Journal of Industrial Engineering and Management

JIEM, 2011 – 4(4):669-698 – Online ISSN: 2013-0953 – Print ISSN: 2013-8423

http://dx.doi.org/10.3926/jiem.329

What do information reuse and automated processing require in engineering design? Semantic process

Ossi Nykänen, Jaakko Salonen, Mikko Markkula, Pekka Ranta, Markus Rokala, Matti Helminen, Vänni Alarotu, Juha Nurmi, Tuija Palonen, Kari T. Koskinen, Seppo Pohjolainen

Tampere University of Technology (FINLAND)

ossi.nykanen@tut.fi; jaakko.t.salonen@tut.fi; mikko.markkula@tut.fi; pekka.a.ranta@tut.fi; markus.rokala@tut.fi; matti.helminen@tut.fi; vanni.alarotu@tut.fi; juha.t.nurmi@tut.fi; tuija.palonen@tut.fi; kari.t.koskinen@tut.fi; seppo.pohjolainen@tut.fi

Received January 2011
Accepted September 2011

Abstract:

Purpose: The purpose of this study is to characterize, analyze, and demonstrate machine-understandable semantic process for validating, integrating, and processing technical design information. This establishes both a vision and tools for information reuse and semi-automatic processing in engineering design projects, including virtual machine laboratory applications with generated components.

Design/methodology/approach: The process model has been developed iteratively in terms of action research, constrained by the existing technical design practices and assumptions (design documents, expert feedback), available technologies (pre-studies and experiments with scripting and pipeline tools), benchmarking with other process models and methods (notably the RUP and DITA), and formal requirements (computability and the critical information paths for the generated applications). In practice, the work includes both quantitative and qualitative components.

Findings: Technical design processes may be greatly enhanced in terms of semantic process thinking, by enriching design information, and automating information validation and transformation tasks. Contemporary design information, however, is mainly intended for human consumption, and needs to be explicitly enriched with the currently missing data and interfaces. In practice, this may require acknowledging the role of technical information or knowledge engineer, to lead the development of the semantic design information process in a design organization. There is also a trade-off between machine-readability and system complexity that needs to be studied further, both empirically and in theory.

Research limitations/implications: The conceptualization of the semantic process is essentially an abstraction based on the idea of progressive design. While this effectively allows implementing semantic processes with, e.g., pipeline technologies, the abstraction is valid only when technical design is organized into reasonably distinct tasks.

Practical implications: Our work points out a best practice for technical information management in progressive design that can be applied on different levels.

Social implications: Current design processes may be somewhat impaired by legacy practices that do not promote information reuse and collaboration beyond conventional task domains. Our work provides a reference model to analyze and develop design activities as formalized work-flows. This work should lead into improved industry design process models and novel CAD/CAM/PDM applications, thereby strengthening industry design processes.

Originality/value: While extensively studied, semantic modeling in technical design has been largely dominated by the idea of capturing design artifacts without a clear rationale why this is done and what level of detail should be favored in models. In the semantic process presented in this article, the utility and the chief quality criteria of semantic models (of technical information and artifacts) are explicitly established by the semantic processing pipeline(s). This constructively explains the significance of semantic models as communication and information requirement interfaces, with concrete use cases.

Keywords: semantic process, design process, engineering design, engineering information management, pipeline data processing

1 Introduction

Designing complex machines involves the creation of design sketches and blueprints of various types. In addition to the primary objective of documenting the machine design for purposes of implementation and production, the design information can be utilized also in other applications. These include simulations, technical documentation, and virtual laboratory applications for training purposes.

According to the current practice, producing material that is not directly related to core machine design is often considered merely as a "secondary" objective. Further, the related secondary tasks are not always directly linked to the primary design activities with clear requirements. As a consequence, producing the artifacts related to the secondary applications is sometimes detached from the primary design process: It might be performed by different people with other tools, perhaps even re-engineering data (implicitly) present in the early design process. However, to reduce development costs, and to meet the needs of the other designers, the requirements and information flows between actors need to be acknowledged throughout the design process. Motivation for this is relatively clear: In an ideal case, many secondary applications, such as part catalogues and visualizations, could be programmatically generated from the original, rich enough design data.

In this article, we present an information processing architecture that captures and reuses the flow of semantics-aware technical information in a design process from design information systems to the primary and secondary applications. Our main use case is semi-automatically generating virtual machine laboratories from existing design information, including simulations for virtual prototyping purposes.

The main contribution lies in elaborating and explaining the underlying *semantic process* related to machine design. Compared with the state of the art, instead of insisting a central data repository or a toolset, we emphasize the information protocols between different design activities in design. This yields certain minimalism in process planning: It captures the critical information flow in a design process, but leaves room for design culture specific organization of individual design tasks and tools. Our work culminates into introducing a novel, semantic process model for analyzing and managing design information flows. This provides an efficient method for reusing design information using semantic data processing pipelines. We also demonstrate this by generating a simulation model from design data, utilizing a library of general-purpose simulations blocks.

While semantic modeling methods are increasingly adopted in managing design information structures, pipeline-oriented semantic modeling is rather new. From our application point of view, semantic process perspective enables the low-cost semiautomatic generation of virtual machine libraries and other work products, secondary to traditional manufacturing. We also believe that the instructional aspect of this work very important, since it provides organizations conceptual tools for understanding and benchmarking their design processes, and promotes individual designer awareness (design as service vs. design as solo activity).

We present our work in the context of a specific Semogen research project (phase I during 2010-2011) which studies industrial virtual laboratory production methods in the context of semantic modeling. Our applications are related to mobile rock-drilling machines with a human operator. The project is mainly funded by the Technology Industries of Finland Centennial Foundation and benefits from the expertise of its industrial partners of different domains, including design and manufacturing, CAD/CAM development, documentation, and engineer training.

The rest of this article is organized as follows: After this Introduction, we outline the background of our work in Section 2. In Section 3, we consider the elements of well-defined and reusable design processes in general. In Section 4, we establish an abstract model for semantic data processing, and consider implementations. In Section 5, we present a case study of generating simulations from hydraulics diagrams, and discuss related experiences. Finally, in Section 6, we conclude the article and make notes about the related trends of engineering design.

2 Background and motivation

In the context of progressive design paradigm (Herrman, 2010), improving design process flow requires first considering the process of information structuring for formal reuse (Troussier, Pourroy, Tollenaere & Trebucq, 1999), and then introducing the modern semantic modeling and computing techniques to provide ICT support for the tasks (Zhang & Yin, 2008; Brandt, Morbach, Miatidis, Theißen, Jarke & Marquardt, 2008). A practical solution of the efficient reuse problem involves at least three components: CAD/CAM and simulation environments, product data management and resource planning systems, and data processing tools. By using modern design tools one may readily establish a link between a design activity (such as mechanics design), product data management, and simulation. Commercial products such as SolidWorks and Dynamic Designer provide tool suites for both designing and rapid simulating of specified artifacts. In addition, general-purpose information architectures such as DITA exist for facilitating reuse in technical documentation (Raaphorst & Johnson, 2007).

Considerable research efforts have been made to develop open-source platforms for high level simulation software, such as the simantics.org project which provides an Eclipse-based framework of tools for structural data management and simulations (Järvinen, Puolamäki, Siltanen & Ylikerälä, 2009; Eclipse, 2010a). In general, the process of accessing data in a machine-readable way, or generating simulations, is usually based on reusable blocks or templates and automation is sought from added descriptions (Lucko, Benjamin, Swaminathan & Madden, 2010). Actual modeling has traditionally been conducted according to the Integration DEFinition (IDEF) modeling principles but the development of Unified Modeling Language (UML), and Extensible Markup Language (XML), specifications has introduced an option of using new, perhaps more easily applicable technologies and tools for the task (Noran, 2000). Using more formal models not only enables data integration but also automated inconsistency handling (Almeida da Silva, Mougenot, Blanc & Bendraou, 2010).

Database-centric design paradigm has also renewed its popularity. The recent TIKOSU project (2009-2011) emphasizes database centric design of machine control systems by defining the linkage between these artifacts across process boundaries, establishing a centralized, single source data repository. As a part of its activities, the project reviewed several system engineering data models such as Föderdal information architecture, AutomationML, PG-Pla-INC project, GENESYS project, Vector informatik’s eAsee tool, CANopen XML specification, PLCopen XML specification, AP233 of ISO 10303, ASAM automotive specifications (Alanen et al., 2011). It is noteworthy that according to the current systems, formal design information models typically emphasize "static" data structures and not to the automated pipeline processing aspect of the design information flow between different designers and design tasks. Indeed, some authors suggest that integrated design should cover both data/knowledge modeling and process planning aspects (Ramana & Rao, 2004). In particular, in design systems, tool integration is often largely accomplished by data transfer or data integration via a central data store, neglecting the requirements of the work processes, and communication in the design team is only supported by generic tools like e-mail, video conferences, etc., which are not integrated with engineering design tools (Marquardt & Nagl, 2004).

However, well-established technologies do exist for asserting general requirements and protocols on the business logic and on the structural information level; consider OASIS Web Services Business Process Execution Language (OASIS, 2007), Ant processor DITA (Raaphorst & Johnson, 2007), or XProc: An XML Pipeline Language (Walsh, Milowski & Thompson, 2010). Combined with traditional and modern engineering design (Pahl & Beitz, 1996; Airila, Pietola & Kuuva, 2001), these provide the necessary processing methods also for design engineers. In turn, however, this shifts the attention from the manually managed global repositories to the information signaling interfaces between design tasks.

In Finland, methods for strengthening design processes have been extensively studied over the past several years. Perhaps the two most significant research initiatives include the national Technology programme of Mechanical Engineering MASINA 2002-2007 (Tekes, 2008) and the national Technology programme of Digital Product Process DTP 2008–2012 (Tekes, 2010). In addition to the particular advancements, the strategic message is relatively clear: First, the simultaneous design of different technical processes is inevitable in order to develop modern, optimized products (Lehtonen, 2006). Second, digitalization of the design (product) processes, in particular during the early stages of design play a key role in establishing competitive advantage in companies of the global market (Ventä, Taklo, & Parviainen, 2007). The entire domain of mechanical engineering research in Finland has also been evaluated in 2000-2007 by international experts (Lensu, 2008). Among other things, the main recommendations include putting more emphasis on fundamental research supporting the long-terms needs of industry and encouraging interdisciplinary collaborations in engineering research.

For us a very practical motivation for studying the semi-automatic generation of virtual machines originates from already (manually) implementing several laboratory environments with industry partners (Ranta, 2005; Palonen, Leino, Koskinen, Ranta, Punki, & Mäkelä, 2007; Markkula, Rokala, Palonen, Alarotu, Helminen, Koskinen, Ranta, Nykänen & Salonen, 2011), and concluded that many of the steps could be automated. However, to do so, the semantic information flows in the machine design process should be elaborated. With this respect, our related work includes, e.g., semantic modeling, computing, and interpretation studies (Nykänen 2007; Nykänen, 2009a; Nykänen 2009b) and research about the various aspects of designing and modeling hydraulics systems (Leino, Koskinen, & Vilenius, 2005; Markkula, 2009; Virta, Aaltonen, Koskinen & Vilenius, 2009).

However, practice shows that the systemic complexity of design information, systems, and methods is a fundamental issue. Also, many design and consulting companies are quite small which raises practical concerns. For instance, about 80% of the 223 Finnish Association of Consulting Firms SKOL (2010) member organizations in Finland have less than 31 employees. Thus, adopting new design practices boils down to process ownership and HR management: Introducing the role of a knowledge engineer to lead the semantic design information process. In practice, this role might be distributed among the project manager and the chief engineers. For brevity, we do not consider the managerial aspects in detail in this article (Malhotra, Heine & Grover, 2001; Danilovic & Browning, 2007).

3 Elements of a well-defined and reusable design process

A stereotypical product life cycle includes product planning and marketing, engineering design, manufacturing, order management, production and procurement, and customer delivery and service, including after-sales and maintenance. An ideal engineering design process may further follow the activities of identification of a need, background research, goal statement, performance specifications, ideation and invention, analysis, selection, detailed design, prototyping and testing, and production (Norton, 2008), often aligned with agreement, technical, and evaluation processes. However, if the success and cost indicators of design projects fail to credit "forward thinking", there is a danger of local (over)optimization of individual design activities and tasks. Thus, to optimize the utility of design information, two central challenges of design projects need to be acknowledged. We call these the *coordinated process challenge* and the *semantically rich modeling and computing challenge*.

3.1 Coordinated process challenge

In progressive design, each design step should ideally aim serving the needs of the other steps: The following step(s) should be served with information, the preceding steps with reasonable requirements. The worst-case scenario involves "re-engineering" artifacts of the earlier design stages (Hislop, Lacroix & Moeller, 2004). Good practices enable not only efficient but also sustainable development (Ramani, Ramanujan, Bernstein, Zhao, Sutherland, Handwerker, Choi, Kim & Thurston, 2010). In both cases, initial design decisions play a central role.

Coordinated process issues typically result from unclear practices or missing protocols for exchanging information. Problems typically culminate when professionals from different disciplines meet. For instance, "hard" machine engineering designers are often unaware of the technical information requirements posed by "soft" after-sales application developers, and viceversa.

Perhaps the most widely applied, structured design process framework is found from software engineering: The (IBM) Rational Unified Process (RUP) (Kroll & Kruchten, 2003; Kroll & Royce, 2005). RUP is an adaptable development process framework that is tailored by the development organizations according to their needs. Note that general frameworks also help identifying concrete tools, such as composer and management tools, wikis, and issue trackers (Eclipse, 2010b).

In the machine design context, we may now use RUP for helping to answer the questions like "What are the elements of a good machine design process in general?" or "What elements is a particular machine design process missing?" RUP points out several nice practices applicable in the machine design context:

- Project lifecycle takes place in four phases (in which iterations may take place): Inception, Elaboration, Construction, and Transition. In particular, the Inception and the Elaboration phases suggest setting clear requirements, constraints, and key features before Construction.
- Work is takes place around Roles (who), Work Products (what), and Tasks (how). This provides concepts for identifying a specific design task in a system of manageable units and interfaces.
- There are six engineering disciplines: Business and Modeling, Requirements, Analysis and Design, Implementation, Test, and Deployment. Thus, system engineering is not "only technical designing"; specific design tasks must be accompanied with e.g. process management which in our case requires the identification of and coordination with other activities.
- There are also supporting disciplines, including Environment, Configuration and Change Management, and Project Management. In our case, the Configuration and Change Management suggests acknowledging the configuration (versioning) management of design information.
- General best practices minimize faults and maximize productivity, namely: Develop iteratively, Manage requirements, Use components, Model visually, Verify quality, and Control changes. These serve as rules of good engineering in general, suggesting active collaborations and seeking means to use external tools and representations to manage the process.

This helps identifying success criteria for well-coordinated design processes:

- The measures of design productivity and costs must be established on an appropriately high management level, based on Business and Modeling and/or Project Management principles. (Not only within a particular phase or "core" engineering discipline.)

- The roles of designers and the work products need to be explicitly identified, setting concrete requirements and protocols between project activities. (Not only agreed informally between expert designers.)

Equipped with this insight, we will next consider our second challenge.

3.2 Semantically rich modeling and computing challenge

The second challenge results from the inherit complexity of an engineering process. Ideally, the design process is properly modeled (OMG, 2008) and the relevant design information can be read from the properly encoded design documents. In practice, processes may be poorly described and reading design documents may be difficult for humans and impossible to software. Writing out implicit information in design documents, however, may introduce additional work. Thus, concrete benefits must be pointed out by project management-level indicators.

The first step in addressing the challenge lies in understanding the typical elements of information in a design task. In practice, even very restricted design tasks involve using lots of models, specifications and model-specific attributes. Troussier, Pourroy, Tollenaere and Trebucq (1999) identify over twenty categories of attributes related to a particular well-defined mechanical design analysis case. Ideally, these specifications would automatically match the needs of the other design activities. In practice, however, this requires sufficiently formal *information interfaces* between design activities. Since information needs may vary upon application (e.g. a simulation model or a simple pass/fail simulation outcome), information interfaces should be introduced upon concrete needs.

In this context, semantic modeling means writing out the meaning of the given structures with respect to a certain application in a machine-processable way, without having to understand them in per se. This typically involves describing the classifications and relationships of the design artifacts with respect to some common domain model and/or theory. For instance, a specific block in a CAD design might be described as a component of a specific type, drawn from machine component ontology (e.g. as a pump). This added piece of information would then allow automatically integrating the block with a simulation model in a later design activity (e.g. with a generic Simulink "pump" model with few parameters). Technically, this requires three things: First, *identifying* the desired (e.g. CAD) block with a well-defined reference, second, *classifying* the identified object with respect to some controlled vocabulary, and third, *acknowledging the context of this information* with respect to a semantic design and application process. Equipped with this information, one can then semantically compute with the data.

By *semantic computing* we mean "computing with (machine processable) descriptions of content and (user) Intentions" (IJSC, 2010). In our context, this means describing data semantics with respect published common data models and theories, and then making queries and other computations on the level of the encoded descriptions. For instance, simply knowing that a component "a10" in a hydraulics design is a kind of a hydraulic pump, that hydraulic pumps are machines (now in the sense of abstract physics), and that machines typically create heat as a byproduct, allows pinpointing components that produce heat in a system.

Typical machine design activity takes place iteratively, starting from problem definition and sketching. In a semantic process, these should capture both design insight and information requirements. For instance, a semantically rich, machine readable sketch of a machine simulation model not only captures the design insight of a particular simulation implementation, but also points out what information is assumed to be requested from the preceding design activities (such as mechanical design, component list, and machine-specific parameters and attributes). Optimizing the global design cost might suggest moving information production responsibilities downwards in the global information consumption food chains, to a point where further delegation would introduce additional costs.

When considering specific technologies for the semantic modeling part, good candidates exist. Considering the modeling part, perhaps the most obvious general-purpose technologies for semantic modeling are provided by the W3C Semantic Web technologies (W3C, 2010; Gómez-Pérez, Fernández-López & Corcho, 2004; Ellemang & Hendler, 2008). Perhaps surprisingly, adopting a good, specific semantic machine description (etc.) vocabulary is much more difficult. While general-purpose technical vocabularies exist (including the licensed ones, see, e.g., (SFS, 2008) and similar standards), precise global vocabularies are not always available or used in hands-on design – particularly when PDM systems does not support or require this and when designers adopts ad hoc terminologies. Thus, unless using standard classifications is systematically insisted on the level of project quality control, at the early stages of adoption, semantic processes are likely to capture "only" the terms of the local design culture. Using standard names, however, is crucial for machine processing and transfer. Once a proper process is followed, benefits cumulate and motivating should become easier.

From the semantic computing perspective, we may perceive the design process in terms of a *semantic data processing pipeline*: The data processing pipeline consists of tasks pointed out by the project coordination, where each task transmits information as requested by the design (sketching) activities. Depending on the technical design of the pipeline and the formality of the data, the data processing pipeline may be used for documenting information interfaces, validating information requirements, and even automatically processing technical design data and thus generating new applications. Pipelines may also encode information that is simply useful for informal designer communication (common map or reference).

We may now outline success criteria for semantic modeling/ computing as follows:

- Semantic modeling that is required to capture design information has two main use cases: Capturing design insight for the purposes of the particular engineering task and asserting formal information requirements for the preceding activities. In other words, design activities are not independent, but linked though well-defined inputs and outputs.
- For purposes of automated information processing, a progressive design process may be technically modeled as data processing pipeline which also points out a framework for particular semantic computing components.
- While acknowledging the coordination challenge, the specific organization of a data processing pipeline should reflect the actual organizational structure. In other words, the pipeline structure is due to both the abstract information requirements and the concretely assigned employees' tasks.

In practice, this may require re-thinking designer roles and responsibilities.

4 Thinking in terms of semantic processes

In a design organization, we may identify three kinds of top-level processes:

- The business process, which sets the general objectives, success indicators and constrains on the organizational level. If the business process fails on a critical level, the entire organization may break down.

- Organizational quality control processes, which among other things define the stereotypical structure of design projects within the organization. This typically includes setting the principles of staffing, process activities and models, documentation practices, tools, project-level progress measures, and organizational learning feedback loops from the completed projects.

- Concrete project processes, which are related to ongoing projects with specific objectives, resources, and timetables. In our case, these are typically engineering design projects for external customers. If the project processes succeed, a learning organization will perform better and is likely to favor the successful project patterns in the future.

Now, semantic process thinking appears both on the level of quality control processes (2) and on the level of concrete project processes (3), within the constraints of the business process (1). In particular, appreciating semantic process thinking on the level of organization quality control (meta) processes requires acknowledging the fact that besides tools, also common project policies and protocols are needed. Further, unless local activity costs are analyzed with respect to global cost savings, there is a risk of trying to (in error) minimize business costs by minimizing each design activity costs individually.

Equipped with this insight, let us then analyze semantic processes on a project level. Assume we would like to generate virtual machine laboratory applications from design information, e.g., prototyping or training purposes. Consider capturing the flow of information in a related machine design process, as depicted in an intuitive example depicted in Figure 1.

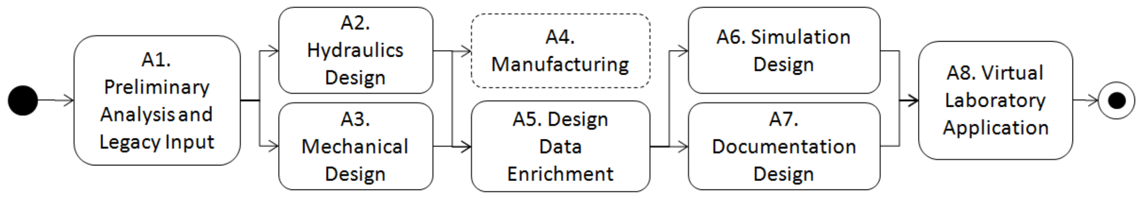


Figure 1. An intuitive example of a semantic design process

Clearly, the example is an oversimplification since, e.g., all activities, feedback loops (e.g. negotiating information requests and responsibilities), and internal iterations with sub-activities are not specified. Nevertheless, it helps pinpointing critical properties of the task: Organize work into modules of activities with clear information interfaces, set requirements for critical information to enable fluent information flow and reuse, establish a working culture (and indicators) to support rational division of labor, and provide tools to designers both for transforming data, generating applications and for validating the information they produce. (Compare, with requirements-based testing in software engineering as in Sommerville, 2004).

Note that semantics appears here in three complementary roles:

- Designer communication and design process management, e.g., sharing a common terminology and asserting well-formed information requirements.
- Automated data processing, e.g., generating drafts of documentation or simulator models utilizing enriched technical design data with machine-processable semantics (reusing data provided by the preceding activities).
- Novel end-user applications, e.g., introducing applications that allow end-users to (semantically) search and visualize information based on the well-defined semantic concepts of technical design. (In our case, a complete end-user application includes a semi-automatically generated virtual machine laboratory including interactive, linked simulation models, diagram/3D views, and component catalogues).

Depending on the management objectives, a semantic process may be interpreted informally (a collaboration metaphor for coordination and management), semi-formally (intuitive information interfaces and best practices for encoding technical data), or formally (data interfaces with machine-readable semantics for purposes of automated validation and processing). Note that conceptually, the semantic search provides a strong basement for various kinds of end-user applications, also in cases when the query language is not visible to the end-users. The semantic search can also cover the current or the recorded state(s) of the dynamic simulation and the semantic model might be modified dynamically (by run-time update), which opens up interesting analysis, prototyping and training possibilities.

A successful semantic process should have positive learning and cumulative knowledge gain effects within organizations. In brief, this means two things. First, we would like to minimize the "extra" work effort in enriching design data (activity A5 in Figure 1), by increasing the reusability and utility of the actual design data. Second, we would like to increase the volume of good and reusable legacy input from the past projects, establishing a positive information reuse cycle between projects (activity A1 in Figure 1).

4.1 An abstract model of a semantic data processing system

Let us next focus onto the specific data processing needs of the semantic process.

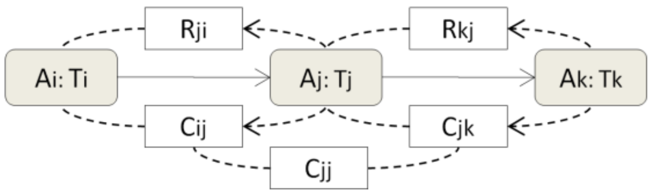


Figure 2. A semantic data processing system: Activities (A), processing targets (T), communications schemas (C), and information requirement schemas (R)

We define a *semantic data processing system* as a quadruple (A, T, C, R) of (directed, feed-forward) graph of *activities* (A), processing targets (T), communication schemas (C), and information requirement schemas (R). For an example, see Figure 2 above.

We may consider a semantic data processing system as a special kind of data processing pipeline (compare with, e.g., Walsh, Milowski & Thompson, 2010). The graph of activities (A) establishes two data processing roles, indicated with the directed arcs: *information provider* and *information requestor* (or consumer). The graph points out the *critical information paths* of the system: When an activity A_j requests information from a providing activity A_i (the graph includes a directed arc from A_i to A_j), processing A_j as a *processing target* T_j depends on the activity A_i.

The *communication schemas* define the basic communication protocols between the directly connected activities. When activity A_j requests information from a providing activity A_i, a formal communication schema C_{ij} may be asserted. This enables validating that A_i transmits information in a format that A_j claims to understand. Besides rudimentary structures and references, communication schemas typically introduce shared controlled vocabularies and formal ontologies.

The *information requirement schemas* (R_{ji}) are used for asserting strong, particular information requirements between the directly connected activities to validate the actual information content. This may include, e.g., checking names, instance data, and cardinalities, and performing arbitrary calculated tests. Note that information requirement schemas are typically more activity-specific than communication schemas. For instance, in a certain application one may adopt the policy of using a single communication schema for the whole network (e.g. RDF/cXML) and then assert specific information requirements per activity. Also, asserting information requirement schemas typically requires a more expressive schema language than asserting mere communication schemas.

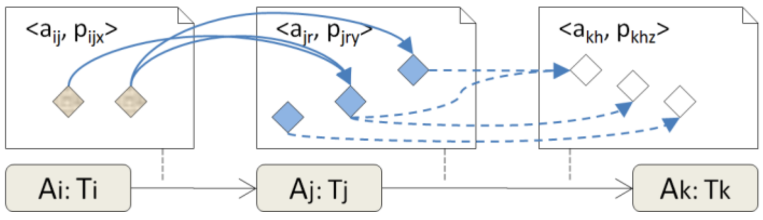


Figure 3. Target processing related to information objects. The arrows represent the transformations of information objects between different activities

An analytical definition of information requirements schemas can be pointed out by organizing the data outputted by an activity A_i into an explicit set of *abstract information objects* a_{ih} associated with (semantic) properties p_{iht}, { $<a_{ih}, p_{iht}>$ } (for an illustration, Figure 3). As a consequence, a processing target T_j may now be perceived as a transformation from the sets of the requested information objects to the set of the provided information objects. This involves both acknowledging the identity of information objects and asserting their properties, where the modeling is due to the actual information requirements.

4.2 Semantic modeling for machine design projects

We may use the term *ontology* to denote an explicit specification of a shared conceptualization. For instance, the classes and instances that are used to describe a machine design might be called ontology. In an abstract treatment, however, the general term ontology should not be confused with a specific knowledge representation language, such as the Web Ontology Language (OWL). In many cases, a less expressive ontology language, such as the SKOS Simple Knowledge Organization Language, RDF Schema, or a even well-designed XML Schema –based markup language, is sufficient. Also, it is possible to interpret some of the formal entailments in applications in a non-orthodox way (Nykänen, 2007).

In engineering, ontologies should be designed with concrete use cases in mind. Following a typical modeling practice (Gómez-Pérez, Fernández-López & Corcho, 2004; Böhms, Leal, Graves & Clark, 2009), we may organize the semantic models related to a semantic process into three layers. These include the top-level management and integration ontologies, various engineering design (domain) ontologies, and specific design (instance) data related to particular design artifacts or products (Figure 4).

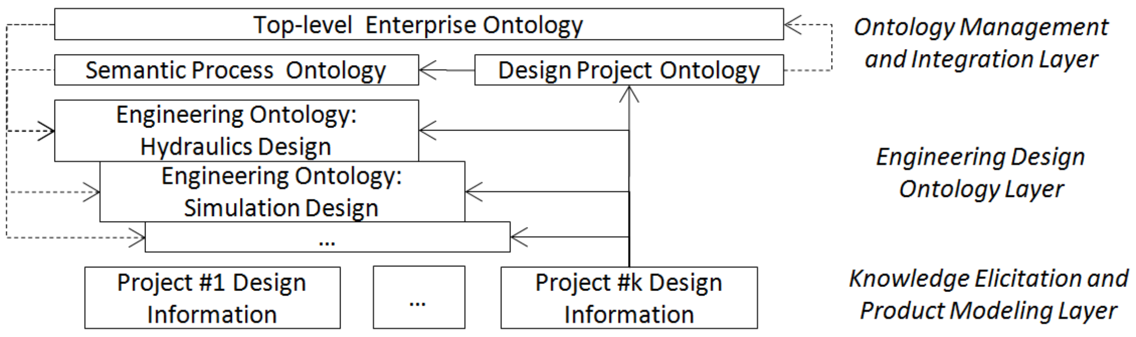

Top-level Enterprise Ontology
Semantic Process Ontology
Design Project Ontology
Engineering Ontology: Hydraulics Design
Engineering Ontology: Simulation Design
...
Project #1 Design Information
...
Project #k Design Information
Ontology Management and Integration Layer
Engineering Design Ontology Layer
Knowledge Elicitation and Product Modeling Layer

Figure 4. A simplified information architecture for machine design

In brief, the *top-level enterprise ontology* provides the common framework for integrating all data within an organization. The *design project ontology* introduces the concepts of the machine design process, such as phases, activities, objectives, work products, tasks, and roles. It also refers to the *semantic process ontology*, which defines the concepts of a semantic process, including targets, critical information paths, and communication and information requirement schemas. Quite obviously, the information captured by the design project and semantic process ontologies should be aligned. Thus, the semantic process should enable capturing *how* to reach the objectives of the project.

The common *engineering design (domain) ontologies* then provide the concepts to capture the design structures related to, say, mechanical, hydraulics, simulation engineering, and virtual machine laboratories. When a formal formulation of the project deliverables and requirements is needed with respect to some application, the design ontologies might be complemented with *application ontologies*. This might be required for validating project deliverables, or simply translating information to end-user concepts in applications.

Finally, the knowledge elicitation and product modeling layer encodes *information specific to various projects*. This typically includes the instance data that is produced and consumed in project activities, and interpreted with respect to the ontology framework. Note that while all information in the above information architecture is clearly subject to change and needs thus to be properly versioned, it is usually assumed that the daily hands-on design activities mostly modify information on the knowledge elicitation and product modeling layer. In particular, the pace of evolution of the underlying semantic standards is typically much slower than of the given instance data (Nykänen, 2009a).

Of course, on the implementation level, the ontology landscape might be more detailed. For instance, it typically makes sense to organize ontologies into several modules, e.g. by separating specific constructions from the general-purpose components. Also, separating the top-level enterprise ontology and design project ontology allows extending the design with new ontology components, e.g. related to life-cycle management. In turn, this also suggests complementing product design information with life-cycle information of deployed products. When the overlap between different (3rd party) engineering ontologies becomes problematic, e.g. when the same concept is defined in several places, one may add mapping or translation ontologies, etc.

The above information architecture is linked to semantic data processing pipelines in two major ways. First, the activity and the requirements structure of the semantic data processing pipeline is designed using the concepts of the semantic process ontology and linked with the design project ontology. Second, the communication schemas are designed using the engineering ontologies.

4.3 Notes about implementation

The basic insight of project management lies in recognizing that teams need to be given a clear goal, educated about the process methods, and providing guidance and tools for successfully applying the methods throughout the project. More formally, abstract semantic processing must be linked with the concrete and well-defined project activities. This boils down to well-defined roles, work products, and tasks: People perform certain activity tasks and are responsible for certain validated work products, in concert with the project timeline and objectives.

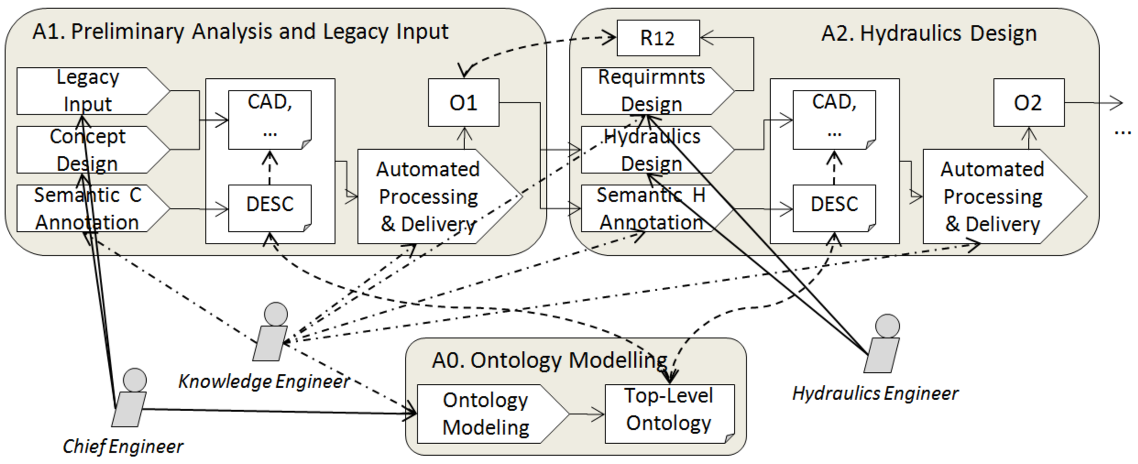


Figure 5. Roles, work products, and tasks in a semantic process

Figure 5 captures a more detailed example of the two activities from Figure 1, namely Preliminary Analysis and Legacy Input and Hydraulics Design, and introduces a previously hidden activity: Ontology Modeling. The basic idea is that three roles are identified (in the scope of the given process fragment): Chief Engineer, Knowledge Engineer, and Hydraulics Engineer. Chief Engineer performs the tasks of Legacy Input identification and Concept Design, and participates in the Ontology Modeling task. Hydraulics Engineer performs the Hydraulics Design task and participates in the Requirements Design task in the Hydraulics Design Activity. Knowledge Engineer performs the tasks of Semantic Annotation, Delivery & Automated Processing, Requirements Design, and Ontology Modeling, in collaboration with the respectful machine design engineers. The identified work products include CAD (etc.) legacy documents, semantic descriptions, formal requirements (as schemas), partially automatically generated activity output documents, and the top-level ontology (etc.). (To make the figure less cluttered, the responsibilities are omitted.)

Perhaps the simplest approach to implement a development process is to model the work breakdown structure and automated data processing in terms of a file system, interpreted with respect to suitable project and semantic process ontologies. In brief, this allows easy versioning of data collaboratively (e.g. with SVN (Apache, 2010a)), applying integrated development environments such as the Eclipse (Eclipse, 2010a), and using general-purpose pipeline data processors such as the Apache Ant (Apache, 2010b) or the Wille Visualisation System (Nykänen, Salonen, Haapaniemi & Huhtamäki, 2008; TUT, 2010). This also allows benchmarking the approach with respect to well-known architectures, such as the Darwin Information Typing Architecture (DITA) (Raaphorst & Johnson, 2007).

Technical semantic modeling may be based on the Semantic Web technologies. Besides the taxonomies for capturing or design cases, we have developed a canonical application profile of RDF/XML called RDF/cXML to capture semantic data in the data processing pipeline. This allows a dual interpretation of data (either XML or RDF interpretation) which means that both XML and RDF schema languages and processing tools are applicable (Nykänen, submitted). Asserting information requirement schemas in ISO Schematron is thus possible. This enables asserting very expressive requirements, even without ontologies in description logic.

Reading data from the legacy file formats requires application-specific adapters. However, the elicitation process involves not only exporting authentic design structures from legacy design systems, but also further describing these structures semantically, when necessary. As a rule of thumb, our application experience suggests that it makes sense to enrich legacy data using relatively simple means: The bottom line is that design practices require that significant structures are appropriately named and grouped for reference. Adding semantic structures to legacy design data (e.g. CAD) makes sense if semantic information can be injected into the editor GUI palette or macro level. A sufficient level of richness of semantic information has been reached when design data, PDM information (etc.), and semantic ontologies can be linked in a programmatic way.

The information requirement schemas provide concrete validation tools for designers, since they allow providing explicit feedback to designers, e.g., as "Your design can be interpreted mechanically only up to 55%", based on an evaluation of related ISO Schematron assertions. However, a major caveat against easily understanding this feedback is that information requirement schemas are typically evaluated with respect to the exported information (in XML), not the editing view in the legacy design system (e.g. a visual CAD graphical user interface).

5 A Technical case study

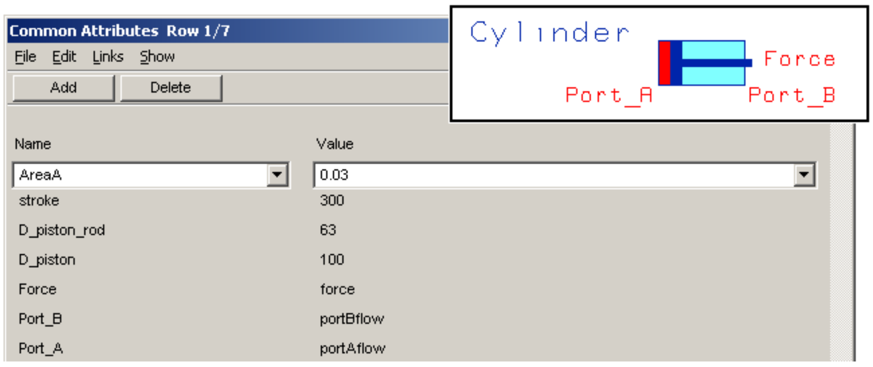
To demonstrate our approach in practice with specific technologies, we shall next discuss a simple design case using the data provided our industry stakeholders.

5.1 Scope and design

We received real-world design data of an operator-driven rock drilling machine. This source covers multiple aspects of the machine's design, including hydraulic diagrams, mechanical models and controller-area network (CAN) design. We also had access to previously implemented simulation models of individual machine components. The material was mostly provided in original formats, including confidential reference information (in Vertex HD, CANopen, and PDF formats). We shall next consider a case study based on a restricted subset of the material, with the general objective to semi-automatically generate simulation models from the semantic information present in the hydraulic diagrams. Simulation models for other design domains such as CAN, can be generated similarly.

The pivotal information source was a hydraulic diagram that was created with the Vertex HD hydraulic design software, with added semantic information. The diagram was drawn to model a simplistic, yet fully functional hydraulic system containing the following parts: a cylinder, a control valve, a pressure relief valve, a filter, a tank, a pump and connections (hoses and pipes) between components. In the HD design activity, the full hydraulic diagram was composed by creating copies (instances) of the macros and connecting these macros together with hydraulic volumes (hoses), where volumes are connection lines with type code. Bridging different models succeeds only if shared schemas are followed. Thus, since hydraulics designers do not typically follow a formal typing or naming scheme for objects, the names of components and properties needed to be corrected.

In this case, the simulations were to be compiled from a pre-defined library of parameterized simulation models of hydraulic etc. components. In brief, this allowed mapping models with the enriched reusable macros of the CAD drawings. Programmatic access to design information was due exporting well-defined structures from the Vertex HD format using a proprietary SVG adapter, thus providing access to data in relatively simple XML.



Name	Value
AreaA	0.03
stroke	300
D_piston_rod	63
D_piston	100
Force	force
Port_B	portBflow
Port_A	portAflow

Figure 6. A depiction of a macro with its common attributes in Vertex HD

To provide semantic data about components that could be used in later stage of generation process, we also included common attributes to the CAD drawings. For instance, in a cylinder (Figure 6), common attributes were used to store information about the related area, diameter and different forces that could be later used as parameters in simulation. In production, some of this information should be included to a mechanical model or PDM since it is typically not included in the hydraulics design. However, the information which connects the structure and the semantics of the hydraulic diagram with the structure and semantics of the simulation models, should be part of the semantic processing pipeline definition.

5.2 Generating a simulation model

Once the semantic process of the case study was specified, it pointed out an automated generation process, performed in several steps. First, the hydraulics diagram designed in Vertex HD was first exported from a proprietary format to XML/SVG for further processing. Second, information from the exported SVG was extracted into a canonical form in RDF/cXML:

```
<!ENTITY semogen "http://www.tut.fi/projects/semogen/" > ...
```

```
<rdf:Description rdf:ID="comp-3_21">
  <rdf:type rdf:resource="&semogen;hydraulicCylinder"/>
  <dc:identifier>E3_21</dc:identifier>
  <dc:title>Cylinder</dc:title>
  <semogen:pistonDiameter>100</semogen:pistonDiameter>
  <semogen:hasPort rdf:resource="#port-2_21"/>
  <semogen:hasPort rdf:resource="#port-2_22"/>
  <semogen:hasPort rdf:resource="#port-2_23"/>
</rdf:Description>
<rdf:Description rdf:ID="port-2_21">
  <rdf:type rdf:resource="&semogen;hydraulicPort"/>
  <dc:title>Piston</dc:title>
</rdf:Description>
```

Hydraulic components, ports and their connections (pipes) are modeled as RDF resources with references to a project-scope RDF-based hydraulics schema. Selected common attributes are translated into RDF properties, in order to carry the information objects encoded into design documents into the semantic model.

As a third step in the generation, the templates of component-specific simulation models (selected from the predefined Simulink library), with generation-specific parameters, are linked to the RDF/cXML data. For instance, components typed as hydraulic cylinders are linked to a cylinder template in the schema (e.g. triple *semogen:hydraulicCylinder semogen:hasSimulationModel "cylinder"*).

Finally, a custom generator component (using Apache Ant and Python) was used to output a script that generates the simulation model. In this step component- and instance-specific parameters are also added. For instance, in order to generate the simulation block for the cylinder, the following code can be generated:

```
add_block('Semogen/Hydraulics/Cylinder', 'system1/comp-3_21');
set_param('system1/comp-3_21', 'D_piston', 100);
set_param('system1/comp-3_21', 'stroke', 300);
```

Connecting signal lines due hydraulic pressure lines can also be added by scripting.

As suspected, the maturity of the resulting Simulink model depends on the well-formedness of the source data. Outside the laboratory environment, it may be tricky to connect signals between Simulink components in a fully automated fashion, since this requires careful book-keeping of names (Figure 7).

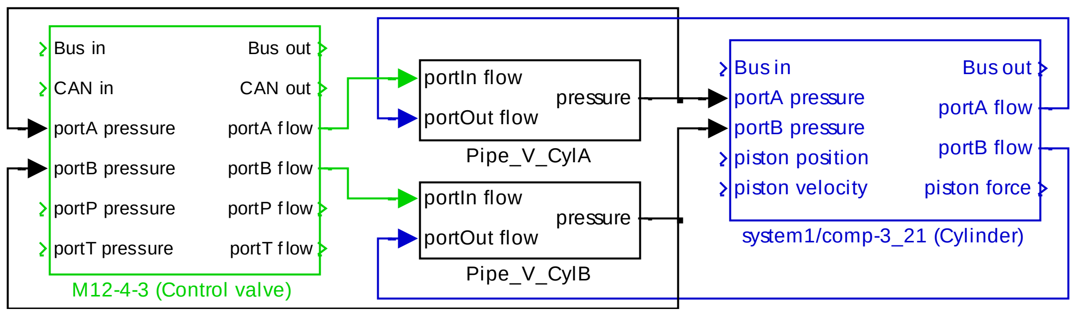


Figure 7. Excerpt of a simulation model in Simulink

However, once the generation pipeline utilizing the semantic process is functional, changing the configuration and the parameters of HD models is relatively easy, and the generation can be quickly recomputed. Designers can also modify the simulation models directly in Simulink. This also provides a natural way to develop new component simulation models, to be added to the library for automatic generation. Note that some of the global properties of the simulation model are partly due to the properties of the individual component models. For instance, an acceptable step size of the global simulation model typically depends on the design of particular simulation component. Once generated, the simulation model does not Simulink since an executable can be generated, including a network interface which provides simulated data for applications, such as browser-based view applications.

5.3 Discussion

While manual engineering is still needed both in the hydraulic diagram design and in design of the simulation components, the above semantic process implementation allows a notable part of the simulation to be automatically generated. As the generation is based on the semantic model, simulation models and the configurations of components can be added in a similar fashion, regardless of their design domain (hydraulics, CAN, mechanics, etc.). We may thus consider the generation application as configuration management problem: Doing things for the “second time” gets much easier since old diagrams may be effectively used as templates, and eventually taken into account in CAD macro design.

With respect to the current (legacy) engineering design processes, some challenges exist. First, rigorous codification of names and structures may require rewriting design macros, and the related features in legacy applications. Second, semantic processes and the generation applications introduce needs for new kind of information that are not explicitly produced in a legacy design process. As a consequence, the designer roles need rethinking (who is responsible to create what information). Also, the information placeholders and tools need to be identified (which tool is used and where the data are stored). In addition, the role of component manufactures as information providers requires rethinking. While it currently does not seem realistic to try to generate simulation models from exact component blueprints, it would still be beneficial to retrieve component data in a precise, semantic form. The challenge, of course, lies in identifying the subtle level of detail in the access to sensitive design information so that enough information is available for purposes of fruitful 3rd-party semantic processing (e.g. generation), but not too much (e.g. for purposes of non-licensed component manufacturing).

Considering the tool chain, it seems that to maximize the amount of automation also the design tools need to evolve. For example, representing semantic structures and performing information validation might be considered as natural properties of CAD and authoring tools. Syntactic and semantic extensibility are also important. In particular, while the use of RDF for data modeling in favor of, e.g., proprietary XML-based file formats requires some additional work in the beginning, it pays off in the semantic integration and validation applications in the end.

6 Conclusion

In this article we have identified and specified the concept of a semantic process for engineering design. The basic idea is to analyze a design process from the perspective of critical (semantic) information paths, and to establish a rational work break-down model with well-defined information interfaces for validating and processing design information. We have also demonstrated semantic process thinking in terms of a restricted simulation generation use case, which enables rapid prototyping and simulation of machines based on relatively simple sketches. In brief, we believe that semantic process thinking may be widely applied, and that it helps improving and implementing positive learning and concrete design information transfer policies within organizations.

Our work continues in two fronts, studying the issues of particular design applications, and adapting the semantic process model in the context of mainstream process tools. In particular, semantic process thinking should become even more attractive when a complete product life is considered: The machine end-user application/after-sales phase is typically much longer (and profitable) than the design phase. In turn, we expect that this work might also lead into improved industry design process models and novel CAD/CAM/PDM applications.

In applications, the main challenge lies in managing the inherit complexity of the task, with respect to both coordination and semantic computing. As such, the semantic process may also be simply used as a model for existing process analysis. Regardless of the level of implementation, however, one *must* take both the technical and the social system properties into account. Experience shows that concerns related to personal credit, intellectual property rights, and protection of business-sensitive information lie in the very core of processes, and need to be acknowledged in concrete projects. Indeed, the significance of proper policy management (clear rules for giving and receiving credit) gets typically highlighted in general data sharing systems research (Smith, Seligman & Swarup, 2008).

Besides the particular applications of the Semogen research project, motivation for this work can be found from the broader context of design engineering. According to the Finnish Association of Consulting Firms SKOL (2010), the volume of services of its member companies (mostly industrial and construction) in 2009 was 1.2 billion euros which means a big design business. Manufacturing and design industries, however, are undergoing significant structural changes in industrialized countries. Large companies tend to move manufacturing to countries "close to customers", and/or to countries with cheaper production costs. Design industry will likely follow this trend, and countries (losing jobs) and organizations (losing customers) suffering from globalization need to rethink their strategies. In Finland, a recent national report about digital design/product process points out that 1) "the more challenging design activities" and 2) the design activities that are able to "increase their productivity/extent of value added", are more likely to remain in Finland (Ventä, Taklo & Parviainen, 2007: page 25). Still, it is estimated that up to 15-20% of the (design) services are at risk. Thus, the importance of introducing more powerful methods to design industry must be acknowledged.

Acknowledgements

This work has been supported by the Technology Industries of Finland Centennial Foundation, and the partners of the Semogen project. We wish to thank both the organizations and the individual experts that have helped us throughout this work.

References

Airila, M., Pietola, H., & Kuuva, M. (2001). *Smart machines and systems: Recent advances in mechatronics in Finland.* Helsinki University of Technology publications in machine design, 1/2001.

Alanen, J., Vidberg, I., Nikula, H., Papakonstantinou, N., Pirttioja, T., & Sierla, S. (2011). *Engineering data model for machine automation systems engineering data model for machine automation systems*. VTT Tiedotteita – Research Notes 2583.

Almeida da Silva, M.A, Mougenot, A., Blanc, X., & Bendraou, R. (2010). Towards automated inconsistency handling in design models. *Lecture Notes in Computer Science*, 6051/2010, 348-362.

Apache (2010a). *Apache subversion home page.* The Apache Software Foundation. http://subversion.apache.org/ - Accessed 20th December 2010.

Apache (2010a). *Apache Ant home page.* The Apache Software Foundation. http://ant.apache.org/ - Accessed 20th December 2010.

Brandt, S.C., Morbach, J., Miatidis, M., Theißen, M., Jarke, M., & Marquardt, W. (2008). An ontology-based approach to knowledge management in design processes. *Computers & Chemical Engineering*, 32(1-2), 320-342. http://dx.doi.org/10.1016/j.compchemeng.2007.04.013

Böhms, M., Leal D., Graves H., & Clark (Eds.) (2009). Product modelling using Semantic Web Technologies. W3C Incubator Group Report 08 October 2009. http://www.w3.org/2005/Incubator/w3pm/XGR-w3pm-20091008/ - Accessed 20th December 2010.

Danilovic, M. & Browning, T.R. (2007). Managing complex product development projects with design structure matrices and domain mapping matrices. *International Journal of Project Management*, 25(3), 300-314. http://dx.doi.org/10.1016/j.ijproman.2006.11.003

Eclipse (2010a). *Eclipse foundation open source community home page.* The Eclipse Foundation. http://www.eclipse.org/ - Accessed 20th December 2010.

Eclipse (2010b). *Eclipse process framework project (EPF).* The Eclipse Foundation. http://www.eclipse.org/epf/ - Accessed 20th December 2010.

Ellemang, D., & Hendler, J. (2008). *Semantic web for the working ontologist: Effective modeling in RDFS and OWL*. Morgan Kaufmann.

Gómez-Pérez, A., Fernández-López, M., & Corcho, O. (2004). *Ontological engineering: With examples from the areas of knowledge management, E-commerce and the semantic web.* Springer, 2004.

Herrman, J.W. (2010). Progressive design processes and bounded rational designers. *ASME Journal of Mechanical Design*, 132, 1-8. http://dx.doi.org/10.1115/1.4001902

Hislop, D., Lacroix Z, & Moeller G. (2004). *Issues in mechanical engineering design management*. SIGMOD Record, 33(2). http://dx.doi.org/10.1145/1024694.1024726

IJSC (2010). *Home page of the international journal of semantic computing (IJSC).* http://www.worldscinet.com/ijsc/ijsc.shtml - Accessed 20th December 2010.

Järvinen, P., Puolamäki, K., Siltanen, P., & Ylikerälä, M. (2009). *Visual analytics: Final report.* VTT. ISBN 978-951-38-7178-9. http://www.vtt.fi/publications/index.jsp - Accessed 20th December 2010.

Kroll, P., & Kruchten, P. (2003). *The rational unified process made easy: A practitioner's guide to the RUP.* Boston: Addison-Wesley.

Kroll, P., & Royce, W. (2005). *Key principles for business-driven development.* IBM DeveloperWorks. http://www.ibm.com/developerworks/rational/library/oct05/kroll/index.html - Accessed 20th December 2010.

Lehtonen, M. (Ed.) (2006). *Simulation-based design process of smart machines.* VTT Tiedotteita - Research Notes 2349.

Leino, T., Koskinen, K.T. & Vilenius, M. (2005). *Modelling of fluid dynamics in water hydraulic seat valve-investigation of pressure distribution.* Proceedings of the 9th Scandinavian International Conference on Fluid Power, *SICFP'05*, Linköping, Sweden, 1-3 June, 2005, 11.

Lensu, M. (2008). *Mechanical engineering research in Finland 2000-2007.* Publications of the Academy of Finland 5/08, ISBN 978-951-715-698-1.

Lucko, G., Benjamin, P.C., Swaminathan, K., & Madden, M.G. (2010). *Comparison of manual and automated simulation generation approaches and their use for construction applications*. Proceedings of the 2010 Winter Simulation Conference, Baltimore, USA, 5-8 Dec., 2010. http://dx.doi.org/10.1109/WSC.2010.5679006

Malhotra, M.K, Heine, M.L., & Grover, V. (2001). An evaluation of the relationship between management practices and computer aided design technology. *Journal of Operations Management*, 19(3), 307-333. http://dx.doi.org/10.1016/S0272-6963(00)00063-2

Marquardt, W. & Nagl, M. (2004). Workflow and information centered support of design processes—the IMPROVE perspective. *Computers & Chemical Engineering*, 29(1), 65-82. http://dx.doi.org/10.1016/j.compchemeng.2004.07.018

Markkula, M. (2009). *Automatic Simulation Model Generation for Hydraulic and Mechanical Systems.* Master of Science Thesis (Automation Technology), Tampere University of Technology.

Markkula, M., Rokala, M., Palonen, T., Alarotu, V., Helminen, M., Koskinen, K.T., Ranta, P., Nykänen, O., & Salonen, J. (2011). *Utilization of the hydraulic engineering design information for semi-automatic simulation model generation*. Proceedings of the twelfth Scandinavian international conference on fluid power, Tampere, Finland, 18-20 May, 2011, 3, 443-457.

Noran, O.S. (2000). *Business Modelling: UML vs. IDEF.* Griffith University School of Computing and Information Technology. http://www.ict.griffith.edu.au/~noran/Docs/UMLvsIDEF.pdf - Accessed 20th December 2010.

Norton, R.L. (2008). *Design of machinery: An introduction to the synthesis and analysis of mechanisms and machines,* (4rd Ed.). McGraw-Hill.

Nykänen, O. (2007). *Interpretation Logics*. Proceedings of the 1st OPAALS conference, Rome, Italy, 26-27 Nov., 2007. http://matriisi.ee.tut.fi/hypermedia/julkaisut/2007-nykanen-ilogics.pdf - Accessed 20th December 2010.

Nykänen, O., Salonen, J., Haapaniemi, M., & Huhtamäki, J. (2008). *A Visualisation system for a peer-to-peer information space*. Proceedings of OPAALS 2008, Tampere, Finland, 7-8 Oct., 2008, 76-86.

Nykänen, O. (2009a). Semantic Web for evolutionary peer-to-peer knowledge space. In Birkenbihl, K., Quesada-Ruiz, E., & Priesca-Balbin, P. (Eds.) Monograph: Universal, Ubiquitous and Intelligent Web. UPGRADE, *The European Journal for the Informatics Professional*, X(1), February 2009, ISSN 1684-5285, CEPIS & Novática. http://www.upgrade-cepis.org/issues/2009/1/upgrade-vol-X-1.html - Accessed 20th December 2010.

Nykänen, O. (2009b). *Understanding data via an RRS in RDF/XML*. Proceedings of the IADIS International Conference Applied Computing 2009, Rome, Italy, 19-21 Nov, 2009, vol. I, ISBN 978-972-8924-97-3.

Nykänen, O. (Submitted). *RDF in Canonical XML (RDF/cXML): A Canonical, Backward Compatible RDF Serialization Syntax in XML*. The 10th International Semantic Web Conference, Bonn, Germany, 23-27 Oct., 2011.

OASIS (2007). *Web Services Business Process Execution Language Version 2.0: Primer.* OASIS. http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsbpel - Accessed 20th December 2010.

OMG (2008). *Business process model and notation, V1.1.* Object Management Group. http://www.omg.org/spec/BPMN/1.1/PDF/ - Accessed 20th December 2010.

Pahl, G., & Beitz, W. (1996). *Engineering design - A systematic approach, (*2nd Ed.). London: Springer.

Palonen, T., Leino, T., Koskinen, K.T., Ranta, P., Punki, J., & Mäkelä, T. (2007). *TI: Learning environment for training the forest machine mechanics.* Proceedings of the tenth Scandinavian international conference on fluid power, Tampere, Finland, SICFP'07, 21-23 May, 2007, 69-83.

Raaphorst, A., & Johnson, R. (2007). *DITA Open toolkit user guide* (3rd Ed.). DITA open toolkit project, OASIS. http://dita-ot.sourceforge.net/doc/ot-userguide131/xhtml/index.html - Accessed 20th December 2010.

Ramana, K.V. & Rao, P.V.M. (2004). Data and knowledge modeling for design-process planning integration of sheet metal components. *Journal of Intelligent Manufacturing*, 15(5), 607-623. http://dx.doi.org/10.1023/B:JIMS.0000037711.00532.38

Ramani, K., Ramanujan, D., Bernstein, W.Z., Zhao, F., Sutherland, J., Handwerker, C., Choi, J.K., Kim, H. & Thurston, D. (2010). Integrated sustainable lifecycle design: A review. *Journal of Mechanical Design*, 132(9), 1-15. http://dx.doi.org/10.1115/1.4002308

Ranta, P. (2005). Metsäkonesimulaatio-opetus kehittyy hiljaisen tiedon avulla. In Muttonen, J. (Eds.) *ITK'05, Missä pedagogiikka? Interaktiivinen tekniikka koulutuksessa -konferenssi*, Aulanko, Hämeenlinna 20-22.4.2005. Hämeen kesäyliopiston julkaisuja, sarja B.

SFS (2008). *SFS-KÄSIKIRJA 174-3 / Technical documentation. Part 3: Preparation of drawings, diagrams, parts lists and instructions.* Suomen Standardoimisliitto SFS Ry. ISBN 978-952-5650-55-6.

SKOL (2010). *SKOL Visio (Kesäkuu 2010).* Suunnittelu- ja konsulttitoimistojen liitto SKOL ry. ISSN 1457-9073.

Smith, K., Seligman, L., & Swarup, V. (2008). Everybody share: The challenge of data-sharing systems. *IEEE Computer Magazine*, September 2008.

Sommerville, I. (2004). *Software Engineering* (7th Ed.). UK: Addison-Wesley.

Tekes (2008). *Masina – Koneenrakennuksen teknologiaohjelma 2002-2007, loppuraportti.* Tekes 4/2008. http://www.tekes.fi/fi/document/42730/masina_loppuraportti_pdf - Accessed 20th December 2010.

Tekes (2010). *Home page of the tekes digital product process programme 2008–2012.* Tekes. http://www.tekes.fi/programmes/dtp - 20th December 2010.

Troussier, N., Pourroy, F., Tollenaere, M., & Trebucq, B. (1999). Information structuring for use and reuse of mechanical analysis in engineering design. *Journal of Intelligent Manufacturing,* 10, 61-71. http://dx.doi.org/10.1023/A:1008968514421

TUT (2010). *Home Page of the Wille Visualisation System.* Tampere University of Technology. http://wiki.tut.fi/Wille/WebHome - Accessed 20th December 2010.

Ventä, O., Takalo, J., & Parviainen, P. (2007). *Digitaalinen tuoteprosessi. (Selvitysraportti Ver 17.8.2007).* Tekes – the finnish funding agency for technology and innovation.

Virta, P., Aaltonen, J., Koskinen, K.T., & Vilenius, M. (2009). *Experiences on the condition monitoring of military aircraft hydraulic systems.* Proceedings of the sixth international conference on condition monitoring and machinery failure prevention technologies, Dublin, Ireland, 23-25 June, 2009.

W3C (2010). *Home Page of the W3C semantic web activity.* World Wide Web Consortium. http://www.w3.org/2001/sw/ - Accessed 20th December 2010.

Walsh, E., Milowski, A., & Thompson, H.S. (2010). *XProc: An XML pipeline language.* W3C Recommendation 11 May 2010. http://www.w3.org/TR/xproc/ - Accessed 20th December 2010.

Zhang, W.Y., & Yin, J.W. (2008). Exploring semantic web technologies for ontology-based modeling in collaborative engineering design. *International Journal of*

Advanced Manufacturing Technology, 36, 833-843. http://dx.doi.org/10.1007/s00170-006-0896-5