances themselves. For instance, the ontology only has the concept of a hex nut, but not the six individual instances in the inspection fitting assembly. The instance data (their number and parameters) stays in the CAD system. This seems like a generally applicable heuristic for the development of engineering domain ontologies.

Note that the theory graph in Figure 7 only shows those parts of a larger OMDoc-represented ontology developed at Jacobs University that are relevant to our running example. In fact, only the eight theories in the top four levels were created for the FfCAD system and only one (flange-bolt-gasket) for the specific example of the inspection fitting; the other seven are about standard machine parts (physical/solid objects, nuts, bolts, and their threads). All the others (units, quantities, prices, legal entities, and elementary math) already pre-

Definition: The sizes of **hex nuts** are standardized by their internal thread (irrespective of the pitch) by the ISO.

thread	flat size	diameter	height
	A/F	A/C	H
M1		2.5 mm	
M1.6		3.2 mm	
⋮	⋮	⋮	
M10	16 mm	19.6 mm	8 mm
⋮	⋮	⋮	

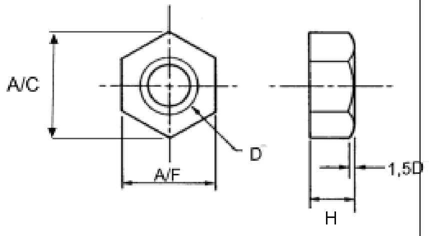
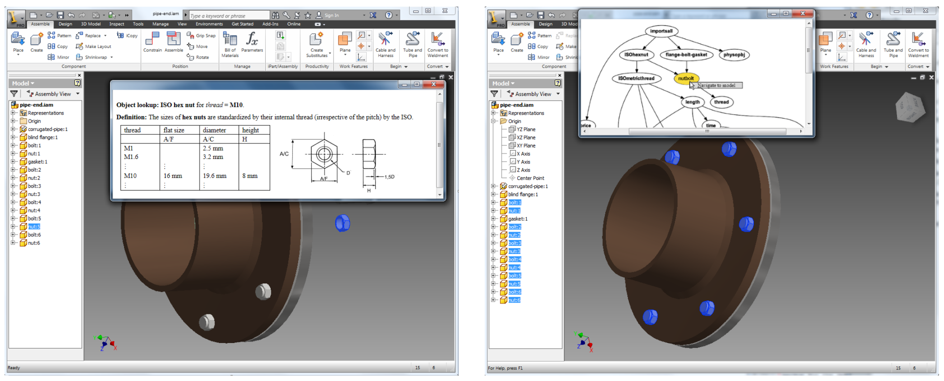


Figure 8: A Definition of an ISO Hex Nut

existed from other case studies. In our experience this is typical, which relativizes the initial investment into an OMDoc ontology needed for employing the `Semantic Alliance` framework. Admittedly, the cost of supplying a semantic mapping remains, but can be eased by custom tools we are currently developing.

4.2 User Assistance Services

As mentioned above, the semantic mapping assigns the nuts in Figure 3 to concepts in the ontology. To specify the instance, it needs to specify the parameters (here only the thread diameter, which fully determines the geometry of hex nuts which are standardized by an ISO norm). Other objects are mapped in a similar way, only that they may require more parameters (e.g. five for the flange fitting) to fully describe their geometry. The availability of the semantic mapping directly translates into a simple (but very useful) service: the definition lookup service from the `Semantic Alliance` framework (see Figure 9a). Clicking on one of the hex nuts produces a pop-over `Theo` window (see Figure 4) near the selected nut in the assembly that shows the definition in Figure 8 and specializes it to the parameter 10 highlighting the particular dimensions of the M10 nuts; this is possible since OMDoc sources contain special markup for the definiendum and the definiens (the M10 case in the table). Similarly, we can use the semantic mapping to call up the concept/prerequisites graph of a selected object to better understand the respective parts or sub-assemblies. This graph can also be used for "semantic navigation", as we can easily link any of the concepts in the graph to all parts of the assembly and highlight them as the user navigates, as illustrated in Figure 9b.


Object lookup: ISO hex nut for *thread* = **M10**.

Definition: The sizes of **hex nuts** are standardized by their internal thread (irrespective of the pitch) by the ISO.

thread	flat size	diameter	height
	A/F	A/C	H
M1		2.5 mm	
M1.6		3.2 mm	
⋮	⋮	⋮	
M10	16 mm	19.6 mm	8 mm
⋮	⋮	⋮	

(a) Definition lookup for a M10 bolt (b) Semantic navigation service

Figure 9: User Assistance Services in FfCAD

4.3 Requirement Tracing

The second feature implemented in the FfCAD system is requirement tracing for the specification documents in Section 2.1. Requirements traceability, e.g. [Jar98] is concerned with the question of how to acquire and maintain a trace from requirements along their manifestations in development artifacts at different levels of abstraction. This is a key technique for software maintenance, because the information about relationships between different artifacts that is provided by these traces allows to assess the impact and scope of planned changes, cf. [GHM98]. A planned change can be propagated along the links that relate dependent items in the artifacts, and thus a work plan of necessary follow-up changes can be acquired. The dependency relationships and the ways in which changes propagate over them is application dependent. Hence common requirement tracing techniques are either very general and thus do not provide sufficient support for propagating changes or they aim at specific phases in the software development process and incorporate a fixed semantics of the managed documents; see [INC] for an overview of existing (mostly commercial) requirement tracing tools.

For FfCAD, we have encoded the requirements documents in a special requirements documents class in sTeX, which provides special environments for requirements and their dependencies. These are translated into special RDFa markup in the OMDoc documents in the Planetary system, which harvests the RDF triples for the integrated triple store (here the Virtuoso system; see Figure 5). With this information we can realize various services that support V-Model-inspired workflows. For instance:

- Individual requirements in the requirements documents displayed in the Planetary system can be instrumented via services that generate the "requirements graphs": a graph that traces the requirements through the chain of specification documents **S1**-**S6** from Section 2. This graph supports checking that all requirements lead to implementations in the embodiment. In our example we would have a trace from **I1** to the seal in the functional specification, on to the gasket in the principle solution and finally to the specific gasket in the embodiment.

- Conversely, we can generate a requirements graph for individual parts or sub-assemblies in the assembly, allowing the designer to trace which requirement justifies it (design methodologies like the V-Model have a requirement that all parts have to be justified).
- Finally, we can integrate requirement checking triggers into CAD system interactions that issue warnings whenever a CAD part or sub-assembly is changed that is linked to a requirement.

5 Conclusion and Future Work

We have presented a knowledge management system for supporting systematic engineering design processes. The FfCAD system is implemented as an instance of the `Semantic Alliance` framework that superimposes semantic services over the user interface of Autodesk Inventor giving users access to knowledge-based user assistance and requirement tracing features. Thus FfCAD acts as a "semantic ally" to Autodesk Inventor.

The present work forms part of a long-term endeavor where we want to rethink the systematic engineering design process as a whole. We note that VDI 2221-based processes only concern the left branch of the V-model (see Figure 1). We conjecture that with the right services, the Planetary system can generate the documentation on the right branch automatically.

In this paper (and the FfCAD system) we have concentrated on largely informal (though structured) specification documents. Safety-critical applications, e.g. in the aerospace industry, may require more formal treatments up to mathematical proofs of security-relevant properties. We are currently working on extending the semi-formal methods reported on in this paper to include formal systems like the HETS system [MML07]. As the OMDoc framework underlying the FfCAD system can already handle formal content, most of the work involves extending the export facilities of the `Alex` for Autodesk Inventor and enrichment of the ontology by formal content, a non-trivial task, since reasoning about the geometry of solid objects is almost uncharted territory in formal methods, though generalizations of results on automated theorem proving in geometry including [Wu94, Grä04] represent first steps.

Finally, we want to adapt the generic change management algorithms for the OMDoc/Planetary suite [ADD$^+$11, Ian12] to the `Semantic Alliance` framework by extending the respective `Alex`es with functionality that intercepts change interaction events and by extending `Sally` with an interaction model for change management. We hope and conjecture that generic change management processes triggered by the dependency relation induced by the background ontology and the specification documents will be sufficient in most naturally occurring applications in engineering design.

Finally note that the integration of semantic services into specific applications is not the only way we can utilize semantic linking which underlies the `Semantic Alliance` framework. The table and images in Figure 8 are typical instances of multimodal documents mixing text with images (here a 2D-sketch that is arguably best produced by a CAD program) and tables (functional blocks best produced by a spreadsheet program). As the content dictionaries and specification documents HTML5 documents are generated from semantically annotated content representations in FfCAD it should be possible to extend the presentation process to generate CAD and spreadsheet objects with their interpretation mappings and use the `Semantic Alliance` framework generate pop-overs from the applications (the dual way the `Semantic Alliance` is used currently). Note that contrary to classical integrations e.g. of the parts of the MS office suite which use object linking & embedding (OLE) or the COM models for integration of software modules this integration would be an mashup of user interfaces integrated by semantic linking.

References

[ADD$^+$11] Serge Autexier, Catalin David, Dominik Dietrich, Michael Kohlhase, and Vyacheslav Zholudev. Workflows for the Management of Change in Science, Technologies, Engineering and Mathematics. In James Davenport, William Farmer, Florian Rabe, and Josef Urban, editors, *Intelligent Computer Mathematics*, number 6824 in LNAI, pages 164–179. Springer Verlag, 2011.

[AEW08] Jürgen Angele, Michael Erdmann, and Dirk Wenke. Ontology-Based Knowledge Management in Automotive Engineering Scenarios. In Martin Hepp, Pieter De Leenheer, Aldo de Moor, and York Sure, editors, *Ontology Management*, volume 7 of *Semantic Web And Beyond*, pages 245–264. Springer, 2008.

[AGGSP08] Samer Abdul-Ghafour, Parisa Ghodous, Behzad Shariat, and Eliane Perna. Towards an Intelligent CAD Models Sharing Based on Semantic Web Technologies. In Richard Curran, Shuo-Yan Chou, and Amy Trappey, editors, *Collaborative Product and Service Life Cycle Management for a Sustainable World*, Advanced Concurrent Engineering, pages 195–203. Springer, 2008.

[AIn12] Autodesk. Product web page: `http://usa.autodesk.com/autodesk-inventor/`, 2012.

[BANS11] Kerstin Bach, Klaus-Dieter Althoff, Régis Newo, and Armin Stahl. A Case-Based Reasoning Approach for Providing Machine Diagnosis from Service Reports. In Ashwin Ram and Nirmalie Wiratunga, editors, *Case-Based Reasoning Research and Development Case-Based Reasoning Research and Development. International Conference on Case-Based Reasoning (ICCBR-2011), September 12-15, London, United Kingdom*, volume 6880 of *Lecture Notes in Computer Science, LNCS*, pages 363–377. Springer Verlag, 9 2011.

[BCGV09] Stefano Borgo, Massimiliano Carrara, Pawel Garbacz, and Pieter E. Vermaas. A formal ontological perspective on the behaviors and functions of technical artifacts. *Artif. Intell. Eng. Design, Analysis and Manuf.*, 23:3–21, 2009.

[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.

[DSS11] Dominik Dietrich, Lutz Schröder, and Ewaryst Schulz. Formalizing and Operationalizing Industrial Standards. In Dimitra Giannakopoulou and Fernando Orejas, editors, *Fundamental Approaches to Software Engineering, FASE 2011*, LNCS. Springer, 2011.

[DTBK09] Olga De Troyer, Wesley Bille, and Frederic Kleinermann. Defining the Semantics of Conceptual Modeling Concepts for 3D Complex Objects in Virtual Reality. In Stefano Spaccapietra and Lois Delcambre, editors, *J. Data Semantics XIV*, volume 5880 of *LNCS*, pages 1–36. Springer, 2009.

[FKST10] Marco Franke, Patrick Klein, Lutz Schröder, and Klaus-Dieter Thoben. Ontological Semantics of Standards and PLM Repositories in the Product Development Phase. In Alain Bernard, editor, *Proc. 20th CIRP Design Conference 2010*. Springer, 2010. To appear.

[GHM98] John Grundy, John Hosking, and Rick Mugridge. Inconsistency management for multiple-view software development environments. *IEEE Transactions on Software Engineering*, 24(11):960–981, 1998.

[Grä04] Hans-Gert Gräbe. The SymbolicData GEO Records – A Public Repository of Geometry Theorem Proof Schemes. In *Automated Deduction in Geometry*, number 2930 in LNCS, pages 67–86. Springer Verlag, 2004.

[Ian12] Mihnea Iancu. Management of Change in Declarative Languages. Master's thesis, Jacobs University Bremen, 2012.

[INC] International Council on Systems Engineering Website. `http://www.incose.org`.

[ISO94] ISO 10303-1, Industrial automation systems and integration — Product Data Representation and Exchange, Part 1: Overview and fundamental principles. International Organization for Standardization, 1994.

[Jar98] M. Jarke. Requirements Tracing. *Communication of the ACM*, 41(12), 1998.

[KCD$^+$11] Michael Kohlhase, Joe Corneli, Catalin David, Deyan Ginev, Constantin Jucovschi, Andrea Kohlhase, Christoph Lange, Bogdan Matican, Stefan Mirea, and Vyacheslav Zholudev. The Planetary System: Web 3.0 & Active Documents for STEM. *Procedia Computer Science*, 4:598–607, 2011. Finalist at the Executable Paper Grand Challenge.

[KK12] Andrea Kohlhase and Michael Kohlhase. Spreadsheets with a Semantic Layer. *Electronic Communications of the EASST: Specification, Transformation, Navigation – Special Issue dedicated to Bernd Krieg-Brückner on the Occasion of his 60th Birthday*, 2012. in press.

[KLSS09] Michael Kohlhase, Johannes Lemburg, Lutz Schröder, and Ewaryst Schulz. Formal Management of CAD/CAM Processes. In Ana Cavalcanti and Dennis Dams, editors, *16th International Symposium on Formal Methods (FM 2009)*, number 5850 in LNCS, pages 223–238. Springer Verlag, 2009.

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

[Lem08] Johannes Peter Lemburg. *Methodik der schrittweisen Gestaltsynthese*. PhD thesis, Fakultät für Maschinenwesen, RWTH Aachen, 2008.

[MLD09] Jose Martinez Lastra and Ivan Delamer. Ontologies for Production Automation. In Tharam Dillon, Elizabeth Chang, Robert Meersman, and Katia Sycara, editors, *Advances in Web Semantics I*, volume 4891 of *LNCS*, pages 276–289. Springer, 2009.

[MML07] Till Mossakowski, Christian Maeder, and Klaus Lüttich. The Heterogeneous Tool Set. In Orna Grumberg and Michael Huth, editors, *Proceedings of the 13th International Conference on Tools and Algorithms for the Construction and Analysis of Systems TACAS-2007*, number 4424 in LNCS, pages 519–522, Berlin, Germany, 2007. Springer Verlag.

[MSS03] Andreas Maier, Hans-Peter Schnurr, and York Sure. Ontology-Based Information Integration in the Automotive Industry. In *The Semantic Web, ISWC 2003*, volume 2870 of *LNCS*, pages 897–912. Springer, 2003.

[Pra01] Michael J. Pratt. Introduction to ISO 10303 – the STEP Standard for Product Data Exchange. *J. Comput. Inf. Sci. Eng*, 1:102–103, 2001.

[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.

[Wu94] W.-T. Wu. *Mechanical Theorem Proving in Geometries*, volume 1 of *Texts and Monographs in Symbolic Computation*. Springer, 1994.