Semantic Enhancement of Virtual Engineering Applications

By

Carlos A. Toro

Submitted to the department of Computer Science and Artificial Intelligence in partial fulfillment of the requirements for the degree of Doctor of Philosophy

At
The University of the Basque Country
Donostia - San Sebastian
2009

Signature of author____

Carlos A. Toro

Certified by : ____

Prof.Dr Manuel Graña Romay (Thesis Advisor)

Certified by : ____

Dr. Jorge Posada (Thesis Advisor)

Semantic Enhancement of Virtual Engineering Applications

by

Carlos A. Toro

Submitted to the Department of Computer Science and Artificial Intelligence on March 10, 2009, in partial fulfillment of the requirements for the degree of Doctor of Philosophy

abstract

The scope of this research is the definition of new methodologies and the creation of new tools that allow the Semantic enhancement of Virtual Engineering Applications (VEA) through the Semantic enhancement of their contained Virtual Engineering Tools (VET). This work combines Virtual Engineering Applications (e.g. Computer Aided Design, Steel Detailing, etc.), and state of the art technologies (e.g. Virtual Reality, Augmented Reality, etc) with Semantic techniques in a Product Life Cycle context.

To my family because they taught me what love really is.

Their support for the always absent son gave me the strength to carry on my path.

To my friends through all these years, you are the force behind this effort.

To music, the most important fuel of my thoughts.

Grams, the imprint of you is left on my soul. I miss you.

Originality Statement

I hereby declare that this submission is my own work and to the best of my knowledge it contains no materials previously published or written by another person, or substantial proportions of material which have been accepted for the award of any other degree or diploma at The University of the Basque Country or any other educational institution, except where due acknowledgement is made in the thesis. Any contribution made to the research by others, with whom I have worked at The University of the Basque Country or elsewhere, is explicitly acknowledged in the thesis. I also declare that the intellectual content of this thesis is the product of my own work, except to the extent that assistance from others in the project's design and conception or in style, presentation and linguistic expression is acknowledged.

The author hereby grants to University of the Basque Country permission to reproduce and to distribute copies of this thesis document in whole or in part.

Signed:____________________________________

"I was born not knowing and have had only
a little time to change that here and there."

Richard Feynman, Letter to Armando Garcia J, December 11, 1985

Acknowledgements

I truly believe the most surprising discovery I made in the course of this research, are the people that have surrounded me over the years. This thesis exists due in great part to my family, friends and advisors. I would like to take the opportuntity, in the following few paragraphs, to express my sincere gratitude. Firstly to my family, despite the distance they have always managed to make me close and never lacking in my need for their love or comfort. Everyday my unconditional love and respect for them grows. From my professors as an undergraduate in Colombia, I learnt the beauty of science. I would like to thank Prof. Oscar Ruiz, my first advisor and friend who gave me the opportunities to follow the path of research. His advice, counsel and particular way in getting things done has helped me to understand the difference in doing things and felling them. Prof. Alberto Rodriguez, Dean of the School of Engineering, EAFIT University has been a role model to me, without his encouragement and support I wouldn't be here. His brilliance in teaching has made me understand that something that cannot be taught easily is not worth teaching at all. I would like to thank my advisors Prof. Manuel Graña and Dr. Jorge Posada, their constant support and guidance shone the light when it was needed most. Jorge for me demonstrates the attitude and character of an outstanding researcher. His personal support and his incisive suggestions, have been decisive at many stages of my research. Thank you.

Prof. Manuel Graña's capacity to quickly see my point of view and offer his contributions during my research have had incalculable effect. Prof. Graña's belief my work, his preparedness to fight for it just as have I, has undoubtedly contributed to the completion of this thesis. I must of course also thank Prof. Edward Szcerbicki for his unwavering support and counsel, not only while I had the privilege to visit his group at the University of Newcastle, Australia, but till this day. It has been an honor to be advised by him and a pleasure to have him on my evaluation board. I want to thank Jon Goya and Fairlight, for the music. I'm certain that without that release I wouldn't have been able to reach this point. Our music, in the most difficult of days, gave me a way to create new places that eased my mind. I must also express my gratitude to my good friends: Isak Bello, Luis Kabongo, John Congote, Carlos Ocampo-Martinez, Carlos Lamsfus, Giuliano Giacovazzi, Will MaCabe, Nacho muñoz and Josune Ramiro for the many hours of their spare time, support and the moments we have shared, they are not only are the my best of buddies but those who gave me the support I need day in and day out. I would like to thank Olatz Irigaray, teaching me that happiness is found in the smallest, slightest of actions. I would like to say thanks to my dear friend Cesar Maldonado-Sanin who opened his house to me while I was in Australia and still works me from a great distance. Without his help many of the core ideas of this thesis would not exist. Cesar made the impossible possible. He thought me that with enough determination nothing is unreachable. My fellow researchers, colleagues, students at VICOMTech, led by Dr. Julian Florez, all of them deserve a special mention. I want to express my gratitude for the myriad of stimulating conversations that take place every day. I would like to thank Iñigo Barandiaran, Yoli Cobos and Cristina Sarasua for their collaboration and inspiration.

Last but not least, I would like to thank the people from the Industrial Applications Department, Sean Gaines, Javi Vaquero, Aitor Moreno and all of the rest of the gang that have since moved on: Maite Termenon, Diana Wald, Verena Wuerbel, Tobias Kammann and Dr. Michael Obach all, you are never far from my thoughts.

Carlos Toro March - 2009

Contents

List of Figures

List of Tables

Chapter 1

Introduction and Overview: Technical Context

In an ever-changing world, new technology is readily and rapidly adopted by industry. Time to Market (from the inception of a new product idea until its implementation) is shortened thanks to the information sharing capabilities offered by information systems. It is common practice that when a new product is launched, the next generation of the product offering enhanced features and lower price, is on track for launch. This rapidly evolving scenario is feasible partly thanks to the facilities provided by the World Wide Web and its companion technologies. As can be seen in Figure 1.1, Internet adoption continues to grow at an impressive rate. With new users, producing new data, that easily surpasses the capacity of our information systems, the problem has evolved from simple data retrieval in the pre-Internet era, to knowledge management in today's networked environment.

WORLD INTERNET USAGE AND POPULATION STATISTICS						
World Regions	Population (2008 Est.)	Internet Users Dec/31, 2000	Internet Usage, Latest Data	% Population (Penetration)	Usage % of World	Usage Growth 2000-2008
Africa	955,206,348	4,514,400	**51,065,630**	5.3 %	3.5 %	1,031.2 %
Asia	3,776,181,949	114,304,000	**578,538,257**	15.3 %	39.5 %	406.1 %
Europe	800,401,065	105,096,093	**384,633,765**	48.1 %	26.3 %	266.0 %
Middle East	197,090,443	3,284,800	**41,939,200**	21.3 %	2.9 %	1,176.8 %
North America	337,167,248	108,096,800	**248,241,969**	73.6 %	17.0 %	129.6 %
Latin America/Caribbean	576,091,673	18,068,919	**139,009,209**	24.1 %	9.5 %	669.3 %
Oceania / Australia	33,981,562	7,620,480	**20,204,331**	59.5 %	1.4 %	165.1 %
WORLD TOTAL	6,676,120,288	360,985,492	**1,463,632,361**	21.9 %	100.0 %	305.5 %

Table 1.1: World Internet usage [int08]

The interest and involvement in Web technologies by today's companies is not surprising, it is a well known fact that substantial investments are made in information management [Coa03]. In engineering companies, the information related to a specific product grows right from the conceptual design stage and continues until the product's disposal or recycling stage, therefore the use of information and knowledge management technologies is an aspect that should not be underestimated. With more information to handle, the chances of losing important information increases. A particular field in which this issue is particularly relevant is product design. There are several computerized tools to support different stages of the *Product Life Cycle* (PLC), these tools provide a good, yet open to improvement, way to handle the data produced and at the same time, reduce the potential information loss. The supporting computerized tools for a *Product Life Cycle Management* (PLM) are called *Virtual Engineering Tools* (VET) and the supporting applications are called *Virtual Engineering Applications* (VEA). It is difficult to assure that the computer programs used in every stage of the product life cycle management (or even in the same stage but from different vendors) are able to share information, this accounts for another potential danger of information loss. Being that Semantics is an important steering technology for the Web, we believe that the intensive use and embedding of Semantic-Based Technologies in VEA, could be a plausible solution to overcome some of the reported problems in [MB06,MB07]. In this Thesis, we will explore the use of Semantic-Based Technologies and the improvements that they introduce in VEA, which can be summarized as follows:

- Improved information and knowledge management
- Enhancements in the search, knowledge and information sharing processes.
- Use of the intrinsic knowledge embedded in the elements being described
- Empowerment of the VEA through better use of his knowledge and embedding of such knowledge in a structured and explicit conceptualization.

1.1 A brief word on Semantics

Semantics is the discipline that studies the meaning of things. The word originates from the Greek term *sēmantikos* that means "significant". The word semantic in its modern form is considered to have first appeared in French as *sémantique* in Michel Bréal's 1897 book, "Essai de sémantique" [Bré05]. Semantic technologies constitute some of the most interesting technologies derived from the World Wide Web revolution. Constantly reviewed in different areas of knowledge (e.g. Linguistics), their greatest improvements of information technologies might still be yet to be discovered. It is true that the concept of the Semantic Web presented by Tim Berners-Lee [Bl07] in his foundational article has still not yet been reached, according to some members of the scientific community, but the improvements due to its introduction in today's Web sites and search engines must not be underestimated. In the present work, Semantics are used to enhance VEA, by using different technologies that empower the user, providing amongst other interesting issues, a higher level of explicit conceptualization of the Product Life Cycle Processes.

1.2 A short introduction to Virtual Engineering Applications

Virtual Engineering (VE) is defined as the integration of geometric models and related engineering tools (such as analysis, simulation, optimization and decision making , etc), within a computerized environment that facilitates multidisciplinary and collaborative product development.

> *"The goal for Virtual Engineering is for the engineer to better focus on solving the problems at hand, without spending undue amounts of time gathering information, modeling the information, and then analyzing it. Virtual engineering is a user-centered process that provides a collaborative framework to integrate all of the design models, simulation results, test data, and other decision-support tools in a readily accessible environment."* [JMLB06]

VE characteristics are similar to software engineering in the sense that different results can be obtained through different implementations. A VE environment provides a user-centered, first person perspective [MB06] that permits users to interact with an engineering system in a natural way, while at the same time, provides a wide range of available and accessible tools to be used. In order to provide such an environment, a fully specified engineering model is required, including models for the physics, geometry and qualitative and quantitative data from the real system. A walkthrough and design-review environment is common in VE in order to extend the perception of the user into a Virtual Reality experience, where the observation of the simulated system can be followed in a first person paradigm. The virtual environment should provide easy-to-understand interfaces that must be appropriate to the users technical background and expertise. Virtual Engineering Applications and their underlying Virtual Engineering Tools should fit into the users environment and allow them to maintain attentive to the problem to be solved. According to McCorkle [MBS04] a key aim of VE is to engage the human capacity for evaluation of complex systems and situations. In order to fulfil the mentioned task, the User-centric Virtual Reality visualization techniques are the key components to take into account.

1.2.1 User-centric Virtual Reality visualization techniques

If the working environment is presented as a 3D interactive virtual world, the complexity can be perceived, rather than imagined, enhancing the speed of response of the user to a problem at hand. This fact, coupled with appropriate expert knowledge (that must be modeled *a priori*), will result in a reduction in design time with a corresponding lowering of cost. Interactive analysis and engineering, supported by today's computer applications, is a key factor in the VE concept. Nearly all aspects of a design require extensive data production and analysis, off-line setups, calculations and simulations are mandatory for new product development which, by its own nature, is an iterative process . The time required between revisions in a product design is highly variable and depends on the artifact being modeled. It can range from one day to several weeks. In this context, interactive collaboration and real-time exploration of "what-if" clauses are essential to the engineering process. An acceptable answer in time is more valuable than a perfect solution that is not available when needed. Often in real life design processes, the time needed to set up, compute and understand a result and to iterate the set of results in a simulation significantly exceeds the available time. Although techniques like Computational Fluid Dynamics (CFD), Finite Element Analysis (FEA) and complex system optimization are applied in new designs, a problem arises when this information cannot be re-used in the PLC at a different stage or even within the same stage. The reason can be found on the design of the programs used, which often do not provide any way to feed their results to another program or process at a later stage of PLC that probably needs or could be benefited from them.

1.3 Thesis motivation

The motivation for this PhD work is to show how Engineering tasks can be steered by the aid of Semantics. This Thesis represents the most important milestone in the continued research line that the author has undertaken for the past several years beginning with [Tor02] where a VEA intended for the modeling of the conceptual design stage was presented. During the research held, contributions from an applied research point of view have been presented, e.g. [TTP+07,TSSP08,PTWS05c]. From a professional perspective, this Thesis is motivated on the belief that the use of Semantic technologies can enhance the PLC in the following aspects:

- **The use of Semantics can shorten product development times:** Because past designs and experiences modeled using a Semantic schema, would serve at initial stages of a new product design, reducing development costs.
- **The use of Semantics provides a better Knowledge handling:** We believe that semantic enhancements of VEA are important because of the nature of the information sharing and reuse capabilities inherent to mentioned technologies. The majority of information needed in a Semantic enhanced VEA will be contained in the objects (seen as meta objects) and not in the users, a fact that enhances information sharing and the know-how of the producers.
- **The use of Semantics derives in direct reduction of costs:** Because

there is no need for an in-house expert every time in order to take risk decisions, those overhead cost will be reduced.

- **The use of Semantics produces better product quality:** When all the systems involved in the PLC share their Knowledge, the overall information will provide a better quality and a fast reaction (e.g. it will reduce the time for a maintenance task).

1.3.1 Problem Statement - Research question

Virtual Engineering Applications are common in today's engineering practice. Although proved powerful, VEA barely exploit the capabilities of contextual information, User requirements (beyond the mere actions paradigm), User Experience, and in general of Knowledge that can be modeled and inferred with the aid of Semantic Based techniques. The research question that motivates this work is:

> How to enhance Virtual Engineering Applications with the aid of Semantic Technologies? is it possible to develop a new theoretical framework and specific recommendations, methodologies and practical tools to allow a better integration of Semantic Technologies in the Virtual Engineering Applications?

1.3.2 Summary of the main contributions

The following is the summary of the contributions of this Thesis to the Semantic enhancement of Virtual Engineering Applications

Methodological contributions

- Recommendations on User Modeling
- Recommendations on Domain Modeling based on Engineering Standards
- The Reflexive Ontologies: A computational methodology for the enhancement of the ontology query process

- An general architecture for the Semantic Enhancement of Virtual Engineering Applications

Technical contributions

- Contributions to the Semantic driven generation of Graphical User Interfaces for CAD
- Contributions to the Semantic simplification of geometries for Large Model Visualization of Industrial Plants and Design Review
- Contributions to Industrial Maintenance from a Knowledge Based perspective

1.4 Thesis objectives

The objectives of this Thesis are:

- To present a review of the most important concepts related to the state of the art in Semantic modeling and some related technologies that can be used to enhance Virtual Engineering Applications with Semantics.
- To present a review of the most important concepts related to Product Life Cycle and Virtual Engineering.
- To present a series of considerations that may help when a User Modeling task needs to be fulfilled.
- To present a methodology for Domain modeling using Engineering Standards.
- To introduce technique for the fast retrieval of information from a Knowledge Base model.
- To introduce an original and generic methodology for the Semantic enhancement of Virtual Engineering Applications.
- To present a technique which uses the advantages of Semantic modeling technologies for the automatic generation of Graphic User Interfaces.

- To present a technique that leverages the advantages of Semantic technologies in the Large Model Visualization domain.
- To propose an architecture and a system implementation for the enhancement of the Industrial Maintenance management using Knowledge Based techniques.

1.4.1 Research environment and context where this Thesis was developed

This Thesis was possible thanks to participation in different projects and research collaborations of the author as full time researcher at VICOMTech Research Centre in San Sebastian, Spain and with the highly valuable support and guidance from Prof. Manuel Graña Romay from the University of the Basque Country.

Endorsing projects

The topics of this PhD were researched whilst participating in the following projects:

- MiroWalk: sponsored and partially funded by the German Academic Exchange Service DAAD (Deutscher Akademischer Austauschdienst).
- MiroView: Co-funded by the Basque Government under the INTEK-Berri program and the Steel Detailer Company, LANIK S.A.
- SEMTEK: funded by the Basque Government under the SAIOTEK basic research program.

Research visits

As part of the doctoral track of the author, one research stay was undertaken at The University of Newcastle (New South Wales, Australia) . The aforementioned visit was supervised by Prof. Dr. Edward Szczerbicki at the faculty of Engineering and Built Environment between June and September of 2007

Endorsing publications

A list of relevant publications co-authored by the PhD candidate can be found at the end of each Chapter that constitutes a contribution of this work.

1.5 Structure of this Thesis

This Thesis will be presented as follows: First we will present a state of the art, containing the basic technologies on which this work will rely on. Such technologies are divided in two groups, (*i*) the Semantic Technologies, presented in Chapter 2, and (*ii*) the Virtual Engineering and Product Design concepts, presented in Chapter 3. Following the presentation of such technologies, we will introduce our contributions to the Semantic Enhancement of Virtual Engineering Applications, such contributions will be divided in two, (*i*) the Methodological Contributions and (*ii*) the Technical Contributions, which will be presented in the following order:

The methodological contributions to the Semantic Enhancement of VEA

- **Chapter 4 - Recommendations on User Modeling**

  Different users have different abstraction abilities depending on their training and education, among other reasons. The user has a profession, certain knowledge about the domain and maybe about the model as well. He may be able to understand technical terminology, some visualizations, symbols, language, etc. Furthermore, the user may have a focus of interest (structural elements, inner parts, intersections, design layers). The interest on the User is also a key aspect to consider. Even in the case of the same field of science, user Semantics may differ, providing interesting information that could be used to adapt the VEA whose interaction is taking place, producing a better experience in the utilization of such VEA. Our approach does not intend to introduce a new methodology for User Modeling, instead of that, we present a series of considerations that may help when carrying a User Modeling task.

- **Chapter 5 - Recommendations on Domain modeling based on Engineering Standards**

  In this chapter we will present our methodology for the Domain modeling based on Engineering Standards, we discuss some benefits of Standards as guidelines for the Knowledge Based modeling and some potential challenges along with possible approaches to overcome them. The benefits of use of Standards as models for Knowledge Bases (Domain ontologies) have been shown valid in previous and related works ([PTWS05b, TPOF07, CTS+08]) and, as it will be shown in our technical contributions, their adoption is a keypoint for us.

- **Chapter 6 - The Reflexive Ontologies: A computational methodology for the enhancement of the ontology query process**

  In this chapter, we will introduce the concept of Reflexive Ontologies (RO). RO is a description of the concepts and relations of such concepts with a set of self-contained queries over instances in the domain of study [TSSP08]. RO can be used whenever an ontology query is needed. The aforementioned technique enhances a Knowledge Base (KB) by (*i*) the speeding up of the query process (*ii*) giving to the ontology the possibility of adding new queries on individuals with the corresponding answers to such queries (a feature that adds knowledge about the domain); and (*iii*) the self containment of the Knowledge Structure in a single file; including the model, the relations between the elements of the model, the individuals (instances) and queries over such individuals. We present in this Chapter a framework that can be used to enhance any existing ontology with the reflexivity approach we introduce. RO are needed because we make massive amounts of queries to our KB when trying to extend a traditional VEA with Semantic technologies. The queries can range from simple questions to the underlying KB up to complex reasoning processes performed programmatically. Moreover, RO can be used outside a Semantic VEA enhancement.

  For the aforementioned reason, we categorize them as a methodological contribution of this Thesis.

- **Chapter 7 - An architecture for the Semantic Enhancement of Virtual Engineering Applications**

  In this chapter, we will introduce an architecture for the Semantic enhancement of Virtual Engineering Applications through its embedded VET. Our architecture uses the concepts presented in the methodological chapters and it will serve as a model for the applications presented in the technical contributions. This chapter will be a conceptual link between the more theoretical part of the Thesis and the implementation part in some real-world VEA framed in the Product Life Cycle concept.

The Technical contributions to the Semantic Enhancement of VEA

Chapters 8 trough 10 will introduce different experiences in the Semantic Enhancement of VET that were undertaken/developed during the work in different R&D projects. Such presentation is divided as follows:

- **Chapter 8 - Contributions to the Semantic driven generation Graphical User Interfaces for CAD**

  This Chapter, presents a technique which uses the advantages of Semantic modeling technologies for the automatic generation of Graphic User Interfaces taking into account three main features: (*i*) the user and his needs, (*ii*) the stage of the PLC in which he is interacting with the CAD and (*iii*) the user intentions. This technique was implemented within the scope of an applied research project with a Steel Retailer (structural design) partner.

- **Chapter 9 - Contributions to the Semantic simplification of geometries for Large Scale Visualization of Industrial Plants and Design Review**

  In this Chapter we present our contributions for the enhancement of the visualization and VR capabilities in generic CAD applications. Our approach uses ontology modeling as well as Engineering Standards in order to conceptualize and exploit the information contained in a CAD model.

- **Chapter 10 - Contributions to Industrial Maintenance from a Knowledge Based perspective**

  Considering some detected challenges in the Industrial Maintenance field, we propose an architecture and a system implementation for the enhancement of the Industrial Maintenance management using Knowledge Based techniques. As an example, we benefited from the User Experience modeling (following our own recommendation on chapter 4) that allow us to consider User Experience, a feature that is not typically taken directly into account for the Industrial Maintenance tasks, at times the Industrial Maintenance databases consider some statistical measurements for a given element lifespan, but the reasons why the element was changed before the end of the cycle are not stored anywhere. We believe that these factors are as important as the fact that the element was changed (or maintained).

Schema of presentation of the Technical contributions

In order to present our implementations, we will present the technical contributions following the next schema:

- **Description of the technical problem and challenges identified**: In this part we will introduce the typical scenario where our contributions will focus on and some challenges still present that we will intend to solve when applying our techniques.

- **Brief description of the contributions**: This part will briefly highlight our contributions. The contributions will be framed in the PLC schema and its main characteristics will be introduced.

- **Background concepts**: In this part we will introduce some core concepts specifically located within the domain of study of the VEA. This is not intended to be a full state of the art, but instead a brief introduction in order to fix the language and the domain specifics.

- **Our approach**: Once we have the problem described and some core concepts introduced, we will address our contribution. We will present the approach we followed in order to solve the detected problems/challenges.

- **Example Case / Implementation of the concepts**: We will exemplify in this part the use of our presented techniques. Some of the examples will have concept validation, depending on the status of the project and the legal facts that allow us in some cases to publish obtained results.

- **Discussion**: Finally, a brief discussion of the contribution will be presented, pointing the main benefits and possible open challenges.

Finally in **Chapter 11**, we will present this Thesis conclusions and future work.

Chapter 2

Semantic Modeling

This chapter is devoted to the review of the state of the art in Semantic modeling and some related technologies relevant to the purposes of this Thesis. First we discuss the idea of Knowledge Engineering, which will serve us as starting point for many of the concepts that will be used along our work, discussing briefly a classical description given in the literature. Following we will present a series of base concepts related to the Semantic Web and in specific on ontology modeling which is the main paradigm we used for Knowledge Modeling.

2.1 Knowledge Engineering

Knowledge is considered an invaluable resource of great benefit for most purposes in life. For this reason, mankind has always attempted to make it part of their assets. Knowledge itself seems to be an attribute of human beings; it may be defined [Pre03] as : (*i*) the expertise and skills acquired by a person through experience or education via a theoretical or practical understanding of a subject, (*ii*) what is known in a particular field or in total related to facts and information or (*iii*) the awareness or familiarity gained by experience of a fact or situation. The acquisition of Knowledge involves complex cognitive processes such as: perception, learning, communication, association and reasoning. The processes involved are the result of the development of the person and closely related with his intelligence. The word Knowledge is also used to mean the confident understanding of a subject with the ability to use it for a specific purpose if appropriate. Philosophical debates in general start with Plato's formulation of Knowledge as "justified true belief". There appears however, that no single agreed definition of Knowledge has been reached at the present, and one of the main reasons for this is the fact that continuously new competing theories are presented by the scientific community. There are theories that come from the medical, the psychological and even the sociological points of view. However in this Thesis we will address a very specific application of the term for the fields of Engineering and Computer Science. According to Feigenbaum:

> *"Knowledge Engineering (KE) is an engineering discipline that involves integrating Knowledge into computer systems in order to solve complex problems normally requiring a high level of human expertise".* [FM83]

Knowledge Engineering, hence, relies on instructional methodologies and Computer Science in general, trying to mimic Knowledge and behaviors that are intrinsically human in a certain Domain and within the scope of an artificial system. This broad definition, implies not only the need of specific technologies, but also the need to overcome related implementation issues.

2.2 The Semantic Web

Probably the field in which computer Semantic analysis is more extensively investigated is the Semantic Web. Leading search engines, such as Google, make use of these techniques in order to obtain better accuracy and performance in their search processes for the answers to the submitted queries. For example, Semantics could be used to recommend sub queries based on a user's search query. If he commits and error while writing a search in a Web enhanced by Semantics, the mentioned Web site could be able to suggest new queries based on relations extracted from the correct words the user typed.

The success of the Semantic techniques is due partially to the help they give for information organization, as they are not restricted to be only information cluster holders, but managers of information concept relations between elements. The Semantic Web is said to be an extension of the traditional Web that is derived from the idea of Tim Berners-Lee (CEO of W3C) in which the known Web acts as a universal medium for data, information and Knowledge exchange. Mainly, the Semantic Web consists of a set of design principles, collaborative working groups, and technologies, of which, some remain unimplemented. Standardization initiatives by the W3C are in fact recommendations, including some of the predominant technologies such as Resource Description Framework (RDF), a variety of data interchange formats (e.g. RDF/XML, N3, Turtle, N-Triples), and notations such as the RDF Schema (RDFS) and the Web Ontology Language (OWL), all of which are intended to provide a formal description of concepts, terms, and relationships within a given Knowledge Domain. Humans are capable of using Web resources in order to find information on a given subject. However, for a computer, this is difficult to accomplish this task without direct human interaction because traditional Web pages are designed to be read by people, and not by machines. In Berners-Lee's vision [BL00], information is understandable by machines and by doing so; a faster and more reliable set of answers can be provided, he quoted:

> *"I have a dream for the Web [in which computers] become capable of analyzing all the data on the Web – the content, links, and transactions between people and computers. A 'Semantic Web', which should make this possible, has yet to emerge, but when it does, the day-to-day mechanisms of trade, machines talking to machines will handle bureaucracy and our daily lives. The 'intelligent agents' people have touted for ages will finally materialize."* [BL00]

2.2.1 Ontologies and their use in the context of this document

In philosophy, Ontology[1] is the most fundamental branch of metaphysics. The word comes (from the Greek $\sigma\upsilon$, genitive $\sigma\upsilon\tau o\zeta$: of being (part. of $\varepsilon\iota\upsilon\alpha\iota$: to be) and -$\lambda o\gamma i\alpha$. This branch of philosophy deals with the study of being or existence and their basic categories and relations amongst such categories. The search of the truth from the Ontological world seeks to determine what entities can be said to "exist", and how these entities can be grouped according to similarities and differences. In

[1] We use intentionally Ontology with the first "O" Upper-cased when talking from a non Computer Science point of view, following Nicola Guarino's recommendation [GP95].

simple words, an Ontology is :

> *"The area of philosophy that occupies itself of the explanation of nature and the organization of reality"* [Mar89].

Recent developments in ontology theory have been introduced to the engineering world through Computer Science. Specifically the Knowledge Engineering community is giving a lot of attention to Ontologies, thanks to the fact that this theory offers an interesting tool to describe objects and the relations amongst them within the scope of a given Domain. In Computer Science, and hence within the scope of this Thesis, ontologies are used as an information support schema for Knowledge representation of a Domain. They provide a vocabulary for representing and communicating Knowledge about some topic and a set of relations holding amongst the terms in that vocabulary.

2.2.2 Ontologies in Computer Science:

With vibrant discussions ongoing between researchers, without agreement being reached, it seems that to provide consensus on the word "ontology" in Computer Science is a very difficult task. Since this work deals with the use of different Semantic technologies where ontologies play an important role, a definition must be somehow expressed; however we would like to clarify that we do not intend to steer the reader towards a given definition or to provide our own. We decided to adhere to the definition provided by Tom Gruber because of the fact that it is powerful enough to exemplify what we want from ontologies and also for its simplicity; Gruber defines an ontology as follows:

> *"An ontology is an specification of a conceptualization"* [Gru93].

A extension to Gruber's definition was provided recently by the Stanford Knowledge Systems Lab defining an ontology as:

> *"The explicit specification of some topic. For simplification purposes, it is a formal and declarative representation which includes the vocabulary (or names) for referring to the terms in that subject area and the logical statements that describe what the terms are, how they are related to each other, and how they can or cannot be related to each other."* [STA08]

This description does however have detractors, but in the words of Gruber, "the definition is consistent with the usage of ontology as set-of-concept-definitions, but more general. And it is certainly a different sense of the word than its use in philosophy" [Gru95].

Foundational (upper) ontologies

They constitute an attempt to create an ontology useful for describing general concepts which offers the possibility of modelling any Domain. The upper or foundational ontologies metaphor suggests a hierarchy of entities and their associated rules (both in the form of theorems and regulations) attempting to describe those general entities that do not belong to a specific problem Domain (rather they are applicable in the description of any Domain). Currently there is debate very interesting debate about whether the concept of using a single, shared upper ontology is even feasible or practical, but we will not address this problem as it is out of the scope of this Thesis. Between the most famous foundational ontologies we can mention Wordnet [Fel98] and DOLCE [MB+03].

2.2.3 Reasons for modeling ontologies

Some of the main reasons for modeling an ontology are:

To share common understanding of the structure of information among people or software agents

This is a common goal of ontology development: the importance of having an equivalent (if not the same) Knowledge base is patent when several actors need to use or re-use resources even in collaborative ways. In this schema a Web provider maintains an underlying ontology and accessing sites publish, share and reuse the structure by providing new information. This schema involves important new paradigms in Computer Science like Ambient Intelligence, sensors and agent technologies.

To enable reuse of Domain Knowledge

The reuse of Knowledge is by far the most important property of a Knowledge structure. When talking about ontologies, this feature involves notions of time and its intervals, dimensions and other kind of physical features that can be measured. The importance of Knowledge reuse implies that by having several sub-ontologies, a large Domain ontology can be created, involving more Knowledge. In other words this means that for example, in an industrial plant, there could be a sub-ontology for the structure, a sub ontology for the piping system, etc., hence the overall Domain ontology is the sum of the sub-ontologies. Another method is a top-down approach creating a Domain general ontology and adding later sub-ontologies.

To make Domain assumptions explicit

Ontologies by nature allow making of Domain assumptions, a feature that eases the otherwise hard coding needed in traditional approaches. The easiness of graphical modeling tools as Protégé [HKR$^+$04] makes Domain assumption modeling even more powerful.

To separate Domain Knowledge from operational Knowledge

Domain Knowledge represents the conceptual classes and relations independently from the configuration of the real objects being modeled, while operational Knowledge is related to instantiations that could be different for each manufacturing. E.g. modeling a computer (Domain) and a laptop made by a certain manufacturer (operational). An ontology is able to represent both schemas offering the possibility to use algorithms developed for operational Knowledge in Domain Knowledge structures.

To analyze Domain Knowledge

When a declarative specification of the terms contained in the ontology is available, analyzing the formal Domain Knowledge is feasible. This formal analysis is valuable for the reuse of ontologies and also when the extension of such ontologies is desired [MFRW00]. Developing an ontology is similar to define a set of data and its structure for other resources (programs) to use. Domain-independent applications, problem-solving methods and software agents use ontologies and Knowledge bases built from ontologies as data. The analysis of ontologies can be done through API tools and Reasoners and it shall be addressed later in this chapter.

2.2.4 Common components of ontology modeling

In Computer Science, and specifically from a modeling point of view, an ontology includes the following components:

Attributes

Used to describe the properties, features, characteristics, or parameters and classes belonging to an abstract object. E.g. "Age". Attributes can be grouped in two groups: The first group of attributes are the ones that describe an intrinsic property of an object that belongs to a class and can be represented by a typical variable type (integer, Boolean, string, etc), for example "name" which is a string (set of ordered characters). The second type of attributes are the ones that relate two classes, like for example "has_son" which could map a class "father" to a class "son", being both subclasses of "person". Ontologies are only true ontologies if the concepts are related between them, in other words, when they have relation attributes. If this is not present, we can have either a taxonomy (if hyponym relationships exist between concepts) or a controlled vocabulary. Although useful, these representations are not considered true ontologies.

Relations (relationships)

Relations describe ways in which classes and objects can be related to one another. E.g. "Has_Father". Typically a relation has a direction of expression that could be inversed expressing the same fact, but with a reversed phrase in a natural language schema. A very important kind of relation is called subsumption, which allows the expression of sub types or super types in hierarchies. The different relations present in an ontology and their subsequent subsumptions describe the expressive power of the ontology language used to model embedded Knowledge. Another common type of relation is called meronymy (part of), used to represent how objects combine together in order to form more complex-composite objects. By using relation types in an ontology, we obtain a structure called Directed Acyclic Graph (DAG)[2] which is of common use in different branches of Computer Science, e.g. parsers (an illustration is shown in Figure 2.1).

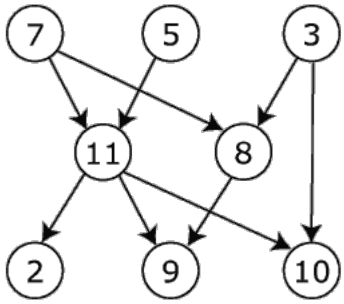


Figure 2.1: A simple DAG schema

Aside from the relations previously, ontologies include additional types of relations that further refine the modeled Semantics. We can define diverse kinds of relations:

- Relations between classes
- Relations between individuals
- Relations between an individual and a class
- Relation types for relations between a single object and a collection
- Relation types for relations between collections

Classes

Classes are used to classify individuals, as a given object must belong to a given class. Even the most abstract object belongs to a generic class called "thing" which answers to the criteria of being a thing, representing a high conceptualization of something excluding at the same time "no-thing" objects (e.g. Part). Classes can

[2]Weisstein, Eric W. "Acyclic Digraph." From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/AcyclicDigraph.html

be grouped into subclasses or super classes (e.g. person can be the super class of father). This subsumption relation is used to create a hierarchy of classes with the more general classes at the top, and the more specific at the bottom leaves of the tree. Sometimes restrictions are applied to a class in order to avoid paradoxes, saying for example that a class “a” is a subclass of a class “b” only when a given condition is met. In order to model restrictions, ontologies use different types of logical models which depending on the ontology language used and hence the degree of Semantic load, express more or less adequately the restriction. It must be pointed out that what is true for a parent class, will be true for its child classes. The above fact should be taken into account when modeling an ontology as it could lead to indeterminate or logical breaks. In order to self-classify an ontology and test for logical consistency, Reasoners can be used as will be discussed later in this chapter. In ontology modeling, multiple parents and multiple child classes are allowed, which leads to a representational graph model rather than a taxonomy (although taxonomies can be represented using ontology modeling languages). In Figure 2.2 , the class “Screw” is a type “Part” via the relation “is.a”, every true or false assertion on the class “Part” will propagate to “Screw” and “Profile”. In the figure, some properties of the modeled classes are depicted e.g. “diameter” in where the data type holder is a double precision float.

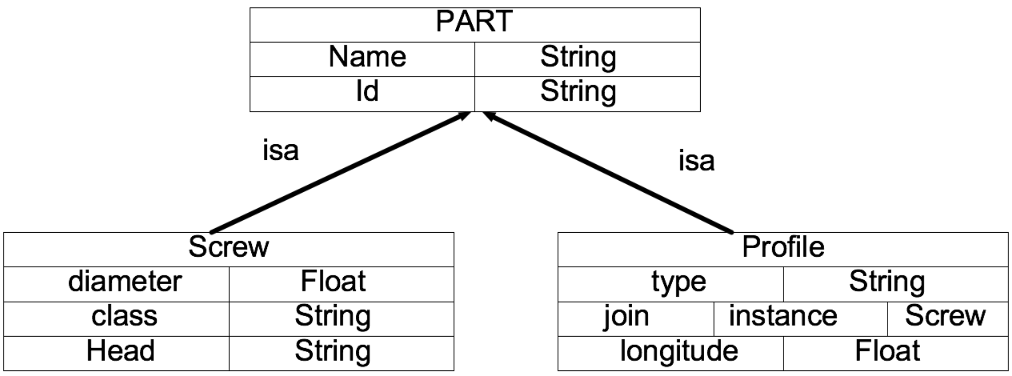


Figure 2.2: A simple Class diagram with example attributes

Individuals

Instances or objects that belongs to a class and hence, share its characteristics (e.g. “screw_2993”). Individuals are also called instances depending on the modeling language used, providing the basic components of an ontology including objects of concrete type (things you can touch) or abstract type (concepts and assemblies). Normally an ontology does not necessarily need to have individuals because the inner logic could be operated within the scope of the classes, but at the same time, one of the main purposes of ontologies is to provide means to classify individuals, hence the use of individuals is common. Figure 2.3 depicts the above presented class subsumbtion with some individuals (in red).

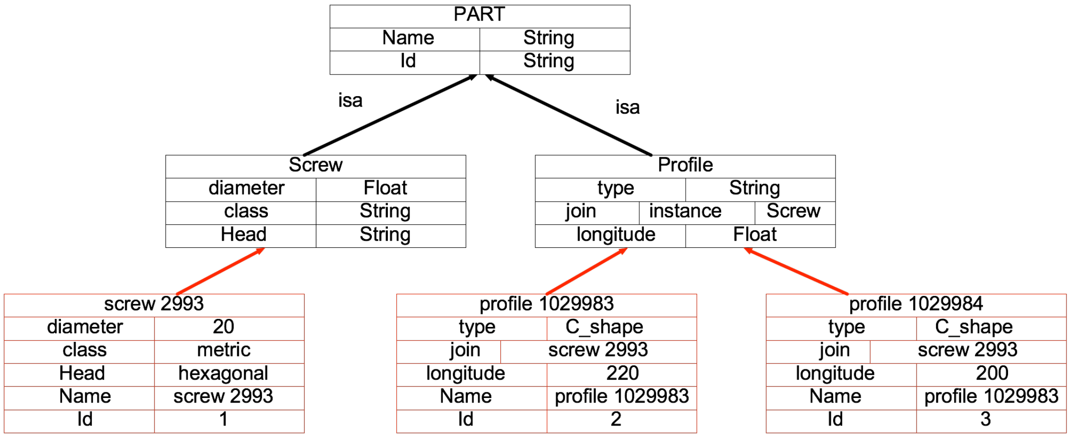


Figure 2.3: Classes + Individuals

Function terms

Which are complex structures formed from certain relations that can be used in place of an individual term in a statement.

Restrictions

Formally stated descriptions of what must be true in order for some assertion to be accepted as input

Rules

Statements in the form of an if-then (antecedent-consequent) sentence that describe the logical inferences that can be drawn from an assertion in a particular form

Axioms

Assertions (including rules) in a logical form that together comprise the overall theory that the ontology describes in its Domain of application. This definition differs from that of "axioms" in generative grammar and formal logic. In these disciplines, axioms include only statements asserted as *a priori* Knowledge. As used here, "axioms" also include the theory derived from axiomatic statements.

Events

They represent the change of attribute values or relations.

2.2.5 Flavors and languages of ontology representation

RDF

The Resource Description Framework (RDF) is a recommendation of the World Wide Web Consortium (The W3C) originally designed as a data model for metadata which has been used for information modeling in a wide variety of schemes. RDF focuses on the idea of making statements about resources present in the Web in a classic subject-predicate-object schema known as a triple. As one can imagine, the subject is the resource itself, and the predicate is the relationship between the subject and the object. For example in the phrase “Diana has a necklace made of mother of pearl” an equivalent RDF triple would have as subject the word “Diana”, as object the words “has a necklace “ and as predicate “made of mother of pearl”. RDF has different serialization formats (file formats); hence the encoding will be kind of different depending of the format used. Historically speaking, the proto-language that originated RDF was called MCF, which was an initiative by R.V Guha and Brai developed when they were working for Apple computer during a sabbatical from Netscape Communications. Some ideas of the actual specification were extracted also from the Dublin Core initiative and also from PICS (an early W3C labeling system). The actual recommendation [W3C97] and its correspondent XML syntax model was introduced in 1999 and later, it was extended in a set of related specifications in 2004 rapidly accepted and implemented by the Internet community. The most common serialization format in use is .rdf (expressed in XML), which acts as a container for FOAF, DOAP and SKOS, amongst others. This implementation should not be confused with the abstract RDF model. Its MIME media type, application/rdf+xml, was registered by RFC 3870. Another format introduced by the W3C is Notation3 (N3) which is a non-XML serialization of RDF and hence easier to follow by humans. N3 is based on tabular notation making the encoding of the triples easier to recognize (ala Turtle or N-Triple) A set of RDF statements is a kind of graph and, hence, suits more to certain types of Knowledge Representation than the relational model commonly used in database modeling. However in practice, a database back-end of the RDF model is often used for persistence reasons. Ontology languages like RDFS and OWL are built upon the RDF schema, and the verification of the triples can be treated as a resource in which additional statements can be made. The association to a Domain through context specification can be also modeled in RDF, giving the option to assert the truth table depending on the context; or in other words, to get assertions that are true in a given context but false in an other. Using First Order predicate Logic (FOL) in RDF, the only meta level relation is negation, but with the possibility to state sets of prepositions in nested contexts, permits RDF to be used to define modal and higher order logic statements and their treatment. To query an RDF model, the most used language is SPARQL, which is an SQL like language.

RDFS (RDF Schema)

As explained before, RDF describes resources with classes, properties, and values. In addition, RDF also offers a way to define application-specific classes and properties. Application-specific classes and properties must be defined using extensions to RDF, one such extension is RDF Schema (RDFS). Its first version was introduced in 1998 and the recommendation is from 2004. RDFS is included in the lower level Semantic flavor of OWL as will be explained later. RDFS does not provide actual application-specific classes and properties. Instead RDFS provides the framework to describe application-specific classes and properties Classes in RDFS is much like classes in object oriented programming. This allows resources to be defined as instances of classes, and subclasses of classes. The following is a piece of code in RDFS:

```
<?xml version="1.0"?>
<rdf:RDF
  xmlns:rdf= "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   xml:base= "http://www.animals.fake/animals#">
        <rdfs:Class rdf:ID="animal" />
            <rdfs:Class rdf:ID="horse">
        <rdfs:subClassOf rdf:resource="#animal"/>
   </rdfs:Class>
</rdf:RDF>
```

OWL

OWL is the acronym of Web Ontology Language, and is a family of Knowledge representation for authoring ontologies endorsed by a recommendation of the W3C [W3C04]. OWL is divided into three sub-languages called flavors: OWL Lite, OWL DL and OWL Full, of which the first two are based on Description Logics [SPG+07] and the third on a Semantic model intended to be compatible with RDF Schema. OWL is serialized using an RDF/XML syntax and has grown to be considered one of the pillars of the Semantic Web idea, attracting academic and commercial partners who use it for several kinds of applications [PLS02, MSJT07, SST07]. The actual OWL recommendation is based on the DAML+OIL Web Ontology Language that was originally developed as a joint effort between the Defense Advanced Research Projects Agency (DARPA) at the United States and Europe's IST projects initiative for research funding. In November 1, 2001, the W3C created the "Web Ontology Working Group" chaired by James Hendler and Guus Schreiber in order to develop a suitable language for the Semantic Web initiative. The first drafts were published in July 2002, becoming a W3C recommendation on February 10, 2004. An OWL model is constituted by "individuals" and a set of "property assertions" which relate the individuals, a set of axioms which place constraints on sets of individuals, called "classes" and the types of relationships permitted between them. Axioms provide Semantics as they allow the systems to infer new information based on the data explicitly provided, characteristics that differentiate ontology modeling from traditional database models. OWL is both syntax for describing and exchanging ontologies, and a formally defined Semantic tool that gives the meaning.

The Open World Assumption

OWL uses Open World Assumption, in contrast to SQL and Prolog, which adopts Closed World Assumption. Under Open World Assumption, if a statement is not explicitly proved true using current Knowledge, it will not draw the conclusion that the statement is false. The following are the types of available OWL ontology schemas (called flavors).

OWL Lite

A flavor originally intended to support users in the need of a classification hierarchy and simple constraints. Cardinality constraints are limited values of 0 or 1. Its implementation in real world systems is not very common and besides from the expressiveness constraints placed on OWL Lite, more syntactic inconveniences exist. The majority of the constructs available in OWL DL can be built using complex combinations of OWL Lite.

OWL DL

It was designed to provide the maximum expressiveness while retaining computational completeness; this means that all conclusions are guaranteed to be computed. The decidability (meaning that all computations will be performed in a finite time) is guaranteed, and the availability of practical reasoning algorithms is determined by Reasoners based on the DL paradigm. OWL DL includes all OWL language constructs, however their use is restricted under certain circumstances (e.g., the number restrictions may not be placed upon properties which are declared to be transitive).

OWL Full

This flavor is based on a different Semantic schema. It was designed to preserve compatibility with RDF Schema. For example, in OWL Full a class can be treated simultaneously as a collection of individuals and as an individual in its own right; a fact that is not permitted in OWL DL or OWL Lite. OWL Full allows an ontology to augment the meaning of the pre-defined vocabulary. For this and other reasons, it is unlikely that any reasoning software will be able to support complete reasoning for OWL Full. Each of these sub-languages is a syntactic extension of its simpler predecessor. The following set of relations hold. Their inverses do not.

- Every legal OWL Lite ontology is a legal OWL DL ontology.
- Every legal OWL DL ontology is a legal OWL Full ontology.
- Every valid OWL Lite conclusion is a valid OWL DL conclusion.
- Every valid OWL DL conclusion is a valid OWL Full conclusion.

Figure 2.4 depicts the OWL flavors schema:

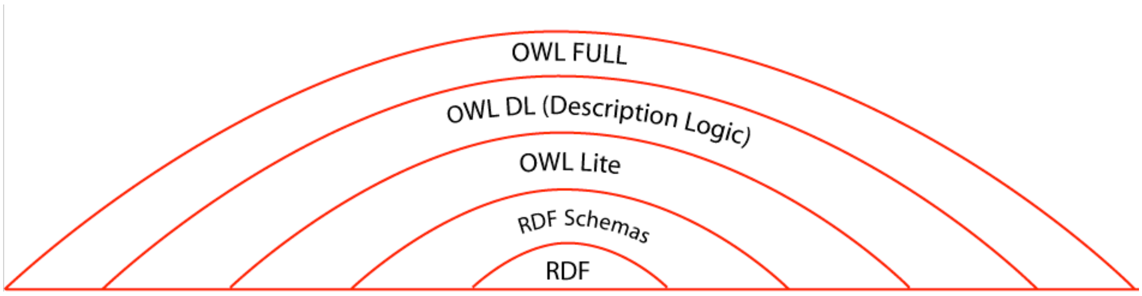


Figure 2.4: OWL Flavors [Ez 06]

2.2.6 Ontology development methodologies.

Currently the field of software development a wide range of methodologies are employed, some examples are the Capability Maturity Model (CMM), the Waterfall Models and various ISO standards, such as ISO 15504 and ISO 9000. In Ontological engineering, an agreement has not been reached yet, and it can be said that there are as many ontology development technologies as ontology engineers. In this section we will introduce two approaches for ontology development in order to exemplify the complexity of the problem.

Uschold and King Methodology [UK95]

The process is divided into a series of steps that are depicted in Figure 2.5.

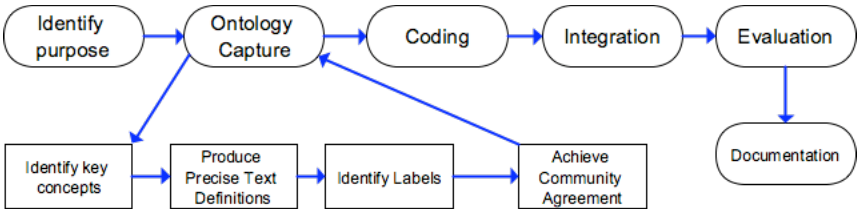


Figure 2.5: Uschold and King ontology development methodology [UK95]

- **Identification of the purpose:** the scope and purpose is facultative to the ontological engineer whose work in this step is focused on the discussion of the Domain with the experts in order to identify the so-called "mission statement" of the ontology that is the basic reason why the ontology is needed. In this step, the ontological engineer thinks of the definition of the vocabulary, the meta-level specification, the primary intention (meaning that it will be used by a small group of users or a large community) the means of use and if the ontology is part of a bigger Knowledge base.

- **Ontology capture:** This process involves (*i*) the identification of the key concepts and relation in the Domain of study that will be present in the ontology, (*ii*) the identification of terms to refer to such concepts and relations, and (*iii*) The achievement of a community agreement over the concepts. The first step of this methodology recommends the use of brain storming techniques in order to obtain and validate the concepts and relations. In a second step, the concepts should be arranged in "work areas corresponding to naturally arising subgroups", in order to decide exclusion or inclusion of terms. Uschold and King, underline the vital importance of having a requirement specification for future reference where the inclusion or exclusion of decisions on the elements can be reflected. "Semantic cross-references" are also identified in this sub-step for concept linking between groups. The third step, involves the identification of meta-ontologies suitable for the Domain of study, being a very important task because it represents a sort of ontological commitment. In a fourth step, a precise definition of the involved terms and concepts is produced. The methodology recommends that the definitions of the more overlapping areas between Semantic concepts should be addressed first, they also recommend that the ontology engineer should pay more attention on the definition of cognitively basic terms as opposed to the most abstract ones, arguing that this facilitates the process of relating terms in different areas. The development of definitions is usually reported best when is described using precise natural language with some sort of examples for better understanding.

- **Ontology coding:** This stage is the actual representation of the conceptualization, using a formal expression language. The representation has three sub-stages (*i*) a commitment to the meta-ontology, (*ii*) the choosing of the ontology language to be used and (*iii*) the coding process itself.

- **Ontology integration:** The possible re-use of existing ontologies has to be taken into account here and this moment is when the typical problems of alignment and mapping, that will be treated later on this chapter may occur.

- **Ontology evaluation:** This stage is in fact the technical judgment of the performance of an ontology with evaluation parameters as the fast response to a given query, the repeatability, that means that if you obtain an answer for a query, the next time the query is performed the answer will only vary if the individuals changed.

- **Ontology documentation:** The last step of this methodology, involves the use of the documents generated along the process in order to allow a future easiness in the integration by third party developers or users of the developed ontology. This stage is commonly forgotten leading to integration problems that in some cases make the use of Domain ontologies difficult.

Grueninger and Fox Methodology [GF95]

The basis of ontology modeling for Grueninger and Fox is the requirement of an application to response to particular problem. In order to solve such a problem, a clear definition must be provided in the form of working scenarios and queries that will be commonplace when the application is running (see Figure 2.6). Grueninger and Fox hence, coined the term "competence question" for defining the competency of the ontology, which in other words represents a clear set of expectations outlining the minimum achievements required by the Knowledge model. This process is performed in the capture of motivation scenarios stage.

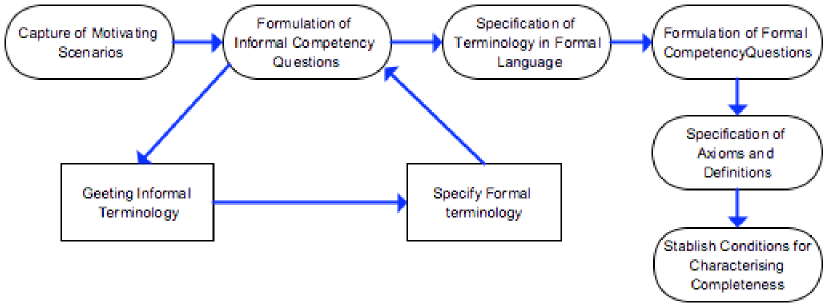


Figure 2.6: Grueninger and Fox ontology development methodology [GF95]

The definition of the terms of the ontology, including objects, attributes and relations, represents a task that when completed produces the language and the constraints. The competence questions can be considered the representation of queries over the ontology, which at this stage have not been formalized. As such, they define the scope and therefore the competency of the ontology, Grueninger and Fox argue that:

> *"Competency questions do not generate ontological commitments; rather, they are used to evaluate the ontological commitments that have been made."* [GF95]

Competence questions at this stage are called "informal" rather than "formal", due to the fact that restrictions are not taken into account yet. In the formalization of the ontology, the expression and definition of the ontology is achieved by using a formal language in which the ontologies can be constructed. In this stage, the formal questions are taken into account, by entailing of consistency questions with respect to the axioms contained in the ontology. Formal competencies model in a way a distinguishing feature between ontologies, as they describe the possible uses of the Knowledge structure in a Domain. It is important to annotate that at this stage, the competency questions do not axiomatize the ontology, but they represent a powerful method for the completeness verification. The specification of axioms is arguably one of the most difficult parts of the ontology engineering process as they provide the Semantics of the ontology terms by restricting or allowing relational data. Grueninger and Fox argue that:

> *"Without the axioms, we cannot express the question or its solution, and with the axioms we can express the question and its solutions."* [GF95]

In the end, the solutions to the competency questions have to be an entailment of the axioms in the ontology, when inconsistencies are found or formal competency questions cannot be answered, the ontology must be reviewed. In the final step, the ontology engineer must provide a set of conditions under which the solutions to the formally stated competence questions are considered complete, providing a base for the formulation of theorems related to completeness.

2.2.7 Handling Ontologies

Although any ontology can be modeled using a simple text editor, the size and complexity of some models and the need for an efficient method of handling, drove the creation of ontology editors. There are many editors available on the market, commercial and non-commercial, the following are some of the more popular editors.

Protégé

Protégé [HKR+04] is a graphical and interactive ontology-design and Knowledge-acquisition environment developed by the Stanford Medical Informatics group (SMI) at Stanford University. It is an open source, stand-alone application that provides a graphical ontology editing environment and an extensible architecture for the creation of customized Knowledge-based tools. Its Knowledge model is OKB-compatible. Its component-based architecture enables system builders to add new functionality by creating appropriate plug-ins. The Protégé plug-in library contains plug-ins for graphical visualization of Knowledge bases, inference-engine for verification of constraints in first-order logic, acquisition of information from remote sources such as UMLS and WordNet, semi-automatic ontology merging, etc. It also provides translators for FLogic, OIL, Ontolingua and RDF(S), and can store ontologies in any JDBC compatible relational database. Plug-ins, applications and ontologies, which have been developed both by the Protégé group and other Protégé users, are available in the Protégé Contributions Library. Protégé supports a wide variety of plug-ins programmed using the provided Java API. The ease of use of Protégé, its multi-platform capabilities (there are versions for Windows, Mac OS, Linux, etc) and a good balance between user interface and programming environments, guided us to toward using this editor in our developments.

OilEd

OilEd [BHGS01] has been developed in the context of the European IST OntoKnowledge project, by the University of Manchester, the Free University of Amsterdam and Interprice GmbH (Germany). It is a simple freeware ontology editor that allows the user to build ontologies using OIL. Consistency checking and automatic classification of the ontologies is possible through the use of an external Reasoner. According to the developers, OilEd is not intended as a full environment for development of large-scale ontologies. It is designed to be the "Notepad" of ontology editors, best suited to tasks such as learning to build ontologies and demonstrating how a Reasoner can be used to check and enrich ontologies.

ONTOEDIT

OntoEdit [SEA+02] is an ontology engineering environment developed at the Knowledge Management Group (AIFB) of Karlsruhe University. It is a stand-alone application that provides a graphical ontology editing environment (which enables inspecting, browsing, codifying and modifying ontologies and supporting the ontology development and maintenance task) and an extensible architecture for adding new plug-ins. The conceptual model of an ontology is internally stored using a powerful ontology model, which can be mapped onto different, concrete representation languages. Ontologies are stored in relational databases (commercial version) and can be implemented in XML, FLogic, RDF(S) and DAML+OIL.

2.3 Information and Knowledge extraction from ontologies.

Ontologies are commonly encoded using ontology languages and with the help of editors, information and Knowledge extraction can be performed manually; however, from a programming perspective, some Application Programming Interfaces (API) are available for the expert user who wants to extend the functionally or embed ontologies in applications easily. Being that this technology is mostly Web-oriented, it is not a surprise that many ontology API are Java based. Chris Bizer and Daniel Westphal from the Freie Universität of Berlin, maintains an excellent relation of the offer in their Web page [BW08]. Within the most widely adopted programming interfaces we can count Jena [CDD+04]and the Protégé OWL API [dameron07protege] which is based heavily on Jena but from an OWL perspective. In this Thesis, we used Protégé as an ontology editor and the Protégé OWL API as the programming toolkit.

2.3.1 A brief word on the Protégé OWL API.

The Protégé OWL API [KFNM04,Knu08] is an open-source Java library for managing OWL and RDF(S) ontologies. Protégé OWL API is designed for two contexts:

- The development of components that are executed inside of the Protégé-OWL editor's user interface which are called protégé Plug-ins.
- The development of stand-alone applications, or embodiment of such applications in the user's own code.

The API provides classes and methods to load and save OWL files, to perform simple queries and to manipulate OWL data models, it also allows connections to DIG-compliant Reasoners (A DIG compliant Reasoner is a Description Logic Reasoner that provides a standard access interface). The API provides a simple procedure to implement and use graphical user interfaces (using JAVA-SWING).

2.3.2 Ontology query using query languages

Ontologies are futile if the information they contain cannot be retrieved and used; for this reason there are special languages for performing ontology queries. These languages are based on the parsing and analysis of the N-Triples in most cases or the logic over an RDF/XML file that contains the Knowledge base.

SPARQL (SPARQL Protocol and RDF Query Language) [PhS07].

SPARQL (pronounced "Sparkle") is an RDF query language of the W3C, characterized as the main query language for ontologies according to the W3C, who released it as candidate recommendation in 2006 and as an official recommendation in 2008. SPARQL allows queries consisting of triple patterns, conjunctions, disjunctions, and optional patterns. The following is an example query that returns all the butterfly valves in an industrial plant model. If the RDF literal is typed, for example as xs:integer as is the case with this generated RDF, then the following query will select the pipes with a hydraulic discharge greater than 150 liters/sec:

```
select ? pipe?discharge where {
   ? pipe rdf:type f: pipe;
   f: Discharge ? discharge.
   FILTER (?discharge > 150)
}
```

The SPARQL query processor will search for sets of triples that match the triple patterns, binding the variables in the query to the corresponding parts of each triple. To make queries concise, SPARQL allows the definition of prefixes and base URIs.

SWRL (Semantic Web Rule Language) [TNDM]

This is a proposal for a Semantic Web rules-language. SWRL combining OWL sublanguages (OWL Lite and OWL-DL) and the Rule Markup Language (RML). The specification was submitted in 2004 to the W3C by the National Research Council of Canada, and Stanford University. SWRL retains the full power of OWL DL, but at the price of decidability and practical implementations. Rules are of the form of an implication between an antecedent (body) and consequent (head). They can be interpreted as: whenever the conditions specified in the antecedent hold, then the conditions specified in the consequent must also hold.

Ontology Query open challenges

A problem that makes the query language cumbersome to work with, is the fact that every query has to parse the triples and analyze them for the query retrieval.

This fact led us to create an alternative in which the queries could be saved in the ontology for faster retrieval, leading to the Reflexive Ontologies (RO) concept that will be explained in Chapter 6. RO can be used whenever an ontology query is needed. The aforementioned technique enhances a Knowledge Base (KB) by (*i*) the speeding up of the query process (*ii*) giving to the ontology the possibility of adding new queries on individuals with the corresponding answers to such queries (a feature that adds Knowledge about the Domain); and (*iii*) the self containment of the Knowledge Structure in a single file; including the model, the relations between the elements of the model, the individuals (instances) and queries over such individuals.

2.3.3 Ontology alignment

The ontology alignment [ES07](also known as ontology matching) is the process of determining correspondences between concepts from ontologies in this context, a set of correspondences is called an alignment. Formally, given two ontologies $i =< C_i, R_i, I_i, A_i >$ and $j =< C_j, R_j, I_j, A_j >$, where C is the set of classes, R the set of relations, $\boldsymbol{I}$ the set of individuals, and A the set of attributes, we define the relationships (inter-ontology) that holds the relations amongst terms. Such relations are called alignments. An atomic homogeneous matching is an alignment that carries a similarity degree $s\epsilon[0,1]$, describing the similarity of two terms of the input ontologies i and j. Matching can be both computed, by means of heuristic algorithms, or inferred from other matching. Formally we can say that, a matching is a triple $m =< id, t_i, t_j, s >$, where t_i and t_j are homogeneous ontology terms, s is the similarity degree of m. A (subsumbtion, homogeneous, atomic) mapping is defined as a pair $\mu =< t_i, t_j >$, where t_i and t_j are homogeneous ontology terms. The Figure 2.7 depicts an example alignment schema.

2.4 Semantic reasoning

Reasoning is crucial for the application of Semantic technologies in real world applications. In order to perform a reasoning process, predicates are used in order to

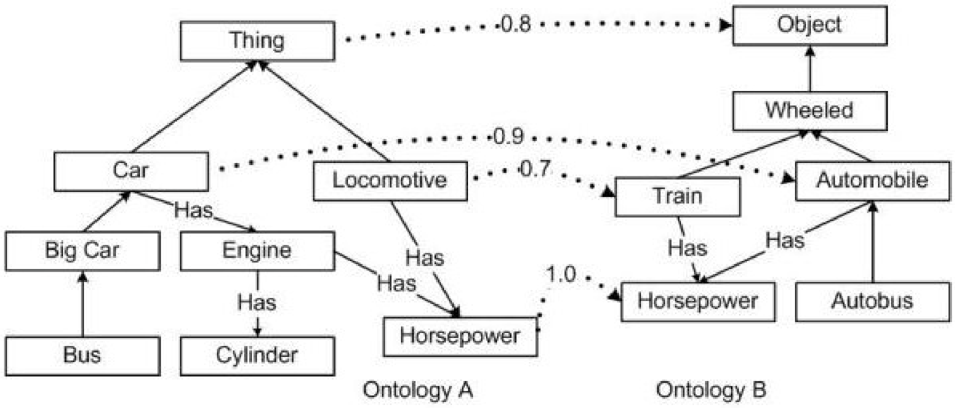


Figure 2.7: Ontology Alignment example [AHH06]

feed the inner logic that will derive conclusions about the set of involved predicates.

A predicate is a feature of language which can be used to make about something, e.g. "The Screw is grey" where "the screw" is the subject and "is grey" the color property. Analyzing the predicate structure, allows us to make use of the internal structure of atomic sentences and hence to understand the arguments that allow the Reasoner to infer Knowledge. The reasoning process is performed by a software tool (called a Semantic Reasoner) that is able to infer consequences derived from a set of asserted facts or axioms. The inference rules are specified via an ontology language in the case of ontology reasoning, First-Order predicate Logic (FOL) being the most common approach. Boolean logic states that propositions are able to take either true or false values, and FOL considers sets of predicates and individuals that when analyzed as atomic sentences produce new facts in the shape of new predicates. FOL allows the reasoning about the propositional connectives and also about quantification ("all" or "some"). A classical example of this is the set of predicates in Table 2.1, from which the conclusion that **"Socrates is mortal"** will be drawn.

Sentence	Remark
All men are mortal	First predicate
AND	Logical connector
Socrates is a man	Second predicate

Table 2.1: Example of a Reasoner characterization

2.4.1 Description Logics (DL)

DL [BHS07] are a family of class-based (concept-based) Knowledge representation formalisms. The kind of logic used, is derived from the first-order logic concept which can be used to represent the terminological Knowledge of an application Domain in a structured and formally well-understood manner. The name description logic refers, on one hand, to concept descriptions used to describe a Domain and, on the other hand, to the logic-based Semantics which can be given by a translation into first-order predicate logic. DL represents an extension to frames and Semantic networks, which were not equipped with formal logic-based Semantics. According to Lenzerini [Len96], a DL system is characterized by four fundamental aspects: the set of constructs used in concept and role expressions, the kind of assertions allowed in the TBox (Terminological Boxes) containing assertions on concepts and the ABox (Asertional Boxes) assertions on individuals, and the inference mechanisms for reasoning on both the TBox and the ABox. OWL DL and OWL Lite, can be viewed as very expressive DL, with an ontology being equivalent to a Description Logic Knowledge base. Description Logic expressivity is shown in Table 2.2.

Symbol	Remarks
$\mathcal{AL}$	Attributive language (This is the base language which allows)
	Atomic negation (negation of concepts that do not appear on the left hand side of axioms)
	Concept intersection
	Universal restrictions
	Limited existential quantification
$\mathcal{FL}^-$	A sub-language of $\mathcal{AL}$ which is obtained by disallowing atomic negation.
$\mathcal{FL}_0$	A sub-language of $\mathcal{FL}^-$ which is obtained by disallowing limited existential quantification.
$\mathcal{C}$	Complex concept negation.
$\mathcal{S}$	An abbreviation for $\mathcal{AL}$ and $\mathcal{C}$ with transitive properties.
$\mathcal{H}$	Role hierarchy (subproperties - rdfs:subPropertyOf).
$\mathcal{R}$	Limited complex role inclusion axioms; reflexivity and irreflexivity; role disjointness.
$\mathcal{O}$	Nominals. (Enumerated classes of object value restrictions - owl:oneOf, owl:hasValue).
$\mathcal{I}$	Inverse properties.
$\mathcal{N}$	Cardinality restrictions (owl:Cardinality, owl:MaxCardinality).
$\mathcal{Q}$	Qualified cardinality restrictions
$\mathcal{F}$	Functional properties.
$\mathcal{E}$	Full existential qualification (Existential restrictions that have fillers other than owl:thing).
$\mathcal{U}$	Concept union.
$(\mathcal{D})$	Use of datatype properties, data values or data types.

Table 2.2: DL-Expressivity

The Protégé ontology editor supports $\mathcal{SHOIN}^{(\mathcal{D})}$. OWL DL provides the expressiveness of $\mathcal{SHOIN}^{(\mathcal{D})}$, and OWL 2 is based on $\mathcal{SROIQ}^{(\mathcal{D})}$.

2.4.2 Reasoner tools:

Reasoner tools are pieces of software that are used for Semantic reasoning purposes. The communication with the ontology editor or the application itself is based on a HTTP-based interface language called DIG (DL Implementation Group). The DIG language is an XML based representation of ontological entities such as classes, properties, and individuals, and also axioms such as subclass axioms, disjoint axioms, and equivalent class axioms. The DIG language contains constructs that allow clients to "tell" a Reasoner about an ontology (e.g., describe an ontology to a Reasoner), and also "ask" a Reasoner about what it has inferred, such as subclass relations, type of relations, etc. Description Logic Reasoners can only work with ontologies that fall into the OWL DL family. DIG interfaces with compliant ontology query languages. In his master Thesis Zhang [Zha05], makes an extensive review of the different ontology query languages for the Semantic Web. The most important are:

- **Bossam:**

  Is a RETE-based rule engine, which is an efficient pattern matching algorithm for implementing production rule systems presented by Forgy [For82] (RETE is is an efficient pattern matching algorithm for implementing production rule systems).Bossam has native supports for reasoning over OWL ontologies, SWRL ontologies, and RuleML (Rule Markup Language) rules. [JS04]. The objective of Bossan is to provide a free, easy to use, Java based tool for reasoning over OWL DL ontologies, the development is still in progress, but initial test results have been successful.

- **Pellet:**

  Is an open-source Java OWL DL Reasoner that includes the option to perform Java calls right from the coding environment without the traditional socket layer connection required in a DIG approach (however includes a DIG compliant structure). Pellet was developed, and commercially supported, by Clark

& Parsia LLC [SPG+07]. Pellet is a commercial tool. Nevertheless, it is, along with RACER and FACT, one of the three most referenced Reasoners in the market.

- **RACER:**

  This is a Semantic Web reasoning system and information repository [HM03] implemented in C++. RACER stands for Renamed ABox and Concept Expression Reasoner, the commercial name of the software is RacerPro. There is an open distribution for students and researchers. Racer uses DL extensively and can be integrated with Protégé and JENA. It can also be used as a Semantic Web information repository with optimized retrieval engine due to its ability to handle large sets of data (e.g., defined using RDF).

- **FaCT++:**

  This is the new generation of the FaCT OWL DL Reasoner [Hor98]; it is implemented in Common-LISP and uses a DIG compliant architecture for the repository communication. FaCT (Fast Classification of Terminologies) is a DL classifier that can also be used for modal logic satisfiability testing.

The FaCT system includes two Reasoners, one for the logic $\mathcal{SHF}$ ($\mathcal{ALC}$ augmented with transitive roles, functional roles and a role hierarchy) and the other for the $\mathcal{SROIQ}$ logic ($\mathcal{SHF}$ augmented with inverse roles and qualified number restrictions). Table 2.3 summarizes key features:

	Bossam	Pellet	Racer	FaCT++
Functionality				
OWL-DL Entailment	not declared	Yes	Yes	Yes
Supported expressivity for reasoning	not declared	SHOIN(D) and SROIQ(D)	n/a	n/a
Reasoning algorithm	not declared	(forward chaining & backward chaining)	not declared	not declared
Updating the OWL-Model with inferred knowledge / Serialization of resulting ontology	Yes / Partial	n/a	n/a	n/a
Consistency checking	not declared	Yes	Yes	n/a
DIG Support	No	Yes	Yes	Yes
SPARQL Support	No	Yes	Yes	No
Rule Support	Yes (SWRL)	Yes (SWRL)	Yes (SWRL)	No
Documentation available	Yes	Yes	Yes	No
Licencing	Free/ closed-source	Free/ open-source	Non-Free/ closed-source	Free/ open-source

Table 2.3: Reasoners comparison

The use of Semantic Reasoners is not only framed to their reasoning capabilities. Semantic Reasoners can be used to check the consistency or in other words to validate the classes and relations of a logic model. This feature is very useful for debugging purposes. An example is depicted on Figure 2.8, showing the inconsistencies report on the Stanford "Pizza OWL ontology" [RDH+].

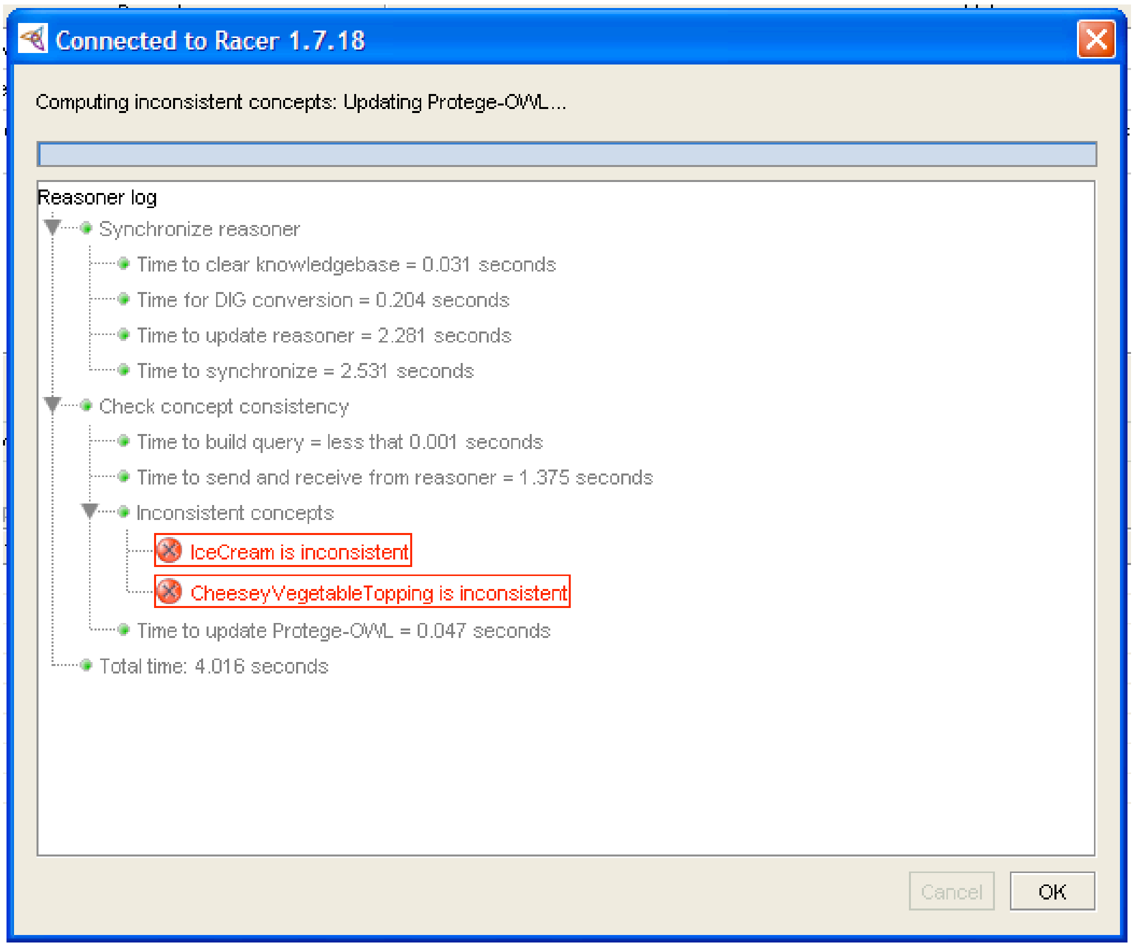


Figure 2.8: Ontology debugging using a Reasoner

Semantic Reasoners can be used also to calculate the equivalent ontology, which is a new enhanced ontology based on the logic rules imposed and the original ontology itself. Figure 2.9, show an example where the Reasoner was used to calculate an equivalent ontology which is generated using the original ontology and the set of rules expressed as DL. The new ontology, called an equivalent ontology is faster to recover and query.

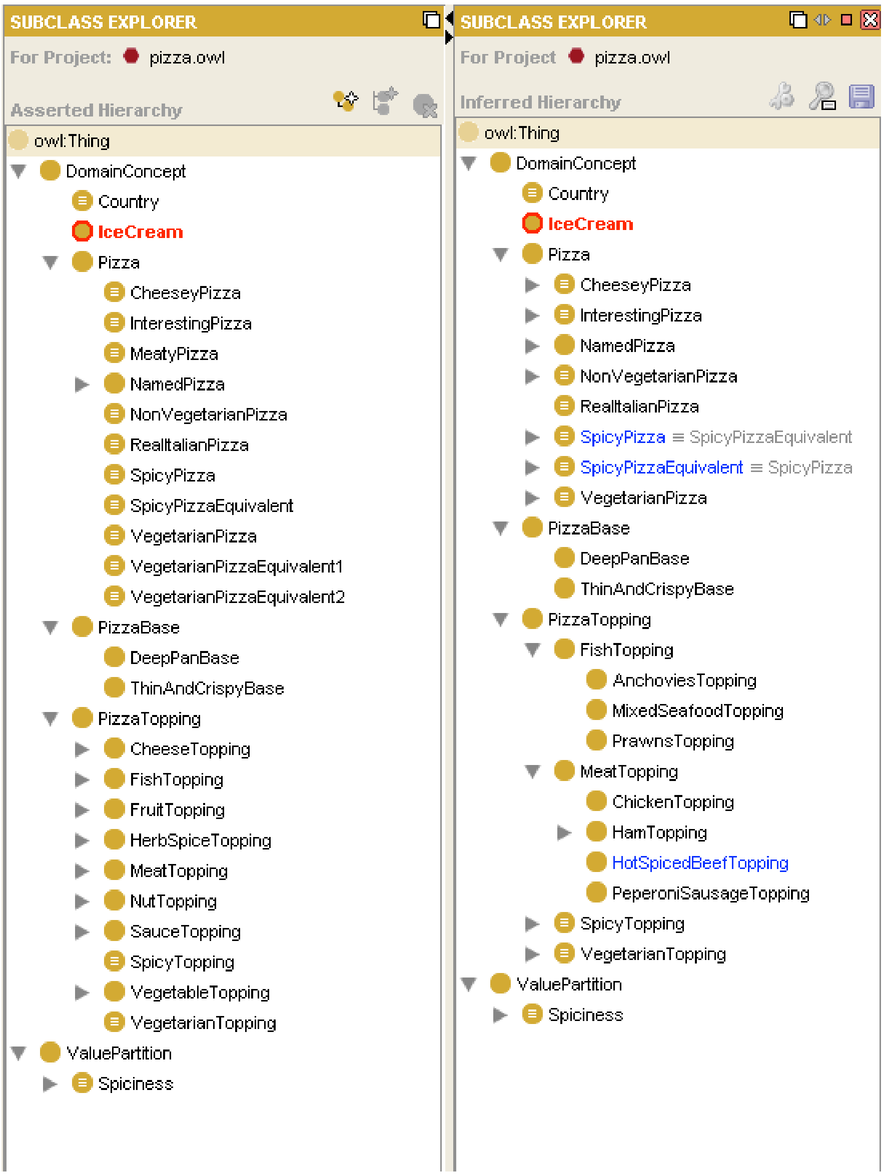


Figure 2.9: Classified taxonomy applying DL.

2.5 Discussion - Other points of view on Semantic Technologies

This chapter was intended as a brief review of the state of the art in Semantic modeling and the Semantic web, along with a set of different technologies relevant to this Thesis. The presented ideas started with a brief discussion of the Knowledge Engineering concept and continued with an introductory explanation of the ontologies concept (from the Philosophy and Computer Science points of view) along with some typical languages used for ontology modeling and handling. In order to have a complete scenario overview, in this section, we want to motivate a discussion about some skeptical reactions to the Semantic Web concept and a criticism to Gruber's classical definition used in Computer Science.

2.5.1 Skeptical reactions to the Semantic Web concept

Practical feasibility

This critique is based on the perspective of human behavior and personal preferences. The Semantic Web deals with the extension of a concept and the categorization and ordering of the data that will be used in order to produce Knowledge. However, people in real life never use good descriptions on the pieces of Knowledge they produce, e.g. lots of documents could be named "noname.doc". This would be very difficult to index, sort and hence to use as data provider for the Semantic Web. Another form of criticism deals with the innovation introduced by the concept itself, arguing that the Semantic Web paradigm existed already, all that has been done was to brand it with a name. Standardization also takes time, and since every large company has their own way to index, name and use their information pieces, the result is a myriad of possible recommendations that must be reconciled in order to produce an unique standard which at the end of the day will be mapped from every company resource schema, making the implementation issue a difficult task.

An unrealized idea

In 2001 Berners-Lee, wrote in Scientific American [Bl07] that the evolution of the World Wide Web will produce the Semantic Web. However in words of the author, this revolution has yet to happen, re-enforcing the idea of the existing paradigm and indicating that Web 2.0 could be an exaggerated rebranding of already existing technologies

Censorship and privacy

This might be viewed as a Big Brother kind of issue. If the Semantic Web is possible, then, for a government or for a large corporation, it will be easy to control even the smallest piece of information on the Web, resulting in a lack of anonymity in what is for some people the last free thinking spot on earth. In the Semantic Web architecture, there is a security layer, but who can be trusted to handle such security over the Web is a question that still has to be answered. Both censorship (for mature audience content for example) and privacy are needed, but arguably much work remains in order to entrust data to a third party.

Doubling output formats

This is a common criticism that addresses the fact that for every document on the Web, there will be a need for two versions, one machine-readable and other human-readable. This is partially solved through the publication of on-request machine-readable data on demand at in real-time. The development of micro formats has been one reaction to this kind of criticism. Specifications such as eRDF and RDFa allow arbitrary RDF data to be embedded in HTML pages. The GRDDL (Gleaning Resource Descriptions from Dialects of Language) mechanism allows existing material (including micro formats) to be automatically interpreted as RDF, so publishers only need to use a single format, such as HTML.

Do we need it?

There is a general consensus on the need for an interpreter between human language to machine language in order to express the queries, and in general, to address the handling of a Semantic Web application, however some researchers argue that with simple HTML this could be done, making the set of Semantic Web technologies unnecessary. There are latent dynamic network models that can, under certain conditions, be 'trained' to appropriately 'learn' meaning based on ordered data.

Criticism of Gruber's definition.

Although we already adhere to Gruber's definition, to be fair we will refer some of the criticisms recently raised against it. Gruber reduces the meaning of an ontology to a mere model in which the concepts or ideas of the world are expressed by a person or a group of persons who in fact model the ontology as an standardization effort. Being an ontology not about peoples' conceptions or interpretations, but about the world, the Knowledge modeled is focused in a generalized user and moreover in a designer's agreement, a fact that leads to possible inconsistencies and a user centered prior Knowledge need. The aforementioned reductive error is based on the tendency of using ontologies as a controlled vocabulary with a hierarchical organization. Gruber's definition does not provide any means for the classification of good or bad ontologies where the first ones fit into an epistemological congruence that offers a more accurate non-user based, or user-Knowledge permeable, description of the relevant Domain. In this schema, real world entities being modeled interact and rely upon. User Knowledge misrecognition is addressed as one of the contributions of this Thesis and it will be explained further in Chapter 4.

Reconciliation between the philosophical and the engineering point of view

Although the broad context of this Thesis is the engineering field, it is convenient to point out some differences between the philosophical and the Computer Science point of view. Philosophers are less concerned with respect to establish fixed and controlled vocabularies, when compared to the engineering field where a major effort in the standardization of names and properties of the elements being described is undertaken. This is due to the fact that the engineers need to describe a given element to have a name upon which a consensus exists, and such name must be sufficient to embed the characteristics needed for the modeling itself and its use in a real world application. However, it must be said that this name and characteristics standardization is not very common except for some initiatives in specific Domains, e.g. CIDOC-CRM [CC06] in the case of cultural heritage. The lack of a generalized naming and properties convention leads to two problems widely addressed in the literature about ontology modeling and handling called "ontology mapping" and "ontology alignment" that will be reviewed later in this Chapter. Philosophers have been debating possible methods to build ontologies without necessarily building them, on the contrary in the Computer Science field, large and arguably robust ontologies like WorldNet have been implemented [Fel98].

2.6 Summary

We strongly believe that a good Semantic Modeling should be almost transparent for the end-user. The concepts presented in this chapter will serve as a technical support for the core ideas of our contributions and they will be used extensively in the rest of the Thesis. There exist several ways to describe and model a Knowledge Base. In the scope of our work we will fundamentally use ontology modeling because of the benefits that such approach give. The aforementioned benefits and the different ways to fulfill an ontology-modeling task were introduced in this chapter. A special attention was given to the Reasoning processes and the possibility to use programming techniques to support the Knowledge Base and to the query processes that are basic for the real-world use of the technologies presented. By using ontology API, the possibility to automate the handling, instancing and even the reasoning processes are increased while at the same time the benefits for the User of such techniques are widened.

Chapter 3

Virtual Engineering and Product Design

In this chapter we will present a short review of some important concepts related to Product Life Cycle and Virtual Engineering. Such concepts will be used regularly along the text.

3.1 Product Life Cycle Management (PLM)

PLM is the process of managing a product's life cycle from its conception to its disposal. CIMdata defines PLM as:

> *"The strategic business approach that applies a consistent set of business solutions that support the collaborative creation, management, dissemination, and use of product definition information, supporting the extended "* [CIM08]

PLM does not comprise a definition of a piece or pieces of technology, its is a business approach for solving the problem of managing the Products Life Cycle, handling the information about the artifact definition, the creation of such information, its managing and dissemination. In the traditional PLM schema, it appears like the processes involved in each stage of development are more important than the information and knowledge produced. We believe that a smarter way of re-using such information and knowledge brings interesting new possibilities to the actors involved. As in every stage of the PLC/PLM different VEA can be found, we consider that enhancing the computer based VEA tools with semantic modeling and reasoning will give great benefits at the diverse stages of PLM and PLC.

The three core, or fundamental concepts of PLM are:

- Universal, secure, managed access and use of product definition information
- Maintaining the integrity of that product definition and related information throughout the life of the product or plant
- Managing and maintaining business processes used to create, manage, disseminate, share and use the information.

Documented PLM benefits include:

- Reduced time to market
- Improved product quality
- Reduced prototyping costs
- Cost reduction through the re-use of original data
- A framework for product optimization
- Reduced waste
- Cost reduction through the complete integration of engineering workflows

While information and knowledge in the PLM includes all media, nowadays PLM is focused primarily in managing the digital representation of that information and knowledge.

3.1.1 History of the PLM concept

PLM started in American Motors Corporation in 1985, when they were looking for a way to speed-up their product development process. The first part of their idea comprised the introduction of CAD software packages to enhance the design productivity and re-use. The second part was the introduction of an internal communication system intended for fast conflict resolution and cost reduction by using shared

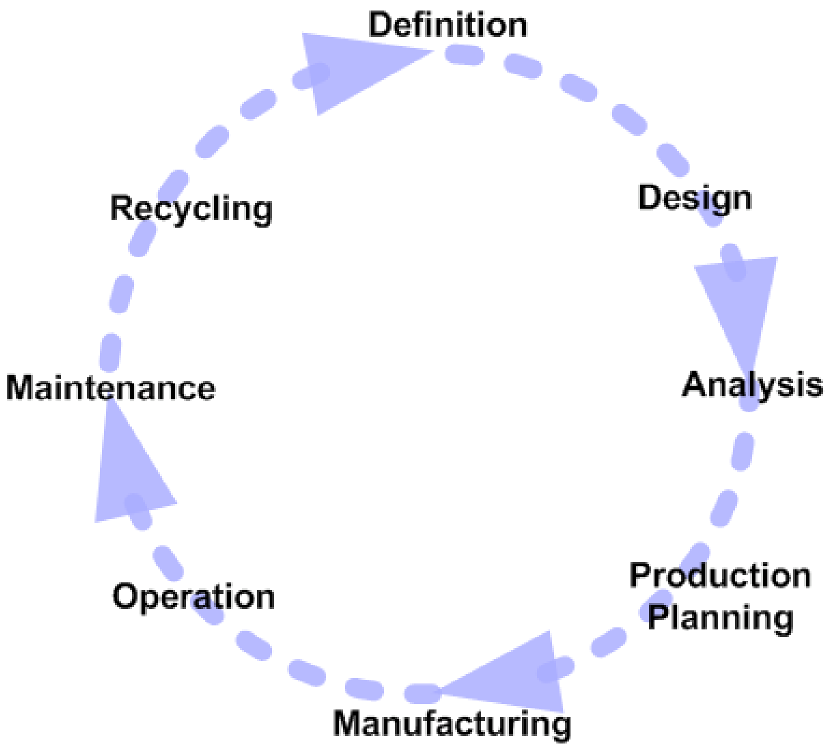


Figure 3.1: Technical Product Life Cycle [TMS+ny]

centralized databases.The concept was further extended when Chrysler purchased AMC and implemented the system in their design schemes. By implementing PLM in their production lines, Chrysler became the lowest-cost producer in the automotive industry, reporting development costs that were half of their competitors.In the early 1990's, the PLM view evolved from mechanical parts management, towards a more expanded conception including electronics and software that were enjoying great interest from a comparative-competitive point of view at that time. The expansion pushed the PLM concept towards a complete comprehension of the processes involved in the product's development ranging from marketing requirements to disposal and recycling, linking information and knowledge by using different authoring tools for production-focused attributes.

3.1.2 Stages of the PLM

As a contribution that combines the view of other authors, in (Figure 3.1) we introduce a PLM cycle that comprises the following stages:

Definition:

In this stage, the necessities for the new product are evaluated, including basic operational issues, User/Maker needs to be fulfilled and the overall desired characteristics. The VET used here are planning tools, word processors and email as the majority of the work in this stage deals with the documentation of the user needs. Some statistical VEA can be used to characterize opinion polls that can be used directly in early design. The definition stage produces a set of characteristics that must be fulfilled and initial sketches of the object, as well as extra information for the remaining stages.

Design:

Design in this schema may be viewed from two perspectives. The first dealing with a conceptualization of the product where the object is seen as viewed from a functional perspective, in this case some VEA handle design theories like TRIZ or design methods. The second point of view is how to materialize a prototype and evaluate it. For this approach, CAD-CAM tools are used in order to produce 3D models and schematics. The output from this stage is a functional prototype.

Analysis:

This stage includes the calculation of mechanical and electrical properties of the object. It uses advanced VEA that contain VET designed to perform analysis on physical characteristics (material stresses, thermal properties, etc). This stage produces a complete set of characteristics that will be present in the final product when it arrives to market.

Production Planning:

In the production planning, the design should be adapted to the plant facilities of the producer, some of the questions addressed at this stage are:

- Can it be done internally in the factory?
- Can we buy parts and adapt them to the design?

- How many units do we need to produce per day/hour?
- Does the plant need to be re-tooled in order to produce the design?

The result of this stage is the workbench and flow of work needed for the design. The usual VET in this stage can be walkthrough visualizers, production planning and cost analysis tools.

Manufacturing:

The manufacturing stage deals with the making of the product and the quantities needed, it involves CAM tools as well as production cost analysis. Stock management VEA are used for the calculation of the materials required, expenditures and warehousing of the manufactured pieces. At the end of this stage, a sales period starts. Quality control VET are very useful to assure the efficiency of the manufacturing process and the overall design excellence.

Operation:

Operation can be the longest stage in the PLM in time, as the object is actually performing what it was designed for. Operation involves in some cases a close monitoring of the design including productivity features and market analysis sales and competence awareness. VEA here are pieces of software that are included with the product such as operating systems, drivers, internal executable code, extensibility, etc. In such cases, the software should not only be maintained, but must also evolve according to the technical reality of the domain in which the product is deployed.

Maintenance:

When the design arrives to market, basic preventive maintenance should be taken into account. The producer has to maintain a stock of replacement parts for a given period of time (legally regulated). Maintenance sometimes takes place within an in-house schema in which the design returns to the factory for service (common under warranty terms). The VET required are contained in VEA databases, VEA for stock (in-out) management and maintenance support applications.

Recycling:

In this stage, the design has reached its end-of-life status and disposal should be handled by recycling what can be recycled and properly disposing of what is not. Recycling can use different VEA like databases. For example, when designing elements for a nuclear power plant, the disposal of these waste elements must be carefully and exhaustively managed (because of the potential of radioactive contamination), then VEA are used to categorize and safely store them.

3.2 Relation of PLM and Modern Visualization Techniques

Virtual Reality (VR) is a virtual environment where the user is immersed in a virtual world in which the elements posses a resemblance to real world objects. Milgram and Kishino created a Virtuality Continuum [MK94] to classify mixed worlds, in where we find Augmented Reality (AR) as a system that, according to Azuma [Azu97], combines real and virtual objects in a real environment enhanced with computer generated graphics, running interactively, in real time, and align real and virtual objects with each other. Augmented Reality provides, for example, user experience enhancements like the viewing of information related to the object in sight that has been matched and the graphical handling of the information contained in the model. Figure 3.2 identifies possible PLM points from Digital Product to Physical Product for these technologies to be part of the VET or VEA:

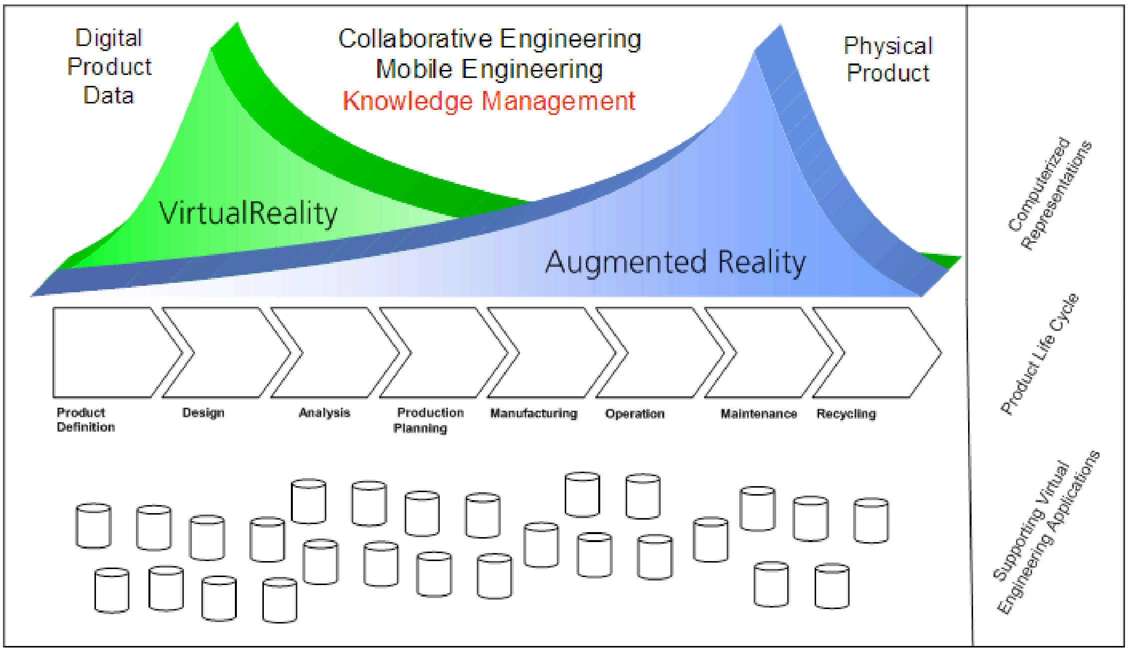


Figure 3.2: Product Life Cycle related to VR and AR, adapted from [Sto]

3.3 PLM 2.0

In early 2008, Dassault Systemes [DS] introduced the concept of PLM 2.0, encompassing a social community approach to PLM, where the re-use information and knowledge produced is based on social networks. In this philosophy, some characteristics are held:

- PLM applications are web-based (Software as a Service)
- PLM applications focus on on-line collaboration, collective intelligence and on-line communities
- PLM expands to new usages like crowd sourcing and real world web, extending the reach of PLM outside of the enterprise
- PLM business processes can be easily activated, configured and used, with on-line access
- Currently, PLM 2.0 is still more a concept proposal than a reality. But increasingly PLM computer support will embrace the concepts that have been listed here.

3.4 Engineering decision support tools (Knowledge engineering in PLM)

Diverse tasks like optimization, cost analysis, scheduling, and knowledge-based tools need to be integrated into the everyday engineering processes. Virtual Engineering Tools based on Virtual Engineering needs allows users to work in the Virtual Environments as if they were in the real world, without thinking about the object's underlying technical information. When an incorrect manipulation takes place in the virtual world, the results of such an action can be investigated experimentally, giving a powerful decision support tool. Such action in the real world could result in large financial losses as the object could be damaged or the production line stopped by a wrong decision. By having decision support tools, several configurations can be simulated in order to choose the best real world implementation, with the confidence that such implementation will behave as planned in the underlying model.

Virtual Engineering (VE) Modules [MBK03, MB07]

According to the literature, in the context of VE, the following concepts are highly relevant:

- **Computed Aided Design (CAD)**

  CAD designates the capability to model a geometry using parametrized, Non Uniform Rational B-Spline Surfaces (NURBS) or other kinds of geometric modeling techniques. The CAD module is in charge of the geometries, their creation, manipulation and handling.

- **Computed Aided Manufacturing (CAM)**

  CAM is in charge of the manufacturing operations needed to generate the geometries in the real world (in other words, to shape them by milling, drilling, casting, etc). The CAM module is able to generate tools paths, tool manipulations and in some cases small stress calculation of importance for the manufacturing processes involved in the creation of physical objects after the models.

- **Computed Aided Engineering (CAE)**

  CAE is the integrator of the above two modules and several other associated technologies, such as engineering analysis, manufacturing plans, etc. CAE oriented tools, make extensive use of the CAD geometries in some cases, but in others, they are more oriented to information management using databases of components, layout manipulations etc. According to McCorkle [MBS04], a methodology for storage and rapid access to engineering analyses, plant data, geometry, and all other qualitative and quantitative engineering data related to plant operation still needs to be developed. From our point of view, a good starting point for such methodology could be the use of semantics and the steering of VEA trough Semantic based technologies.

3.5 Discussion - Toward Semantically enabled VEA.

The ideas and technologies that support the Semantic Web provide an interesting platform to support the Virtual Engineering Applications. This can aid in the development, use and re-use of knowledge from a networked environment in which the information is meaningfully distributed. According to McCorkle the underlying question that must be answered today is:

> *"How will information be integrated in a manner that will allow commercial and proprietary software tools to remain separate while also being integrated so that the end user can control and query these tools with little to no knowledge of the tools implementation or inner-working details?"* [MB06]

The answer to this question will depend largely on the ability to implement open interfaces and schemes that can evolve over time as well as open source toolkits that enable development teams to collaborate at a high level. The enhancement of VEA by Knowledge-oriented (or Knowledge-supported) technologies has been arguably tangentially reached by some independent efforts. In [MB07], a discussion about potential applications of the Semantic Web to explore the above question is given, presenting some specific capabilities that should be fulfilled by Semantically oriented Virtual Engineering Tools. Huang [HB05] presents a decision support platform for the interactive design that integrates mathematical optimizations with human interactions based on VET. In this approach, the designer's interaction causes the optimization process to dynamically change by adding, deleting, and modifying objectives, constraints, and other parameters that govern processes. As an illustration, a coal pipe design case was used to demonstrate the platform's capabilities. Huang's case study demonstrated that adding user interaction into the design process has the potential to improve design efficiency and quality. Kuntz's [KCC+98] research was focused on the application of computational simulation models used in other branches of engineering to project planning, the authors propose a model, called the Virtual Design Team (VDT), that represents the structure and capabilities of organizational entities, such as teams (called actors in the model), activities, and both direct and coordination work. The model links the organization chart and the activity diagram of projects, and can be used to predict the effect of different organizational structures or of the use of different project constraints. In this respect, the proposed method outperforms planning tools based on the Critical Path Method, as it explicitly incorporates the necessary communication and coordination among the actors assigned to different project activities. As mentioned previously, some research into the topic has been carried out, however, at the present time, the scientific community has not reached consensus in how to extend Virtual Engineering Tools with Semantics. An architecture to solve the mentioned scenario is an open problem.

3.6 Summary

This chapter presented the basics of the Product Life-Cycle Management concept. In the context of this thesis, the PLM concept will serve as a link between the different contributions from a technical and applicative perspective. As we introduce our technical contributions in Chapters 8 to 10, we will frame each contribution in the PLM cyclic graph introduced in this chapter. By presenting our contributions using this reference frame, we can provide a better understanding of the VEA enhancement while at the same time, as a consequence we gain an extra argument in the fact that the Semantic enhancement of a VEA through its VET leads to an enhancement of the PLM stages where the VEA is used with the economical and technical benefits that such improvement brings.

Chapter 4

Recommendations on User Modeling

User Modeling (UM) is a very broad research field. Despite the many approaches to this topic that have been presented in the literature, the scientific community has yet to reach a common agreement yet. Generally the UM is addressed using a lot of different synonyms: e.g. behavioral User models, Individual User model, feature-based User model, etc. Such names seek to describe the User as he interacts with a system from slightly different points of view that can be aggregated in the same research field. Several introductory surveys on UM have already appeared in the last few years ([KF88], [NS89], [Kok91]). These surveys are mainly focused on the User interaction with a computer system or have been developed from a natural language point of view. In our research, however, we focus our attention on the more functional characteristics of the User interaction; such characteristics (according to our experience) are able to affect or modify the behavior of a VET. Our approach does not intend to introduce a new methodology for UM, instead, we present a series of considerations that may help when a UM task needs to be fulfilled.

4.1 User Modeling definition

According to Wes-Nam, User Modeling is:

> *"One way of obtaining predictive evaluation of real-world tasks by trying to represent some aspect of the User's understanding, knowledge, intentions or processing."* [Nam]

In Wes-Nam's approach, User Models are divided in three categories:

- Hierarchical representation of the User's tasks and goal structure
- Linguistic and grammatical models
- Physical and device level models

As pointed out in the introduction, several previous attempts to create a conceptualization of Users have been presented by the research community. Dolog [DN] models the User from the point of view of Engineering Standards in a similar manner to our approach for Domain models (see chapter 5). In Dolog's approach the flexibility of the User model description can be somewhat restrictive, but undoubtedly favored by the benefits of its use of Standards. Gomez-Perez et.al. [GPFC04] defined a User model as a set of hierarchies of personal and demographic characteristics with domain independent data and context and low-level sensor data. As indicated by Yudelson [YGB05], Dolog and Gomez-Perez approach place an emphasis on decomposing the User model itself, but they do not explicitly consider a methodology for UM. Yudelson's approach proposes a meta-ontology of the User-modeling field. In his work, the ontology is meant to structure the state-of-the-art in the field and serve as a central reference point and as a tool for index systems, papers and learning media. From the point of view of User interaction, there are several models presented being GOMS, explained below, one of the most important.

The GOMS model

The GOMS Model is a model of human information processing that attempts to predict what skilled Users will do in seemingly unpredictable situations ([CNM83], [JK96]). This model is one of the most recognized approaches for Human Computer Interaction (HCI). In general GOMS, is used by software designers to model User behaviors when they interact with the programs, such behaviors are characterized in terms of four components: Goals, Operators, Methods and Selection rules (see Figure 4.1).

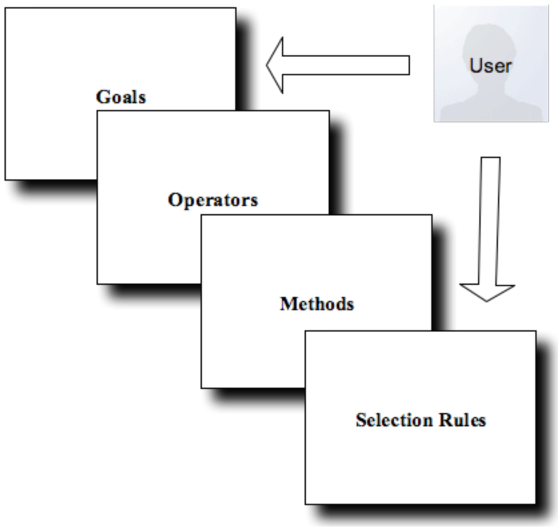


Figure 4.1: GOMS methodology components

Goals:

Goals describe what a User wants to accomplish, it can be a high level desire such as "write a Doctoral Thesis", for instance or a low level intention like "write a character". Usually the differentiation between Goals is expressed as a set of accomplishments divided into low and high level distinctions.

Operators:

In GOMS, operators are the basic perceptual, cognitive, or motor actions used to accomplish goals, or the actions that the software allows the User to take e.g. to press a keyboard key. In the world of VET, operators are equivalent to the concept of Tools however in such case, their representation is more perceptual and cognitive than physical.

Methods:

These are the procedures (also called sequences) of sub-goals and operators that are needed in order to accomplish a goal. Methods are normally considered as a set of actions needed in order to fulfill a desired task e.g. to perform a stress calculation on a particular beam, the elements must be geometrically described and the properties of the material must be known. In the mentioned case, any CAD software could be used to obtain the geometries, but the non-geometric features e.g. engineering properties tables etc, should be obtained from a more specialized VEA like a Finite Element Analysis (FEA) program for example.

Selection Rules:

These are the collection of personal rules that the Users follow when deciding what method to use in a determined circumstance. This is basically a strategy of problem confrontation; it means that for a certain task, divided in sub-tasks, documented actions can be taking place focusing on the overall solution of the problem. The selection rules are NOT experiences in the strict sense of the word; instead they look more like recipes in the form of "if X is needed and Y is available, do Z". An interesting application of the selection rules principle from the point of view of visualization was presented by Posada in his doctoral dissertation [Pos05]. In this work, he introduced a technique that allows the selection of the best algorithmic approaches for a User/task-model-resources schema. Besides, Posada's work is relevant to this Thesis because the Semantic techniques are used in order to provide the best strategy for visualization.

Recent applications and open challenges of the GOMS model

In recent times, the GOMS model has been used to extend and enhance tasks in the well-known technique of the Critical Path Method (CPM) in a technique presented by Grey and John called CPM-GOMS [GJA92]. CPM-GOMS was used in project Ernestine carried by New England Telephone. In this implementation, newly ergonomically designed workstations were compared to old workstations in terms of improvement telephone operators' performance. CPM-GOMS analysis estimated a 3% decrease in productivity. Over the four-month trial 78,240 calls were analyzed and it was concluded that the new workstations produced an actual 4% decrease in productivity. As the proposed workstation required less keystrokes than the original it was not clear from the time trials why the decrease occurred. However CPM-GOMS analysis made apparent that the problem was that the new workstations did not utilize the workers' slack time. Not only did CPM-GOMS give a close estimate, it but also provided more information on the situation. GOMS has some open challenges, the most relevant are the following ones:

- Tasks must be analyzed in terms of procedural Knowledge.
- GOMS is used to represent only skilled behavior. It means that is not very useful when the problem is poorly defined. For such cases cognitive walkthrough techniques can be used.
- It needs to follow a top-down approach, describing the more general tasks first and the low level operations at the end of the modeling process.

4.2 User Modeling Recommendations for VEA

In this section we will present our recommendations for UM. Our recommendations can be considered when a User model task needs to be fulfilled, no mater which modeling technique has been chosen (e.g. GOMS, etc). The suggestions we make in this part of the thesis have been tested and found useful in our work through different projects and specifically, they will play an important role in the chapters of this thesis presenting our technical contributions. As pointed before, we do not intent to present a methodology for UM, nor a guide, but just a collection of suggestions that we have followed when User-modeling tasks are needed and that have been useful for us. It is up to the User modeler, which ones of these recommendations will be considered in his model, very specifically in the VEA application field, taking into account Semantics in that modeling. The application of one of our recommendations does not imply or need the application of other.

4.2.1 Consider User Roles in your User types

A sensible approach to UM is to classify the users into types and sub-types, e.g. technical, non-technical, etc. in a tree-like schema. We recommend considering User Roles for such kind of approaches.

Constantine points out that:

> *"A User role is a relationship with a system. Users within roles perform tasks.*

> *Tasks are what Users do; roles are about how they do it. The key to succinct characterization of User roles is differential description."* [Con06]

User roles can be modeled as a kind of User behavior conceptualization on a domain. Such behaviors could be valuable for modifying the behavior of a VEA or they could be used to recommend the best VET for the User's goal completion. In the mentioned scenario, the same User is able to perform different duties in the Domain, even when these duties are not within his areas o expertise e.g. a technical User performing basic administrative tasks. When performing such duties, the User can take advantage of the VET that an expert usually works with, but as the system knows that he is not an expert, such VET can adapt itself to the user in order to complete the goal with the minimum effort. Moreover the VET itself can recommend the use of an expert. Roles can be the result of overlapping characteristics within different User profiles. There are some things that must be kept in mind:

Overlapping of characteristics can be wide, but degrees of knowledge on them could be narrow.

It is not easy to foreseen the degree of knowledge needed in order accomplished a certain goal with the actual User knowledge. No matter how much urge exist to perform a highly technical task or goal, the User needs a minimum degree or certification. A VET requires a minimum User Knowledge of the Domain in order to be handled by any User. In the current state of the art, VEA are not capable to take decisions autonomously. However, they can provide options. We believe and maintain that a human User will always be needed to select between the various possibilities offered by a VEA. Of course VEA can be enhanced, in fact this thesis presents ways of semantically doing so, but no matter how much enhanced they might be, the human User should be the only one responsible to make decisions. An enhanced VEA value, relies on the fact that such VEA can provide quickly the most accurate options possible for the task at glance. We also base our contributions that such enhancement of VEA is performed within the contained set of tools (VET) and by using those VEA in a PLM scenario the Semantic enhancements provide better information systems, cost reductions, re-usability and accuracy in the technical processes involved in the product.

Roles are not profiles

The concept of a profile is related to the sets of characteristics that belong to a User. Such characteristics can be areas of expertise, knowledge, common tasks, etc. It can be argued that those characteristics are in a way, areas of expertise of the User, e.g. a Mechanical Engineer "knows how to perform stress calculations in beams". Roles are more related to possible overlapping characteristics, e.g. a Mechanical Engineer "is able to create a Gantt chart". Figure 4.2 depicts an excerpt of a User Model that considers the Roles recommendation; Figure 4.3 depicts the same excerpt using ontology representation as the modeling paradigm, as can bee seen the Role "CAD designer" is shared between both the technical Users Architect and Engineer.

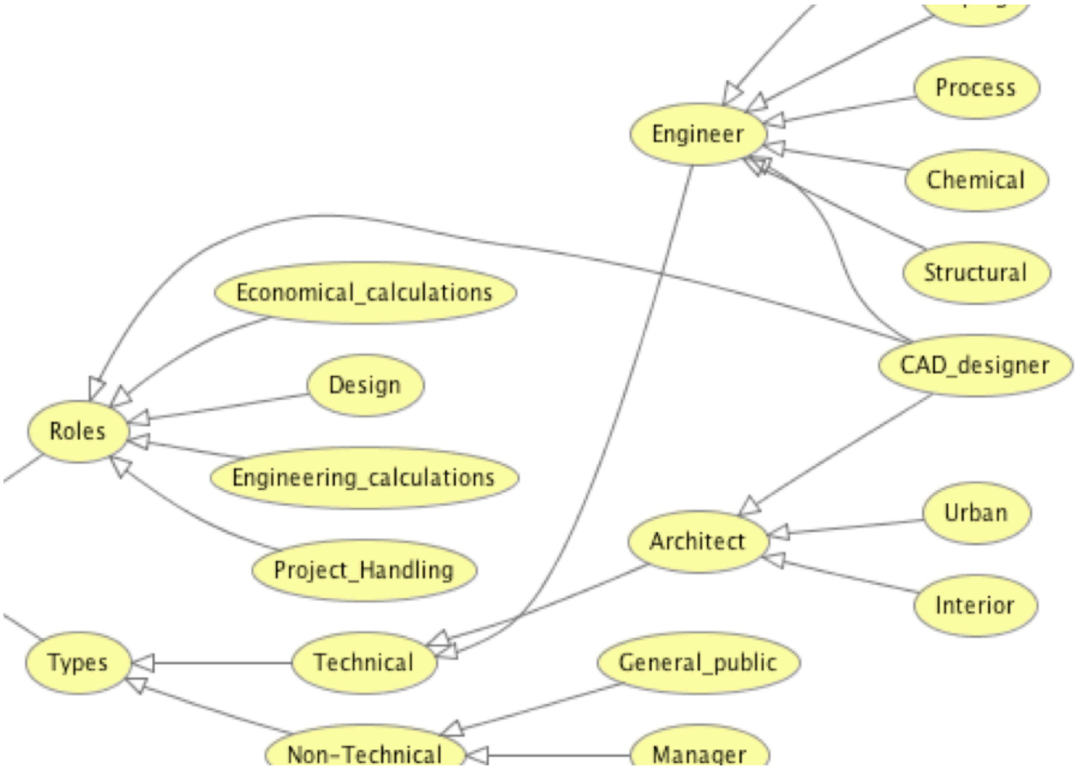


Figure 4.2: Considering Roles in UM (excerpt)

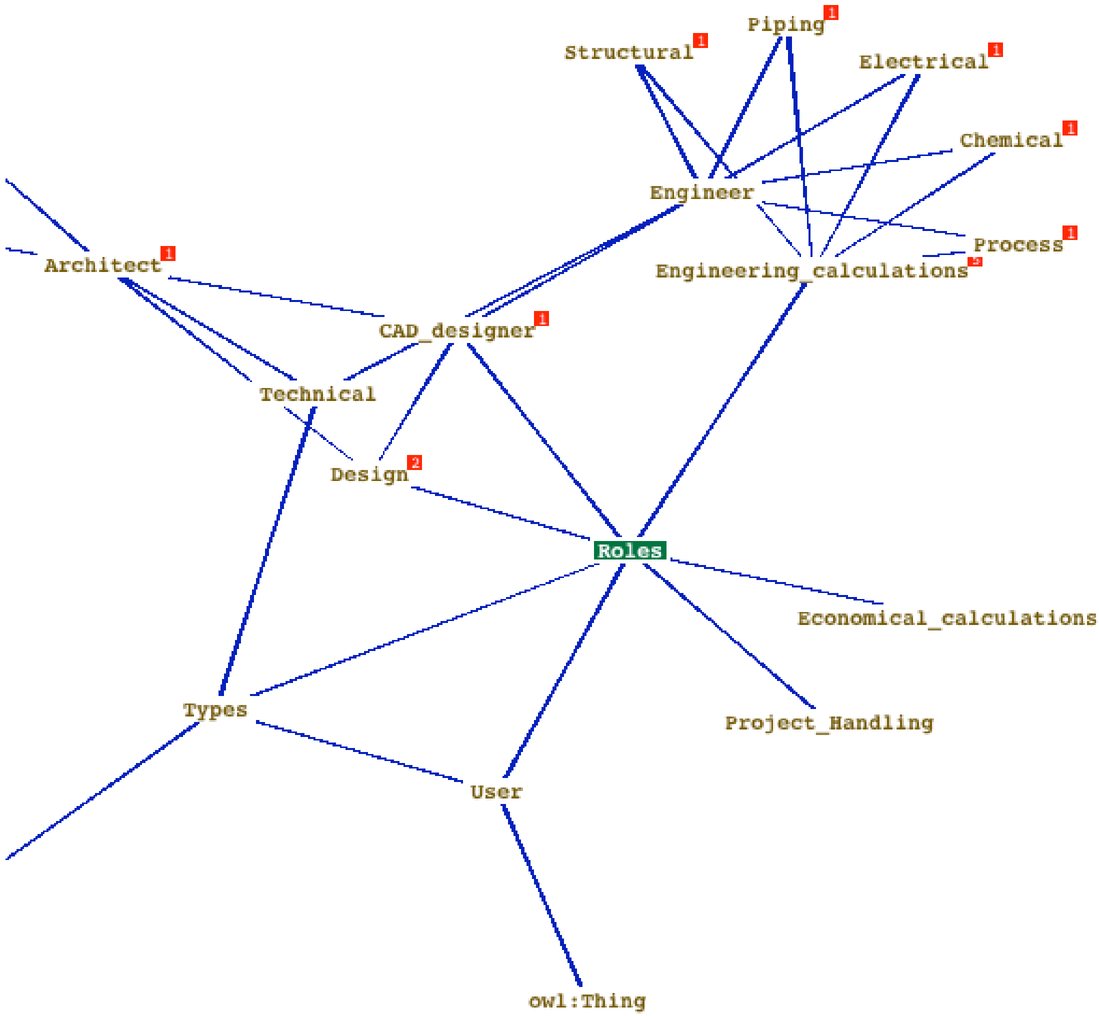


Figure 4.3: Considering Roles in UM, Excerpt modeled using ontologies

4.2.2 Consider implicit Semantics and explicit Semantics in the UM

User Semantics

In our approach to the Semantic Enhancement of VEA, we consider the set of tools belonging to such applications (the set of VET), the natural point from where an enhancement of the VEA can be done. Such VET are in many occasions driven by Users that provide additional contextual information. The overall result of this User-oriented approach enhances the User experience, providing him with the "best tool for the case" amongst other other benefits. User Semantics in our approach refers to the Knowledge that can be obtained from the User through the User-VET interaction. Such Knowledge is embedded in the VET and hence in the VEA in two different ways: Explicit and Implicit.

Explicit Semantics

Explicit Semantics means that the Knowledge can be extracted from related information already present in the VEA [Pos05]. In other words, explicit Semantics are the Semantics that can be accessed directly from the model or the VEA when acting upon the model. Apart from the geometric and topological data inherited from models handled by VEA, the inner representations of such models often contain extra information. Such information can be in the form of a layer-structure applied during the design phase, a hierarchical tree-like-structure that gives important topological information on how parts relate to each other or the existent grouping information of cells inserted from part-libraries. Furthermore, parts of such models may have been annotated with explicit information about their function, their specifications, bills-of-material data, domain specific data, etc. All this gives valuable information about the model that can be used by a proper User model to adapt it to produce an enhancement to the VEA. Explicit Semantics are the result of User interactions with the model mainly during the initial stages of the PLM, for example by structuring the model into levels, naming and coloring parts, annotating parts with product information, etc during the Design stage.

Implicit Semantics

On the contrary, implicit Semantics is when a User (typically an expert) is required to identify and reconstruct certain Knowledge contained in a VEA. Such Knowledge may not be stored in an explicit manner, but instead, it must be identified using the User. For example, when dealing with a certain VEA that focuses on Visualization. The exploitation of implicit Semantics, allows a better Virtual Reality model (as will be presented in Chapter 9). Such enhanced experience is oriented to the User, who introduced the semantic information and whose Knowledge is required in the visualization walkthrough process. In Figure 4.4, User Semantics considerations have been taken into account. The image on the left depicts a section of an elbow component in full detail, while the rightmost part is the same element, but in this case, the visualization has changed knowing that the User has a technical background (the number of elements have been minimized by representing the elbow using descriptive geometry techniques ergo minimizing the needed number of triangles). In order to process Implicit Semantics, the VET must be enhanced (often programmatically) and on this occasion, it is when the use of API, Reasoners and Semantic handling tools in general become very useful.

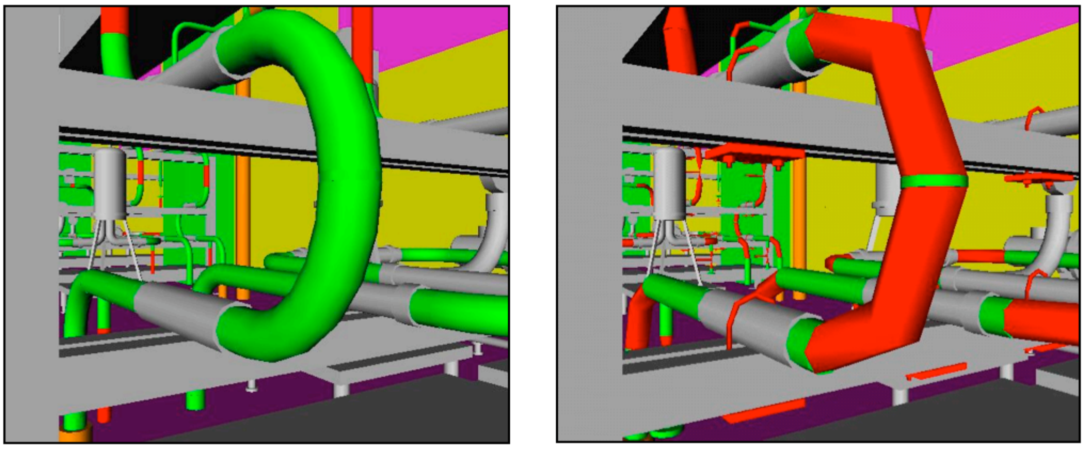


Figure 4.4: User Semantics used for visualization: Different valid graphical representations for a Pipe element

4.2.3 Consider the User Experience in the UM

Experienced Users can be considered as highly trained decision-makers who base their decisions on situation assessments [D.N98]. In other words, decision-makers principally use experience for their decision. When a decision event emerges, Experienced Users, select actions that have worked well in previous similar situations. Similar situations at this stage operate at an abstract level. Nevertheless, when applying similarity in a realistic level, it depends upon the knowledge representation used. Advanced trained or highly experienced Users, extract the most significant characteristics from the current circumstances, and relate them to similar situations and actions that have worked well in the past. This reasoning strategy stresses the importance of keeping records of decision events. As a consequence, tools for representing and storing formal decision events in a knowledge-explicit way are evidently necessary, understanding that a formal decision event is a decision occurrence that was made following procedures that make it structured and formal [SS05b]. Considering Experience issues in a UM, is not a simple task, indeed not many approaches have been presented. However one interesting possibility to consider is the User of the Set of Experience Knowledge Structure (SOEKS), presented by Sanin and Szczerbicki ([SS04, SS05b, SS06b]) and extended by Sanin and Szczerbicki and Toro [SST07].

The Set of Experience Knowledge Structure (SOEKS)

Arnold and Bowie claim that:

> *"The mind's mechanism for storing and retrieving knowledge is transparent to us... somehow, during this process, all the essential qualities of an object are stored. Later, when someone mentions it, our senses are activated from within, and we see, smell, touch, and taste the object all over again"* [AB85]

Computers, unfortunately, are not yet capable of forming representations of the world in this way, and even simpler, for representations of just formal decision events. This presents a problem of how to adequately, efficiently, and effectively represent information and knowledge inside a computer. The SOEKS (see Figure 4.5) has been developed to store formal decision events explicitly [SS05a]. It is a model based upon existing and available knowledge, which must adjust to the decision event domain upon which it was built from. Four basic components surround decision-making events: variables, functions, constraints, and rules. They are stored in a combined dynamic structure that comprises a Set of Experience. Variables typically involve representing knowledge using an attribute-value language (that is, a vector of variables and values) [Llo03]. This is a traditional approach, used since the origins of knowledge representation, and is the starting point for a Set of Experience. Identification of the Variables that intervene in the process of decision-making is the first step in the construction of the Set of Experiences. These variables are the root of the structure, because they are the origin of the other components. Based on the idea of Malhotra [SSeT] who maintains that "to grasp the meaning of a thing, an event, or a situation is to see it in its relations to other things", variables are related among them through functions. Functions, the second component, describe associations between a dependent variable and a set of input variables; moreover, functions can be applied for reasoning optimal states, because they come out from the goals of the decision event. Therefore, a Set of Experiences uses Functions, and establishes links among the variables constructing multi-objective goals. According to Theory of Constraints (TOC), Goldratt [GC86] affirms that any system has at least one constraint; otherwise, its performance would be infinite. Thus, constraints are another form of relationships amongst the variables; in fact, they are also functions. A Constraint, the third component of a Set of Experiences, is a restriction of the feasibility of solutions in a decision problem, and a factor that limits the performance of a system with respect to its goals. Finally, Rules are suitable for associating actions with the conditions under which the actions should be performed. Rules, the fourth component of set of experience, are another form of expressing relationships among variables. They are conditional relationships that operate in the universe of variables. Rules are relationships between a condition and a consequence connected by the statements IF-THEN-ELSE. In conclusion, the Set of Experiences consists of Variables, Functions, Constraints and Rules, which are uniquely combined to represent a formal decision event. SOEKS can be used in platforms to support decision-making, and new decisions can be made based on sets of experience. SOEKS is able to collect and manage explicit knowledge of different forms of formal decision events [SS05a]. The SOEKS has been developed as part of a platform for transforming information into knowledge named Knowledge Supply Chain System.

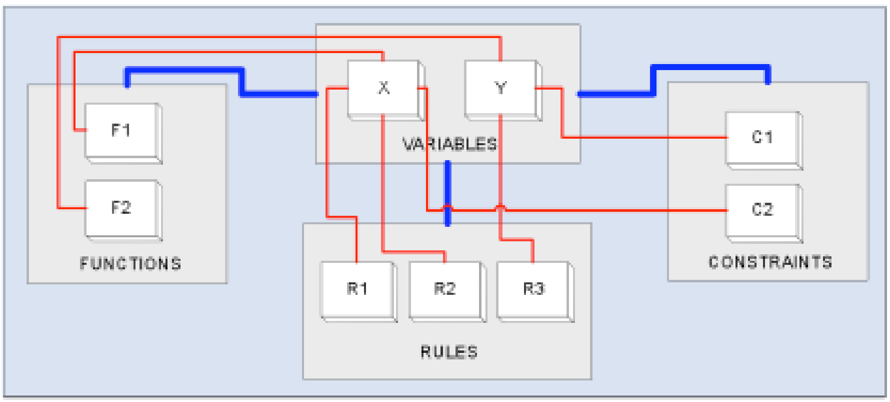


Figure 4.5: Components of the Set of Experience Knowledge Structure

The combination of the four components of the SOEKS offers distinctiveness due to the elements of the structure that are connected among themselves, imitating part of a long DNA strand. A SOEKS produces a value of decision, called efficiency. Besides, it is possible to group sets of experience by category, that is, by objectives. These groups could store a "strategy" for such category of decisions.

Contribution of SOEKS in the UM tasks

As pointed before, SOEKS introduces a novel way to model User Experience from a conceptual perspective. The modeled experience can be used in the UM as a valuable feature that adds in fact Knowledge obtained from past experiences. The aforementioned benefit, provides an enhanced User model that as a novelty includes considerations that are outside the scope of traditional UM techniques. We believe that the use of SOEKS as presented by Sanin and Szczerbicki is one of the most important key tools that this thesis advantages. We argue that UM Before SOEKS was an incomplete process, however we reckon that work needs to be done in order to see the actual technology at a market level.

4.2.4 Distinguish between high and low level Goals when modeling the User

In general, it is common wisdom that goals or tasks must be defined with the best detail possible in order to make an efficient UM. Goals describe what a User wants to accomplish. As it was stated in the GOMS methodology, dividing Goals into high level and low level groups is a technique that helps in the understanding of the overall expectations of the User. Such issue must be reflected in the User Model. Dividing Goals into sub-goals will help to identify a series of needed VET and such identification derives in the fact that some VET must not be available in the in-house VEA.

4.3 Discussion - UM recommendations and their relation with the GOMS model

In this Chapter we presented a series of recommendations for UM. Our recommendations were not intended to be used as a modeling methodology, instead of that, we presented them as a set of concepts that could be taken into account when the UM task needs to be fulfilled. Between other topics covered, we over viewed a classical methodology for User Modeling called GOMS. As a conclusion of such exposition, in Figure 4.6, a mind-map containing the relation of the concepts presented in the Chapter with the GOMS model is depicted.

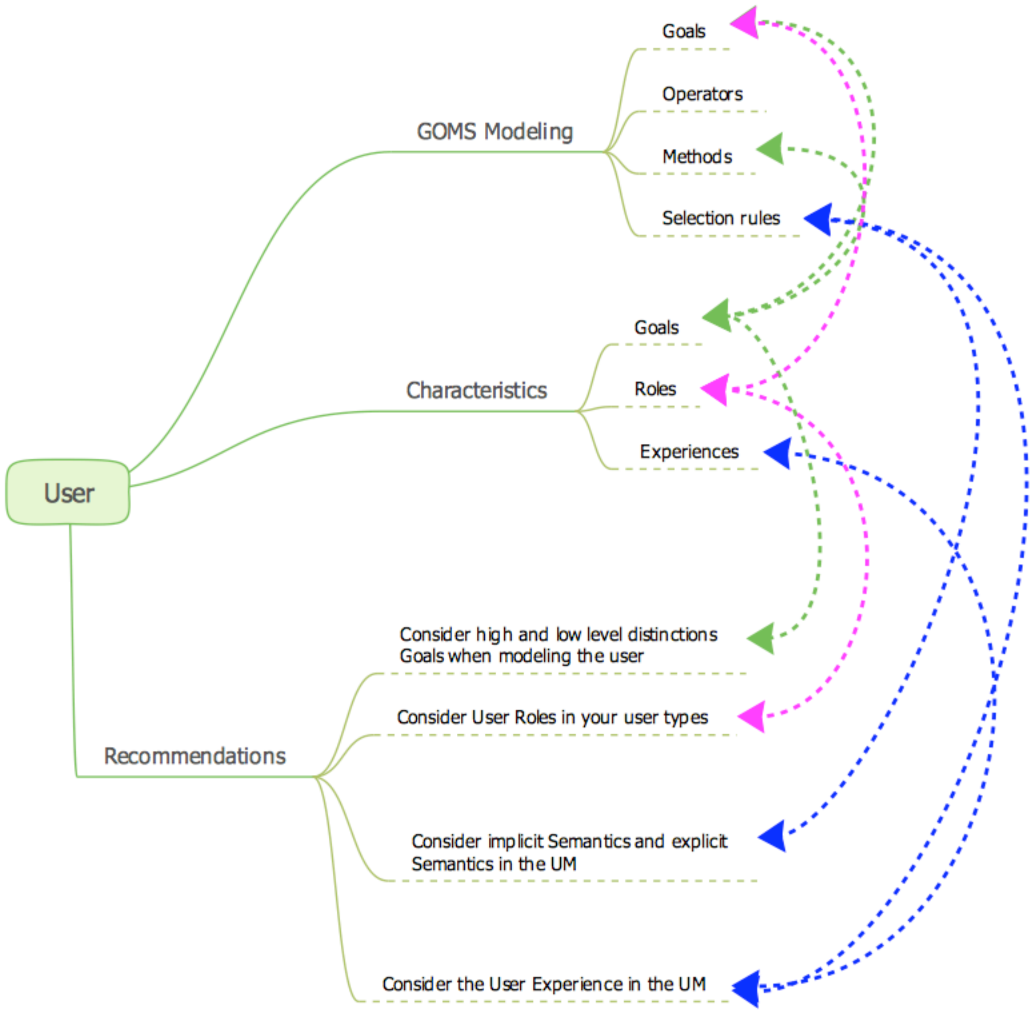


Figure 4.6: Relation of our recommendations and GOMS

As can be seen, the mind-map is divided in three sub-topics: (i) the GOMS-modeling, relating the concepts of the same name methodology, (*ii*) the User Characteristics, showing a basic user characteristics division and our (*iii*) recommendations. From GOMS description, it is known that the User Rules are the collection of personal rules that the Users follow when deciding what method to use in a determined circumstance. Our recommendations about the differentiation and use of explicit and implicit Semantics, and the user Experience modeling, clearly point to an enhancement of the User Rules modeling. The enhancement comes when not only the classical set of rules are applied for the User Rules modeling (we call classical to a set of typical "tools of the trade" actions). By adding User Experience to the Rules set, we can have a sort of adaptive set that can grow in time, tightening decisive factors while reducing the amount of Rules to be considered. The differentiation between explicit and implicit Semantics forces the modeling process to be designed with a query system in mind. The aforementioned fact links directly to the need of a fast query system that will be presented in Chapter 6. The recommendation for the consideration of User Roles, can be used to enhance and tighten the Goals step in the GOMS model, because the more we known about the User characterization, the better work can be done in order to model a correct set of Goals. Such Goals are once again dependent of the GOMS methods step, as by a logical approach, in order to obtain a greater goal, small sub-goals could be performed during a period of time.

4.3.1 UM recommendations translated to the real world

It is a decisive step in the use of any Knowledge Based technique the modeling of the User as it will be shown in the technical contributions to of this thesis. However, we believe that such modeling at the current state of the art was missing some interesting points that our work focused in by providing the recommendations presented in this chapter. As a proof of concept of the above asseveration, we can mention the case of Industrial Maintenance discusses in Chapter 10, where the UM model enhanced with our recommendations played a decisive role. From a conceptual perspective, a proper User Modeling provides a series of benefits that can be easily foreseen; however the relation of such benefits and the real world is also important. Conceptual models exist for many reasons, between them we can mention the generalization efforts, the standardization of processes and the elements involved and in general an abstract understanding of the real world. User Modeling is not indifferent to these needs, by having an enhanced conceptualization of the User, the processes in which such user takes part are better understood while at the same time, the experience of using VEA is enhanced by the exploitation of the characteristics, Knowledge, Past experiences and even expertise of the User interacting. From a PLM point of view, the better understanding of the User, which is a direct consequence of a good UM; empowers the processes involved and provides better products with the User in mind. The paradigm of design for usability [CR92] is centered in the idea of the User and his interactions, such perspective can be advantaged from the recommendations described in this Chapter.

Supporting publications

This Chapter was supported by the following Papers and Journals in International conferences:

- Toro, C. ; Termenón, M ; Posada, J ; Oyarzun, J ; Falcón, J: Ontology supported adaptive user interfaces for structural CAD design In: Digital Enterprise Technology, Perspectives and Future Challenges, in Cunha, Pedro F.; Maropoulos, Paul G. (Eds.) ,2007, XVIII, 593 p., Hardcover, ISBN: 978-0-387-49863-8

  Remarks: In this book chapter, th UM recomendations are applied for the specific case of User Interfaces for CAD.

- Sanin, C. ; Toro, C. ; Szczerbicki, E. : On the Semantics of a Knowledge Structure. International Conference on Information Systems Architecture and Technology, ISAT'2006, September, Wroclaw, Poland, pp. 239-248

  Remarks: In this work, the use of ontologies for the implementation of the set of experience knowledge theory, introduced by Sanin and Szczerbicki, is addressed. The Set of Experience Knowledge (SOEKS for short) is a methodology that allows an engineering system to have expertise knowledge embedded within.

- Sanin, C. ; Szczerbicki, E. ; Toro, C. : An OWL Ontology of Set of Experience Knowledge Structure in Journal of Universal Computer Science , Volume 13, number 2, pages 209-223, 2007

Remarks: This Journal paper introduced an OWL version of the initial ontology intended for the embedding of user experience in a Virtual engineering Application using the ideas presented by Sanin and Szczerbicki in their foundational works.

- Ortiz, A ; Oyarzun, D ; Carretero, M ; Toro, C. ; Posada, J : Avatar behaviours due to the user interaction in IADAT Journal of Advanced Technology on Imaging and Graphics. Volume 1, number 1, pp. 28-31. Portsmouth, United Kingdom, 7-9 September 2005

  Remarks: This journal paper was the extension of the conference paper entitled "Animation Techniques for Achieving Avatar Interaction in a Virtual Environment", in such work, a methodology to embed an avatar in a Large CAD model is presented, giving the avatar the possibility to interact with the environment.

- Marcos, G. ; Smithers, T. ; Jimenez, I. ; Toro, C. : Meta Level: Enabler for Semantic Steered Multimedia Retrieval in an Industrial Design Domain in Systems Science, Vol. 33 No. 2: 15-22, ISSN: 0137-1223, Wroclaw, Poland 2008.

  Remarks: This Journal paper introduced a semantic steered multimedia retrieval tool intended for the search of physical features that could aid in the design of a new car. One interesting fact is the user-oriented schema followed in the presented architecture that allows an extra enhancement of the provided solutions.

Chapter 5

Recommendations on Domain Modeling based on Engineering Standards

Webster Dictionary defines a Domain as:

> *"A sphere of knowledge, influence or activity"* [mer09]

In Computer Science, we consider a Domain as a sphere of Knowledge identified by a name; where the contained information is a collection of facts about entities. In other words, a Domain describes the elements and characteristics belonging to a Knowledge Base (KB). As mentioned earlier in this thesis, a domain ontology (or domain-specific ontology) models the Knowledge in a specific Domain, representing particular meanings for the contained terms. For example, in the Plant Design Domain, the concept *elbow* is a specific type of bend pipe used to change the direction of the fluid, while in the anatomy Domain the same concept represents a part of the body between the forearm and the hand. Considering similar Domains, the degree of specialization and conceptualization even with the same concept can slightly vary (e.g. a piping engineer will consider a wider definition of an elbow when compared to an structural engineer). The Domain problem does not stop only in the concept definition. Any concept in a domain needs also the characterization of its properties e.g. for an elbow in the Plant Design Domain, distinctiveness like the radius, the curvature, etc. In this chapter we will present a methodology that can be used to aid in the Domain modeling of a KB using Engineering Standards.

Reasons for using Engineering Standards (ES) as models for Knowledge Bases

The use of Engineering Standards as models for the underlying KB, provides the following benefits:

- **Consensus:** Building the standard takes a lot of effort establishing a consensus about terminology, organization and logic of the Domain. Therefore KB models based on standards will be widely accepted and recognized as relevant.
- **Format support:** Many VEA support Standards as input/output formats (e.g. CAD software with STEP modules [ISO01]) this fact helps in the categorization of elements and the mapping of such elements into he Knowledge Base.
- **Avoidance of Semantic loss:** The modeling of ES, usually considers not only the element's concept in isolation, but also, the relation of such element with surrounding objects. The aforementioned fact is indeed a very valuable feature that helps in the conservation of the Semantics properties of such elements.
- **Easiness of a new Domain modeling based on an existing Standards:** If there is no existing ES for a given Domain, a ES complying with similar characteristics can be used. An example to this is the use of STEP application protocol 227 (plant design) [ISO01] for the case of ship design.
- **ES are revised on a regular basis:** The nature of an ES is eminently evolutionary due to the development of new technologies for fabrication or the typical evolution of engineering paradigms. When using ES as base for KB, there exist an intrinsic guarantee of using the most recent data models (if the Knowledge Base is updated accordingly).

The above benefits have been proved in diverse approaches ([PTWS05b, TPOF07, CTS+08]) and as it will be shown in our technical contributions, the use of Engineering Standards is a core technique for our work.

5.1 Engineering Standards definition

According to BSI (British Standards organization) [BSI], a standard is an agreed, repeatable way of doing something. It is a published document that contains a technical specification or other precise criteria designed to be used consistently as a rule, guideline, or definition. Standards help to make life simpler and to increase the reliability and the effectiveness of many goods and services we use. Standards are created by bringing together the experience and expertise of all interested parties such as the producers, sellers, buyers, users and regulators of a particular material, product, process or service. Standards are designed for voluntary use and do not impose any regulations. However, laws and regulations may refer to certain standards and make compliance with them compulsory. For example, the physical characteristics and format of credit cards is set out in standard number ISO/IEC 7810:1996. Adhering to this standard means that the cards can be used worldwide. Any standard is a collective work. Committees of manufacturers, users, research organizations, government departments and consumers work together to draw up standards that evolve to meet the demands of society and technology. A comprehensive list of standards organizations can be found in [Dav], between the most relevant standardization organizations we can mention: DIN [DIN], which is the German Standards organization, whose work is commonly used as the base of many European Standards, CAM-I, which is the international association for Manufacturing Standards [CAM], BSI, the British Standards organization (the oldest organization of this type in the world) [BSI] and ISO, which is the International Standards Organization [ISO].

5.1.1 Standards development process

According to BSI [BSI]: “the task of drafting formal full consensus standards (BS, EN, ISO) is usually delegated by a technical committee or subcommittee to a drafting group or panel. Occasionally a committee may commission a consultant to complete the drafting. There are specific rules for drafting standards that must be adhered to. These are designed to ensure that standards meet their aim of providing, for common and repeated use, rules guidelines or characteristics for activities. They are founded on usability, verifiability and commonality". Lead times for standards vary from a matter of months to several years. As an example to this, British Standards [BSI] are usually developed within 12 and 15 months, whilst international standards take around 3 years. Commissioned standards such as PAS and PS can be developed within months to meet customer requirements.

5.1.2 Benefits of using Standards

The ability to demonstrate compliance with widely recognized and respected standards is an effective mean of differentiation in a competitive marketplace. In addition, manufacturing products or supplying services to appropriate standards maximizes their compatibility with those manufactured or offered by others, thereby increasing potential sales and widespread acceptance. As consumers become increasingly informed about their choices, conformity to recognized standards becomes pivotal. An example is the international standard for environmental management (ISO 14001), increasingly used by businesses to demonstrate environmental responsibility.

5.2 A methodology for using Engineering Standards as models for Knowledge Bases

In this section, we introduce our methodology for the domain modeling based on ES. Our methodology is divided in a series of logical steps that must be performed to assure a correct modeling of the domain (see Figure 5.1). The presented methodology has been used in some research projects, as it will be shown in the technical contributions of this thesis. We divide our methodology in 4 layers: Define, Identify, Model, Instantiate, encompassing eight stages as follows (an example of use can be found in the annexes of this thesis) :

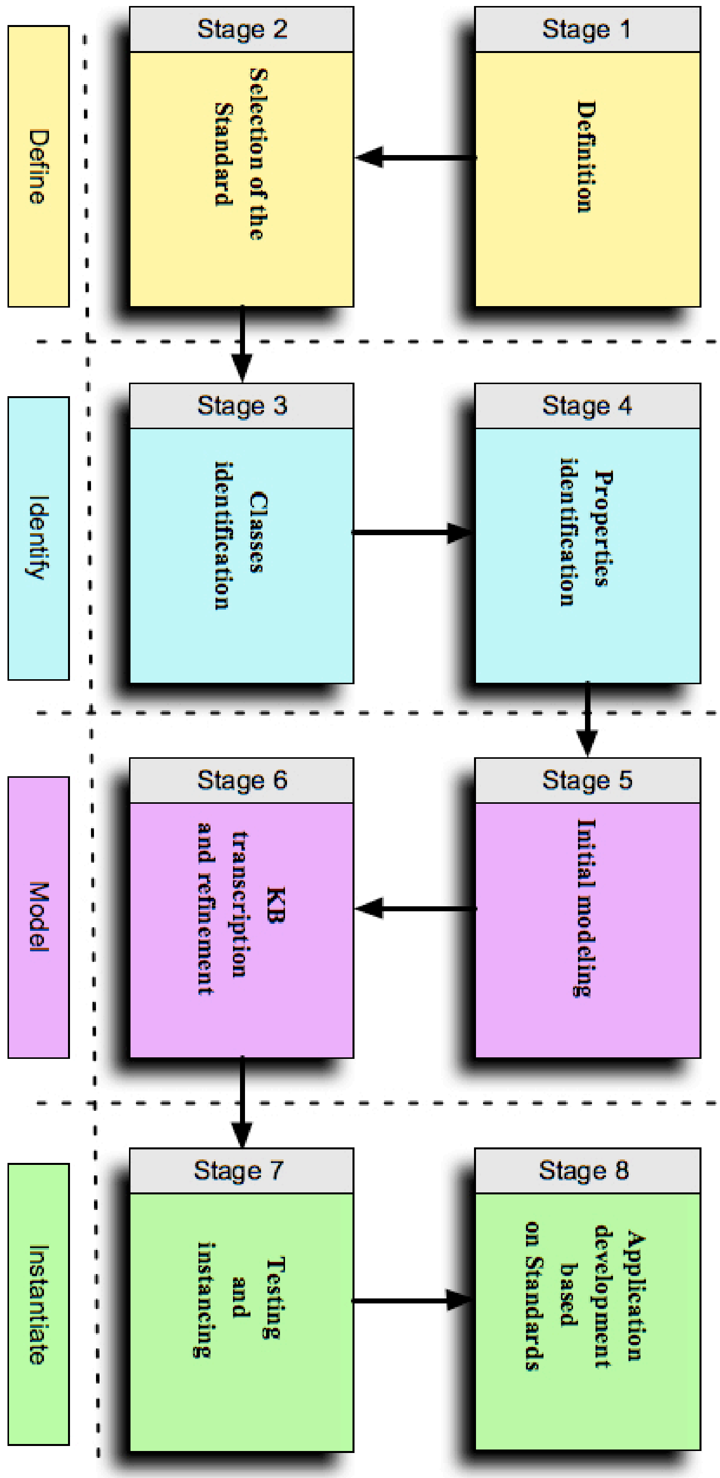


Figure 5.1: A methodology for KB domain design based on ES

Define:

- **Stage 1- Definition:**

  In this stage, an identification of the purpose and requirements of the domain is made. What is the purpose of the KB, the information that will be stored and the needed level of detail of such information.

- **Stage 2- Selection of the Standard**

  In this stage, there is a search of a standard that suits the defined needs. As a result of this selection, the chosen standard must be studied in detail, how it is constructed, what can be done in order to extend it, etc

Identify:

- **Stage 3- Class identification**

  In this stage, an identification of the possible classes that model the Domain is performed, Classes are categorized in a tree-like structure of the more general terms.

- **Stage 4- Property identification**

  At this stage, the characteristics that can me measured or determined by data types (string, Boolean values, integers, etc), are identified in each class, e.g. outside diameter, length, etc. Then we identify the characteristics that relate a class with other classes (relation types). In general data type characteristics are easily recognizable and obtained by simple interrogations (e.g. to a geometric model). Relation types are a little bit more difficult to find, because generally when talking about a geometric model whose characteristics will be obtained, those sets of elements are categorized as geometric primitives rather than functional objects. For the aforementioned case, solutions like the process of branding and matching presented by Posada, could be used [Pos05].

Model:

- **Stage 5- Initial modeling**

  In this stage, a subset of the domain is chosen in order to verify the complexity of the overall modeling and the real capabilities of the KB. Since the elaboration of a KB is an iterative process by nature, this small test must answer initial modeling needs.

- **Stage 6- KB transcription and refinement**

  Sometimes the initial modeling is enough for the KB to fulfill the design requirements in stage 1, however is highly recommended to perform a verification of the transcription, by using for example the capabilities of a reasoner to check the congruence of the KB (as it was discussed in chapter 3). Once the transcription is done, a refinement process takes place. In such stage any needed extension of the standard take place.

Instantiate:

- **Stage 7- Testing and instancing**

  In this step the test of the instances and the creation of an automatic or semi-automatic instancing mechanism (if needed) is performed. As a final step, some individuals conforming to the specification of the classes can be manually modeled using again an editor. Such process can be automated if needed if any API tools are available and the elements that must conform ca be interrogated e.g. modeling an industrial plant KB that has a 3D counterpart in CAD, the CAD API can be used to interrogate the elements and the editor API to semi-automatically "fill" the individuals in the KB. This step does not strictly fall into the modeling process, but it is the required step for the KB model to be of practical use.

- **Stage 8- Application development based on Standards**

  In this last step the Virtual Engineering Application using the Domain model is developed, this last stage comprises the actual usability of the Domain where the VEA advantages from Semantics via the enhancement obtained by having a better described and consensual Domain model.

5.3 Discussion - Challenges in using Standards as models for Domain Knowledge Bases

Using Engineering Standards n this scenario, could be very beneficial for the reasons explained earlier in this chapter, however, there are possible challenges in such an approach, being the following the most notorious:

- **The design of the Standard could be Functional-side balanced:** In some cases, the ES is functional oriented, a fact that leads to potential semantic loss, as the Standard doesn't include the needed parameters for a complete Domain modeling. The mentioned case is exemplified by the STEP 103-03 AP227 element "Valve" [ISO01]. In such element, the standard reveals only functional parameters (actuator_type, operation_mode, type), but omits about geometric parameters that are needed and also easy readable from a CAD model (diameter, length, etc). For those cases an extension of the class should be performed in order to obtain a complete KB. When the extension of the Domain is needed, it is advisable to double check if the parameter is a fundamental. Sometimes, the ES offers a way to obtain the needed parameter by interrogating neighbor elements (in the case of the Valve, the input and output pipes could be used for such purpose). If the parameter need is fundamental and the extension is unavoidable, it should be clearly specified as an "outside the standard feature" and it must follow the ES architecture, e.g. it should be part of the correct element and moreover derived from the correct parent class.

- **The Standard can disappear or being absorbed by other standard:** Maybe for a lack of use or because of administrative reasons, some standards disappear (e.g. the case of CAM-I AIS [CAM]), in such cases; the use of a KB based on such ES could be continued, however it would be advisable to migrate the KB to a new paradigm when available. In the case of absorption by other standard, the advice is to review the model in order to check its robustness.

- **The Standard falls short for the domain needs:** This indicates a possible immature ES, or an inappropriate election by the Domain designer. For both cases, it is advisable the reading and understanding of the Standard and an extensive review of the problem's characterization (domain requisites).

- **The Standard could be used to model different domain:** This feature is not a bad situation at all, it means that the same Domain modeling could be shared by different VEA without the need of further specifications. A situation like this, involves the possibility of using a VET that was not intended for a given task as such VET can be applied in the shared Domain.

Engineering Standards and VEA

The use of Engineering Standards as models for Domains is clearly beneficial for the Semantic enhancement of Virtual Engineering Applications through the Semantic enhancement of their Virtual Engineering Tools because of the fact that a consensual Domain gives the capability of more accurate real world-to-VEA mapping tools that ease the identification and pairing of real world elements with the virtual objects that belong to the VEA. As an example to this, we can mention a certain element that we know is clearly described by a Domain modeled after an Engineering Standard, if the VET "brands" such element as for example "Pipe", a VEA adhering such Standard will have for certain a way to represent a "Pipe", moreover in such VEA, the element will be called "Pipe" and not "Tube" or "Cylinder" amen of the fact that the element's characteristics (both physical and functional) will be represented as well. The Domain based Engineering Standard constitutes then the quintessential Knowledge Based model in terms of matching Virtual and Real world elements.

Supporting publications

This Chapter was supported by the following Papers and Journals in International conferences:

- Toro, C. ; Graña, M. ;Posada, J.; Sanín, C. .; Szczerbicki, E. ;Vaquero, J (2009). Domain Modeling based on engineering Standards. Submitted to KES 2009 - Knowledge-Based Intelligent Information and Engineering Systems, Springer, Heidelberg, Santiago, Chile

Remarks: This paper is directly obtained from the ideas expressed in this chapter, at the time of this thesis delivery, the paper is in the state Submitted for publication and awaiting the answer from the international committee of reviewers of KES 2009.

- Toro, C. ; Sanín, C. ;Vaquero, J.; Posada, J.; Szczerbicki, E. (2007). Knowledge Based Industrial Maintenance Using Portable Devices and Augmented Reality. Knowledge-Based Intelligent Information and Engineering Systems, (Apolloni, B., et al., Eds.), Vol. 4692, pp. 295–302. Springer, Heidelberg, Vietri Sul Mare, Italy.

  Remarks: This work addresses a Virtual Engineering Application intended to the enhancement of the industrial maintenance area by the use of semantics, the Domain modeling played an important role (Industrial Maintenance), the techniques discussed in this chapter were successfully applied in this VET. A more comprehensive description will be discussed in Chapter 10 of this Thesis.

- Posada, J. ; Toro, C. ; Wundrak, S. ; Stork, A.: Ontology Modelling of Industry Standards for Large Model Visualization and Design Review using Protégé. in Proceedings of the Eighth International Protégé Conference, Madrid, Spain, July 2005.

  Remarks: This short paper is the result of an invitation from the Protégé Group (Stanford University, USA) in where the use of an ontology modelling software is addressed in order to help in the development of a real world engineering solution that addressed the use of ontologies in the verification of steel based designs (known as steel detailing).

- Posada, J. ; Wundrak, S. ; Toro, C. ; Stork, A.: Semantically Controlled LMV Techniques for Plant Design Review, in Selected Readings in Computer Graphics 2005, by Fraunhofer IRB Verlag, 2004, ISSN 0948-3950 , ISBN 3-8167-6356-1 Darmstadt, Germany 2005.

  Remarks: This journal paper originally appeared in the 2004 ASME conference, and it was chosen to participate in the annual best paper award of the INI-Graphics NET in 2005, being shortlisted in the 8 best papers of the year from a total of 400 presented papers by an international committee of reviewers

- Posada, J. ; Wundrak, S. ; Toro, C.. ; Stork, A. : Using Ontologies and STEP standards for the semantic simplification of CAD models in different engineering domains in Journal of Applied Ontology, Volume 1, Number 3-4;2006, pp. 263-279

  Remarks: This journal paper introduced the use of engineering standards (e.g. STEP and CIS/2) for the ontology modelling of engineering based applications that later evolved to Virtual Engineering Applications.

Chapter 6

The Reflexive Ontologies: A computational methodology for the enhancement of the ontology query process

In previous Chapters, great attention has been given to ontologies as a result of their flexibility to model a domain and hence, to conceptualize the portion of reality to which such domain refers. Is not enough to have a good-modeled ontology fed with real world instances (individuals) from trustable sources of information. It is also of great importance to use the produced Knowledge Base (KB). Using the KB implies the capability to obtain information contained in the model within a reasonable time. Querying information stored in ontologies is typically performed via a programmatic approach using API calls, Reasoners and even the ontology editor GUI paradigms. However, any person, who has dealt with query processes, even within a non-semantics approach (e.g. Databases), knows that computationally speaking, some query processes can be very expensive in terms of time consumption. Nowadays, then, it is of the utmost importance to enhance ontologies with the capability of querying in a fast and trustable way while at the same time preserving their integrity extending their embedded information with new knowledge. In this section, we introduce the concept of Reflexive Ontologies (RO) as a technique that can be used to add self-contained queries to domain ontologies. The enhancement of having self-contained queries relies on:

- The speeding of the query process.
- The possibility of the ontology itself to add new queries on individuals with the correspondent answers to such queries (a feature that indeed adds new Knowledge of the domain).
- The self-containment of the enhanced Knowledge Structure in a single file; including the model, the relations between the elements belonging to it, the individuals (instances) and of course the queries over such individuals.

6.1 Background Concepts

Reflexivity

The concept of Reflexivity is used in multiple fields; in this thesis, we approach Reflexivity from the mathematics (logics) and the sociological points of view.

Mathematical concept of reflexivity

Reflexivity, in mathematics, refers to the logics based concept that every mathematical statement, which is known as a proposition, does implies itself. The former is written: $p \rightarrow p$ and usually is reads as “p implies p”. With sufficient information, mathematical logic can often categorize a proposition as true or false, although there are various exceptions (e.g., "This statement is false"). Any relation in mathematics is referred as a subset of a Cartesian product. For instance, a subset of $A \times B$, called a "binary relation from A to B", is a collection of ordered pairs (a, b) with first components extracted from set A and second components extracted from set B. In particular, a subset of $A \times A$ is called a "relation on ". For a binary relation R , one often writes (aRb) to mean that (a, b) is in R. The reflexivity property can be defined as a relation R on a set S . Such relation is reflexive provided xRx that exists for every $x \in S$. In our approach, a mathematical simile corresponds well to the self-containing nature of the reflexive ontology, as it will be explained later.

The mathematical simile will permit us in the future to express in a structured way lemmas and demonstrations useful for the formalization of the concept.

Sociological concept of reflexivity

The mathematical idea is not distant from definition used in sociology, where the concept of Reflexivity takes an epistemological flavor as it is used to identify the foundations of knowledge and the implications of any findings. Reflexivity [Wik07], in sociology, is the action of self-referencing, and addresses and affects the subject by the means of examining or acting on the object itself. Reflexivity is a bidirectional relationship between cause and effect. In such a case, the reflections of the object about itself are not independent of its status quo as a reflexive object. Moreover, any object or agent in a real-world social system possesses characteristics of reflexivity and self-enquiry. Thus, the object or agent being reflexive discovers or determines something about itself and abstracts out that aspect. It may reflect on beliefs, memories or knowledge, but this reflection does not produce that belief, memory or knowledge. Flanagan [Fla81] argues that reflective agents support the traditional roles played by the classical science: control, explanation and prediction. In order to help in the control, explanation and future prediction of domains, our approach uses the capacity of self-interrogation, and hence, the capacity of obtaining conclusions, which are elements characterized by the sociological point of view.

Autopoiesis

Autopoiesis literally means "self-creation or auto-creation". It expresses a fundamental dialogue between structure and function. The term was originally introduced by Maturana and Varela [MV80], and according to them, an Autopoietic system represents a network of processes or operations that define the system and make it distinguishable among others. Any Autopoietic system is able to create and destroy elements of the system itself in response to the environment perturbations. Although the system changes at a structural level, the network is invariant along the system's existence, holding the inner system integrity. The auto-production capacity of a system is Autopoietic and constitutes the basic property of living creatures as they are always determined by their structures; in other words, they are systems such that when an external factor affects them, the resulting effects depend solely on themselves, on their structure at that moment and not on the external factor itself. Living creatures are autonomous in the sense that they have an auto-referencing property as systems in continuous production of themselves. The most important thing for this theory is not the properties of the components of the system, but the processes and relationships between processes made via their components. Luhmann [Luh97] argues that Autopoiesis is not a limited property of biological or physical systems and he defines it as the "Universal Capacity of every system to produce self states well differenced between each other that are tied to the system own operations due to the self-organization capacity of systems". The structure and function relation of the Autopoietic definition allows us to refer to our reflexive ontology as an Autopoietic system that is in constant evolution in the sense that every new query being asked and stored will extend the knowledge base.

6.1.1 Reflexive ontologies, Formal Definition:

In our case, the de facto property of being reflexive addresses the property of an abstract structure of a knowledge base (in this case, an ontology and its instances) to "know about itself". When an abstract knowledge structure is able to maintain, in a persistent manner, every query performed on it, and store those queries as individuals of a class that extends the original ontology, it is said that such ontology is reflexive. Thus, we propose the following definition for a Reflexive Ontology:

> *"A Reflexive Ontology (RO) is a description of the concepts, and the relations of such concepts in a specific domain, enhanced by an explicit self contained set of queries over the instances."* [TSSP08]

Considering that any abstract knowledge structure of this kind is essentially a set of structured contents and relationships. The mathematical concept of a set and its properties can be applied to the knowledge structure for its formalization and handling.

Properties Of The Reflexive Ontologies

A RO is, basically, an ontology that has been extended with the concept of reflexivity. This can be seen in Figure 6.1, where C_i represents a class with r_j relations, I_k characterizes an instance of C_i (right side of the image), and Q_p is a query represented as an extension of the base ontology (left side of the image). In order to be compliant with the RO concept, an extension of a base ontology must fulfil the next set of properties:

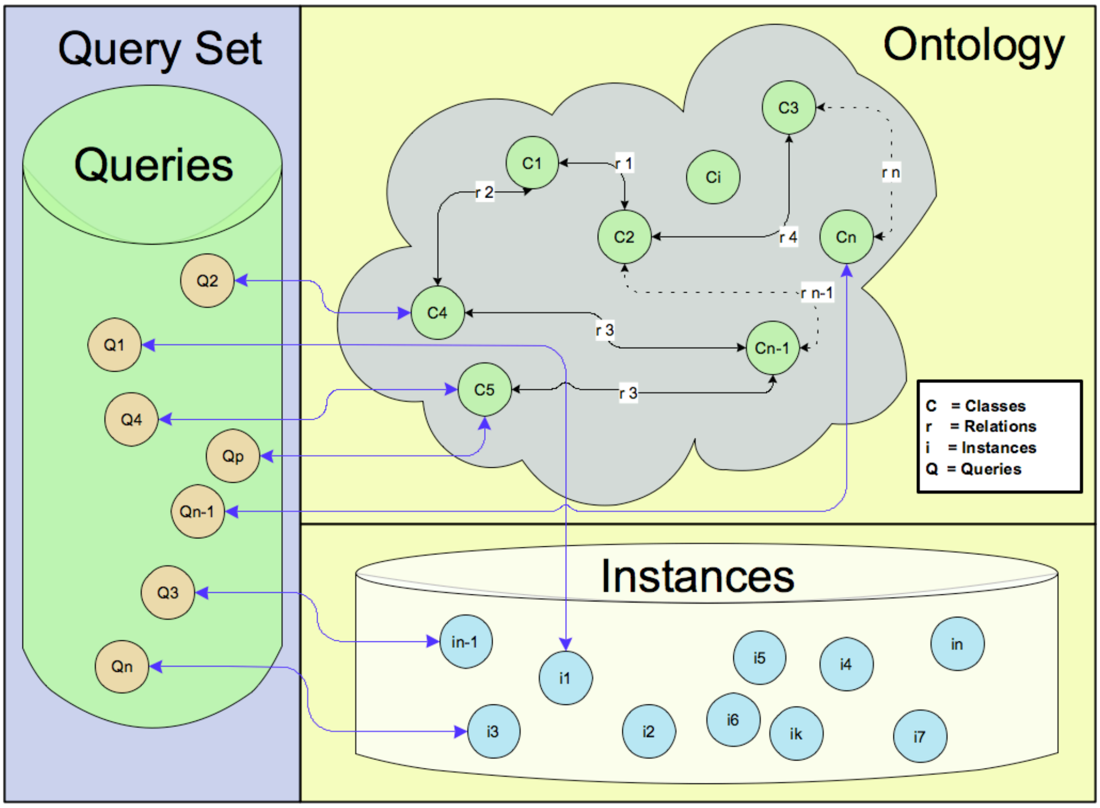


Figure 6.1: Simple structure of a RO, adapted from [TSSP08]

- **Property 1 – Query retrieval**

  It refers to the RO faculty of storing every query performed in order to return it, when re-queried. The query storage happens at two different levels: it could be in its atomized (simple query) or complex form (that is, containing a boolean operator between atoms). Queries can be referred to the structure of the ontology (data type) or to the instances (value type).

- **Property 2 – Integrity update**

This property refers to the RO faculty of updating structural changes in the query retrieval system. In other words, when a new individual is added or removed from the ontology, the query system actualizes the queries that contain such individual. This property is similar to the database integrity property in a traditional Data Base Management System (DBMS).

- **Property 3 – Autopoietic behavior**

  This property refers to the Autopoietic capacity of self-creation or auto-creation derived from the fact that for every new query launched, the knowledge structure will grow. The quality of the knowledge embedded in the system is increased as the system is in a constant auto-building process that can reflect the auto production capacity of an Autopoietic system. Moreover, it is important to mention that this property allows the RO to store the history of the queries while, at the same time; it provides the schema with a repeat ability and integrity mechanism.

- **Property 4 – Support for logical operators**

  This property provides the RO with the mechanisms of set handling from the logistic point of view; that is, the inclusion of AND, NOT, and OR logical operators.

For example, lets us suppose that we have a simple ontology (O) that describes the genealogy of the Jones family with some properties (p) like for example the age of every individual ($p.age$). After the instantiation with some elements, a query with two elements can be launched: Elements whose age is greater than 5 expressed formally as $Q_1 = (\forall p \in O \mid p.age \geq 5)$ and elements that belong to the Jones family : $Q_2 = (\forall p \in O \mid p.family = "jones")$, a query like "retrieve the members of the Jones family whose age is greater than 5 years" on the ontology O will give as a result a set of individuals that match the abstract properties $p.age > 5$ and $p.family = "jones"$. So the final answer to the query called indistinctly reflexivity, will be: $Q_t = Q_1 \cap Q$, where Q_1 and Q_2 defined as above, with p being the predicate and O the ontology.

A question like this will be instantiated in the ontology in its atomized form (Q_1, Q_2) and, of course, in the complex form Q_t. Therefore, next time a question is asked, the system will search trough the reflexivity instances in order to retrieve the answer (being linear the worse case computing time). If the exact question or (as it will be seen later) a similar question cannot be found (similarity), a traditional ontology interrogation process is executed.

- **Property 5– self-reasoning over the query set**

  This property refers to the RO capacity of performing some logic operations over the query system in one of the following scenarios:

  – To discover patterns of queries.

  – To suggest the need of ontology refinement, as some elements are more queried than others; meaning that, probably, they should be taken into special consideration as the queries performed are being focused on specific sections of the query system that cause a possible asymmetric behavior.

  – To discover new non-explicit relationships, meaning that some queries could be different, but their sets of solutions are the same. Hence, this could imply a possible undercover relationship between sets of queries (a possible serendipity).

6.1.2 Advantages of the Reflexive Ontologies

The enhancement of having self-contained queries relies in the following main aspects:

- **Speed:** (uhe speeding up of the query process) As every query is handled and stored in either its atomized form and its complex form, including the logical operators, the interrogation of the ontology is in the worst case time linear if the query exist (it has been previously stored). If the query has not been asked before, then a typical ontology interrogation via an API takes place and when the new set of answers is retrieved it is added to the reflexivity class. This advantage is related to the RO Property 1. The actual state of the technology in API based query systems for ontologies is far away from optimal in the efficiency aspect (at least when compared to Database systems). With few classes and instances, even simple queries could take an enormous amount of time. By using RO, the efficiency in speed of the query process is enhanced even in complex ontologies.

- **Incremental nature:** the possibility of the ontology itself to add new queries on individuals with the correspondent answers to such queries: This is in fact a feature that adds knowledge about the domain as the more questions are asked, the more knowledge can be stored in the ontology. This advantage is related to the RO Property 5. The questions and answers are in fact a guideline to infer "things" about the ontology.

- **Self-containment of the Knowledge Structure in a single file:** This feature includes the storage of the model, the relations between the elements of the model, the individuals (instances) and queries over such individuals. This advantage relies on fulfilment of the functional purpose of a RO.

6.2 Implementation of the Reflexive Ontologies concept

We define in Figure 6.2 schema for the implementation of the RO concept.

This schema is divided in three layers as follows:

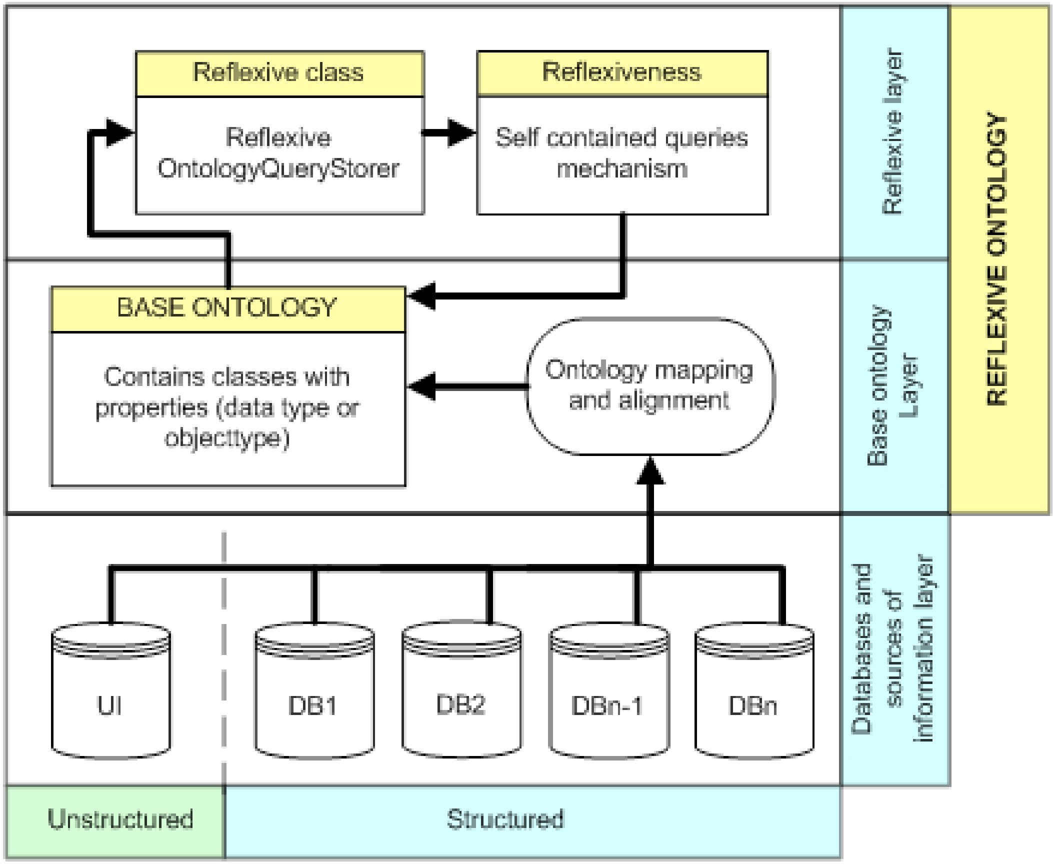


Figure 6.2: Reflexive Ontology architecture

The databases and sources of information layer: contains the real world elements that "feed" the ontology, the information contained can be structured or unstructured and by means of a process of ontology alignment and mapping they become instances of the base ontology (traditional ontology approach). A typical ontology has the knowledge definition represented as classes and properties, being the last ones of two possible types: (*i*) object type, mapping a class into a class and (*ii*) data type, that map a class into a characteristic represented by a traditional computer type like a string, and integer, etc. The next layer contains two modules, the first one is the extension which adds a new class to the base ontology with the needed schema for the reflexivity; in our implementation, we call this the "ReflexiveOntologyQueryStorer class" (see Figure 6.3).

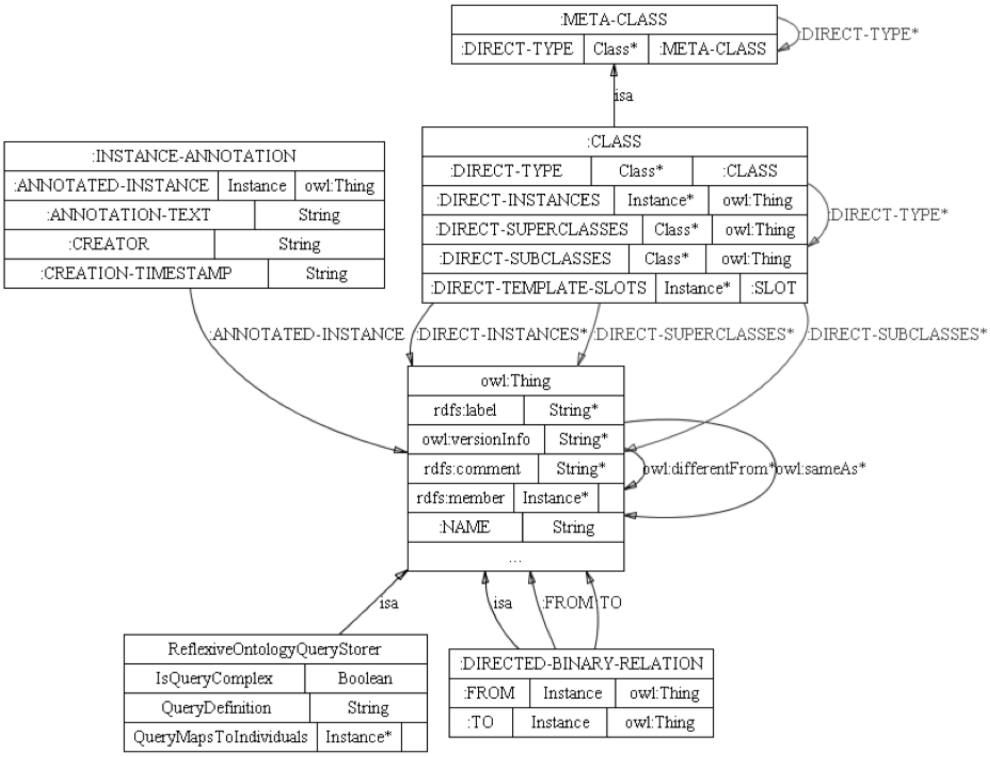


Figure 6.3: Reflexive Ontology Query Storer schema

The extension hangs from the OWL:Thing super class and it has the following OWL properties (see Table 6.1).

Property	Type	Comment
isQueryComplex	Data	Boolean
QueryDefinition	Data	String
QueryMapsToIndividuals	Object	Collection of individuals

Table 6.1: Reflexive class properties

The last module inside our schema is the reflexiveness itself and it provides the ontology (programmatically) with a mechanism to perform queries and some logic on the queries that allows the handling of the reflexiveness.

6.3 Reflexive Ontologies in the context of Virtual Engineering Applications

Reflexive Ontologies can be very useful in the context of Virtual Engineering Applications when enhancing such applications with Semantics because of the fact that many queries are needed in the interrogation of the underlying Knowledge Bases. When we consider the fact that also the queries are performed not only within an intensive rate, but those queries are in many occasions very complex, the use of RO is a great advantage. For the Semantic enhancement of VEA, the RO play a key role also in the fact that every new query indeed increases the explicit semantic information contained in the KB, providing not only speed, but reliability.

6.4 Discussion - RO open challenges

In this Chapter we introduced the concept of Reflexive Ontologies (RO). The presented idea can be applied to an existing KB in order to improve its query capabilities. We presented the RO properties and some benefits of having self contained queries. Among the future work some additional features must be developed in order to offer a more user friendly environment. Such features include: the implementation of an automatic graphical paradigm which will help in the transformation of an ontology into a RO and the implementation of a better interface for the enhanced query engine with its corresponding extension of the query 'logic' elements into a more human like language. The aforementioned implementation could be added to traditional ontology modeling systems as Stanford's Protege.

Supporting publications

This Chapter was supported by the following Papers and Journals in International conferences:

- Toro, C. ; Sanin, C. ; Szczerbicki, E. ; Posada, J. : Reflexive Ontologies: Enhancing Ontologies with self-contained queries in Cybernetics and Systems: An International Journal, 39: 1–19, ISSN: 0196-9722 print, 1087-6553 online, USA 2008.

  Remarks: This journal paper introduced the concept of the Reflexive Ontologies, their basic properties and a simple application example for the SOEKS architecture presented by Sanin in his doctoral work.

- Cobos Y. ; Toro, C. ;. Sarasua, C. ; Vaquero, J. et al. : An architecture for fast semantic retrieval in the film heritage domain. In 6th International Workshop on Content-Based Multimedia Indexing (CBMI), pages 272–279. IEEE, June 2008.

Remarks:In this paper, we propose an architecture that takes advantage of the concept of Reflexive Ontologies (RO) in order to achieve timely semantic retrieval. The proposed architecture is illustrated by a case study in the Film Heritage domain, showing a performance improvement and providing ground for further research and discussion. The concept presented in this paper is easy extensible to VEA as it will be shown later in this thesis.

Chapter 7

An general architecture for the Semantic Enhancement of VEA

7.1 Introduction

Virtual Engineering (VE) is defined as the integration of geometric models and related engineering tools (such as analysis, simulation, optimization and decision-making, etc), within a computerized environment that facilitates multidisciplinary and collaborative product development [JMLB06]. According to McCorkle [MBS04] "a key aim of Virtual Engineering is to engage the human capacity for evaluation of complex systems and situations". In this thesis, we consider a Virtual Engineering Application (VEA), any engineering-focused software able to perform the aforementioned task. Also, a Virtual Engineering Tool (VET) as any of the components of a VEA that offer a user or machine interaction capability. In this chapter, we will introduce an architecture for the semantic enhancement of VEA through their embedded VET. Our architecture uses concepts already presented in previous chapters and it will serve as a model for the supporting applications discussed in the technical contributions sections.

7.2 Virtual Engineering Tool definition

We model a VEA in terms of the different components depicted in Figure 7.1. The question that we want to address is: How can Semantic Technologies work with the Virtual Engineering Application components? To find answers to such a question, we need first to describe the VEA elements: (*i*) characteristics, (*ii*) requirements and (*iii*) interaction.

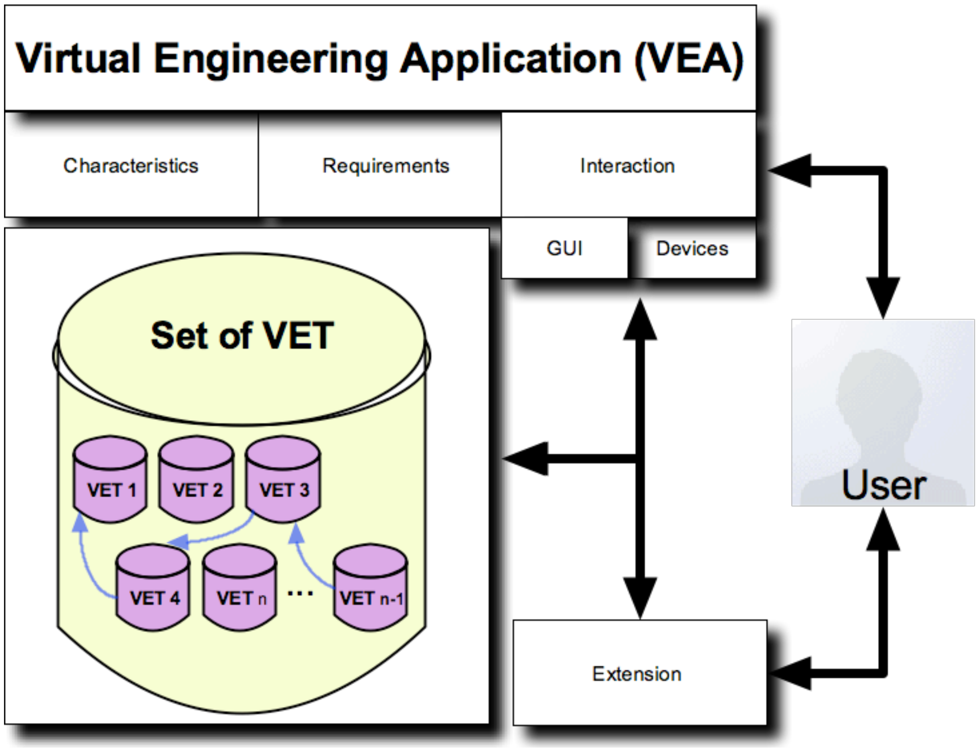


Figure 7.1: Components of a VET

7.2.1 Set of Characteristics:

Usually the set of characteristics describes the expected benefits of the VEA in terms of capabilities, features, compatibility, accepted formats for input/output interactions and the overall benefits of using the VEA. The set of characteristics defines a general overview of the competences of the VEA, and the comparison of such competences against the requirements, is in most of the cases the main reason to use the VEA. However, characteristics are usually presented with a marketing focus. For the aforementioned reason, it is wise to perform an evaluation of the VEA before the making the purchase decision. Aware of such a scenario, some VEA providers offer the possibility to test before buying. Figure 7.2, presents a typical set of characteristics extracted from an advertising pamphlet

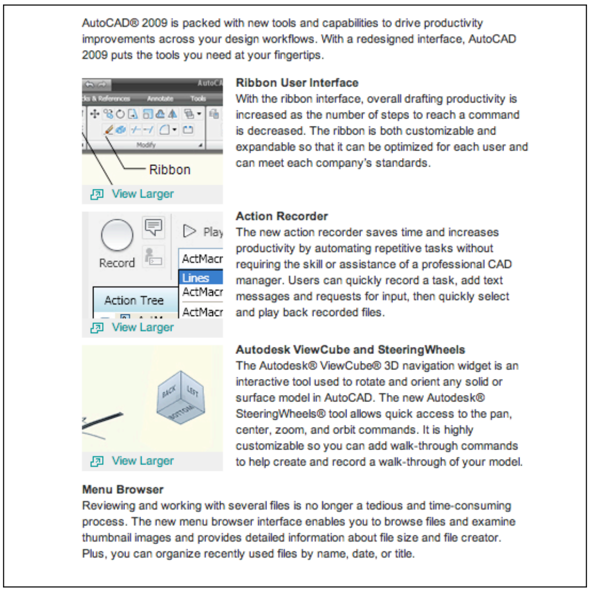


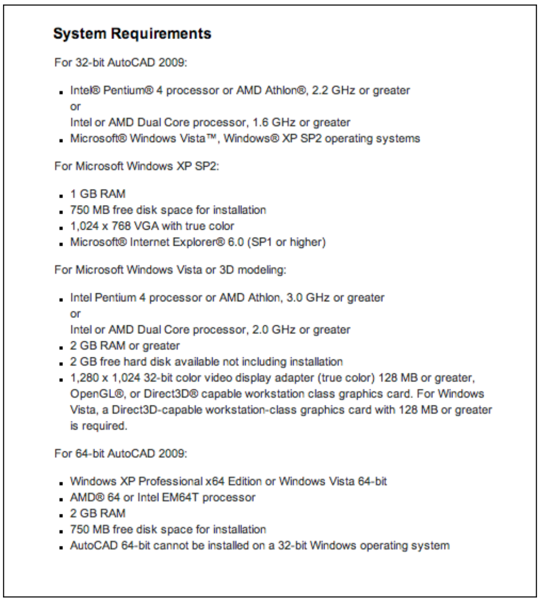
Figure 7.2: VEA characterization and advertising [Aut]

7.2.2 Set of Requirements:

The set of requisites are the minimum requirements by the computer system where the VEA will be used. Usually the requisites are given as the specification of a minimal configuration. The requirements are a good way to balance the VEA needs as they relate to hardware and software requisites. Figure 7.3 depicts a typical requisite sheet excerpt.

7.2.3 Set of Interaction paradigms:

The set of interaction paradigms, include different GUI's, and input/output device characteristics and requirements e.g. a mouse, an output screen, etc. The interaction paradigms are the places where the user executes the input and output interface actions. Some example interaction paradigms possible are: (*i*) the use of multiple views in the same display, (*ii*) the capability to accept not traditional GUI interactions e.g. voice commands, and (*iii*) the use of alternative input devices. In Friedell's work [FC], input devices and display devices for the CAD Domain are



System Requirements

For 32-bit AutoCAD 2009:

- Intel® Pentium® 4 processor or AMD Athlon®, 2.2 GHz or greater
  or
  Intel or AMD Dual Core processor, 1.6 GHz or greater
- Microsoft® Windows Vista™, Windows® XP SP2 operating systems

For Microsoft Windows XP SP2:

- 1 GB RAM
- 750 MB free disk space for installation
- 1,024 x 768 VGA with true color
- Microsoft® Internet Explorer® 6.0 (SP1 or higher)

For Microsoft Windows Vista or 3D modeling:

- Intel Pentium 4 processor or AMD Athlon, 3.0 GHz or greater
  or
  Intel or AMD Dual Core processor, 2.0 GHz or greater
- 2 GB RAM or greater
- 2 GB free hard disk available not including installation
- 1,280 x 1,024 32-bit color video display adapter (true color) 128 MB or greater, OpenGL®, or Direct3D® capable workstation class graphics card. For Windows Vista, a Direct3D-capable workstation-class graphics card with 128 MB or greater is required.

For 64-bit AutoCAD 2009:

- Windows XP Professional x64 Edition or Windows Vista 64-bit
- AMD® 64 or Intel EM64T processor
- 2 GB RAM
- 750 MB free disk space for installation
- AutoCAD 64-bit cannot be installed on a 32-bit Windows operating system

Figure 7.3: VEA System Requirements [Aut]

presented. Friedell's focuses on the question of what is the fundamental nature of the interaction between a human and a computer in the design process supported by the CAD system, and presents a discussion on the best approaches.

7.2.4 Set of Virtual Engineering Tools:

The set of VET provided is arguably the most important argument to use the VEA. We base our approach to Semantically enhancing a VEA in the fact that a given VET belonging to the set of Virtual Engineering Tools can be Semantically enhanced by means of considering a User-Domain-Intention approach. Figure 7.4 depicts an excerpt that characterizes a VET from a VEA intended for Steel Detailing.

The set of Virtual Engineering Tools can be composed of complementary tools that are used in conjunction with other tool or isolated tools that can be used upon necessity. When we introduced the GOMS methodology for User Modeling, we described the Operators, the Set of available VET are equivalent to such Operators in the sense that they represent the means for accomplishing a oal or sub-goal.

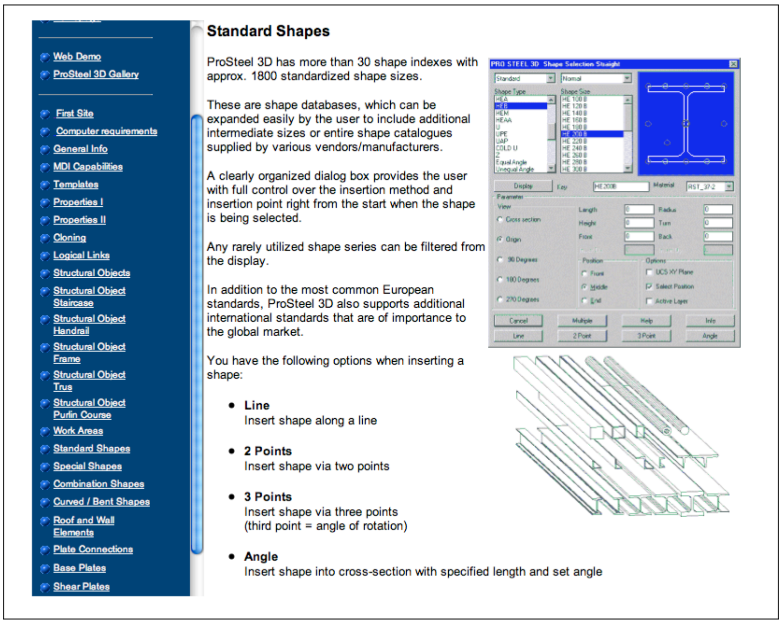


Figure 7.4: VEA Tools (excerpt from Prosteel 3D manual) [Kiw]

7.2.5 Extension capabilities:

We propose our VET enhancements by means of using the programming capabilities of the VEA. Such programming capabilities allow the extension of the functionality of the VEA and its VET. Generally, the capability to extend a VEA is provided via an Application Programming Interface (API) or by means of scripting languages. As a general rule, the more open the API, the easier it is to enhance it. The most common computer languages used for API interfaces are C/C++, Java and Visual Basic. The extension capabilities of the VEA, also allow the ease of repetitive actions. Figure 7.5 depicts a sample code excerpt from the AutoCAD C++ extension API (called ObjectARX).

```
// create a new line
AcDbObjectPointer<AcDbLine> line = new AcDbLine();

// set the properties for it
line->setStartPoint(AcGePoint3d(10,10,0));
line->setEndPoint(AcGePoint3d(20,30,0));
```

Figure 7.5: Extending a VEA using an API

Undoubtedly the two ways of User-VEA interaction are (i) the direct manipulation of the component Virtual Engineering Tools that belong to the VEA via the

GUI and Devices and (*ii*) the use of the extension capabilities of such VEA. As we are interested in semi-automatic features, our approach focuses on the second interaction method, being the Extension Capabilities of the VEA a fundamental stepping-stone for their Semantic enhancement. Of course we do not categorically say that a VEA without extension capabilities cannot be Semantically enhanced. However such scenario is not considered in our approach and it would be a possible extension to our work as a future development.

7.3 Methodology for the Semantic Enhancement of VEA

In this section we will present a methodology for the Semantic Enhancement of Virtual Engineering Applications. Our methodology uses the recommendations previously introduced in the methodological chapters of this thesis e.g. User Modeling, Domain Modeling and Interaction Modeling. We consider a simple, yet efficient procedure to enhance any API extensible VEA by dividing our methodology in three layers: (*i*) the Conceptualization layer, (*ii*) the Semantic Abstraction layer and (*iii*) the Virtual Engineering Application layer. In our methodology, different modules and sub-modules compose the layers following a bottom-up approach where the sub-modules are bottom concepts to be modeled and the layers are the top level of conceptualization. Our approach makes strong use of ontology modeling for the diverse reasons explained in previous chapters, the most important being the benefits related to the subsumption relations that allow us to take advantage of implicit Semantics. Sub-modules are able to share information between each other when they belong to the same module and the connections between the different layers are made through module-to-module bidirectional connections. In the following sections we give a detailed description of the different elements of our methodology. A general view of the architecture is depicted in Figure 7.6.

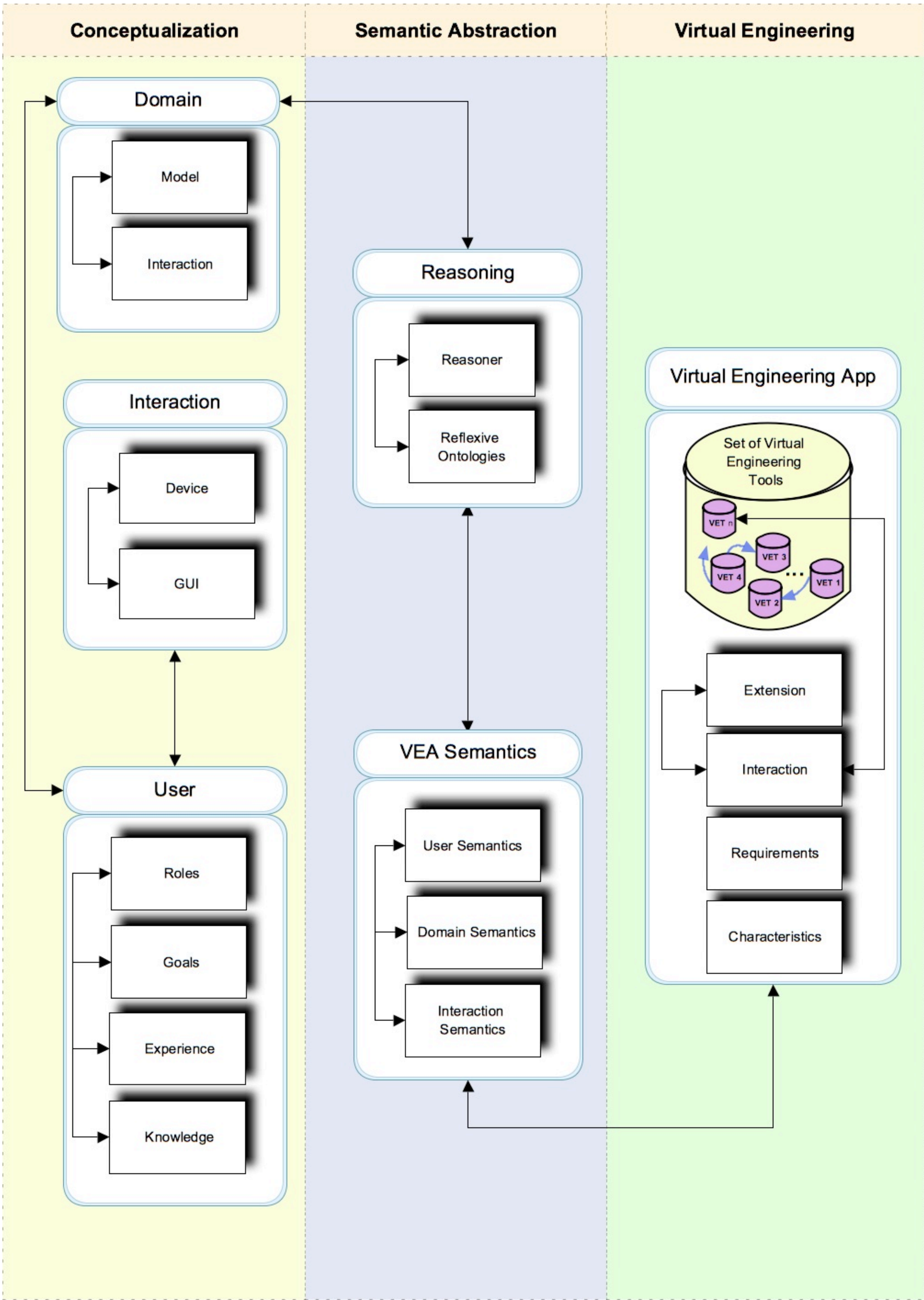


Figure 7.6: Architecture for the Semantic Enhancement of VEA

7.3.1 The Conceptualization layer

The general idea of this layer is to provide a conceptual model of the User goals, his expertise, how the user interacts with the VEA and the scenario where the interaction is taking place. The conceptualization layer comprises three modules: The domain Module, the User Module and in the Interaction Module. The importance of conceptualizing the characteristics of the User was discussed in Chapter 4; the Domain recommendations were commented in Chapter 5. For such reason, in this part we will briefly describe the overall characteristics of such modules. To help the reader to follow our explanations, Figure 7.7 depicts only the Conceptualization Layer components.

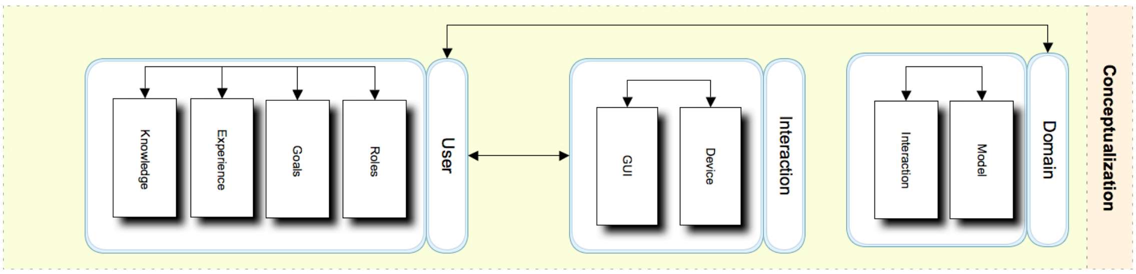


Figure 7.7: The Conceptualization Layer

7.3.1.1 The User Module

The user is modeled from a Goals-Experience and Knowledge-Role points of view, following the recommendations presented in the User Modeling chapter. Each consideration is modeled as a sub-module featuring Knowledge Bases or each one using an ontology approach. The component sub-modules are:

- **Role sub-module:** Defining the different Roles (that the user is able to perform following the recommendation presented in chapter 4). The user role sub-module can be modeled as a sub-ontology or as a full ontology.

- **Goals sub-module:** This sub-module considers high and low level goals, being the over-all describing what a user wants to accomplish (or is needed to accomplish). User goals are lists of requirements on the domain.

- **Experience sub-module:** This sub-module is very important in order to determine recommendations based on previous users experiences. This User Experience sub-module has been modeled in various presented approaches using the SOEKS and it will play an outstanding role in Chapter 10 where the User Experience modeling is the base for the Semantic enhancement of an Industrial Maintenance Domain-based VET.

- **Knowledge sub-module:** This sub-module is equivalent to the User Type definition described in Chapter 4. It contains the Knowledge about the Domain that defines the basic differences between Users. In other words, what is known in other methodologies as the User Type. We intentionally make User Knowledge and the User Type synonymous because our approach is oriented towards the functional aspect of the User. If the User "has a particular Knowledge of the Domain" which means that such a User can be categorized in the Domain. The User Role sub-module described before will help tighten such User Type categorization by extending the Knowledge model in the sub-module. Another tightening factor is the Experience, that enriches the Knowledge sub-module with previous experiences that are not necessary from the User being modeled, but from sets of Users in similar Domain restricted situations.

7.3.1.2 The interaction module

This module is in charge of conceptualization of the user-VEA interface handling. Two sub-modules compose it, the device sub-module that is a list of hardware interaction features and the GUI that contains Graphical User Interface paradigms (software). The main function of the interaction module is to serve as interface between the conceptualized model of the interaction and the VET, which is where the interaction finally takes place. The different action-reaction situations between the user (programmer or final user) with the rest of the model and the VEA through its VET are modeled from a User Perspective in terms of usability and applicability.

- **Device sub-module:** As already mentioned previously, this sub-module handles the modeling of different hardware that is capable of offer User-VET interactions. Not only typical interaction devices are considered (e.g. screen, mice, etc.) but also the device as a whole is modeled here (e.g. PDA, sub-laptop, etc).

- **GUI sub-module:** This sub-module contains the modeling of the GUI paradigms that allow a better recommendation of a VET from the set of VEA, or an enhancement to the VEA GUI itself by means of enhancements of its embedded VET GUI. The GUI sub-module is treated in more detail in Chapter 8, where we present a methodology for the Semantic Enhancement of GUI.

7.3.1.3 The Domain module:

This module was discussed in depth in Chapter 5, where we used Engineering Standards as the base for our Domain Modeling. As can be seen in Figure 7.7 this module has two sub-modules, the Domain Model sub-module and the Domain Interaction sub-module. The Domain module relates to the remaining of the architecture trough the User Module. The reason why we do this is because in our approach we consider a User Oriented concept for the VEA interaction.

- **Domain model sub-module:** In this sub-module, an identification of the purpose and requirements of the Domain is made. Such identification and purpose tasks are modeled based on a suitable Engineering Standard that complies with such requirements.

- **Domain interaction sub-module:** This sub-module contains the model of the different interactions with the Domain. Such interactions are important for the User Roles correct determination and use.

7.3.2 The Semantic Abstraction Layer

The general idea of this layer is to use all the models gathered at the conceptualization layer and performs a reasoning process over them in order to enhance the VEA using the knowledge obtained directly, or by means of semantic reasoning from the modeling of the User, the Domain and the Interaction. Figure 7.8 depicts the Semantic Abstraction Layer components.

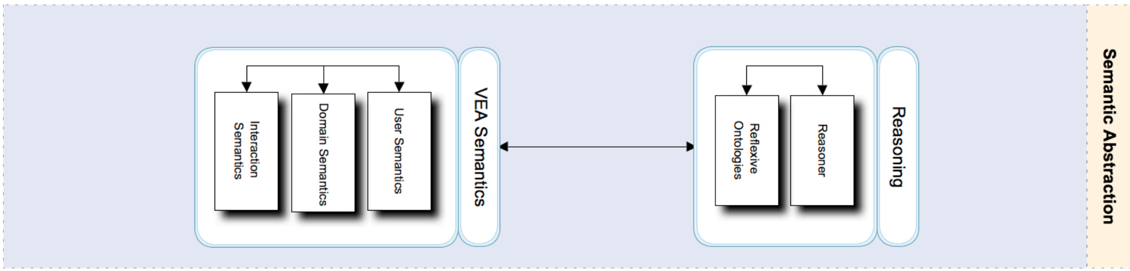


Figure 7.8: The Semantic Abstraction Layer

The Semantic Abstraction module is the place where the Semantic Enhancement takes place. This layer has two modules: The Reasoning Module and the VEA Semantics module.

7.3.2.1 The Reasoning module

The Reasoning module is in charge of performing the Semantic Reasoning of the gathered Knowledge from the Conceptualization Layer and also from the VEA semantics sub-module. The reasoning process is performed by a third party reasoner tool handled by the reasoner sub-module and a query engine (implemented using the Reflexive ontologies) which is needed to maintain the Knowledge reasoning in a sort of cyclic status. As can bee seen the Knowledge is gathered also from the Virtual Engineering Layer through the VEA semantics sub-module in order to feed such Knowledge from the VEA being enhanced.

- **Reasoner sub-module:** As explained in the introduction to this thesis, reasoners allow the discovery of non-direct patterns (e.g subsumptions), in other words, this module is in charge of the indirect queries and their handling. The reasoner module can be implemented using the ontology editor API if the reasoning is not complex, but it is advisable to use more complex tools such as those already mentioned in the introduction (e.g. Pellet, FaCT++) as they cover more complex semantic scenarios.

- **Reflexive ontologies sub-module:** This module follows the recommendations discussed in Chapter 6, where we presented the Reflexive Ontologies

(RO) concept as a way to perform better and faster query processes to the KB. The generation of the RO idea was directly related to the fact that we needed a fast response tactic in order to maintain the cyclic nature of our architecture. As can be seen, it is quite important to have a good flow of information between all the modules and sub-modules of our architecture and since all the KBs are modeled as ontologies, the most natural approach in order to enhance the massive use of queries over such KB was to develop a technique that provides us with a mechanism that save time by accessing the KB only when needed. The RO concept was introduced in [TSSP08] and the concept was also used by Cobos et.al in [CTS+08].

7.3.2.2 The VEA Semantics module:

This module serves as a logical link between the conceptual modeling and the real world actions in the VEA. The module is composed by three sub-modules each one capable of handling the semantic information reasoned in the Reasoner module and the real world feeds provided by the VEA. The three components handle the three different aspects we consider in our Semantic Enhancement: User, Domain and Interaction.

7.3.3 The Virtual Engineering layer

This layer contains the Virtual Engineering Application as described earlier in this chapter. Figure 7.9 depicts the Virtual Engineering Layer components.

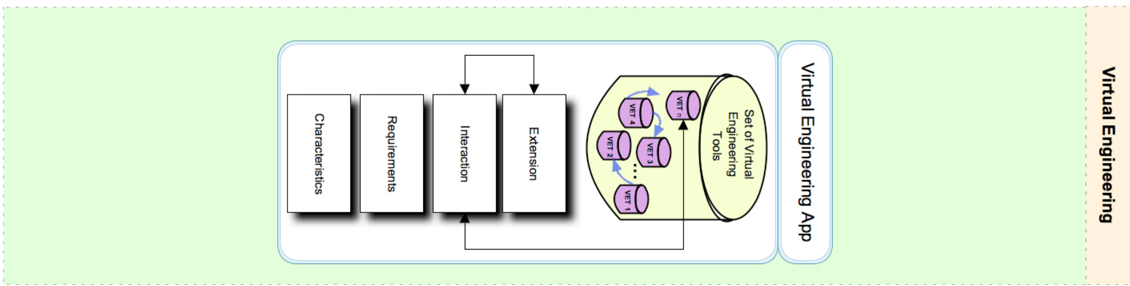


Figure 7.9: The Virtual Engineering Layer

7.3.3.1 The Virtual Engineering Application module:

This module is divided in four sub-modules, those being the Extension and the Interaction sub-modules in a cyclic schema that allows enhancements from the programmatic point of view provided by the Extension sub-module, reachable by parts of the interaction sub-module whose main function is to communicate the interactions to the VET closing the logic chain between the first conceptual layer and the VEA. Interactions are not only considered at a VET level, however, in most cases, the interactions happen here and not in a VEA. The above comment guides us to interesting conclusions that will be discussed in Chapter 11. As stated before, this layer is composed by three sub-modules: (*i*) The Extension, (*ii*) The interaction, (*iii*) the requirements and (*iv*) the Characteristics.

7.4 Discussion - Architecture

A novel modular architecture has been proposed which allows the implementation of the different conceptual steps in the methodological part as connected modules. Each module helps in having a more explicit representation of the semantics inherent to the model, but also to the user and the domain. Thus, the methodology starts from a Conceptualization Layer and passes the gathered Knowledge about the User, the Domain and the requirements to a Semantic abstraction Layer that performs the Reasoning processes that reflect the Semantic enhancement in the VEA. This architecture will serve as the base for the Technical Contributions presented in the following Chapters, proving its value as a generic and extensible way to semantically enhance a VEA through the enhancement of its VET. In each technical contribution Chapter, we will present how this architecture was implemented.

Supporting publications

This Chapter was supported by the following Papers and Journals in International conferences:

- Toro, C. / Graña, M. / Posada, J / Sanin, C. / Szczerbicki, E. :An architecture for the Semantic Enhancement of Virtual Engineering Applications accepted for publication in the

Springer book Intelligent Systems for Knowledge Management, Studies in Computational, series, Springer - Germany, 2009

Remarks: This book chapter is extracted from the concepts presented in this section of the Thesis in a summarized form intended for dissemination of the concepts.

Chapter 8

Contributions to the Semantic driven generation of Graphical User Interfaces for CAD

This Chapter, presents a technique which uses the advantages of Semantic modeling technologies for the automatic generation of Graphic User Interfaces taking into account three main features: (*i*) the user and his needs, (*ii*) the stage of the PLC in which he is interacting with the CAD and (*iii*) the user intentions. This technique was implemented within the scope of an applied research project with a Steel Retailer (structural design) partner.

8.1 Description of the technical problem and challenges identified

In modern industry, the use of CAD applications is of critical importance for the efficient handling of 2D and 3D models present in most stages of the PLM. The ease of geometric data sharing, the enhancement model comprehension and the evolution of computerized tools, both in hardware and software, are facts that are not be underestimated by an active research field that focuses on the enhancement of CAD based technologies. Doubtless, CADs are the most widespread supporting tool for engineering and arguably the most easily recognizable VET. However, it can be said that the complexity of interaction as seen from the users perspective is in direct relation to the amount of resources provided by the CAD software. The User-CAD Interaction paradigm is defined by the manipulation of the model via components of the Graphic User Interface (GUI), e.g. Drop down menus, pop-up and command windows, buttons and keyboard combinations that present, among others, the following challenges:

Challenge 1: The same functionality can be found in different menus or even different GUI paradigms

Any command can be accessed in many different ways, so the user is faced with a redundant set of choices (see Figure 8.1). Sometimes the different ways to interpret the GUI metaphor renders the CAD working process highly inefficient.

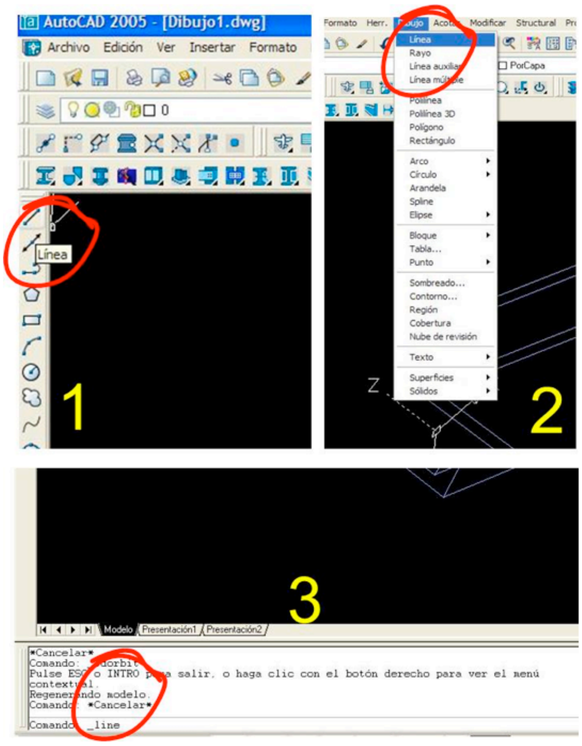


Figure 8.1: Accessing a simple CAD command, 1. Through a button, 2. Drop Down menu, 3. Command line

This issue has causes originating at the very beginning of the development of the CAD technology. At that time, the commands were typed in a console that has been kept for backward compatibility while the systems enhanced their graphical capabilities. Although highly trained and expert designers use combinations of command line inputs, buttons and drop menus, the reduction in the interface redundancy would increase the productivity of the CAD software use.

Challenge 2: Finding a needed functionality is not always an easy task

Usually a CAD software offers thumbnail buttons to give access to all the available user commands in the same window. This issue is easy to detect in general purpose CAD's like AutoCAD and MicroStation because of their general nature. It must be noted also that in high end CAD programs, like CATIA or Pro/E, finding of a certain functionality may be a difficult task in this case due to the degree of specialization of the program. In these situations, the user is forced to navigate through several menus while searching for a specific tool, slowing down the design process hence increasing the cost of the design stage. Finding the needed tool could be difficult for a novel user learning the operational issues of a CAD with a direct consequence on the learning curve. Another problem is the reduction of the available working area as can be seen in Figure8.2.

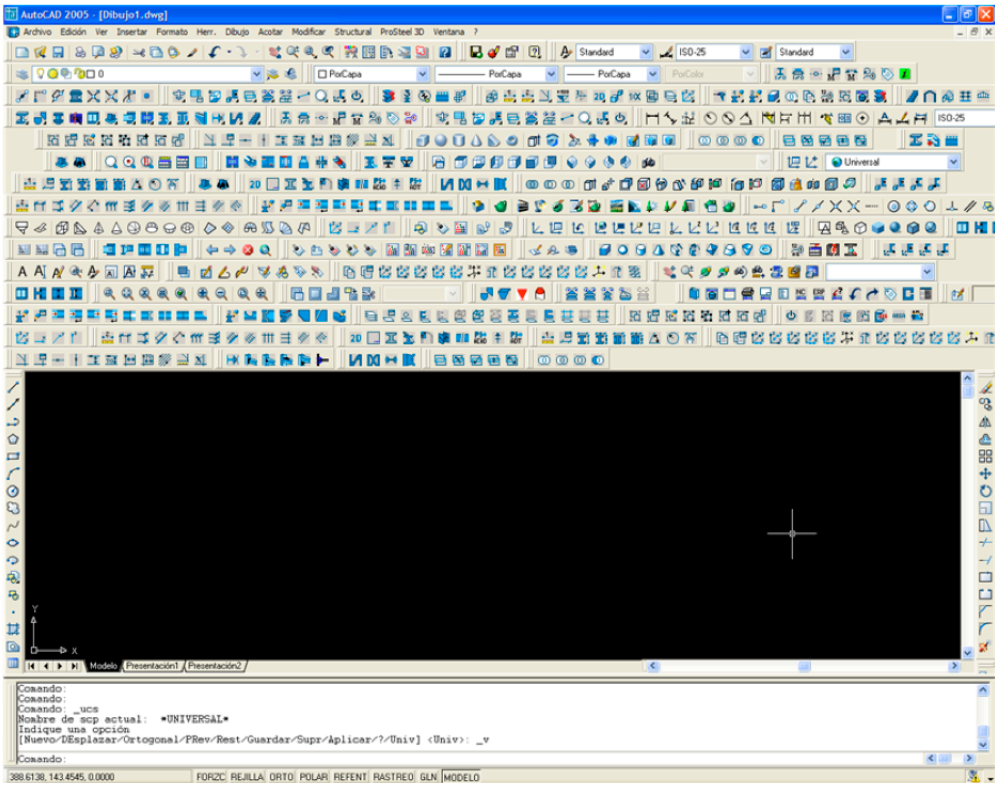


Figure 8.2: Reduction of the available working area in AutoCAD

In these challenging cases, the amount of options to choose from is very high, so it can be argued from a GUI point of view, that high generalization or high specification could be counterproductive and, moreover, that there are opportunities to improve the interface usability if the user profile and the PLM stage are taken into account in the GUI generation.

Challenge 3: There are functionalities (tools) that are not used or rarely used for a certain design task or even for certain stages of the product life cycle

As the CAD is not aware of the user needs, all functionalities must be available to the user at all times, consuming the visual/display working space while confusing the GUI (which leads to problems already described). This challenge has an extra implication when dealing with the usefulness of a tool at different stages of the PLM. For example, the dimensioning tools are highly demanded in the manufacturing stage but unneeded in the Recycling stage. The need of the PLM stage recognition arises in order to face this challenge.

Challenge 4: The User behavior (tailoring)

When learning a new CAD, a typical user inherits some habits from previous experiences with similar programs or personal tastes. For example a user coming from the Unix/Linux world is more accustomed to working with the command line than a Windows user who is more confident with the use of on-screen dialogs. The same happens with CAD software where the familiarity with the software speeds up work processes. Tailoring the software configuration to a User behavior can be counterproductive when different users operate the same computer because every CAD session ends up with completely different configurations to the settings of another user. To illustrate the point, one might consider two hypothetical users: User A is command line oriented hence, tends to remove all the buttons and the drop down menus to maximize the working space. On the contrary, User B is dialog oriented, hence he makes extensive use of drop downs, windows in-out dialogs etc. In this case, the working space is reduced and the command windows are probably absent.

When those two users work on the same computer, reaching an agreement over a common GUI could be hard, because it is difficult to find a mid point between their behaviors.

8.2 Brief description of our contributions

Considering the previously described challenges, we developed a technique which uses the advantages of Semantic modeling technologies for the automatic generation of GUI taking into account three main features: (*i*) the user and his needs, (*i*i) the stage of the PLM in which he is interacting with the CAD and (*iii*) the user intentions. This technique was implemented within the scope of an applied research project with a Steel Retailer (structural design) partner. For this reason, the examples and the implementation itself are strongly related to the domain of Steel Detailing. However, we believe that an extrapolation of the presented techniques to other domains will not be a difficult task. In brief, this Semantic Enhancement, can be summarized as follows:

- It is Domain centered in Steel Detailing, Structural Design.
- Makes use of the CIS/2 Standardization effort as the model of the underlying Knowledge Structure (an ISO STEP based protocol, that will be discussed later on this chapter).
- The Application presented here, can be used in the Definition, Design, Analysis, Production Planning, manufacturing and Maintenance stages of the PLM, see Figure 8.3.

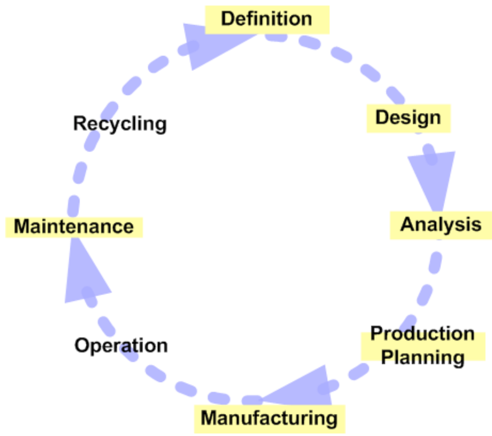


Figure 8.3: PLM placement of this Semantically Enhanced VET

- The main contribution in this VEA is the use of Semantics for the creation of user menus on-the-fly that adapt to the user needs and actual stage of the PLM.

8.3 Current Graphic User Interface design approaches

In recent times, the problem of producing a good GUI has been a recurrent topic in research papers and conferences. To highlight some approaches, we can mention the work by Furtado [FFS+01] who presented an ontology based methodology to produce GUI where multiple users carry out multiple tasks in a universal context. In this approach, they divided the GUI design in three layers: a conceptual layer where a domain expert defines an ontology of concepts, relationships and attributes of the domain; a logical layer where a designer specifies multiple models based on the ontology and a physical layer where a developer derives multiple user interfaces from the previously specified alternative models . This is not a completely automatic approach as it requires a developer to program the alternatives systematically (while our approach only requires the expert at the beginning). The interface development is focused in the modeling of the user but ignores the stage of the PLM. Furthermore, Furtado's approach is focused on window-based dialogs disregarding forgetting other interaction paradigms common in CAD programs. Puerta [PEGM94] based his approach on a model ontology, showing a GUI development environment and system implementation with both, static and dynamic behaviors. In this approach, the domain model allowed the creation of interfaces for medical applications. In our approach, we model the user as the executor of a design task, meaning that the design process is divided into stages where the user performs actions that require tools as it will be shown later on this chapter. Hovestadt [HGD95] presented a prototype implementation of a graphical GUI intended for the architectural design process. In this approach, the interface integrates CAD-like object manipulation and navigation through large data sets oriented to help the user in the process of changing directly from construction tasks to navigation tasks. In this approach, the main focus is to change the representation of the menus from the traditional pull down types to a hyperbolic shape. Igarashi [Iga00] presented a methodology to design a GUI focused on visual thinking activities (creativity design). The basic idea is to use transparent interfaces so that the user can directly interact with the target visual representations without using menus and buttons in a manner akin to a predictive approach. For the structural design problem, it would be difficult to predict tools as the process itself is engineering based, although some creativity is allowed. The area of intelligent and Adaptive User Interfaces (AUI) has been of interest to the research community for a long time. Much effort has been spent in trying to find a stable theoretical base for adaptivity in human-computer interaction and to build prototype systems showing features of adaptivity in "real-life" interfaces, an interesting state of the art of AUI was presented by Brown in [Bro90]. in contrast to the aforementioned authors, we present a technique which uses the advantages of Semantic modeling technologies for the automatic generation of GUI taking into account three main features: (*i*) the user and his needs, (*i*i) the stage of the PLM in which he is interacting with the CAD and (*iii*) the user intentions.

8.3.1 Specific VEA case: CAD Steel Detailing Design based on CIS/2 LPM/6

Steel Detailing is a branch of design that focuses on the production of detailed drawings for steel fabrication and steel erection. The steel detailer prepares blueprints, drawings and other documents for the manufacture and erection of steel members (columns, beams, braces, trusses, stairs, handrails, joists, metal decking, etc.) used in the construction of buildings, bridges, industrial plants, and non-building structures. A steel detailer's projects are usually commercial, industrial or municipal; residential developments are rarely large enough to require significant amounts of structural steel. CIS/2 (CIMsteel Integration Standards, Version 2) [CIS08] is an effort led by Leeds University (UK) and SCI (Steel Construction Institute). This approach is based on the ISO-STEP (10303) protocol. CIS/2 extends STEP so it encompasses conditions encountered in structural steel design. The CIS/2 data model is called the Logical Product Model (LPM) and it is currently in its sixth revision (LPM/6). The main objective of CIS/2 is to support the engineering of low, medium and high rise constructions for domestic, commercial and industrial contexts. Although CIS/2 has been developed primarily to enable the engineering of building frames, it can also be applied to other types of steel frames, such as process plant installations, transmission towers, and (to some degree) offshore structures.

To refer to some examples of use of CIS/2, we can mention the work by Danso-Amoako [DAOI04] who presents a case study for the support of steel supply-chain processes, showing that the standard can be used not only for information exchange in design, but also for information management during lengthy stages of the supply chain. Lipman [Lip03] presents an approach to using CIS/2 as a kind of storage model for an immersive virtual reality system inspiring one of our test tools. Alda [ABH04]uses the CIS/2 framework for the support of collaborative design on a peer-to-peer basis, showing the importance of a common base in a Virtual Reality (VR) design scenario. Our work does not focus on collaborative design, but could be extended in the near future to support such a scenario. The structural steelwork lifecycle supported by CIS/2 is shown in Figure8.4. This is a high level overview of the standard process model for Structural Design. CIS/2 describes three major stages needed to design a steel frame structure for a building: (*i*) Analysis, (*ii*) Design and (*iii*) Fabrication, each phase has its own data model that serves for information exchange between each other and external applications. The three different models of a steel structure are called "Views" [CIS08].

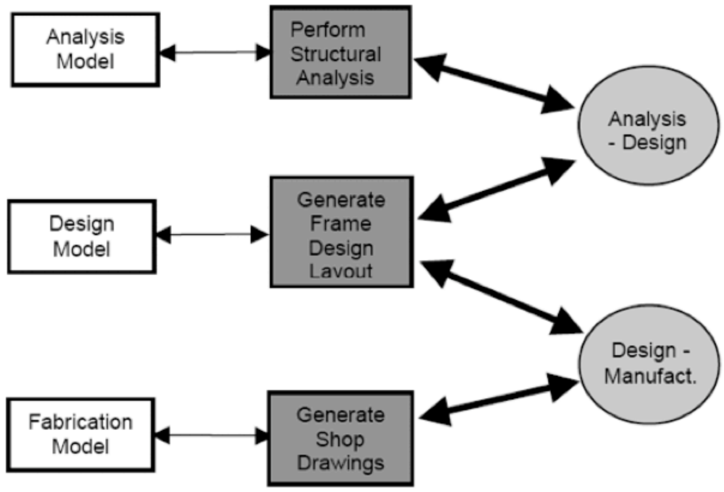


Figure 8.4: Simplified process model identifying the various CIS/2 views and major exchanges, supported by high-level conformance classes

8.4 User Interface generation - Core concepts

In our framework for the steel detailing case, the adaptive GUI generation is a core contribution. For this, we define a CAD tool as:

> *"The means whereby some act of design, drafting, display or modification of entities in a graphically oriented environment is accomplished"* [TPOF07]

In other words, a tool is an active part of the GUI that executes an action inside the CAD (e.g. the command "line"). Since our Semantic enhancement of VEA is PLM oriented; we consider the element designed as an entity that passes through different stages of the life cycle, in each stage of the PLM a CAD VET uses different tools that can be shared between stages. An example of this is the tool "*line*" that can be used in the design or the analysis stages. Our approach for the GUI generation starts with the prerogative that every stage of a structure life cycle can also be sub-divided into sub-processes. In Figure 8.5 a design sub-process is shown. In each sub-process a set of CAD tools can be applied by the user (also any CAD tool can be placed in more than one sub-process).

The steps in our approach can be summarized as follows (see Figure 8.6):

- **Tools Categorization (TC-Stage)** : In this stage, the available toolset in the CAD VET are categorized by an expert taking into account the stage of the PLM in which they can be used.

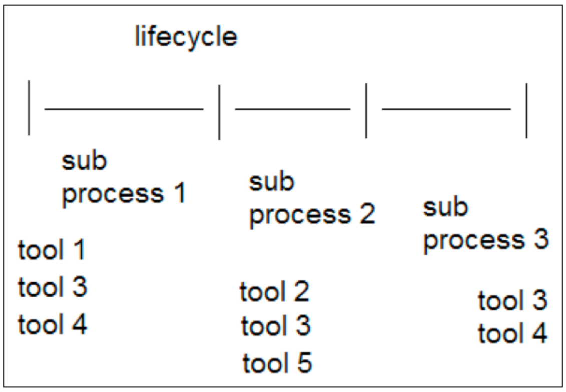


Figure 8.5: Sub processes View schema

- **Process Categorization (PC-Stage)** : In this stage, the different processes of the PLM are described with sub-stages when possible (according to the domain of study).

- **User Intention and background Categorization (UIC-Stage)** : In this stage the different types of users are defined, taking into account possible experience categorizations (according to the domain).

- **Supporting Knowledge Base Modelling (SKBM-Stage)** : In this stage, the conceptualizations made in the first three stages are modeled in a suitable Knowledge Base container.

- **Knowledge Base Reasoning (KBR-Stage)** : In this last stage, a reasoning process is made in order to determine the set of allowed tools based on information stored in the Knowledge Base. The Reasoning and handling process is commonly performed through a programmatic approach.

Tools, Process and User Intention and background Categorizations (TC, PC and UIC-Stages).

The three stages above can be implemented using a categorization table. This type of table is essentially a set of conditions to be met in order for the tool to take effect. A domain expert whose main task is to classify the available set of CAD tools does the initial categorization process. Part of the domain expert's job is to

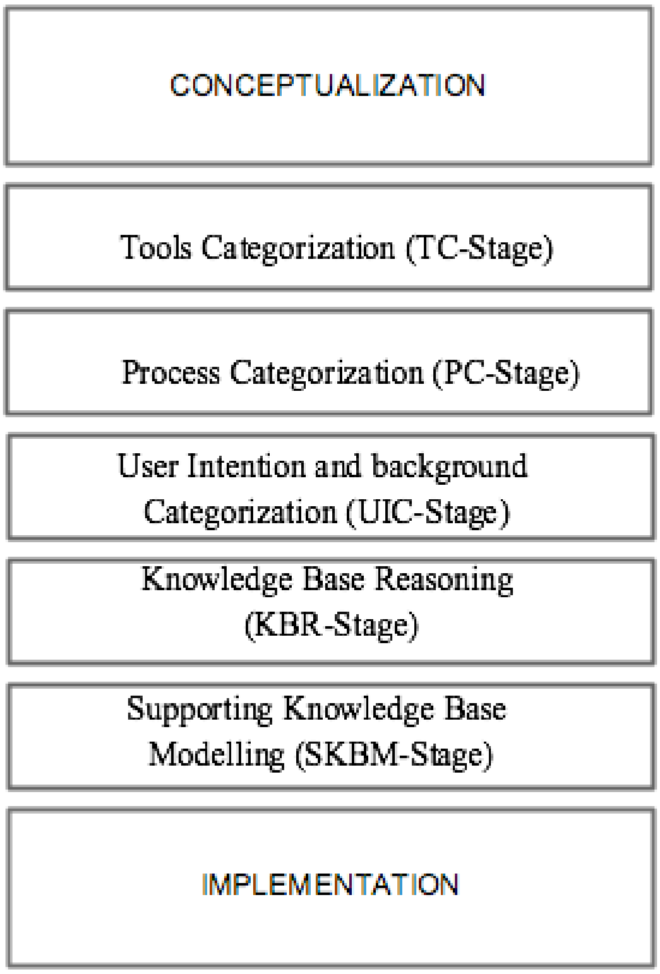


Figure 8.6: Stages in the generation of a Semantically-Oriented GUI

catalog the different types of target users, taking into account possible experience categorizations (according to the domain). User categorization is seminal and commonly absent in various knowledge-based architectures. A good example of this is the case of the "line" command, which can be accessed by an expert user via polar coordinates(*@lenght<angle*). A non-technical user, would likely be more familiar with a simple initial-end point paradigm (*<Pix,Piy,Piz>* , *<Pfx,Pfy,Pfz>*). Both calls will produce the same geometric entity, but the user expertise and knowledge is utilized to produce a better user experience. Figure 8.7 depicts a portion of a categorization table.

Model_type	Process_stage	for	User_intention	User Type	T1	T2	T3	T_{n-1}	T_n
Steel_based	design		assembly	expert					
Steel_based	fabrication		assembly	novice					
Steel_based	analysis		assembly	technical					
Steel_based	design		detailing	non-technical					
Steel_based	fabrication		detailing	designer					
Steel_based	analysis		detailing	not_espeficied					
Wood_based	design		review	expert					
Wood_based	fabrication		review	novice					
Wood_based	analysis		review	technical					
Wood_based	design		other	non-technical					
Wood_based	fabrication		other	designer					
Wood_based	analysis		other	not_espeficied					

Figure 8.7: Excerpt from a categorization table

Supporting Knowledge Base Modelling (SKBM-Stage)

Once the initial categorization by the domain expert is done the system is initialized and the ontologies are instantiated with the information provided. The Knowledge Base can benefit from the use of standards in the domain, when available. The Knowledge Base should be modelled with consideration for account restrictions imposed by the **PC** and **UIC** stages.

Modeling of the Process ontology according to CIS/2 standard

Our Process Ontology implementation is divided in two main classes: Available Semantic Tools and Tasks. The first class contains the VET CAD available tools, or in other words, the set of options that will be offered to the user. The Task class includes the overall actions, as a composition of a Model type (steel or wood), sub/process (analysis, design or fabrication) and User Intention (assembly, detailing or review), these three parts are links to OWL compliant Ontologies. In Figure 8.8 the detail of the Process ontology is shown.

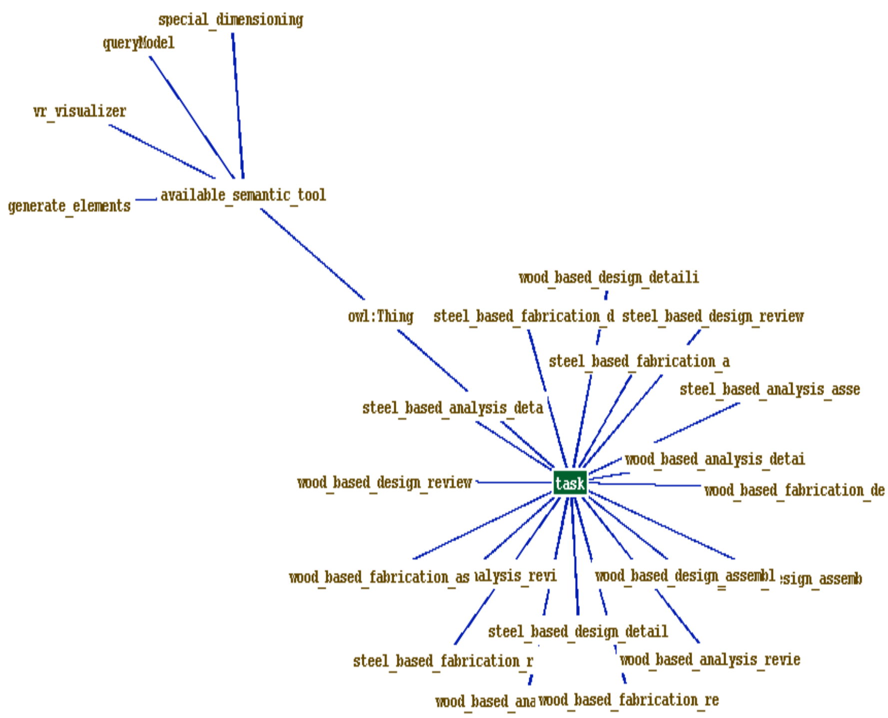


Figure 8.8: Detail of the Process Ontology based on CIS/2

Knowledge Base Reasoning (KBR-Stage)

For a given Model m_i, Process p_j , User Intention i_k and User Type u_o, the goal is to produce a set of Recommended Tools (RT) derived from a set of Available Tools (T) (Figure 8.9).

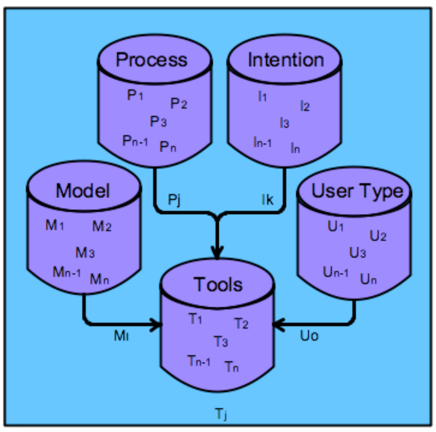


Figure 8.9: Schematic of the Model-Process-Intention-User type View

According to the sub-process schema, let us define the following variables: m_i=

Model, p_j = **Process**,i_k = **Intention** and U_o= **User Type**.

We define an injective relation $\mathfrak{R}$ that assigns every quadruple (m_i, p_j , i_k , u_o) to one or more tools T_n belonging to the set of Available Tools T. Which is expressed formally as: $\mathfrak{R} \subseteq X \times T$ where :

$$X := M \times P \times I \times U = \{(m_i,\ p_j\ ,\ i_k\ ,\ u_o) : m_i \in M,\ p_j \in P,\ i_k \in I\ ,\ U_o \in U\ \}$$

In this expression, a set of restrictions R will map to a subset of the available tools T as follows:

$$R_{(m_i,p_j,i_k,u_o)} = t_i \cup\ t_j\ \cup\ \ldots .\ \cup\ t_r$$

As a result, the Recommended set of tools (RT) is then described by the following expression:

$$RT_{(m_i,p_j,i_k,u_o)} = \overline{(T \cap R(m_i, p_j, i_k, u_o))}$$

The above is implemented, for example, using an ontology editor. To implement $\mathfrak{R}$ (the injective relation) in the ontology, the open world assumption and OWL description logics are used in order to relate both parts of the ontology via restrictions. In order to model the mapping from the Available Tools (AT) subclass of the ontology and the task class, we restrict an object type property using OWL DL restrictions. At this point we have a set of ontologies instantiated with values from the real world and with a reasoning model capable of recommending the most relevant tools to the user according its needs. Every new interaction with the system refines the initial model (the one that the expert gave at the very beginning) by using the reflexive ontologies concept. The user is presented with a simple set of questions and starts the reasoning process over the ontologies as described above. Once that the reasoning process ends, a modification of the CAD GUI Graphical User Interface (buttons, drop down menus, command lines, etc) is produced using the CAD API.

8.5 Execution example

In this section we will present a simplified execution example of the previously described technique. According to CIS/2 we can divide the types of structures (Model Type) into Wood Based or Steel Based (materials categorization) with their corresponding subdivisions. The process stage related to the PLM cycle of the structure itself but with slightly different names for the stages follows the CIS/2 model and is divided in three global categories with their corresponding subdivisions called: Analysis, Design and Fabrication. The Domain was modeled using an ontology paradigm and followed the recommendation presented in Chapter 5.

For the User Intention (part of the sub-goal division we recommended in Chapter 4), we modeled five categorizations: *Detailing*, *Fabrication*, *Assembly*, *Design Review* and a general-purpose intention that we called *Intention_not_specified*. Since this implementation was made for an applied research project being conducted with a well-known steel detailer in Spain, we asked them for their categorization of their User types and we found that their users were specialized and trained to perform specific tasks (Intentions). In our customer's case all the users were considered technical (CAD detailers) which allowed us to simplify the categorization questions. Figure 8.10 depicts the implemented categorization questions interface.

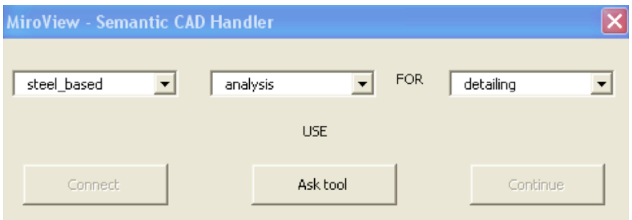
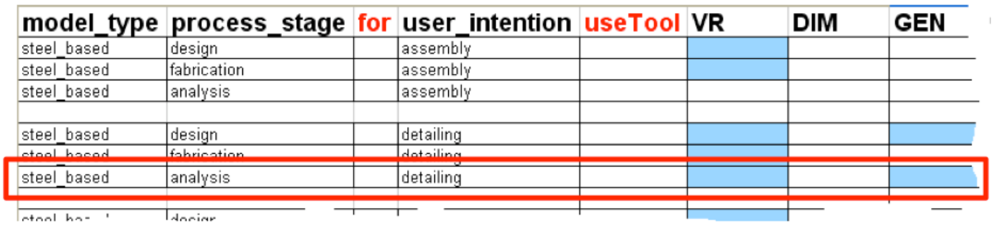


Figure 8.10: Session Characterization

We asked the experts in the Steel Detailing Company to initialize the model providing them with a small subset of the CAD tools and specifically with some other tools we implemented within the scope of the research project where the contribution was developed.

Figure 8.11 presents an excerpt of the categorization table obtained and used to instantiate our CIS/2 based ontology.

In order to model the mapping from the Available Tools (AT) subclass of the



model_type	process_stage	for	user_intention	useTool	VR	DIM	GEN
steel_based	design		assembly				
steel_based	fabrication		assembly				
steel_based	analysis		assembly				
steel_based	design		detailing				
steel_based	fabrication		detailing				
steel_based	analysis		detailing				

Figure 8.11: Detail of the relations table, VR= Visualization tool, GEN=Element Generation Tool, DIM=Dimensioning tool

ontology and the task class, we restrict an object type property using OWL DL (description logics) restrictions. The restriction for the case shown is as follows:

$R = Dimensioning\ Tool$

$T = Visualization\ Tool \cup Element\ Generation\ Tool \cup Dimensioning\ Tool$

$RT = \overline{(T \cap R)} = Visualization\ Tool \cup Element\ Generation\ Tool$

Once the restrictions are applied to the original set of available tools, the reasoned outputs the set of recommended tools and their GUI paradigm (command, drop down, etc). The automatic tool recommendation also adds an extra command for handling purposes; this is called "GUI Handling options" and is used for maintenance operations (destroy, rebuild, add a non recommended tool, etc) Figure 8.12 depicts a screen-shot of the test case.

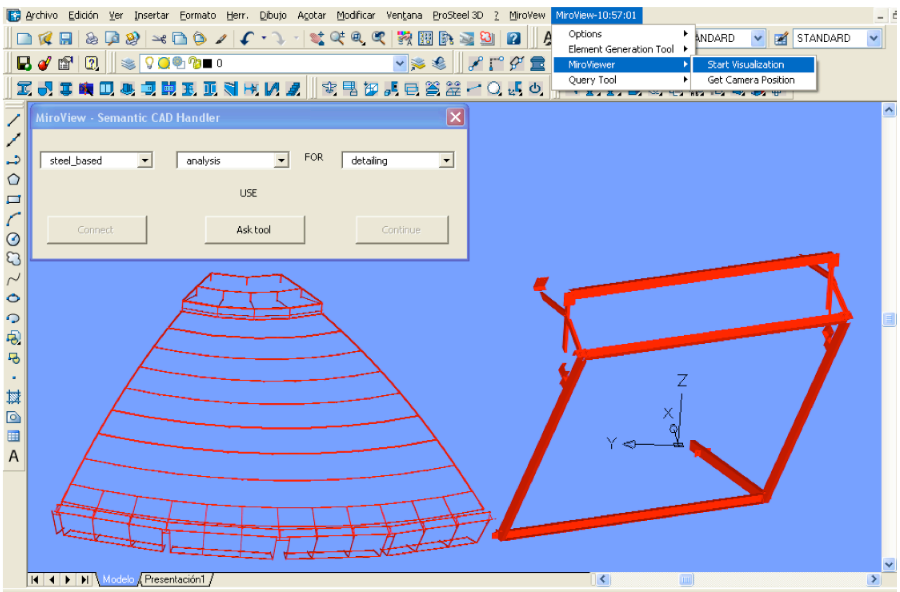


Figure 8.12: A Semantically generated UI

Not only were the drop-down menus created, but also shortcuts, buttons and relevant GUI features which in fact are (according to the expert modelling) the best set of tools to the task at hand. A feature hidden to the user is also performed as the recommended set of tools and the process of obtaining such a set are handled via a Reflexive Ontology query. The use of Reflexive Ontologies enriches the Knowledge Base while at the same time accelerates the reasoning process because the next time that a similar scenario is presented, a reduced reasoning process must be done.

8.6 Discussion - Semantic User Interfaces

In this Chapter, we presented a technique which uses the advantages of Semantic modeling technologies for the automatic generation of GUI taking into account three main features: (*i*) the user and his needs, (*ii*) the stage of the PLC in which he is interacting with the CAD and (*iii*) the user intentions. The presented approach was used for the generation of graphical user interfaces on a structural CAD software. Most of the concepts introduced in this section have been published in the author's scientific paper "Ontology supported adaptive user interfaces for structural CAD design" [TTP+07]. Our approach deals directly with the structural design process in order to suggest the user with the most appropriate tools for every stage of the design. The presented schema was modeled using an ontology that allows the semantic interrogation of the different parameters needed to produce the GUI. This Chapter technical contribution are identified in the framework of the VET Semantic enhancement general architecture presented in Chapter 7 as shown in Figure 8.13.

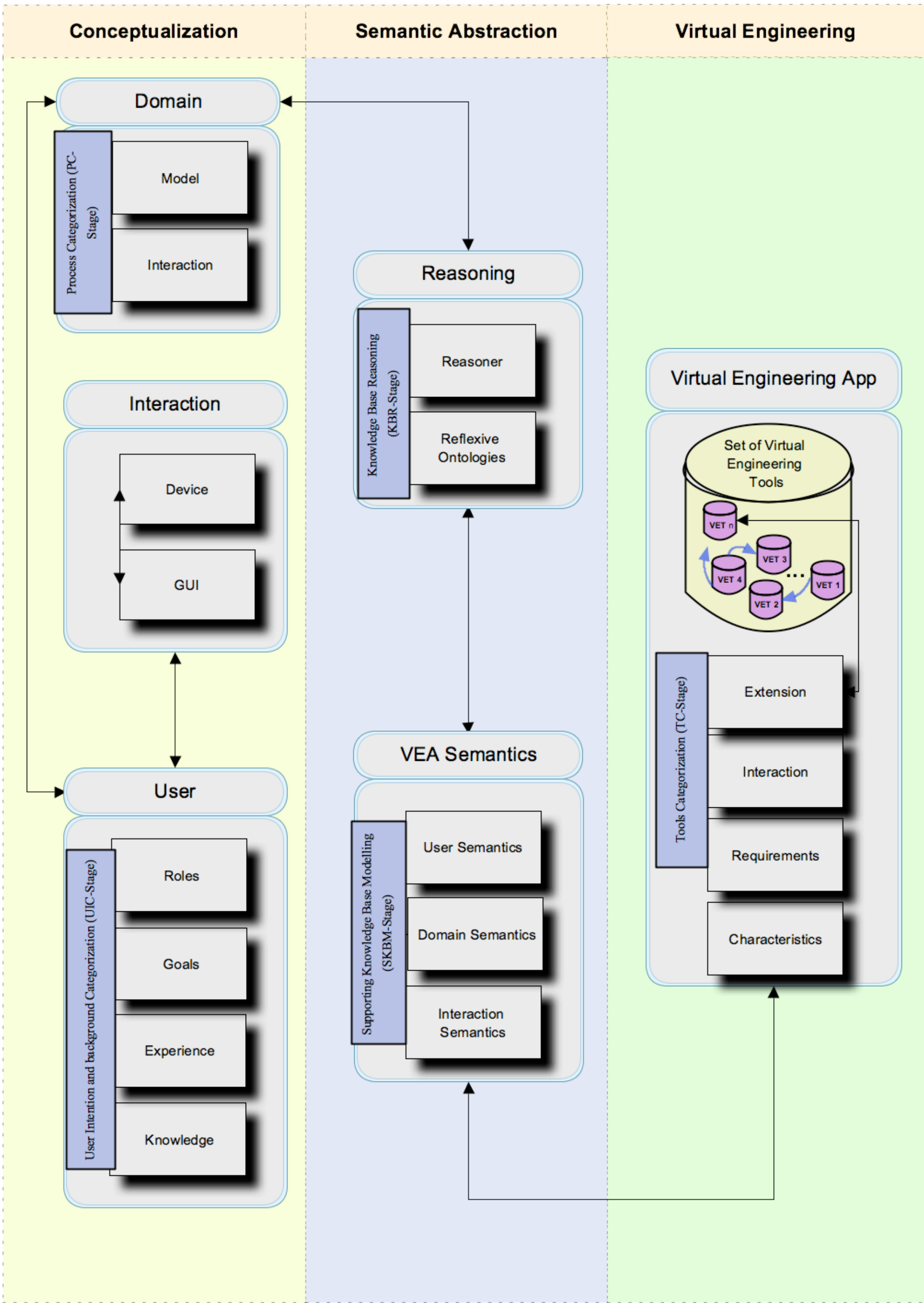


Figure 8.13: Relation of this VET with the general architecture

Supporting publications

This Chapter was supported by the following Papers and Journals in International conferences:

- Toro, C. /Termenón, M / Posada, J / Oyarzun, J / Falcón, J: Ontology supported adaptive user interfaces for structural CAD design In: Digital Enterprise Technology, Perspectives and Future Challenges, in Cunha, Pedro F.; Maropoulos, Paul G. (Eds.) ,2007, XVIII, 593 p., Hardcover, ISBN: 978-0-387-49863-8, Springer, Germany.

  Remarks: This Book Chapter introduced the concept of semantic-based user interfaces that adapt themselves in response to a user need or previous knowledge.

Chapter 9

Contributions to the Semantic simplification of geometries for Large Scale Visualization of Industrial Plants and Design Review

In this chapter we present our work in semantic simplification for a VEA of Large models for Design review of Industrial Plants. This work has been the result of a collaborative work ([PTWS05a], [PTWS05c], [PTWS05b]) ; however the portions described here are more representative of the author's specific contributions.

9.1 Description of the technical problem and challenges identified:

The possibility of having interactive walkthroughs (Design Review), for very large geometric data-sets, offers the promise of clear benefits as it reduces design times while at the same time allowing engineers and designers to detect early any potential construction problems that may appear ([BAN+02], [FTK96], [PWTS05]). When dealing with massive-geometric data for the purposes of visualization and review, a collection of compression algorithms and diverse simplification techniques are used in order to produce a VR output suitable for a walkthrough experience. In general this field is known as Large Model Visualization (LMV). While the principal approaches presented in the literature in the field of LMV for Design review purposes are generally related to algorithms and compression methods applied to a CAD model via Computer Graphics (CG) techniques ([PLS02], [Man00]) the following challenges have been identified:

Challenge 1: No special attention is given to models containing parts arranged in exact and often repeated shapes.

In an LMV model, a common issue is the existence of sets of repeated parts, e.g. valves, pipes, elbows, etc. This characteristic, that we shall refer to as element redundancy, can be exploited by taking into account the nature of the models and the information implicitly contained. An approach to achieve the aforementioned task does not imply only the ability to identify features, but also, to identify groups of repeated elements and to treat them according to a given criteria. Knowing the elements, their function in the design, and their neighbor relationships is a desirable feature that can be used in conjunction with traditional CG techniques to enhance the visualization and understanding of Large Models. In the case of Industrial Plants, there is knowledge can be used to perform a better Design Review of the model.

Challenge 2: User needs and characteristics, prior knowledge (background) are underrepresented or not present in traditional CG approaches.

Typical LMV handling techniques not only misuse redundancy, but also they do not adequately consider user characteristics, needs or background. For example, let us consider a Plant CAD model used for promotional purposes, in such a schema, the information about structural details (e.g. clamps) may be less important than good rendering (e.g. photo-realistic). Considering that a clamp is a highly repetitive element, the set of clamps and its component triangles, could not be drawn in favor of a better rendering quality. This stated approach will only be possible if the user needs and profile are a known factor. In such a scenario, the user "needs" condition the appearance of the elements in the model in the form of a type of selective reduction sort (a CG technique that will be discussed later on this chapter).

Challenge 3: Domain based models usually are not taken into account (although can be argued that some high end CAD programs feature a type of domain approach).

Even when containing the same set of elements, the functionality and importance of such set, is related to the Domain. For example, the element "valve" has a different meaning for an engineer focusing on the structure of the plant, or for an engineer focusing on the fluids transport (functional process). Valves in industrial plants are features of utmost importance when considering the functional process, but from the point of view of the structural engineer, they are relatively uninteresting. As already demonstrated in this thesis, a good approach for Domain modeling is to base the KB structure on engineering standards when available.

Challenge 4: Traditional CG based techniques have a theoretical limit.

Even in current computers, the number of triangles drawn (measured in kTris or thousands of triangles) affects the fluidity of the walkthrough experience. The vast majority of current technologies are focused on the reduction of triangles; however such reduction can be calculated and reaches a limit imposed by the limitations of the hardware and the algorithm used. A new LMV technique, or an enhancement of an existing CG technique for LMV, could be steered and enhanced from an orthogonal perspective, such as the technique that semantics provides.

9.2 Brief description of the contributions

Considering the previously described challenges, we developed a technique that leverages the advantages of Semantic technologies in the LMV problem, taking into account: (*i*) the user and his intentions, (*ii*) the model characteristics (Domain), (*iii*) the available resources of the machine where the walkthrough experience is taking place. This technique was implemented within the scope of the MiroWalk applied research project. We base our approach on the categorization, simplification and semantic compression techniques for the engineering parts contained in the model. In our case study we based our Knowledge Structure on the ISO initiative STEP (ISO 13013-AP227) [ISO01]that will be introduced in this chapter.

In brief, our contribution may be summarized as:

- It is Domain centered in Industrial Plant design and Virtual Reality.
- Underlying Knowledge Base is based on ISO Standards (STEP,CIS/2).
- The presented can be applied mainly to the Definition, Design, and Analysis stages of the PLM, see Figure 9.1.

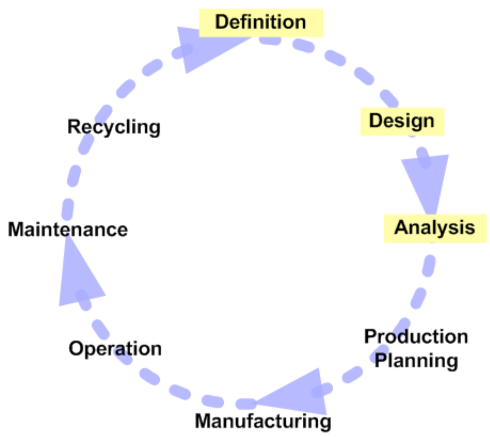


Figure 9.1: PLM placement of this Semantically Enhanced VEA

- The main contribution here is the implementation and extension of the semantic visualization concept presented by Posada [Pos05] in where the traditional visualization techniques of a Large CAD model were enhanced using semantic criteria. Our contribution is based on an architecture that allows three supporters of information, a CAD file (seen as the VET to extend) a Virtual Reality (VR) viewer and the underlying Knowledge structure (consisting in a STEP based ontology).

In the following section we will highlight some relevant concepts.

9.3 LMV concepts and CG approaches

The problem of large model management has many different aspects sides: not only is it necessary that the computer deals with the model, but the visualization must also be easy to understand. It is true that high-end CAD systems already have excellent visualization tools, but several widespread CAD systems still depend on converting the model to a Virtual Reality format, such as VRML. An automatic (or preferably, a semiautomatic) conversion to VR models is still a valid approach. But to do this conversion process well remains a problem, especially where a typical Workstation class computer are concerned. When the models are not very large, the tools available for direct conversion usually work acceptably. However, this conversion process is not addressed with sufficient detail for Large Models in normal working environments. As a result, in many cases, the interactive visualization of such a Large Model fails, since it approaches the resource limitations of computer (memory, processing power, etc.), making it unusable in normal circumstances. A problem that usually arises is the loss of information during the conversion process to the VR model. CAD models store a large amounts of information, including, but not limited to, the geometrical and visual representation. In some cases, a relevant part of the dataset is invested in complementary information that is not directly visible but very useful (e.g. the organization of the graphic elements in a level-tree or the information associated to the characterization of the different parts). In certain CAD systems, after the conversion, a complete model becomes a mere graphical scene in which it is only possible to access the visual geometric information; access to the rest of the information is only possible in the CAD format, so the VR model is not as useful as it could be.

9.3.1 Brief review of some relevant CG techniques

In the scope of this work, the most relevant topic regarding LMV is the research area of interactive walkthroughs of large models. As pointed out by Manocha in a classical SIGGRAPH course [Man00] there are basically four families of topics to consider in the interactive walkthroughs of large models: (i) Techniques dealing with rendering acceleration, (*ii*) Techniques dealing with Database management, (*iii*) Techniques dealing with the Interactive collision detection, and (*iv*) Techniques dealing with System integration There is a similar exposition by [Bar01] with a slightly different formulation: accordingly, the main rendering techniques focus on database management, architectural aspects of large computing systems, parallel computing, and the most important aspect, rendering techniques for visualization. Several authors have explored the main rendering acceleration techniques with good results so far. As comprehensively explained in [BAN+02] and extended in [CCS00]. Amongst the above techniques, the most widely used are: (*i*) Culling ([Bax02], [CCS00]), (*ii*) Geometric Simplification ([LRC02], [Man00]) and (*iii*) Image-based representations ([BAN+02, Man00]).

Reported Semantic technologies approaches to LMV

Other approaches have been investigated in the past for Semantic modeling in the Plant Design Domain. The main reference in this area is the work of Mizoguchi (Osaka University) [MKSK00], whose group has investigated how Industrial Plant Ontology can be modeled and used for functional processes (called tasks) such as diagnosis, monitoring and scheduling. The ontology-modeling environment is called "Hozo", used for building the plant ontology and model. It is composed of a graphical interface, editor and ontology/model server in a client/server architecture. This work is essentially of a complementary nature to this research in several ways: Firstly, it focuses on operational processes, in order to have a common representation and sharing of knowledge between agents (computer or human), and not in standard physical components for visualization walkthroughs, as we do. Secondly, it is not developed using existing standards (the ontology tools used, as well as the types of operations, components of the plant, etc. are their own independent development). This is mainly an understandable decision if the goal is to investigate how plant ontology can improve current knowledge sharing between agents in a well-constrained scenario (Oil refinery plant), and not other goals such as interoperability with other systems.

9.3.2 ISO STEP 10303 Protocol - Overview [ISO01].

In the 1980s, several international standards bodies came to the conclusion that the (then) current generation of translation methods, based on files formats such as DXF and IGES were not capable of supporting complex data models of engineering products. These organizations included the American National Standards Institute (ANSI), ISO and other national standards agencies in Europe. The US and European efforts began in parallel, but soon merged. They adopted the following, now relevant, goals: (*i*) Incorporate new programming language concepts, especially those dealing with object-oriented programming, (*ii*) Incorporate formal specifications of the structures defined, independent from the implementation, using the new data-modeling languages., (*iii*) Separate the data model from the physical file format and support mapping into multiple implementation formats, including files and databases, (*iv*) Support subsets of an overall model, so that clusters of applications could be integrated without the overhead of having to deal with parts of a model irrelevant to a task, and (*v*) Incorporate reference models that are common shared subsets of larger standard models. The major advantage resulting from this new architecture was the commercial development of a number of software toolkits, which facilitated the implementation of interfaces. The STEP toolkits also implement a standard file format for data storage, allowing reading and writing to/from a file. The ISO STEP standard is divided in application protocols (AP) as a logical division between different aspects of the standard. In the case of Industrial Plant Design, the correspondent application protocol is the number 227. To point an interesting fact, this protocol is been used also for ships design, this is because of the similarities found in both schemas and the fact that for the time being an AP for this domain is not available. ISO 10303-227 was prepared by the Technical Committee ISO/TC 184, Industrial automation systems and the integration Subcommittee SC4 (Industrial data). The information in ISO 10303-227 includes the shape and spatial arrangement characteristics of piping system components as well as the shape and spatial arrangement characteristics of other related plant systems (i.e., electrical, instrumentation and controls, heating, ventilation and air-conditioning, and structural systems) that impact the design and layout of piping systems. In the design and fabrication of a piping system, the piping layout must be evaluated with respect to the spatial characteristics and arrangement of these related plant systems, and the requirements for clearances between systems. The complete specification of these other systems is not required, but sufficient spatial information is needed to support the layout of the piping system. Users of this standard should understand the basic principles and concepts of plant and piping system design. The main focus of the AP is on piping systems and the shape and spatial arrangement of systems including plant items ensuring the physical integrity of piping systems.

9.4 An Architecture for the Semantic enhancement of Visualization in Large Models

In this section we present our architecture. This architecture is adapted from the general architecture presented in Chapter 7. The relation of the adaptation for this specific VET and the general schema will be presented in the discussion section of this chapter. In our approach for this VET, we take as a starting point any proprietary geometric 3D CAD representation belonging to the Industrial plant Design Domain. We consciously assume that no other information is available (e.g. from a modern PIM system). We then automatically reconstruct the families of engineering parts in the model; associate those families to the Domain standard and then apply both geometric and semantic object simplification techniques in order to present an adapted VR model in an interactive walkthrough system which is the VET. We use semantic compression added to geometrical data simplification techniques to increase the efficiency of the VR model and to complement the traditional Computer Graphics methods that are applied in the classic approaches. This architecture can be seen in Figure 9.2.

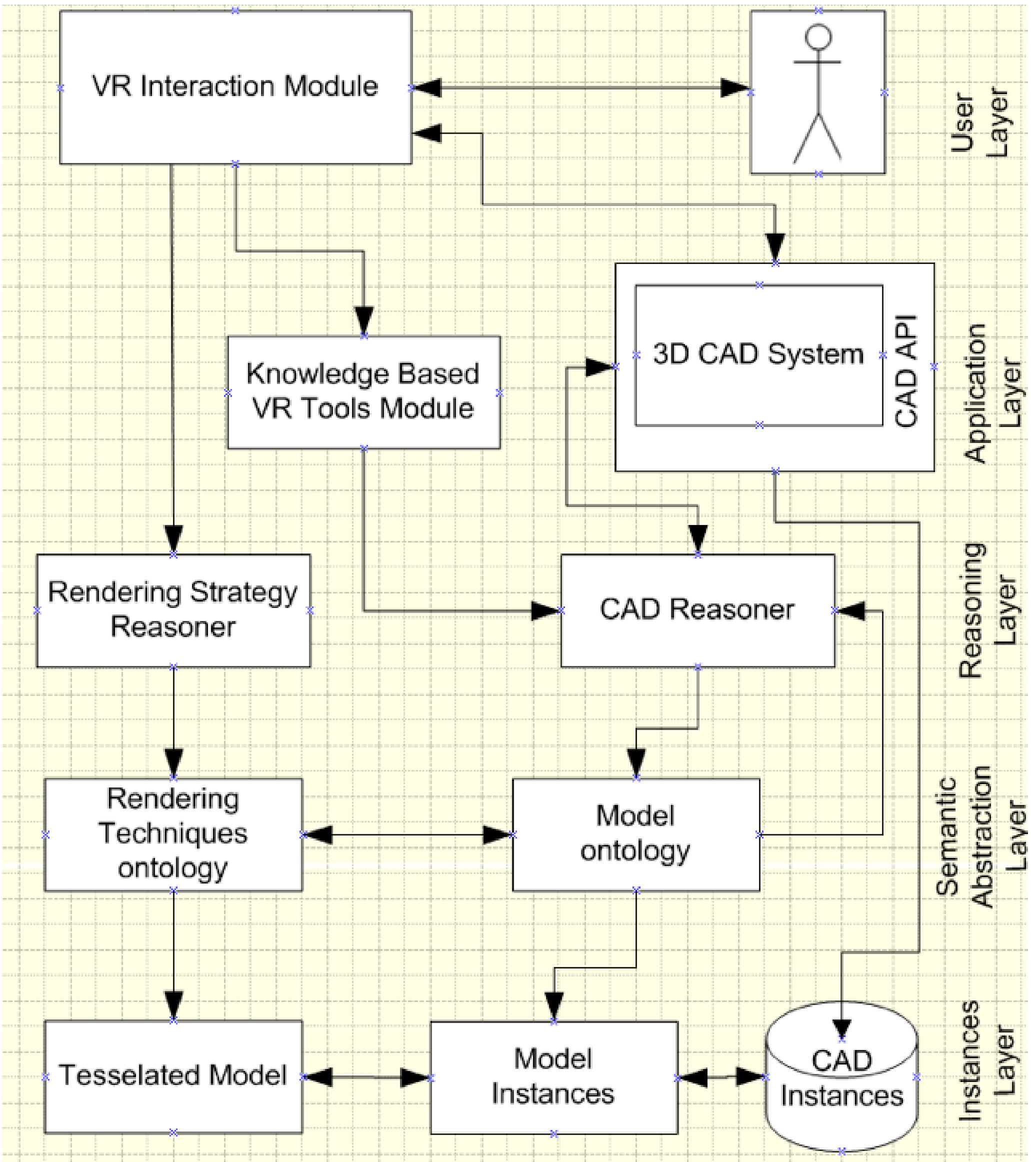


Figure 9.2: Semantic walkthrough architecture

Modules description:

The User Layer is in charge of the bidirectional interaction between the User and the different application modules of the system. It can be implemented as an external application able to share data and communicate between the VR environment and the CAD kernel, in our approach we used the recommendations in Chapter 6 for the modeling of the User and the interactions. The second layer is called the Application layer and contains two modules, the first one being the 3D CAD system itself (with its API) and the knowledge-based VR tools module, that contains a knowledge model of the VR tools available (e.g. stereo capabilities, specialized hardware handling, etc). In the third layer called the Reasoning layer, our architecture includes two modules that are in charge of the semantic reasoning located in this Layer. One is the rendering strategy module, which contains a reasoning engine whose main inputs are the user type, the computational available resources, processor speed, available RAM, max Ktris per second. The second module in this layer is called the CAD reasoner, whose main activity is to perform geometric reasoning on the elements contained in the actual CAD model. Amongst others, some of the geometric reasoning activities include geometric similarity between elements, handling of cells or groups of elements and handling of parts at a parametric level. The reasoning layer uses a query engine based on the Reflexive Ontologies (RO) concept presented in Chapter 6. We decided to use an RO approach because when we tested the first implementations, they behaved well with a test (small number) of 3D geometries, but when the model grew up in size, the query process required more time due in part to the computational complexity increase. By implementing the queries using RO, the time consumed was lessened because of the fact that many of the questions made to the ontologies were stored in the RO model. The next layer is called the Semantic abstraction layer. Its main purpose is to store the ontologies that will be instantiated with the information gathered from the CAD model. This layer is divided into two modules: the first one is called the Rendering Techniques ontology which contains an ontology model of the different (CG and semantic oriented) techniques available to produce the VR output. The second module within this layer is called the Model ontology which contains the standard based compliant conceptualization of the model Domain (ISO STEP, CIS/2). The last layer is called the instances layer and contains three modules; the tessellated model module, which is the triangulated model, generated using the chosen rendering techniques suggested by the reasoning layer. The Model Instances module contains the categorized list of elements from the model according to the geometric definition given in the model ontology. That is, a collection of elements whose characteristics are similar (e.g. in the context of Industrial Plants, groups of elements of the ISO STEP 10303 AP227 modeled Domain). The third module contains the CAD instances, in other words, the actual elements that belong to the CAD model (VEA).

Domain modeling

For this technical contribution, we have modeled our Process and Model ontologies based on Engineering Standards (ES) under a Domain perspective following the recommendations given in Chapter 5. The reason to use ES is because our ultimate objective is to have a system where the concepts and relations of such Domain can be handled and queried using Semantic criteria, beyond the mere data modelling structures of the norm. This improves the generality of the presented approach, as the elements and their characteristics are described following a widespread agreement between a well recognized group of users, engineers and industries. Of course, some properties have to be completed/extended, as we found out that certain elements described by the standard are viewed from a mere functional point of view (e.g. valve element in ISO STEP 10303 AP227). The aforementioned challenge was also discussed in Chapter 5.

9.5 Example of use of the proposed Architecture

In this section, we present two VETs based on our proposed architecture. These test cases where implemented during two research projects held in cooperation with leading industries and research centres in Spain and Germany. In the first project, the main idea was to develop a Large model Visualization system for the real time navigation of industrial plant facilities using user, model and resources information following a semantic approach. The second project's intention was the development of a series of semantic based tools (VET) to support the structural design (Steel Detailing) process of which one was a specific the development of an advanced VR viewer to extend a generic CAD-VEA.

Using Semantics for visualization purposes of large industrial plant models

As mentioned before, in Large model Visualization, the basic idea is to navigate smoothly through a CAD model containing a large data set of 3D elements (most of them repetitive). In order to extend the VEA, the following steps were taken.

Semiautomatic reconstruction of conceptual components/parts (Catalog Reconstruction)

The Domain ontology in this case was modeled using STEP, the CAD reasoner performed geometric similarity operations between the modeled parts and the CAD instances and the 3D CAD used was AutoCAD extended via the provided API which allows the extension of the VET from a programmatic perspective. In this case, the process ontology was modeled to reflect design review needs. However, the final CAD model itself does not keep this information in an explicit manner. Based on the model characteristics, we look for conceptual parts/components that match the Domain modeled elements. The geometric objects in CAD models of Plant Design are produced mainly by occurrence in the application Domain (in this case, Plant Design). This knowledge is used to store the information about instances of elements in the model and to set up a relation between repeating elements and their possible matching concept in the ontology. An example of partial catalog can be seen in Figure 9.3. The catalog has to be semi-automatically filled using structural information of the CAD model and the knowledge of the user, who interacts in the process. We used for the aforementioned process an algorithm for catalog filling presented by Posada et.al in [PTWS05c].

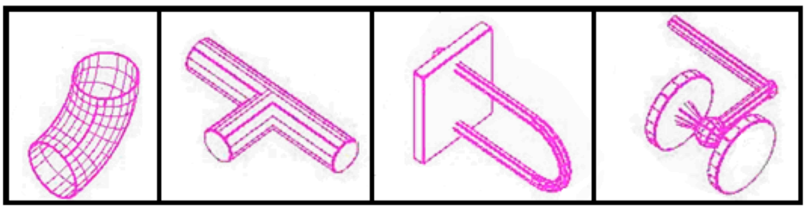


Figure 9.3: Classified parts within the reconstructed catalog. Left to right: elbow, T-joint, Clamp, Valve

Selective part simplification on a Per-Part Basis

Probably the most effective technique of the current implementation is the ability to define rules for applying different optimization techniques on a per-part basis. Using the information of the reconstructed catalog, the export system has control over the export process with a much finer granularity; instead of applying simplifications techniques globally it can now be decided on a per-part basis which technique with which parameters to apply. The decisions are made considering information available through the semantic-triangle [PWTS05]. The logic for exploiting this decision is encoded in a set of semantic rules. Two schematic examples of rules (not following any particular notation) are:

IF part = part.piping.valve.* AND user = engineer THEN REPLACE part WITH valve-symbol

IF part = part.piping.clamp.* AND user = manager THEN DROP part

The CAD reasoner performed geometric similarity operations between the modelled parts and the CAD instances and the 3D CAD.

Conceptual 3D Symbols

Once the catalogue reconstruction is done, semantic information about parts of the model is recovered. In other words, the available implicit information has been made explicit. A technique that exploits this fact is the replacement of complex parts with conceptual 3D symbols that are much faster to render and nevertheless hold the semantic information associated by the user. The conceptual symbols are, so to speak, synonyms for parts. Figure 9.4 shows a part of the model containing an element from the class elbow. In the upper image, the elbow is drawn with full detail (as if it was a normal VR output), in the lower left image, the number of segments is reduced keeping the semantics (to the user it is still an elbow) but reducing visual quality in order to fasten the visualization. In the lower right image, the same elbow is rendered for a non engineer user; the clamps (the pieces that grab the elbow to another sub-structure) are dropped from the model.

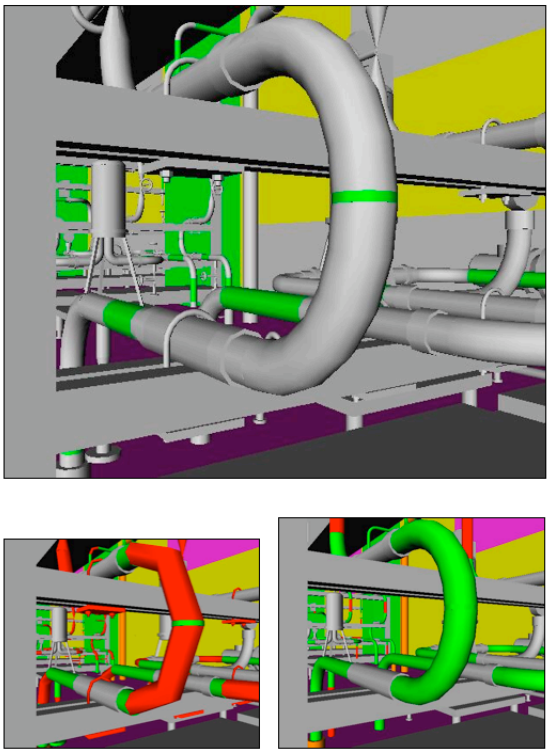


Figure 9.4: Elbow complete (above), reducing the number of segments (lower – left) and dropping holding clamps (lower-right)

The semantics are still kept as the element is still recognizable as an elbow, the sole fact that we are not drawing the clamps save us around 300 triangles per clamp and taking into account that the these clamps do not add information for a non technical user, the overall result is a real triangle saving technique as in an average model the number of clamps could easily exceed one thousand. For this Tool, the ontology reasoner plays a decisive role as it is in charge of the logic related to the correct choice of the semantic symbols. The Model instances and CAD instances are strongly related as the actual CAD model changes upon the ontologies recommendation. The concept of the semantic triangle (user-resources-model) and its implications is explained in our paper [PWTS05]. In Figure 9.5, the set of modules present in the implementation of the tools in both test cases for the Plant design test case are depicted.

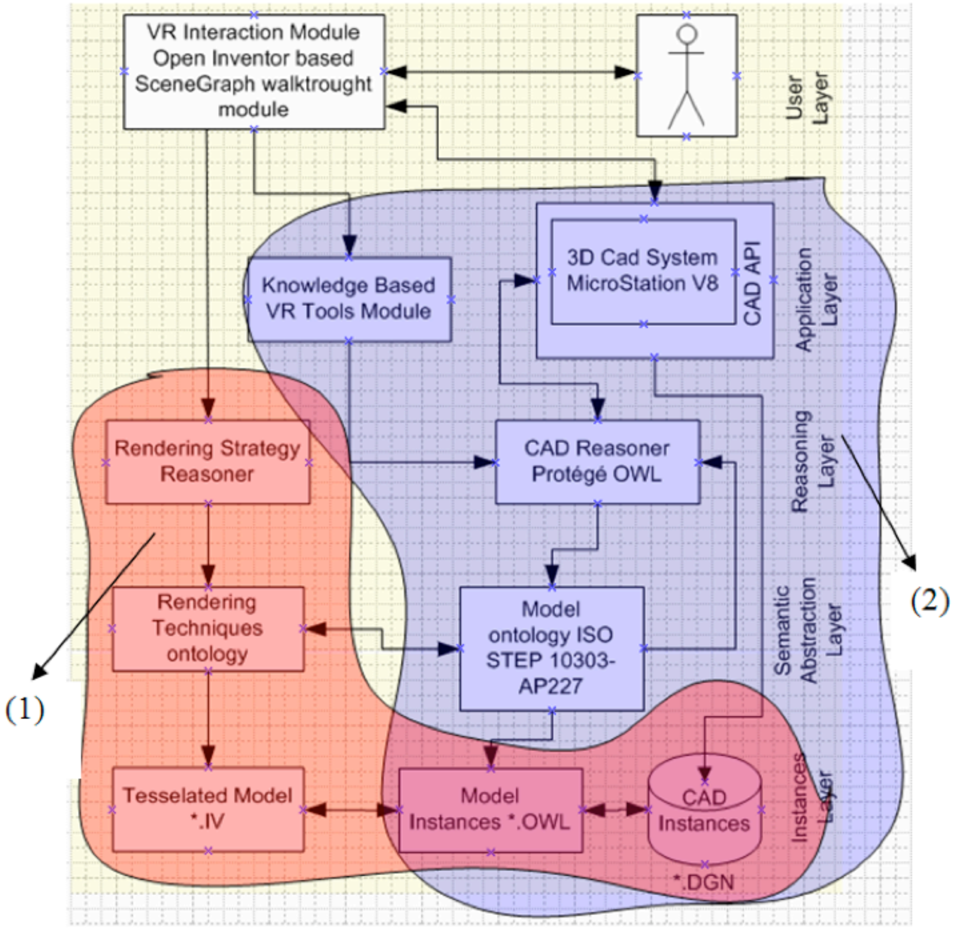


Figure 9.5: (1) Modules needed in the selective part simplification and. (2) Semiautomatic reconstruction

Model simplification and camera dependent synchronization.

In this test case the VET enhancement is a synchronization between the cameras present in the walkthrough and the CAD itself, in a seamless approach to handle the movements in the VR extension, and obtain the same view in the CAD, once the user reaches a desired point of view. In Figure 9.6, the modules of our architecture that allow such outcome are depicted. This VET scans the 3D model and classifies the CAD elements into Domain compliant classes and then simplifies the geometry representation by choosing the appropriate rendering techniques (according to semantic criteria) in order to produce a tessellated model suitable for a enhanced VR output. Most of these structures come from the integrated resources that are common to all ISO-STEP product models.

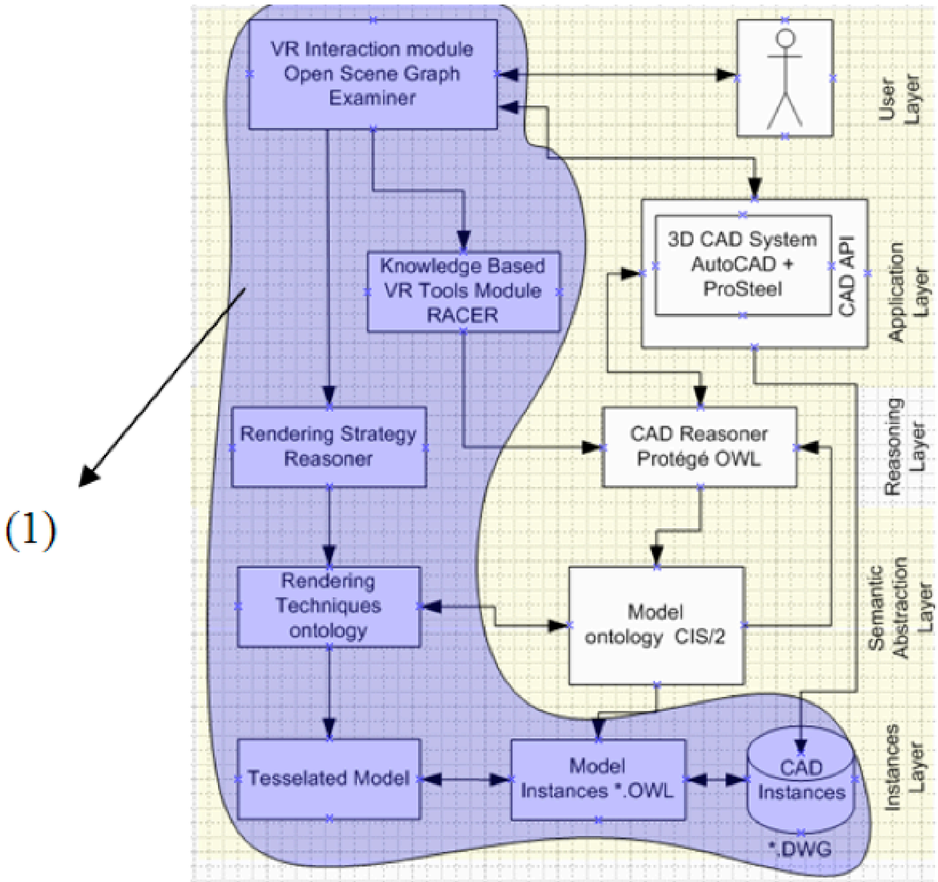


Figure 9.6: (3) Model simplification and camera dependent synchronization tool

This VET allows us to solve a technical problem experimented when a User is working on a massive design and needs to reach a certain part of the model. On a typical CAD, the objects in sight are always displayed with minimal optimization (the CAD is oriented in this case to modeling rather than visualization). When the designer needs to place a new part on the model, the usual approach is to place the 3D model which is for practical reasons displayed as a wireframe, in a position that allows him to attach the new element (e.g. a screw in a steel frame). Depending on the complexity and the available computational resources, some time for re-drawing the model is needed. This time might range from a few seconds up to several minutes for more complex scenarios. The problem arises when for perspective reasons, or because of the wireframe display, the new position of the 3D model does not satisfy the designer's needs. According to our customer, the situation described presented itself often and resulting in a loss of productive design time and a corresponding increase in the design stage cost. As the knowledge in the model is contained in the ontologies, and the branding allows us to characterize the elements contained, we decided to use some of the stored knowledge to provide a solution to this problem by extending the Design Review Walkthrough module with a little interaction with the 3D CAD. The implemented extension allows the simultaneous interaction with the user interface inside the 3D CAD system and with the VR environment returning non-geometric information, which in this case corresponds to the VR camera characteristics (Camera Position, Camera Target, Camera Up Vector, Camera View Angle and Type of Projection). When returning the Camera properties to the 3D CAD, we can assure that the final view will correspond to the user needs. Figure 9.7, shows an example of this enhanced VET; the upper right image depicts a Dome (300 MB file size), the upper left shows the VR visualization in the user-desired position. Such that the position is returned to the 3D CAD, as shown in the lower left image (Wireframe) and lower right (Rendered)

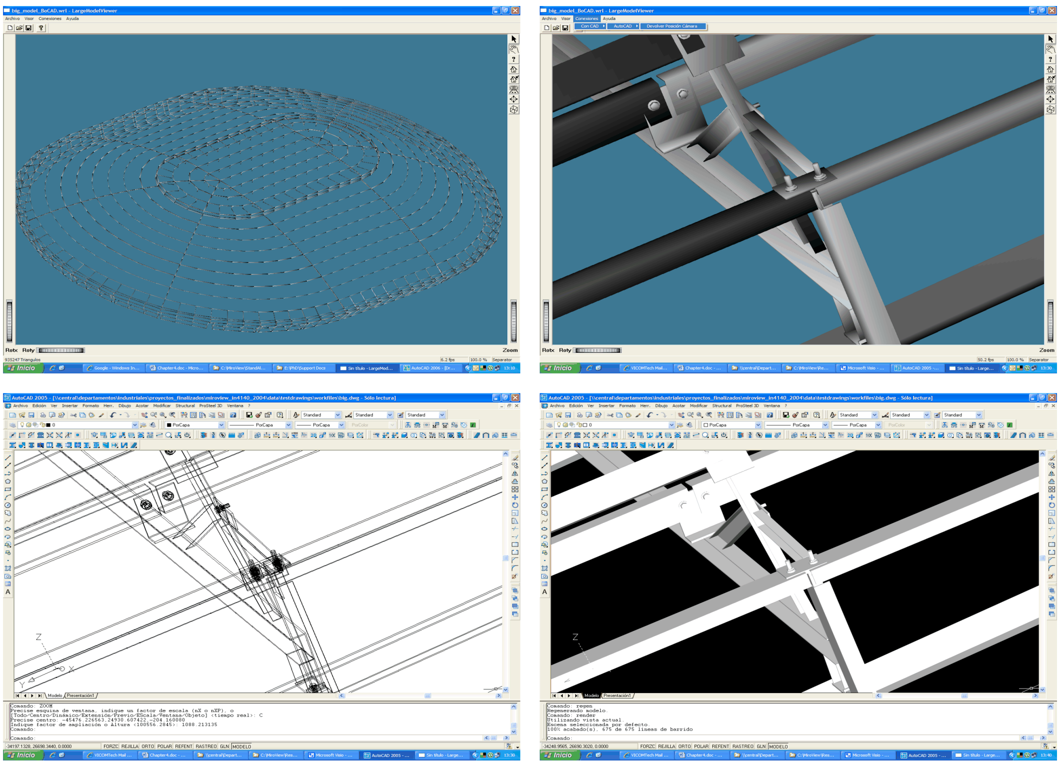


Figure 9.7: Using the KB to support the designer

9.6 Discussion - Semantics Enhancement of LMV VET.

In this Chapter we presented an architecture and a system implementation for the Semantic enhancement of VET used for the purpose of visualising a Large CAD model. By using Semantics as well as spatial information, the system defines which components can be placed, their location, interactions and how the overall design should be configured. The outcome is an easiness of the design process for expert and non-expert users alike. Our architecture implementation allows the semi-automatic recognition, and Semantic simplification of the elements contained in the model. This new approach could lead to an improvement in quality and performance of most current techniques in visualization. Moreover, Semantic-based shape representations and Semantic-oriented VET foresees a new generation of shapes in which Knowledge is explicitly represented and therefore retrieved, processed, shared, and exploited to construct new Knowledge that ultimately enhances the User experience and lowers the price of different stages of the PLC where the visualization of the model is needed. The relation of this chapter's technical contribution with the general architecture for the semantic enhancement of VET presented in chapter 7 can be seen in Figure 9.8.

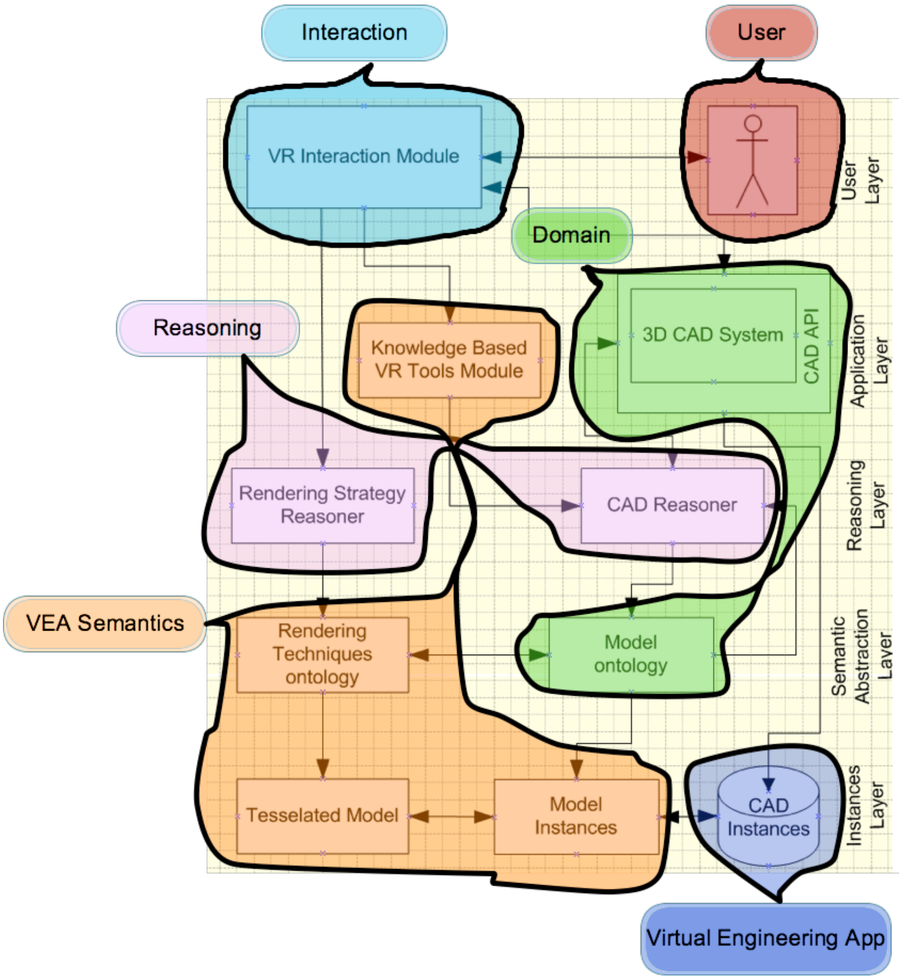


Figure 9.8: Relation of the Semantic LMV architecture and the general architecture

Supporting publications

This Chapter was supported by the following Papers and Journals in International conferences:

- Toro, C. / Posada, J / Oyarzun, J / Falcón, J: Supporting the CAD Structural Design Process with knowledge based tools, Journal of Cybernetics and systems, ISBN 1087-6553 (electronic) 0196-9722 (paper), Taylor and Francis publishers, 2007.

  Remarks: This Journal paper is an extension of the original paper entitled Knowledge Based Tools to Support the Structural Design Process.

- Posada, J. / Wundrak, S. / Toro, C. / Stork, A. : Using Ontologies and STEP standards for the semantic simplification of CAD models in different engineering domains in Selected

Readings in Computer Graphics 2006, Veröffentlichungen aus dem INI-GraphicsNet, Editor(s): Encarnação, J.L.; Fellner, D.W.

Remarks: This journal paper originally appeared in the 2005 Applied Ontology Journal, and it was chosen to participate in the annual best paper award of the INI-Graphics NET, being shortlisted in the 8 best papers of the year from a total of 400 presented papers by an international committee of reviewers

- Posada, J. / Toro, C. / Wundrak, S. / Stork, A.: Ontology Supported Semantic simplification of Large Data Sets of industrial plant CAD models for design review visualization. in Lecture Notes in AI (part of the LNCS/LNAI series) Springer-Verlag, Germany 2005 (also in Proceedings of KES05 (Knowledge-Based & Intelligent Information & Engineering Systems), Melbourne, Australia, September 2005)

  Remarks: This journal paper presented an initial architecture for the embedding of semantics in a Computer Aided Design environment.

- Posada, J. / Wundrak, S. / Toro, C. / Stork, A.: Semantically Controlled LMV Techniques for Plant Design Review, In Proceedings of ASME DECT/CIE 05, Salt Lake City, USA, 2004.

  Remarks: This paper addressed the initial ideas on the inclusion of semantics for the Large Model Visualization (LMV) problem that was one of the main focuses of the MiroWalk project.

Chapter 10

Contributions to Industrial Maintenance from a Knowledge Based perspective

In this Chapter we propose an architecture and a system implementation for the enhancement of the Industrial Maintenance management using Knowledge Based techniques. As an example, we benefited from the User Experience modeling (following our own recommendation on chapter 4) that allow us to consider User Experience, a feature that is not typically taken directly into account for the Industrial Maintenance tasks, at times the Industrial Maintenance databases consider some statistical measurements for a given element lifespan, but the reasons why the element was changed before the end of the cycle are not stored anywhere. We believe that these factors are as important as the fact that the element was changed (or maintained).

10.1 Description of the technical problem and challenges identified

Amongst the different stages of the PLM, one of utmost interest is Maintenance. This phase covers the perpetuation (or at least extension) of the product's life span through a series of actions, some of them predictive, some of them reactive, commonly known as Industrial Maintenance (IM). IM can be defined as the combination of all technical and administrative actions, including supervision actions, intended to retain an entity in, or restore it to, a state in which it can perform a required function [ELE08]. According to Honkanen [Hon04], machines have to be maintained in order to increase reliability and thereby avoid production disturbances. It is, therefore, assumed that the purpose of a single maintenance action is to increase reliability. In Honkanen's work [Hon04], industrial maintenance activities are studied from a systemic point of view, considering maintenance as an information processing system. There are several techniques and theories that can be applied to the Industrial Maintenance field. An explanation of such techniques in depth is outside the scope of this thesis, however, the most common approaches in IM are the following:

- **PM - Preventive maintenance:**

  According to the US Federal Standard 1037C and the US Department of Defense Dictionary of Military and Associated Terms, PM is defined as"Any activity, such as tests, measurements, replacements, adjustments and repairs, intended to restore or retain a functional unit in a specified state in which the unit can perform its required functions" [Mai], such activity involves:

  - The care and servicing, for the purposes of maintaining equipment and facilities in a satisfactory operating condition. Such care is achieved by providing support for systematic inspection, detection, and correction of incipient failures either before they occur or before they develop into major defects.
  - The actions, including tests, measurements, adjustments, and parts replacement, performed specifically to prevent faults from occurring.

- **PdM - Predictive maintenance:**

  PdM is a collection of techniques for the analysis and determination of the condition of in-service equipment in order to predict when maintenance should be performed. PdM is based on the fact that most machine components warn before failure. To detect the symptoms that are triggering a machines warning alarms, requires several types of non-destructive testing, such as oil analysis, wear particle analysis, vibration analysis, and temperature measurements. Use of these techniques to determine the machine condition results in a more efficient use of maintenance effort compared to any earlier types of maintenance. Predictive maintenance allows plant management to control the machinery and maintenance programs rather than vice versa. In a plant using predictive maintenance, the overall condition of machinery at any time is known, and more accurate planning is possible. This approach offers cost savings over routine or time-based preventive maintenance, because tasks are performed only when warranted. In [CL07] some standard examples of wear processes and lifetime distributions are obtained and the cost consequences of certain maintenance schemes are investigated.

- **RCM - Reliability Centered Maintenance:**

  RCM is an engineering framework that enables the definition of a complete maintenance regime. It regards maintenance as the means to maintain the functions that a user may require of the machinery in a defined operating context. RCM was introduced as a maintenance technique by Matteson, Stanley and Heap while working for United Airlines, and then published in its generalized form as a report to the US Department of Defense who sponsored the authoring of both a textbook and the evaluation report itself in 1978 [MNH78]. RCM describes the processes used to determine the optimum maintenance requirements emphasizing the use of Predictive Maintenance (PdM) techniques in addition to traditional preventive measures.

From an application point of view, many different research initiatives have been presented by the scientific community, ranging from seminal work presented in 1929 by Wilson [Wil29] and up to IM techniques (predictive, TPM, etc) [Nak91]and the use of modern computer-based technologies such as AR/VR [WKS03]. To our knowledge, most of these approaches however, miss out on the potential of using domain specific knowledge-based theories that might enhance the user's experience. This user, in our case, is the maintenance worker in a typical industrial facility whose special needs include mobility, fast response and immediate access to the relevant data, like specifications, historical records, etc. We show in this VET that the use of Semantics and AR techniques provide additional support to maintenance tasks, by improving the user-understanding of the elements under maintenance. The enhancement is realized when the knowledge and the user experience related to the maintenance system is embedded in the AR environment as an important aid for the user, providing him with a sense of immersion. Some of the identified challenges in IM are:

Challenge 1: Domain modeling:

Since IM takes place in many differing environments and for diverse elements, the modeling of the domain represents a difficult task to accomplish. The set of actions needed to perform IM in a Chemical Plant and the actions needed in a Nuclear Plant, although similar, are at the end different (the later requires even more careful inspection). For the this reason, we believe that the environment itself should provide the information for IM. Moreover, the domain modeling must be consistent and easily adaptable and extensible for new requirements (reasons that strengthen the use of ontology modeling using available maintenance databases as data providers). To our acquaintance, there is no agreement reached on the modeling of the IM domain, specially taking into account Ambient Intelligence scenarios to model Knowledge Bases for industrial Plant maintenance. in our approach we use Ambient Intelligence modeling for our Knowledge Base.

Challenge 2: User immersion and use of portable devices:

This challenge has been previously addressed by various R&D projects. There have been difficulties reported with RFID tags and further difficulties in employing optical devices. The former relate to interference and the need for close proximity to a tag, and the later due to certain characteristics of the environment where the IM is taking place (darkness, contrast, etc). We do not intend to solve any of these issues; instead we concentrate on a simple VR/AR output that can be easily and cheaply integrated into any facility. We deem the benefits of portable devices in this specific

VET. For this reason we intentionally included their use within our architecture. However we believe strongly that some technical problems must be solved in order to make their use beneficial.

Challenge 3: User expertise:

This challenge is related to the fact that the user experience is not typically taken directly into account with the IM tasks. At times the IM databases consider some statistical measurements for a given element lifespan, but the reasons why the element was changed before the end of the cycle are not stored anywhere. We believe that these factors are as important as the fact that the element was changed (or maintained). The possibility to store and handle User expertise is therefore desirable.

10.2 Brief description of the contributions

Considering the aforementioned challenges, we propose an architecture and a system implementation for the enhancement of the Industrial Maintenance management using Knowledge Based techniques that has the following characteristics:

- It is centered in Plant Management and Industrial Maintenance Domain.
- It uses the Standard Ontology for Ubiquitous and Pervasive Applications (SOUPA) [CPFJ04] for the Ambient Intelligence modeling and the Set of Experience Knowledge Structure (SOEKS) [SST07]as the underlying Knowledge Structure for the User Experience modeling.
- The Enhanced VET presented here, can be used in the Operation and Maintenance stages of the PLM (see Figure 10.1).

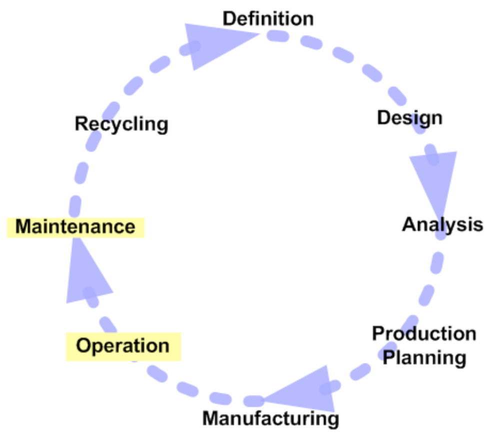


Figure 10.1: Placement of the Semantic Enhanced VET in the PLM

- We present an architecture that we call UDKE, which uses Augmented Reality technologies in order to provide a better user interaction and is designed with portable devices in mind.
- Here the main contribution is the use of Knowledge Structure in the domain of Industrial Maintenance that combines Augmented Reality, portable devices and User Experiences modeling that allows a better understanding of the maintenance events.

10.3 Use of visualization and inmersion tools in IM

10.3.1 Virtual Reality and Augmented Reality in the IM context

Virtual Reality (VR) is a virtual environment where the user is immersed in a virtual world in which the elements have the physical resemblance to objects possibly belonging to the real world. Milgram and Kishino created a Virtuality Continuum [MK94] to classify mixed worlds, where we find Augmented Reality (AR) as a system that, according to Azuma [Azu97], combines real and virtual objects in a real environment enhanced with computer generated graphics, running interactively, in real time, and aligns real and virtual objects with each other. In this work we use Augmented Reality in order to enhance the user's experience/know-how (maintenance engineer) by giving him contextualized information extracted from an ontology-reasoning kernel. Augmented Reality in an industrial maintenance environment provides, for example, user experience enhancements like viewing information related to the object in sight that has been matched with the marker; this information can be extracted from a knowledge system or web page and is contained or handled by the ontologies. Moreover, the handling and use, in a graphical manner, of the information contained in manuals and operation guides is more natural. With such ideas in mind, the German project ARVIKA [WKS03] aimed its research at the use of some advanced AR techniques (e.g. marker-less tracking) within different application scenarios (one of them being IM). All of the initiatives presented make an outstanding use of the different VR/AR technologies; we believe that User Experience modeling and Semantics can be beneficial as an enhancement to such approaches.

10.3.2 Ubiquitous Computing and Ambient Intelligence

Ubiquitous Computing is a model of human-computer interaction in which information processing is integrated into everyday objects and activities. In opposition to the desktop paradigm, in which a single user consciously engages a single device for a specialized purpose, a user embedded in a ubiquitous computing environment engages many computational devices and systems simultaneously and sometimes without full knowledge about their functionalities. In Adam Greenfield's book "Everyware: The Dawning Age of Ubiquitous Computing" [Gre06], the interaction paradigm of ubiquitous computing as "information processing dissolving in behavior" is described. This computing paradigm makes extensive use of sensors and RFID (Radio Frequency IDentification) tags. Ambient Intelligence ([Lug06], [WRA05]) refers to environments that are sensitive and responsive to the presence of people. Ambient Intelligence is a vision on the future of consumer electronics, telecommunications and computing that was originally developed in the late 1990s for the time frame 2010–2020. In an Ambient Intelligence world, devices work in concert to support people in carrying out their everyday activities, tasks and rituals in easy, natural way using information and intelligence that is hidden in the network connecting these devices. As these devices grow smaller, more connected and more integrated into our environment, the technology disappears into our surroundings until only the user interface remains perceivable by the users.

The Ambient Intelligence paradigm builds upon Ubiquitous Computing and human-centric computer interaction design and is characterized by systems and technologies that are [MA03]:

- Embedded: many networked devices are integrated into the environment
- Context aware: these devices can recognize the user and your situational context
- Personalized: they can be tailored to the user needs
- Adaptive: they can change in response to the user
- Anticipatory: they can anticipate your desires/needs without your conscious mediation.

The concept is strongly related with the long-term vision of an intelligent service system whenever the technologies are able to automate the platform, embedding the required devices for powering context aware, personalized, adaptive and anticipatory services. An interesting application of these technologies was presented by Ishii [IWB+98], in this work, a design of what they call an "ambientROOM" was discussed. The former is an interface to information for processing in the background of awareness, which integrates Ambient Media and Architectural Spaces. There are different approaches for ontology modeling of ambient intelligence events, some of the most relevant includes the work presented by Preuveneers et.al. [PVdBW+04] and Chen et.al. [CPFJ04], this being the one we used for our model as a result of a design requirement in the project in which our contribution was developed.

10.3.3 SOUPA (Standard Ontology for Ubiquitous and Pervasive Applications)

SOUPA [CPFJ04] is a shared ontology expressed using OWL, designed to model and support pervasive computing applications. It consists of two distinctive but related sets of ontologies, called SOUPA Core and SOUPA Extension as can be seen in Figure 10.2. The set of the SOUPA Core ontologies attempts to define generic vocabularies that are universal for different pervasive computing applications, vocabularies for expressing concepts associated with person, agent, belief-desire-intention (BDI), action, policy, time, space and event. The set of SOUPA Extension ontologies define additional vocabularies for supporting specific types of applications and provide examples for future ontology extensions. The Core and Extension of SOUPA have been applied in several research projects like the Context Broker Architecture (CoBrA) [CTA03], this is an agent-based architecture for supporting context-aware systems in smart spaces. Central to this architecture is an intelligent agent called context broker (majordomo) that maintains a shared model of context on behalf of a community of agents, services, and devices in the space while providing privacy protections for the users. We will show in this chapter an extension to SOUPA based on the CoBrA architecture that implements a context aware (ubiquitous) system that helps a user in an Industrial Maintenance environment.

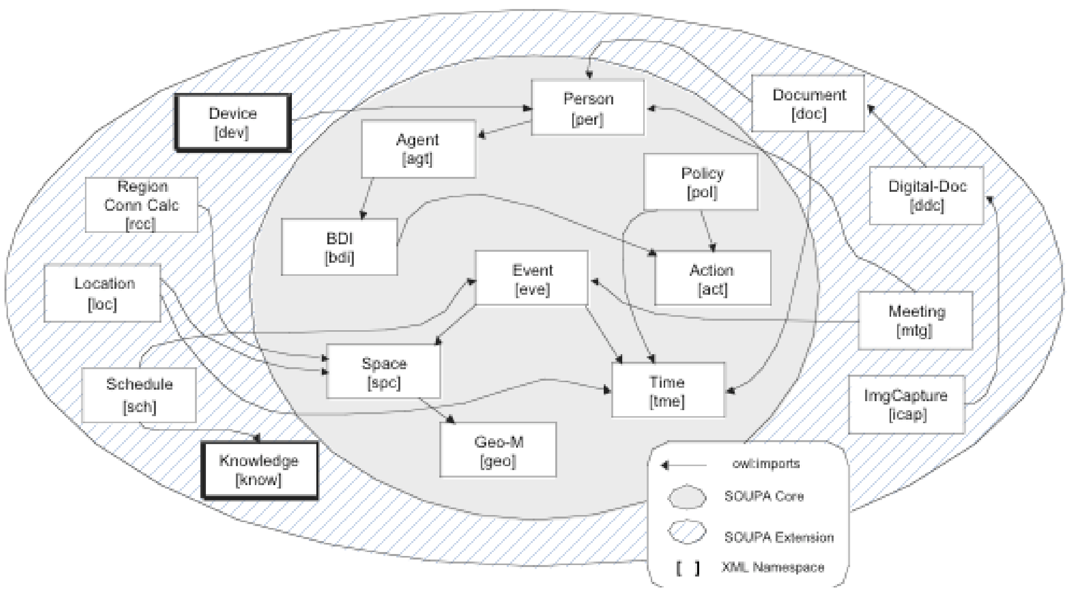


Figure 10.2: The SOUPA group of ontologies (Core + Extension)

10.4 An approach to support IM task with Semantics in an Ambient Intelligence scenario

10.4.1 Modeling an ontology for the SOEKS

SOEKS was introduced in Chapter 4 of this thesis as a technique that allows the modeling of user experience. In the case of IM, the stored sets of experience are produced by past decisions made by maintainers with the same or similar elements. E.g. the user is checking during her/his duties a butterfly valve. Since it is a fluid-restricting element, possible decisions taken in the past on such valve, or a similar element (ball valve), could be suggested. An OWL ontology for the SOEKS was presented by us in [SST07]. In order to obtain our ontology, we started from the XML set of experience model presented by Sanin and Szczerbicki ([SS05b], [SS06a]), where they established an initial shareable model for SOEKS. For every first level tag of the XML-SOEKS a concrete class of the ontology is created (role concrete), that is, variables, functions, constraints, and rules, among others. For second level tags, a slot with the proper cardinality and data type is created. A tag from the XML version of set of experience knowledge structure can be seen in Figures (10.3,10.4,10.5,10.6)

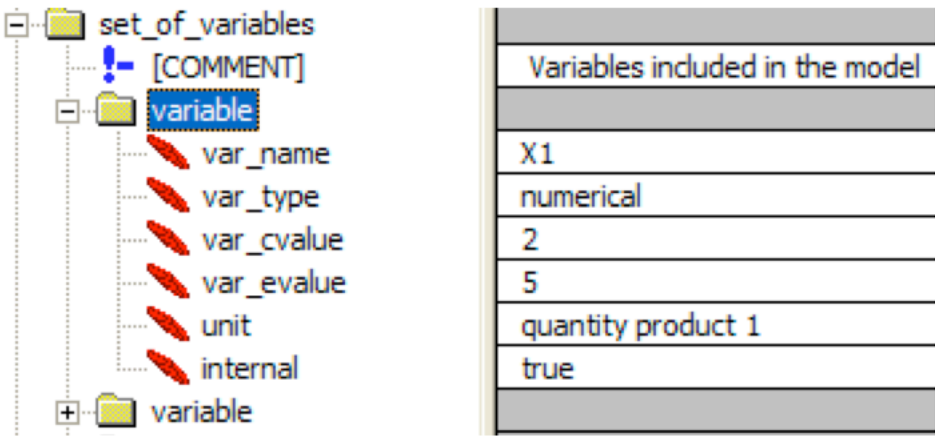


Figure 10.3: Variable in the XML version of the Set of experience Knowledge Structure

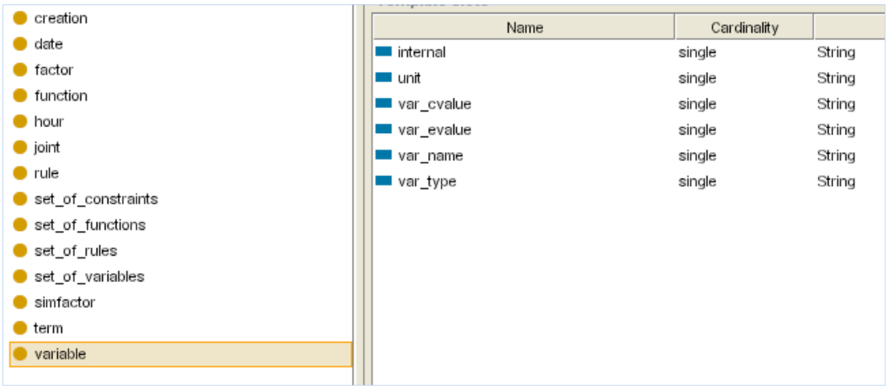


Figure 10.4: Tag Variable in the Ontology version of the Set of experience Knowledge Structure

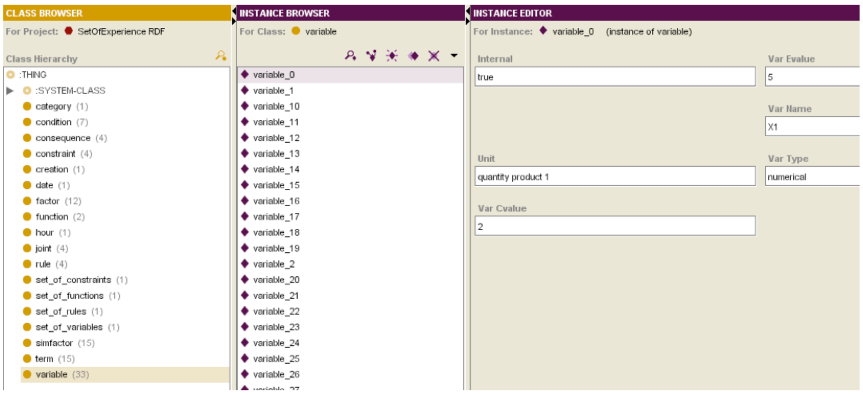


Figure 10.5: Tag Variable instanced in the Ontology version of the Set of experience Knowledge

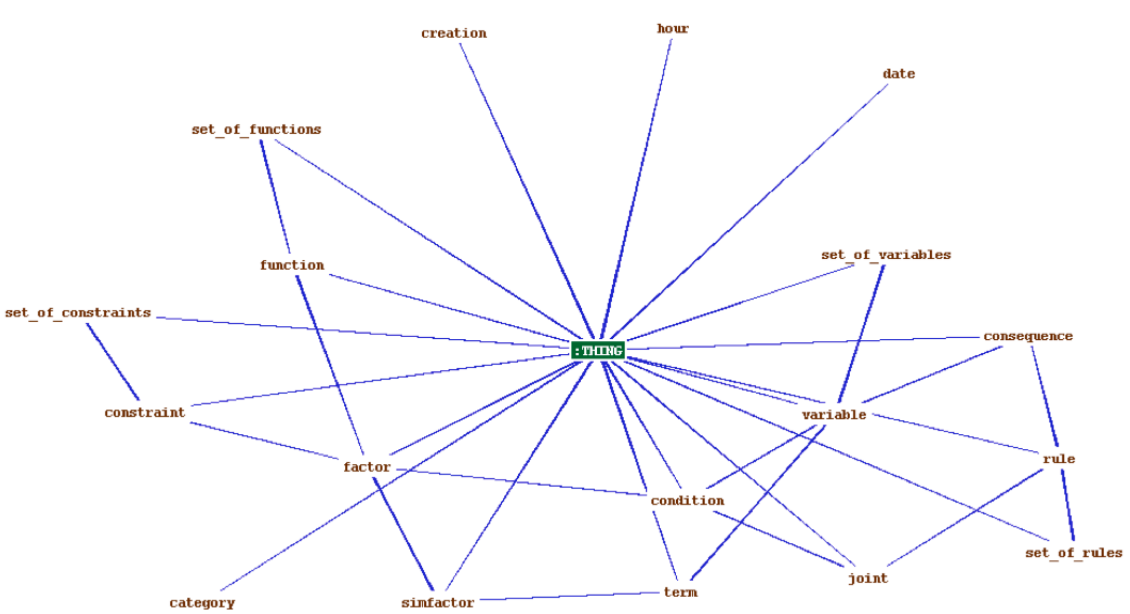


Figure 10.6: Graphical view using Protégé OWLVIZ plugin

Our developed ontology was added to the SOUPA set of ontologies, with the purpose of extending its functionality in a way that formal decision events can be suggested to the Maintenance User during his/her work, as it will be discussed later.

10.4.2 Supporting IM tasks using Knowledge Based techniques - The UDKE Architecture.

In this section, we describe our architecture for supporting IM tasks with Knowledge Based techniques. We call our architecture UDKE (User, Device, Knowledge and Experience). Such architecture was introduced in our paper "Knowledge based Industrial Maintenance using portable devices and Augmented Reality" [TSV+07] which will be extended in this part of the thesis. Our approach takes into consideration the four challenges found in the IM supporting problem that we discussed in the introduction of this chapter . It can be said that UDKE provides a possible conceptual model for a Maintenance System that combines Knowledge, User experience and AR/VR techniques and with the capability of running in portable devices like PDA's or sub-laptops, and in general in a vast majority of devices with Java support. We acknowledge that our approach is not new in the use of advanced VR/AR techniques for IM. As discussed earlier, projects like ARVIKA [WKS03] utilize more advanced techniques in such fields; however, we believe that our approach can be used to enhance the aforementioned efforts with Knowledge and User Expertise.

The novelty of our contribution relies on:

- The extensible nature of our architecture (based on available technologies).
- The User-centered approach (User experience modeling, user needs, etc).
- The use of Ambient Intelligence concepts to model a Knowledge Base (SOUPA).

As can be seen in Figure 10.7, UDKE architecture is divided into four layers, each one describing a fundamental view from which the Maintenance task accomplishment could may acquire an advantage.

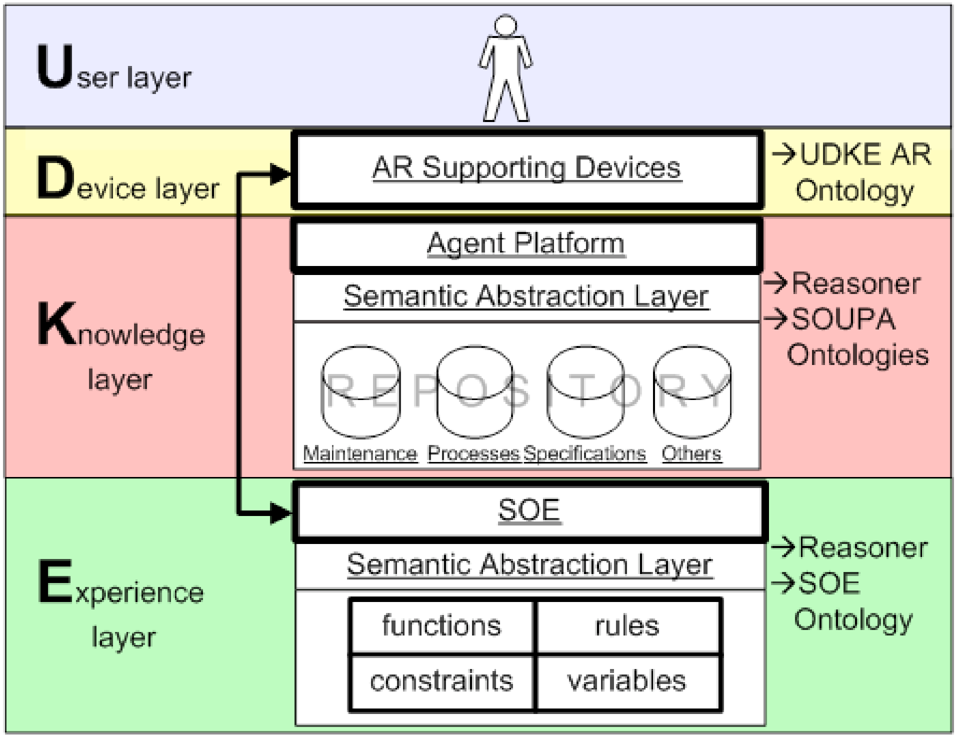


Figure 10.7: The UDKE architecture

The User Layer

This layer contains the user modeling; it handles the different user profiles and maintenance sessions. The different sessions are modeled in order to have the means for monitoring different decisions that are common in maintenance duties, for example an expert maintainer, decides to replace a fire nozzle before reaching the end of its life cycle. In such case, that maintenance action obeys to expertise factors rather than operation manuals. Such a decision has inherent consequences, for example, the fire nozzle records have to be updated and the replacement nozzles checked by the corresponding authorities (for correct sealing verification). The user in our architecture is fundamental in the sense that the rest of the architecture, reacts to the modeled characteristics of this layer, providing reactions for "depending-to" questions. In this case the user modeling, was based on the ontology we had for the LMV problem discussed previously in this thesis with some minor enhancements as to adapt it to the Maintenance field. When using available resources (such as already available ontologies for example); we found that our development was accelerated thanks to the inherent properties and simplicity of ontology modeling. For the User Layer modeling, we used the different recommendations presented in Chapter 4. Specifically the User Experience modeling and the Roles differentiation. Such recommendations allowed us to tighten our User Layer model.

The Device Layer

The Device layer contains the modeling of the different devices used to capture the environment (e.g. a camera in a portable device). For this layer we decided to model a new ontology that we call the UDKE-AR, as an extension to SOUPA because it was not directly available on the standard knowledge base. Our extension to SOUPA is depicted in Figure 10.8, derived from the SOUPA class: **class dev:device**).

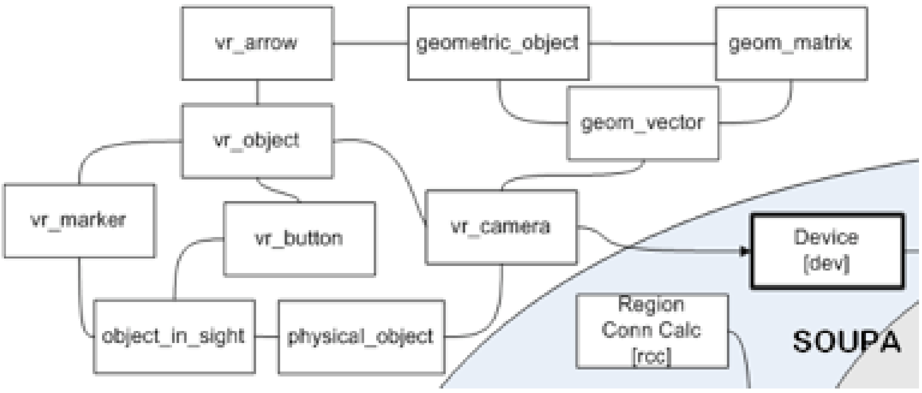


Figure 10.8: The UDKE AR extension to SOUPA

The main characteristics we modeled in the device layer, were related to VR actions or concepts e.g. VR_Marker which stores the characteristics of a Virtual Reality marker such as the ones used in the AR_Toolkit library [KBBM99] depicted in Figure 10.9 along with the concept. The aforementioned library was used in our implementation. The device Layer also stored the different user interactions that come from the User Goal and Sub-goal divisions as explained in Chapter 4.

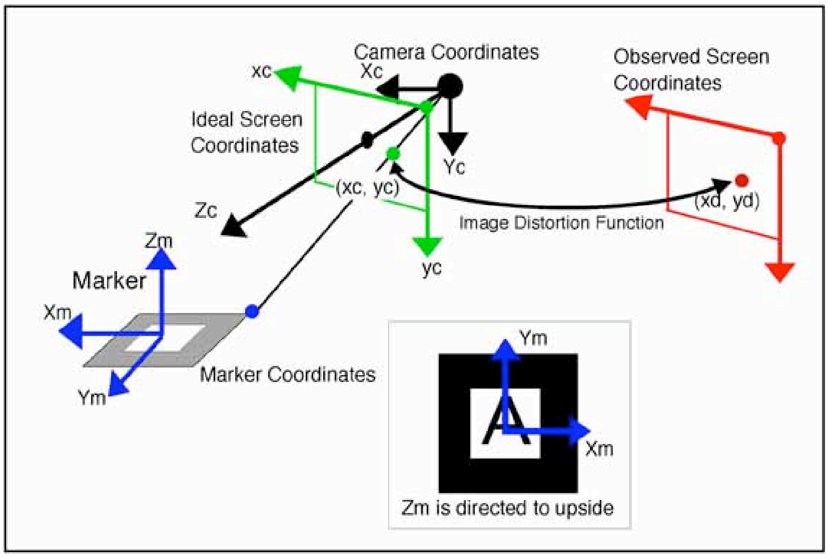


Figure 10.9: AR Toolkit concept and some of the characteristics modeled in the Device Layer.

The Knowledge Layer

The Knowledge layer contains an agent platform whose main objective is to interact via majordomo messages with the Semantic abstraction engine. The majordomo is contained in an Agent Platform in a similar way to the approach presented by O'Hare et.al. [ODC04]. This layer contains the SOUPA Core and the SOUPA Extension ontologies as well as a Reasoner system that is responsible for performing the Semantic queries. The ontologies feed their instances from different data repositories relevant to the maintenance domain (historical data, programmed stops, cycles, etc).

The Experience Layer

In the Experience Layer, the SOEKS is enclosed as an OWL ontology with different data bases that feed the reasoning system with functions, rules, constrains and variables used to specify new decisional events or even to contain past decisional events taken over similar elements (where similar refers to the object in sight element or in other words the element in which the maintenance engineer is considering at a given moment). In Figure 10.10, we show the extension place of the SOUPA set of ontologies in which we derived the Set of Experience Knowledge Structure ontology, the extension class is **know:Knowledge**.

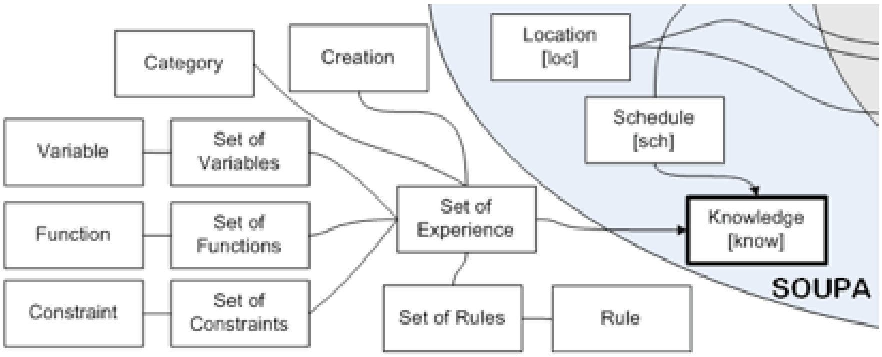


Figure 10.10: Extension of SOUPA with SOEKS

10.5 Implementation example

In our scenario, the user during his maintenance patrol uses a portable device (PDA, UMPC or Tablet PC) with a camera connected. For every object to be maintained there exists a VR marker (following the sensor concept in ambient intelligence). Every marker is an unequivocal gray-scale pattern that can be easily printed on white paper by a regular PC printer. When the camera recognizes a marker, a matching element to be maintained is identified according to the context (user, task, priority) and a set of information is extracted from the repositories. The output video stream of the camera is mixed with 3D objects and other relevant information and is displayed to the user in his portable device screen. As can be seen in Figure 10.11, the user is in front of an element (in this case a fire extinguisher) and when the system recognizes the matching marker, the user receives on the device's display, information such as the name of the element, the next programmed change, the maintenance procedure etc. All information is obtained from the repositories in the Knowledge layer and is maintained by the Experience layer.

Figure 10.11: AR Enhanced user view (UDKE Platform)

10.5.1 Implementation issues

The system was tested using different portable devices, our implementation uses JAVA as the core language for the prototype implementation. The AR engine used was the JAR Toolkit library [GRSP02]. All the ontology modeling was done in Protégé and the API used was the Protégé OWL API. The agent platform used in our implementation was JADE [BCG07], and for reasoning over the ontologies we used RACER. When a marker is detected, the system calculates the matrices necessary to place the augmented information via JARToolkit calls. Following the application flow, the Agent platform begins its work starting a majordomo service whose main function is to serve as an intermediary between the user and the rest of the architecture. The majordomo handles the events in the knowledge layer databases through reasoning over the SOUPA ontologies. The majordomo also handles the Experience layer through reasoning over the SOEKS ontology in order to obtain knowledge from past experiences or similar devices being maintained. Once all of the information is obtained/inferred and possible experiences are acquired from the SOEKS using the reasoning system, a final step is performed by returning the information to the device (UMPC, Pocket PC, etc) and displayed (streamed) in its graphical output.

10.6 Discussion - A new paradigm for IM

In this Chapter we presented a framework and a system implementation for the exploitation of embedded knowledge in the Domain of industrial maintenance in a mobile context, using Augmented Reality techniques. We based this Semantically enhanced VEA in the SOUPA group of ontologies (Standard Ontology for Ubiquitous and Pervasive Applications). Our approach extended SOUPA with two new ontologies (i) the Set of Experience Knowledge Structure, used to model the user's experience and (ii) the AR ontology which models an Augmented Reality environment that is used to enhance the maintenance experience through virtual elements. As test case, we implemented our approach in different portable devices with video input capabilities such as UMPCs, PDAs and Tablet PCs. Figure 10.12 depicts the relation of the UDKE architecture presented in this Chapter with the general architecture discussed in Chapter 7.

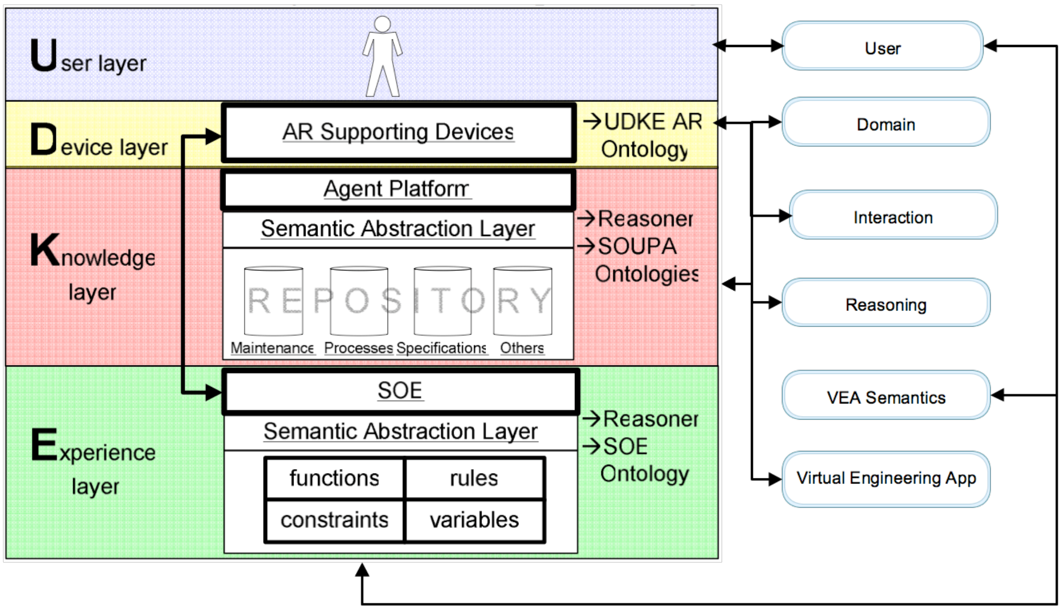


Figure 10.12: UDKE-General Architecture relation

The benefits of using a Semantically enhanced VET for Industrial Maintenance tasks are evident, however we believe that some challenges are still open. One of such challenges is that the discussed approach depends on the full capability of viewing all the regions on the marker (for identification purposes and calculation of the camera parameters). The problem arises when such marker cannot be seen partially or in total, as the information that relates such marker to the actual Maintenance Datasets and hence with the Domain embedded Knowledge will not be reached. Such problem is due to poor light quality or deterioration of the marker and even because of the quality of the camera's CMOS sensor. We believe that a logical next step in our methodology would be to consider marker-less approaches such as projects like ARVIKA [WKS03] or in the work presented by Comport [CMC03].

Supporting publications

This Chapter was supported by the following Papers and Journals in International conferences:

- Toro, C. / Sanín, C. /Vaquero, J./ Posada, J./ Szczerbicki, E. (2007). Knowledge Based Industrial Maintenance Using Portable Devices and Augmented Reality. Knowledge-Based Intelligent Information and Engineering Systems, (Apolloni, B., et al., Eds.), Vol. 4692, pp. 295–302. Springer, Heidelberg, Vietri Sul Mare, Italy.

  Remarks: This work addresses a Virtual Engineering Application intended to the enhancement of the industrial maintenance area by the use of semantics.

- Sanin, C. / Toro, C. / Vaquero, J. / Szczerbicki, E. / Posada, J. : Implementing Decisional DNA in Industrial Maintenance by a knowledge SOUPA Extension in Systems Science, Vol. 33 No. 2: 61-68 - ISSN: 0137-1223, Wroclaw, Poland 2008.

  Remarks: This journal paper focused on the implementation issues of the Decisional DNA in an Industrial Maintenance application.

Chapter 11

Conclusions and main contributions

11.1 Research question and objectives accomplishment

Semantic technologies can indeed improve the current state of the art of Virtual Engineering Applications from diverse perspectives. In particular in this thesis we proposed a theoretical framework with specific recommendations, methodologies and practical tools that allows the integration of Semantics in VEA trough the Semantic enhancement of their embedded VET. In order to present our contribution, we started by reviewing the most relevant concepts related to the state of the art in Semantic modeling and some related technologies that could be useful in the enhancement of Virtual Engineering Applications trough Semantics. We continued our presentation by reviewing the most important concepts related to Product Life Cycle and Virtual Engineering was introduced. Following the objectives proposed in the first chapter, we presented a series of considerations for User Modeling, a methodology for Domain modeling using Engineering Standards and a novel technique for the fast retrieval of information from a Knowledge Base model. The integration of the aforementioned recommendations, allowed us to introduce an original and generic methodology for the Semantic enhancement of Virtual Engineering Applications. The stated architecture and recommendations were used for the technical contributions which included (i) a technique that uses the advantages of Semantic modeling technologies for the automatic generation of Graphic User Interfaces, (ii) a technique that leverages the advantages of Semantic technologies in the Large Model Visualization domain and (iii) an architecture and a system implementation for the enhancement of the Industrial Maintenance management using Knowledge Based techniques.

11.2 General conclusions

The main contribution of this work is the presentation of different methodologies and recommendations that allow the Semantic enhancement of Virtual Engineering Applications (VEA) through the Semantic enhancement of their contained Virtual Engineering Tools (VET). These methodologies take into consideration User modeling recommendations and Domain modeling based on Engineering Standards (CIS/2, ISO-STEP 10303-AP227, etc) for the purpose of providing not only the best VET for the case, but to enhance the User interaction through Semantic based generated graphical user interfaces and the Semantic enhancement of the VET themselves. This work combines Virtual Engineering Applications (e.g. Computer Aided Design, Steel Detailing, etc.), and state of the art technologies (e.g. Virtual Reality, Augmented Reality, etc) with Semantic techniques in a Product Life Cycle context. Our contributions are divided in two conceptual sets: (i) Methodological recommendations and (ii) Technical contributions.

In the Methodological part we presented the following recommendations:

- Recommendations on User Modeling
- Recommendations on Domain modeling based on Engineering Standards
- The Reflexive Ontologies: A methodology for the enhancement of the ontology query process
- An architecture for the Semantic Enhancement of Virtual Engineering Applications

In the Technical part we presented the following contributions:

- Contributions to the Semantic generation Graphical User Interfaces for CAD
- Contributions to the Semantic simplification of geometries for Large Scale Visualization of Industrial Plants and Design Review
- Contributions to Industrial Maintenance from a Knowledge Based perspective

11.2.1 Methodological recommendations overview

In the Methodological part, we introduced a series of recommendations for User Modeling (UM). Such recommendations were not intended to be used as a methodology, instead we presented them as a set of concepts that could be taken into account when the UM task is fulfilled. A good User modeling is the direct result of better User considerations. However, such considerations must be taken into account with a Domain perspective in mind. For the aforementioned reason, we introduced a series of recommendations and a methodology for using Engineering Standards as models for Knowledge bases. Such methodology was followed in the technical Chapters 8 and 9 where we used CIS/2 and ISO-STEP with outstanding results. The use of Engineering Standards was found very useful when it comes to the typical problems of branding elements (e.g a CAD 3D object) as we have found that even the final users no matter their expertise on the field, tend to “call” the elements using many different names, not to mention how the create and describe such elements. By using Engineering Standards as bases for our Knowledge Bases, we used the meta-language provided by the Engineering Standards in order to have a common vocabulary for the elements being described. When the use of Engineering Standards was applied in the case study presented in Chapter 8 (related to GUI) we had a result which was a chance finding, in that the CAD-VET tool used for our test offered import/export capabilities to the CIS/2 (our chosen Engineering Standards for the case study). Such capabilities, when treated as the VET they are, allowed us to quickly map element information to the Domain ontologies, saving us time as we did not required any ontology mapping tool. Given the fact that our Knowledge Bases are modeled using the ontology paradigm in order to take advantage of their benefits (described in Chapter 2), we found the needed of a better query process. For such reason, we introduced the Reflexive Ontologies (RO) as a framework that can be used to enhance any existing ontology with a set of self-contained queries over instances in the domain of study. A novel modular architecture has been proposed which allows the implementation of the different conceptual steps in the methodological part as connected modules. Such architecture was used as the base for some of the implementations shown in the Technical contributions.

11.2.2 Technical Contributions overview.

In the technical part, we presented a set of contributions that used the recommendations introduced in the methodological sections. Firstly we introduced a methodology to the Semantic generation Graphical User Interfaces for CAD in the relevant chapter, we presented the advantages of Semantic modeling technologies in order to support the process described. We took into consideration the User and his goals, the stage of the PLC where the interaction is taking place and the User intentions. Such intentions (considering User Roles) where the base for the contributions to the Large Model Visualization (LMV) problem commented in Chapter 9. The contributions for the enhancement of the visualization and VR capabilities in generic CAD applications through semantics, we were able to improve a VET using an ontology based approach were the Engineering Standards were decisive for the Domain modeling. Such Domain consideration was the key feature when we presented our contributions to Industrial Maintenance from a Knowledge Based perspective where we targeted some detected challenges in the Industrial Maintenance field. We proposed an architecture called UDKE that was basically an implementation of the general architecture presented in chapter 7 for the semantic the enhancement of the Industrial Maintenance management process. We used User Experience modeling (following our own recommendation on chapter 4) that allowed us to use past Experiences for tightening decisional events in the Domain.

11.3 Future Work and Open Challenges.

In this section we will discuss some remaining open challenges that we consider important for a future treatment in the scope of the Semantic Enhancement of VEA. When we presented our User modeling recommendations, we found that all User modeling methodologies are based in the fact that the user characterization is possible. But what if such characterization is not possible with the means at hand? Such question provides an interesting point for a future work implying Semantic technologies. Facts like proving without doubt how tight a UM supports the Design specifications are also yet to be considered. User Knowledge gathering is still not addressed from a Semantic Point of view. SOEKS is able to model it, but the Knowledge gathering process could be somehow tricky. What would be the best way to treat the Knowledge gathered in the User Modeling process? Such Knowledge would constitute an invaluable resource for the enterprise that has it; the need of security profiles for such information handling and use must be between the next steps in the development of these technologies. In our approach we gave a comparison with the GOMS modeling procedure, in order to fully test the series of heuristic recommendations we made, they should be proven against other User Modeling techniques. We believe that our recommendations would pass such a test, but this has to be proven in a future work. Context modeling and context design are very important techniques in immersive paradigms such as the case of Ambient Intelligence. However such systems are based on the fact that the environment (Domain) is equipped with sensor capable of sensing the User and predicting his needs (Goals). Such theories are quite possible, but the real situation is that the actual state of the art systems are not able to provide such environments yet. This search for a seamless User-environment integration is highly dependable on the hardware availability in actual Ambient Intelligence installations (which are more like laboratory set-ups than real world application sites). Typically the architectures presented in the literature consider a total-immersive status of the User in an Ambient sensing him automatically. At time of writing, such a scenario is far away, because for that architecture to work, the User still needs to connect to the environment via a Bluetooth and Java enabled phone that must fulfill certain minimum requirements that are not easy found in every terminal. Given the evolution on the state of the art in mobile phones and portable devices, one can be certain that all the paradigms promoted by technologies like Ambient Intelligence are not as distant as we might first think. We believe that Context must be considered in future Domain paradigms and it should be derived from the Interaction stages of the User model. The mentioned context models could be greatly benefited from Reflexive Ontologies (RO). Although proven efficient for query handling [CTS+08], a RO needs to feed upon an initial set. Such initial set's complexity has to be filled out with initial queries that must be performed over the ontology. It must not be forgotten that the whole idea behind the RO is to query the actual ontology at a minimum. For such reason, when using the RO as the query engine, the first questions will not represent any difference than a classical query (even using a Reasoner). Once the RO is fed with enough Knowledge, the set of answers or the set of implicit answers that can be Semantically derived from the stored Knowledge will make a big difference. Especially when cyclic processes are in action (as the ones involved in our general architecture) the use of the RO is a valuable tool to consider. There is a consideration derived from the above fact: when using a RO, basically the set of queries is stored in a structure that roughly speaking can be considered an "on-steroids" array. When a query is asked and such query was already made, the time needed to look for such query would be equivalent in the worst case scenario to walk trough all the items in the array. If such array were big enough, the classical query (trough a reasoner or ontology API) would be a better approach. At present, there is no mechanism to predict how complex the RO can be in order for it to remain to be considered as a valid option. We believe that such mechanism would be a logical next outcome to fulfill.

Appendix A

Example Domain model based on ES

The following is the example of use of the methodology for Domain Modeling based on ES.

Stage 1- Definition

Let us consider the problem of modeling an Industrial Plant and in particular we will focus in the modeling of a Flange element. As our KB modeling paradigm, we choose to use ontology modeling.

Stage 2 - Selection of the Standard

By performing an Internet search, we find that there is an ISO Standard that could be used for our needs; this case is ISO 10303 AP 227 (Industrial Plants) [ISO01].

Stage 3 - Classes identification

Searching in the standard we find in page 64 that a description of a Flange element exists, such description is as follows:

4.2.84 Flange:

A Flange is a type of Fitting (see 4.2.83) that is an annular collar that permits a bolted connection to a similar collar. Each Flange contains two end connectors, one of which shall be a Piping_connector of type Flanged_end. Each Flange may be one of the following: a Blind_flange (see 4.2.3), an Expander_- flange (see 4.2.75), an Orifice_flange (see 4.2.146), or a Reducing_flange (see 4.2.202). Each Flange may be one of the following: a Lap_joint_flange (see 4.2.118), a Slip_on_flange (see 4.2.222), a Socket_- weld_flange (see 4.2.224), a Threaded_flange (see 4.2.252), or a Weld_neck_flange (see 4.2.266). The data associated with a Flange are the following:

— end_1_connector;

— end_2_connector;

— hub_through_length;

— hub_weld_point_diameter.

Figure A.1 depicts a typical weld-neck Flange:

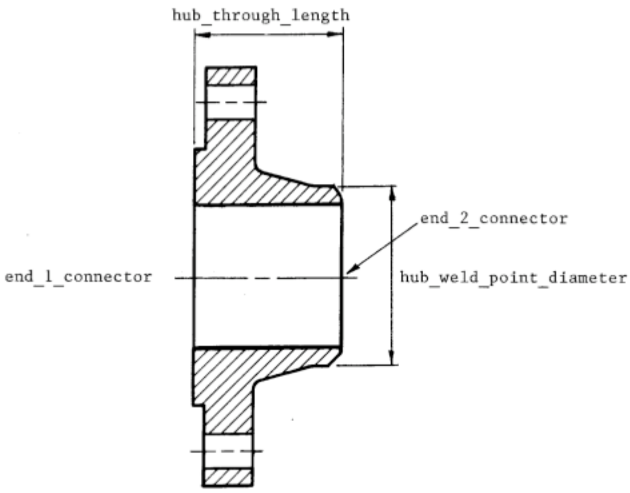


Figure A.1: A Flange according to the ISO 10303 AP227 Standard [ISO01]

Stage 4 - Properties identification

By looking at the properties in the Flange element, we create a classification as can be seen in Table A.1.

Name	Property_type	Value
hub_through_length	Data	Double
hub_weld_point_diameter	Data	Double
end_1_connector	Relational	Element
end_2_connector	Relational	Element

Table A.1: Properties classification

Stage 5 - Initial modeling

We use the Protégé ontology editor to model the element as can be seen in Figures A.2,A.3, A.4,A.5.

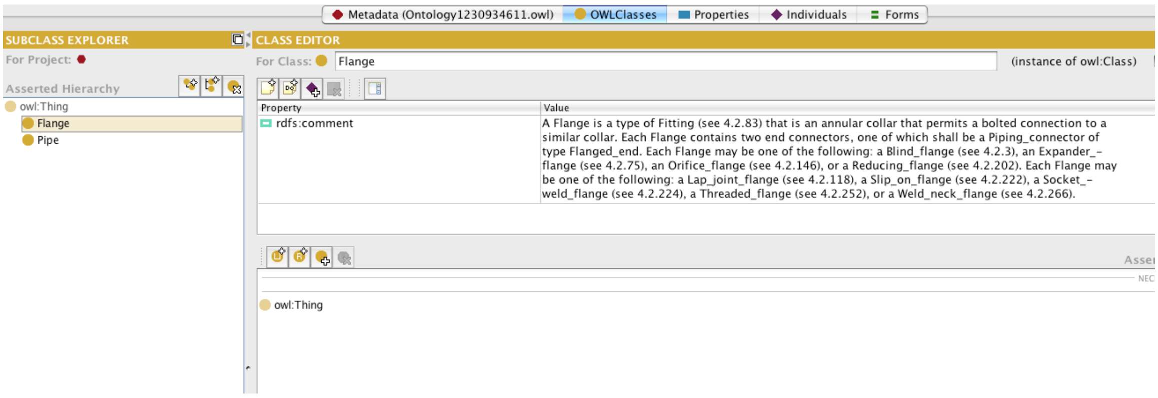


Figure A.2: Modeling of the Flange class

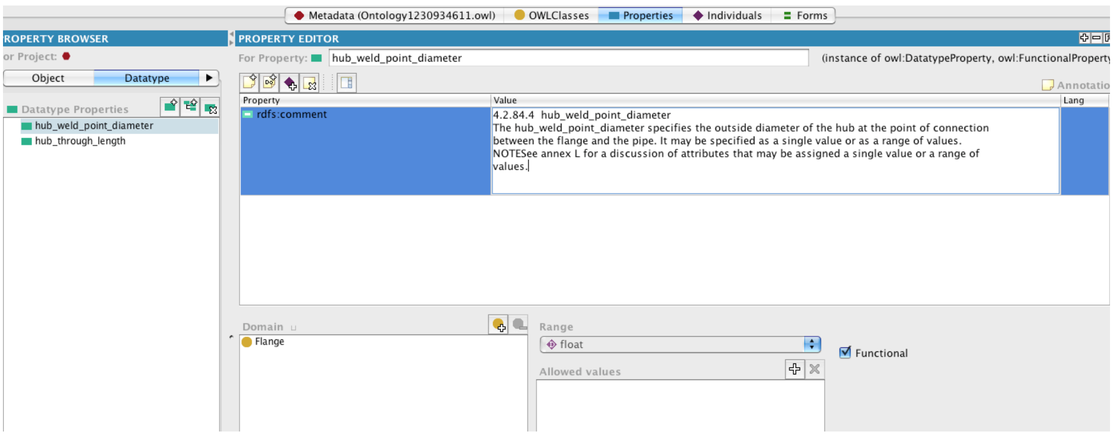


Figure A.3: Modeling of the Flange, data type properties

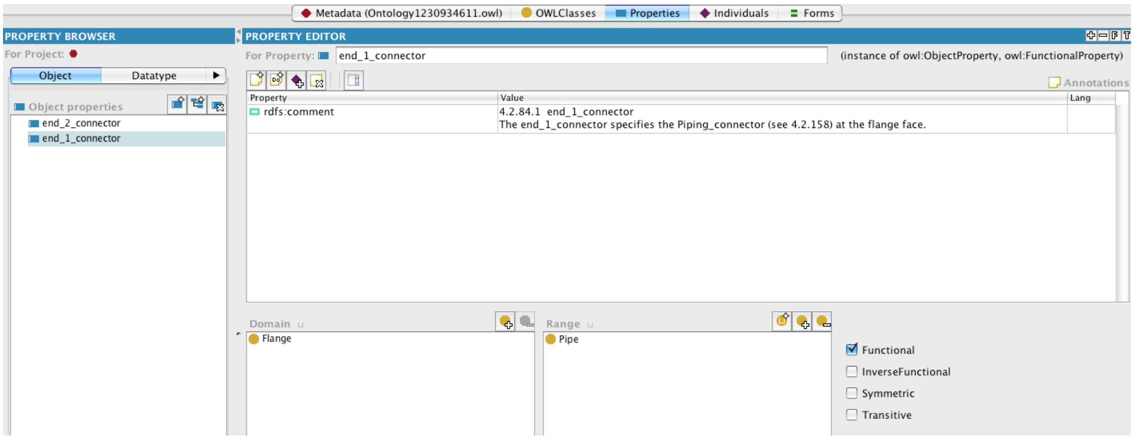


Figure A.4: Modeling of the Flange, relational properties

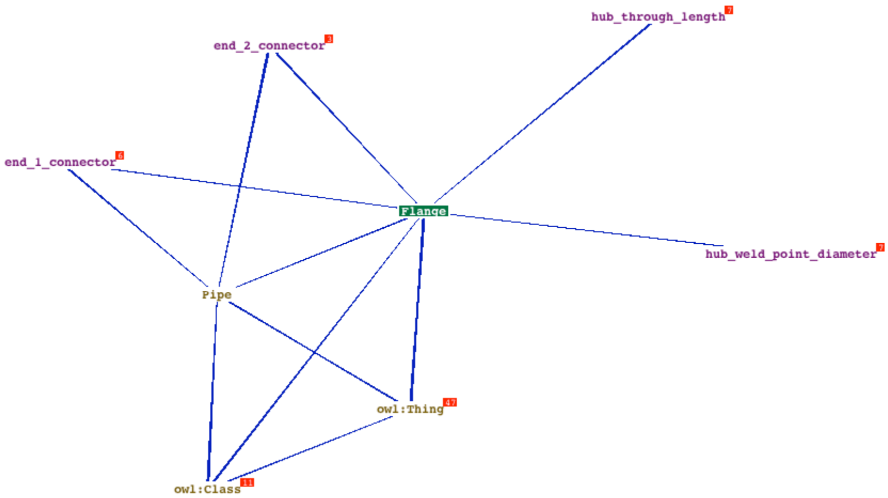


Figure A.5: Ontology graph for the Flange Class

Stage 6- KB transcription and refinement

For the example case, we decide than the ES contains enough information for our modeling needs, hence no extension is needed. The process is finalized by running a reasoning process to check the ontology for any problems at a logical level (not shown here).

Stage 7- Testing and instancing

In our case we use the Protégé OWL API [Knu08] for the generation of a java code suitable for the semi-automatic instancing of individuals, Figure A.6 depicts a piece of such code in the eclipse editor:

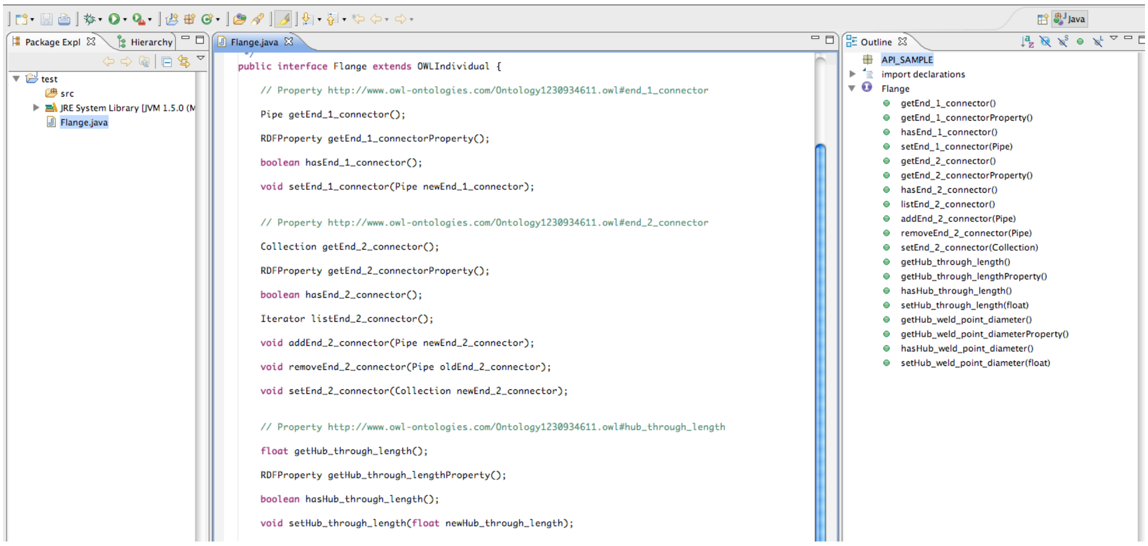


Figure A.6: Flange Class handling code

Glossary of terms

The following is a glossary of key terms related to the Semantic Web in general.

- **CLASS** – A named set, which serves to describe and distinguish some set of INDIVIDUALS. CLASSES may have one or more sub-classes, and one or more super-classes – in this way classes can combine to form a lattice-type graph (i.e. a structure similar to a hierarchical taxonomy, but which supports the idea of multiple parent or super-classes – sometimes described as multiple inheritance).

- **DATATYPE PROPERTY** - the kind of PROPERTY which specifies an attribute of the CLASS or CLASSES in the DOMAIN of the PROPERTY. An example is Name, which might use a number of characters to describe the name of the CLASS or CLASSES which are in the DOMAIN of the PROPERTY.

- **DOMAIN** – a DOMAIN is the CLASS or CLASSES which a given PROPERTY can be said to describe. A PROPERTY like IsStudent, intended to describe INDIVIDUALS belonging to a Human CLASS, has the DOMAIN of Human.

- **INDIVIDUAL** – A thing in the world, an object or item. INDIVIDUALS may be explicitly declared to be members of CLASSES, or may belong to CLASSES implicitly by virtue of the PROPERTIES they hold.

- **INCONSISTENCY** - An ONTOLOGY is said to be inconsistent when by deductive reasoning the axioms of the ONTOLOGY lead to contradiction. If for instance an INDIVIDUAL such as this author is declared to be a member of an Author CLASS, and the Author CLASS is declared as a sub-class of a WealthyPerson class, but it is also asserted that the author cannot possibly be a WealthyPerson (a reasonable assertion) – then deductive reasoning will demonstrate that the axioms thus declared lead to contradiction, and so the ONTOLOGY must be inconsistent.

- **KNOWLEDGE ENGINEERING (KE)** - is an engineering discipline that involves integrating Knowledge into computer systems in order to solve complex problems normally requiring a high level of human expertise

- **OBJECT PROPERTY** - the kind of PROPERTY which specifies a relation between two CLASSES, or two groups of CLASSES. In an ONTOLOGY describing people, an example could be isParentOf, which might specify a Person CLASS in both the DOMAIN and the RANGE of the PROPERTY.

- **ONTOLOGY** – An specification of a conceptualization.

- **ONTOLOGY MATCHING** – the process of matching two ONTOLOGIES. A MATCH can be made between any two CLASSES, or any two PROPERTIES, of the ONTOLOGIES. A match may contain one of four possible RELATIONS: EQUIVALENCE, GENERALITY, SPECIFICITY and DISJUNCTION. Furthermore, it may also have a CONFIDENCE LEVEL, which is specified by a matching agent (person or algorithm). An ONTOLOGY MATCH therefore may have a series of four-part structures, consisting of: CLASS or PROPERTY 1 (deriving from the first ONTOLOGY); CLASS or PROPERTY 2 (deriving from the second ONTOLOGY); a RELATION; and a CONFIDENCE LEVEL.

- **PROPERTY** - a PROPERTY is either an attribute (DATATYPE PROPERTY) or relation (OBJECT PROPERTY). A PROPERTY must always have one or more CLASSES in its DOMAIN; if it is a relational PROPERTY, it must also have one or more CLASSES in its RANGE. PROPERTIES, like CLASSES, may have sub-properties and super-properties, and therefore are also formed in a lattice-type graph.

- **RANGE** - a RANGE is the CLASS or CLASSES for which a given relation, or OBJECT PROPERTY, can take INDIVIDUALS as values. A PROPERTY like Studies, with a DOMAIN of Human CLASS, could take a RANGE of, for example, a Subject CLASS. Any INDIVIDUALS belonging to the Subject CLASS would be valid values for Studies PROPERTY.

- **RESTRICTION** - a RESTRICTION is a rule which limits which INDIVIDUALS may be said to belong to a given CLASS. Without RESTRICTIONS, all INDIVIDUALS belong to all CLASSES within an ONTOLOGY – CLASS membership is assumed, rather than required to be declared explicitly. Restrictions specify the conditions under which CLASS membership is to be restricted. There are various forms of RESTRICTIONS: the most common is PROPERTY RESTRICTION. A CLASS such as Mammal can be said to restrict INDIVIDUALS to those which contain specific values for PROPERTIES like HasBackbone (true); FemaleProducesMilk (true), and so on. Any INDIVIDUAL which does not contain these PROPERTIES, and these values for these PROPERTIES, cannot be a member of the Mammal set so defined.

- **SUBSUMPTION** - one CLASS (or PROPERTY) can be said to subsume another CLASS (or PROPERTY) when it is a generalisation of that CLASS (or PROPERTY). For example, both a Human CLASS and a Dog CLASS can be said to be subsumed by an Animal CLASS, since the Animal CLASS is a generalisation of these other CLASSES. Such a relation is declared one of subsumption in the literature.

- **USER MODELING (UM)** - One way of obtaining predictive evaluation of real-world tasks by trying to represent some aspect of the User's understanding, knowledge, intentions or processing. And there are many different techniques that are used to build User models

- **VIRTUAL ENGINEERING (VE)** - is defined as the integration of geometric models and related engineering tools (such as analysis, simulation, optimization and decision making , etc), within a computerized environment that facilitates multidisciplinary and collaborative product development.

Bibliography

[AB85] W Arnold and J Bowie. *Artificial Intelligence: A Personal Commonsense Journey*. Englewood Cliffs, 1985.

[ABH04] S. Alda, J. Bilek, and D Hartmann. Support of Collaborative Structural Design Processes through the Integration of Peer-to-Peer and Multiagent Architectures. In *Proceedings of ICCCBE 2004*, Weimar, Germany, 2004.

[AHH06] Hassan Abolhassani, Babak Bagheri Hariri, and Seyed H. Haeri. On ontology alignment experiments. Available online at: http://www.webology.ir/2006/v3n3/a28.html, 2006.

[Aut] Autodesk. *AutoCAD 2009 - Features, available online at: http://images.autodesk.com/adsk/files/autocad _2009_product_ overview_ brochure.pdf.*

[Azu97] R.T Azuma. A survey of augmented reality. presence. *Teleoperators and Virtual Environment*, 6(4), 1997.

[BAN+02] William V. Baxter, Iii Avneesh, Sud Naga, K. Govindaraju, and Dinesh Manocha. Gigawalk: Interactive walkthrough of complex environments. In *Proc. of Eurographics Workshop on Rendering*, pages 203–214, 2002.

[Bar01] Dirk Bartz. Large model visualization: Techniques and applications. Technical report, Ph.D. Thesis. Universität Tübingen, Wilhelm-Schickard-Institut für Informatik / Arbeitsbereich Graphisch-Interaktive Systeme, 2001.

[Bax02] W.V Baxter. Occlusion Culling for Walkthroughs of Large Virtual Environments. Technical report, University of North Carolina at Chapel Hill, http://www.cs.unc.edu/ baxter/projects/occlusion.html, 2002.

[BCG07] Fabio L. Bellifemine, Giovanni Caire, and Dominic Greenwood. *Developing Multi-Agent Systems with JADE (Wiley Series in Agent Technology).* Wiley, April 2007.

[BHGS01] Sean Bechhofer, Ian Horrocks, Carole Goble, and Robert Stevens. OilEd: a reason-able ontology editor for the semantic web. In *Proceedings of KI2001, Joint German/Austrian conference on Artificial Intelligence*, pages 396–408. Springer-Verlag, 2001.

[BHS07] Franz Baader, Ian Horrocks, and Ulrike Sattler. Description Logics. In Frank van Harmelen, Vladimir Lifschitz, and Bruce Porter, editors, *Handbook of Knowledge Representation*. Elsevier, 2007.

[BL00] Tim Berners-Lee. *Weaving the Web: Origins and Future of the World Wide Web*. Texere Publishing, 2000.

[Bl07] Tim Berners-lee. Architecture of the semantic web. 2007.

[Bré05] Michel Bréal. *Essai de sémantique*. Editions Lambert-Lucas, 3 edition, 1897, Re Ed. 2005.

[Bro90] Scott M. Brown. Adaptive user interfaces. Technical report, Intelligent Distributed Information Systems Laboratory, University of Connecticut, 1990.

[BSI] BSI Standards, available online at: http://www.bsi-global.com/.

[BW08] Chris Bizer and Daniel Westphal. Developers guide to semantic web toolkits. Technical report, in: http://www4.wiwiss.fu-berlin.de/bizer/toolkits/, Last Visited September 2008.

[CAM] CAM-I Standards, available online at: http://www.cam-i.org/.

[CC06] CIDOC-CRM. CIDOC Conceptual Reference Model, 2006.

[CCS00] Daniel Cohen, Yiorgos Chrysanthou, and Cláudio T. Silva. A Survey of Visibility for Walkthrough Applications. Technical report, EUROGRAPHICS, course notes, 2000.

[CDD+04] Jeremy J. Carroll, Ian Dickinson, Chris Dollin, Andy Seaborne, Kevin Wilkinson, Dave Reynolds, and Dave Reynolds. Jena: Implementing the semantic web recommendations. pages 74–83, 2004.

[CIM08] CIMData. PLM Definiton. *http://www.cimdata.com/plm/definition.html*, Last Visited September 2008.

[CIS08] CIS/2. CIS/2 – CIMSteel Integration Standard, July 2008.

[CL07] Martin Crowder and Jerald Lawless. On a scheme for predictive maintenance. *European Journal of Operational Research*, 176(3):1713–1722, February 2007.

[CMC03] Andrew I. Comport, Eric March, and François Chaumette. A real-time tracker for markerless augmented reality. In *In ACM/IEEE Int. Symp. on Mixed and Augmented Reality, ISMAR'03*, pages 36–45, 2003.

[CNM83] Stuart K. Card, Allen Newell, and Thomas P. Moran. *The Psychology of Human-Computer Interaction*. Lawrence Erlbaum Associates, Inc., Mahwah, NJ, USA, 1983.

[Coa03] Elayne Coakes. *Knowledge Management: Current Issues and Challenges*. IRM Press, London, 1 edition, 2003. Used in IRMA-L chapter.

[Con06] Larry Constantine. *Users, Roles, and Personas, in The Persona Lifecycle: Keeping People in Mind Throughout Product Design*. Morgan-Kaufman, San Francisco, 2006.

[CPFJ04] H. Chen, F. Perich, T. Finin, and A. Joshi. Soupa: Standard ontology for ubiquitous and pervasive applications, 2004.

[CR92] John M. Carroll and Mary B. Rosson. Getting around the task-artifact cycle: how to make claims and design by scenario. *ACM Trans. Inf. Syst.*, 10(2):181–212, April 1992.

[CTA03] Harry Chen, Finin Tim, and Joshi Anupam. An Intelligent Broker for Context-Aware Systems. In *Proceedings of Ubicomp 2003*, pages 12–15, October 2003.

[CTS+08] Yolanda Cobos, Carlos Toro, Cristina Sarasua, Javier Vaquero, Maria Teresa Linaza, and Jorge Posada. An architecture for fast semantic retrieval in the film heritage domain. In *6th International Workshop on Content-Based Multimedia Indexing (CBMI)*, pages 272–279. IEEE, June 2008.

[DAOI04] M. Danso-Amoako, W.J. O'Brien, and R. Issa. A case study of IFC and CIS/2 support for steel supply chain processes. In *Proceedings of the 10th International Conference on Computing in Civil and Building Engineering (ICCCBE-10)*, 12p , Weimar, Germany, 2004.

[Dav] Leroy Davis. Standards and Publications for Engineering - Standards Organizations. available online at: http://www.interfacebus.com/Standard_org.html.

[DIN] DIN Standards, available online at: http://www.din.de.

[DN] P. Dolog and W. Nejdl. Challenges and benefits of the semantic web for user modelling.

[D.N98] D.Noble. Distributed situation assessment. In *Proc. FUSION '98 International Conference*, 1998.

[DS] Dassault-Systemes. PLM 2.0. http://www.3ds.com/es/products/.

[ELE08] IEC 60050-191 - International Electrotechnical Vocabulary – Chapter 191: Dependability and quality of service, 2008.

[ES07] Jérôme Euzenat and Pavel Shvaiko. *Ontology matching.* Springer-Verlag, Heidelberg (DE), 2007.

[Ez 06] Ez az előadás a Magyarországi Web Konferencia. *Szemantikus Web: egy rövid bevezetés.* Available online at: http://www.w3.org/2006/Talks/0318-Budapest-IH, 2006.

[FC] Moderator Mark Friedell and Lotus Development Corporation. Interaction paradigms for human-computer cooperation in design.

[Fel98] Christiane Fellbaum, editor. *WordNet: An Electronic Lexical Database (Language, Speech, and Communication).* The MIT Press, May 1998.

[FFS+01] Elizabeth Furtado, João José Vasco Furtado, Wilker Bezerra Silva, Daniel William Tavares Rodrigues, Leandro da Silva Taddeo, Quentin Limbourg, and Jean Vanderdonckt. An Ontology-Based Method for Universal Design of User Interfaces. 2001.

[Fla81] O. J Flanagan. Psychology, progress, and the problem of reflexivity: a study in the epistemological foundations of psychology. *Journal of the History of the Behavioral Sciences*, 17:375–386, 1981.

[FM83] E Feigenbaum and P McCorduck. *The Fifth Generation.* Addison-Wesley, 1983.

[For82] Charles Forgy. Rete-a fast algorithm for the many patterns and many objects match problem. *Artificial Intelligence*, 19(1), September 1982.

[FTK96] Thomas Funkhouser, Seth Teller, and Delnaz Khorramabadi. The UC Berkeley system for interactive visualization of large architectural models. *Presence*, 5:13–44, 1996.

[GC86] E.M Goldratt and J Cox. The goal. Technical report, Grover , Aldershot, Hants, 1986.

[GF95] Michael Gruninger and Mark S. Fox. Methodology for the design and evaluation of ontologies. 1995.

[GJA92] Wayne D. Gray, Bonnie E. John, and Michael E. Atwood. The precis of Project Ernestine or an overview of a validation of GOMS. In *CHI '92: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 307–312, New York, NY, USA, 1992. ACM Press.

[GP95] Nicola Guarino and R. Poli. Formal ontology, conceptual analysis and knowledge representation. *International Journal of Human and Computer Studies*, 43:625–640, 1995.

[GPFC04] Asuncion Gomez-Perez, A. Fernadez, and Oscar Corcho. *Ontological Engineering with examples from the areas of Knowledge Management, e-Commerce and the Semantic Web*. Springer, 2004.

[Gre06] Adam Greenfield. *Everyware : The Dawning Age of Ubiquitous Computing*. New Riders Press, March 2006.

[GRSP02] C. Geiger, C. Reimann, C. Sticklein, and J. Paelke. Jartoolkit - a java binding for artoolkit. In ISBN: 0-7803-7680-3, editor, *Augmented Reality Toolkit, The First IEEE International Workshop*, 2002.

[Gru93] Tom Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199–220, 1993.

[Gru95] T.R. Gruber. Toward principles for the design of ontologies used for knowledge sharing. *International Journal of Human-Computer Studies*, 43(5-6):907–928, 1995.

[HB05] Gengxun Huang and Kenneth Mark Bryden. Introducing virtual engineering technology into interactive design process with high-fidelity models. In *WSC '05: Proceedings of the 37th conference on Winter simulation*, pages 1958–1967. Winter Simulation Conference, 2005.

[HGD95] Volkmar Hovestadt, Oliver Gramberg, and Oliver Deussen. Hyperbolic user interfaces for computer aided architectural design. In *CHI '95: Conference companion on Human factors in computing systems*, pages 304–305, New York, NY, USA, 1995. ACM.

[HKR+04] Matthew Horridge, Holger Knublauch, Alan Rector, Robert Stevens, and Chris Wroe. A practical guide to building owl ontologies using the protege-owl plugin and co-ode tools edition 1.0. August 2004.

[HM03] Volker Haarslev and Ralf Muller. Racer: An OWL reasoning agent for the semantic web. In *In Proc. of the International Workshop on Applications, Products and Services of Web-based Support Systems, in conjunction with 2003 IEEE/WIC International Conference on Web Intelligence*, pages 91–95. Society Press, 2003.

[Hon04] T Honkanen. *Modelling Industrial Maintenance Systems and the Effects of Automatic Condition Monitoring*. Technical, Helsinki University of Technology, 2004.

[Hor98] Ian Horrocks. Using an expressive description logic: FaCT or fiction. In *Proc. of the 6th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'98)*, pages 636–647, 1998.

[Iga00] Takeo Igarashi. Supportive interfaces for creative visual thinking. In *Collective Creativity Workshop*, 2000.

[int08] World internet usage, available online at http://www.internetworldstats.com/stats.htm. Technical report, 2008.

[ISO] ISO Standards, available online at: http://www.iso.org/.

[ISO01] ISO. *International Standard ISO 10303, Industrial Application Systems and Integration – Product Data Representation, Application protocol 227, Plant Spatial Configuration*. First edition, 2001.

[IWB+98] Hiroshi Ishii, Craig Wisneski, Scott Brave, Andrew Dahley, Matt Gorbet, Brygg Ullmer, and Paul Yarin. ambientroom: integrating ambient media with architectural space. In *CHI '98: CHI 98 conference summary on Human factors in computing systems*, pages 173–174, New York, NY, USA, 1998. ACM Press.

[JK96] B. John. and D. E Kieras. Using GOMS for user interface design and evaluation: Which technique? In ACM, editor, *ACM Transactions on Computer-Human Interaction*, volume 3, pages 287–319, 1996.

[JMLB06] C. Q. Jian, D. McCorkle, M. A. Lorra, and K. M. Bryden. Applications of virtual engineering in combustion equipment development and engineerin. In ASME, editor, *ASME International Mechanical Engineering Congress and Expo*, IMECE2006–14362, Chicago, November 2006. ASME, ASME Publishers.

[JS04] Minsu Jang and Joo-Chan Sohn. *Bossam-An Extended Rule Engine for OWL Inferencing*, volume Volume 3323/2004 of *Lecture Notes in Computer Science*. Springer, 2004.

[KBBM99] H Kato, M. Billinghurst, B. Blanding, and R. May. Artoolkit. Technical report, Hiroshima City University, December 1999.

[KCC+98] John C. Kunz, Tore R. Christiansen, Geoff P. Cohen, Yan Jin, and Raymond E. Levitt. The virtual design team. *Commun. ACM*, 41(11):84–91, 1998.

[KF88] Robert Kass and Tim Finin. Modeling the user in natural language systems. In *Journal of Computational Linguistics*, volume 14, pages 5–22. MIT Press, 1988.

[KFNM04] Holger Knublauch, Ray W. Fergerson, Natalya F. Noy, and Mark A. Musen. The Protégé OWL plugin: An open development environment for semantic web applications. pages 229–243. Springer, 2004.

[Kiw] Kiwisoft. ProSteel 3D Manual, available online at: http://www.strucsoftsolutions.com/prosteel3d/index.htm.

[Knu08] Holger Knublauch. Protégé-OWL API programmer's guide. web, 2008.

[Kok91] A. Kok. A Review and Synthesis of User Modeling in Intelligent Systems. In *The Knowledge Engi- neering Review*, volume 6, pages 21–47, 1991.

[Len96] Maurizio Lenzerini. TBox and ABox reasoning in expressive description logics. In *In Proc. of KR-96*, pages 316–327. Morgan Kaufmann, 1996.

[Lip03] R. Lippman. Immersive Virtual Reality for Steel Structures. In *Conference on Construction Applications of Virtual Reality (CONVR 2003).*, September 2003.

[Llo03] J.W. Lloyd. *Logic for Learning: Learning Comprehensible Theories from Structure Data*. Springer, Berlin, first edition, 2003.

[LRC02] D Luebke, M Reddy, and J Cohen. *Level of Detail for 3D Graphics*. Computer Graphics and Geometric Modeling ISBN 1-55860-838-9. Morgan Kaufmann, 2002.

[Lug06] A. Lugmayr. The future is 'ambient'. 6074:172–179, February 2006.

[Luh97] Niklas R Luhmann. *Organización y Decisión, Autopoiesis y Entendimiento Comunicativo. Barcelona*. Anthropos, 1997.

[MA03] Stefano Marzano and Emile Aarts. *The New Everyday View on Ambient Intelligence*. Uitgeverij 010 Publishers, 2003.

[Mai] Federal Standard 1037C, Glossary of Telecommunication Terms. Technical report.

[Man00] D Manocha. *Interactive Walkthroughs of Large Geometric Datasets*. AMC Siggraph, 2000.

[Mar89] Herbert Marcuse. *Hegel's Ontology and the Theory of Historicity*. MIT press, October 1989.

[MB+03] Claudio Masolo, Stefano Borgo, , Aldo Gangemi, Nicola Guarino, Alessandro Oltramari, and Luc Schneider. The WonderWeb Library of Foundational Ontologies and the DOLCE ontology. Technical report, 2003.

[MB06] D. S McCorkle and K. M Bryden. Using the Semantic Web to Enable Integration with Virtual Engineering Tools. In Springer London, editor, *Proceedings of the 1st International Virtual Manufacturing Workshop*, March 2006.

[MB07] D. S. McCorkle and K. M. Bryden. Using the semantic web technologies in virtual engineering tools to create extensible interfaces. *Virtual Real.*, 11(4):253–260, 2007.

[MBK03] D. S McCorkle, K. M Bryden, and S. J Kirstukas. Building a foundation for power plant virtual engineering. In *8th International Technical Conference on Coal Utilization and Fuel Systems*, pages 63–71, April 2003.

[MBS04] D. S McCorkle, K. M Bryden, and D. A Swensen. Using virtual engineering tools to reduce nox emissions. In *Proceedings of ASME Power 2004*, pages 441–446. POWER2004-52021, ASME Publishers, March 2004.

[mer09] Merrian-Webster dictionary. http://www.merriam-webster.com/dictionary/domain, January 2009.

[MFRW00] Deborah L. Mcguinness, Richard Fikes, James Rice, and Steve Wilder. An environment for merging and testing large ontologies. pages 483–493. Morgan Kaufmann, 2000.

[MK94] Paul Milgram and Fumio Kishino. A taxonomy of mixed reality visual displays. *IEICE Transactions on Information Systems*, E77-D(12), December 1994.

[MKSK00] Riichiro Mizoguchi, Kouji Kozaki, Toshinobu Sano, and Yoshinobu Kitamura. Construction and deployment of a plant ontology. 1937, 2000.

[MNH78] Tom Matteson, F. Stanley Nowlan, and Howard Heap. Reliability-Centered Maintenance. Report AD-A066579, Department of Defense, Washington, D.C, 1978.

[MSJT07] Gorka Marcos, Tim Smithers, Iván Jiménez, and Carlos Toro. Meta level: Enabler for semantic steered multimedia retrieval in an industrial design domain. *Systems Science*, 33(2):15–22, 2007.

[MV80] Humberto Maturana and Francisco Varela. *Autopoiesis and Cognition.* Reidel, Boston, 1980.

[Nak91] Seiichi Nakajima. Introduction to TPM. Technical report, Japan Institute for Plant Maintenance, 1991.

[Nam] K Wes Nam. User Modeling, available online at: http://www.cc.gatech.edu/classes/cs6751_97_winter/Topics/user-model/.

[NS89] A. F. Norcio and J. Stanley. Adaptive Human-Computer Interfaces: A Literature Survey and Perspective. In *IEEE Transactions on Systems, Man, and Cybernetics*, volume 19, pages 399–408, 1989.

[ODC04] G. M. P. Ohare, B. R. Duffy, and A. G. Campbel. Nexus- mixed reality experiments with embodied intentional agents. In *In Proceedings of Computer Animation and Social Agents - CASA 2004*, 2004.

[PEGM94] Angel R. Puerta, Henrik Eriksson, John H. Gennari, and Mark A. Musen. Model-based automated generation of user interfaces. In *National Conference on Artificial Intelligence*, pages 471–477, 1994.

[PhS07] Eric Prud-hommeaux and Andy Seaborne. SPARQL Query Language for RDF (Working Draft). Technical report, W3C, March 2007.

[PLS02] J. Posada, A. Larzabal, and A. Stork. Semantic-based parametric control of CAD model conversion for large model visualization in virtual reality. 2002.

[Pos05] Jorge Posada. *A Methodology for the Semantic Visualization of Industrial Plant CAD Models for Virtual Reality Walkthroughs.* Phd thesis, Technische Universität Darmstadt, Darmstadt, Germany, December 2005.

[Pre03] Oxford University Press. *Oxford Dictionary of English.* Oxford University Press, 2rev ed edition (21 aug 2003) edition, 2003.

[PTWS05a] Jorge Posada, Carlos Toro, Stefan Wundrak, and AndrÃ© Stork. Ontology Modelling of Industry Standards for Large Model Visualization and Design Review using Protege. In *Eighth International Protege Conference*, Madrid, Spain, 2005.

[PTWS05b] Jorge Posada, Carlos Toro, Stefan Wundrak, and AndrÃ© Stork. Ontology Supported Semantic Simplification of Large Data Sets of Industrial Plant CAD Models for Design Review Visualization. In R. Khosla, R. Howlett, and L. Jain, editors, *Knowledge-Based Intelligent Information and Engineering Systems, 9th International Conference, KES 2005*, volume 3681 - Proceedings Part I of *Lecture Notes in Artificial Intelligence*, Melbourne, Australia, 2005. Springer-Verlag.

[PTWS05c] Jorge Posada, Carlos Toro, Stefan Wundrak, and AndrÃ© Stork. Using Ontologies and STEP Standards for the Semantic Simplification of CAD Models in Different Engineering Domains. In *Proceedings of FOMI2005 (Formal Ontologies Meet Industry)*, Verona, Italy, 2005. Mateo Cristani.

[PVdBW+04] Davy Preuveneers, Jan Van den Bergh, Dennis Wagelaar, Andy Georges, Peter Rigole, Tim Clerckx, Yolande Berbers, Karin Coninx, Viviane Jonckers, and Koen De Bosschere. *Towards an Extensible Context Ontology for Ambient Intelligence*. 2004.

[PWTS05] Jorge Posada, Stefan Wundrak, Carlos Toro, and AndrÃ© Stork. Semantically Controlled LMV Techniques for Plant Design Review. In *ASME DECT/CIE*, Salt Lake City, 2005. ASME.

[RDH+] Alan Rector, Nick Drummond, Matthew Horridge, Jeremy Rogers, Holger Knublauch, Robert Stevens, Hai Wang, and Chris Wroe. OWL Pizzas - Practical Experience of Teaching OWL-DL, Common Errors & Common Patterns. *Engineering Knowledge in the Age of the SemanticWeb*, pages 63–81.

[SEA+02] York Sure, Michael Erdmann, Juergen Angele, Steffen Staab, Rudi Studer, and Dirk Wenke. Ontoedit: Collaborative ontology development for the semantic web. pages 221–235. Springer, 2002.

[SPG+07] Evren Sirin, Bijan Parsia, Bernardo C. Grau, Aditya Kalyanpur, and Yarden Katz. Pellet- A practical OWL-DL reasoner. *Web Semantics: Science, Services and Agents on the World Wide Web*, 5(2):51–53, June 2007.

[SS04] Cesar Sanin and Edward Szczerbicki. Knowledge supply chain system: A conceptual model. In A. Szuwarzynski, editor, *Knowledge Management: Selected Issues*, pages 79–97. Gdansk University Press, Gdansk, 2004.

[SS05a] Cesar Sanin and Edward Szczerbicki. Set of experience: A knowledge structure for formal decision events. *Foundations of Control and Management Sciences*, 3:95–113, 2005. Poznan University of Technology, Poznan.

[SS05b] Cesar Sanin and Edward Szczerbicki. Using XML for Implementing Set of Experience Knowledge Structure. In R. Khosla, R. Howlett, and

L. Jain, editors, *Knowledge-Based Intelligent Information and Engineering Systems, 9th International Conference, KES 2005*, volume Proceedings Part I - 3681 of *Lecture Notes in Artificial Intelligence*, pages 946–952, Melbourne, Australia, 2005. Springer-Verlag. Berlin, Heidelberg.

[SS06a] C. Sanin and E. Szczerbicki. Extending set of experience knowledge structure into a transportable language xml (extensible markup language). *Cybernetics and Systems: An International Journal*, 37(2-3):97–117, 2006.

[SS06b] Cesar Sanin and Edward Szczerbicki. Similarity metrics for set of experience knowledge structure. In R. Khosla, R. Howlett, and L. Jain, editors, *Knowledge-Based Intelligent Information and Engineering Systems, 9th International Conference, KES 2006*, Lecture Notes in Artificial Intelligence, pages 663–670, Bournemouth, UK, 2006. Springer-Verlag. Berlin, Heidelberg.

[SSeT] Hidebound Systems, In K. Srikantaiah, M. E. D. Koenig (eds, and Knowledge Management For The. From information management to knowledge management: Beyond the 'hi-tech hidebound ' systems malhotra, y. (2000). from information management to knowledge management: Beyond the 'hi-tech.

[SST07] Cesar Sanin, Edward Szczerbicki, and Carlos Toro. An owl ontology of set of experience knowledge structure. *J. UCS*, 13(2):209–223, 2007.

[STA08] Stanford Knowledge Systems, AI Laboratory, June 2008.

[Sto] Andre Stork. Semantics in Modeling, Interaction and Visualization, Invited Talk Fraunhofer IGD 20 Years.

[TMS+ny] Carlos Toro, Aitor Moreno, Alvaro Segura, Iñigo Barandiaran, and Jorge Posada. Semantics and Graphics in Product Life Cycle Management (PLM). Bringing Virtual Engineering to the Real World. In *iTEC08 - Your Technology Hotspot*, pages 37–47, November 2008, Darmstadt (Germany).

[TNDM] Samson Tu, Csongor Nyulas, Amar Das, and Mark Musen. Querying the Semantic Web with SWRL.

[Tor02] Carlos Toro. Representacion de estructuras funcionales de diseño conceptual usando XML. Master's thesis, Universidad EAFIT, June 2002.

[TPOF07] Carlos Toro, Jorge Posada, Joaquin Oyarzun, and Juanjo Falcon. Supporting the CAD Structural Design Process with Knowledge-based Tools. *Cybernetics and Systems*, 38(5):575–586, 2007.

[TSSP08] Carlos Toro, Cesar Sanín, Edward Szczerbicki, and Jorge Posada. Reflexive Ontologies: Enhancing Ontologies with self-contained queries. *Cybernetics and Systems: An International Journal*, 39(1–19), 2008.

[TSV+07] Carlos Toro, Cesar Sanin, Javier Vaquero, Jorge Posada, and Edward Szczerbicki. Knowledge based Industrial Maintenance using portable devices and Augmented Reality. In R.J. Howlett, and L.C. Jain (Eds.): Proceedings Part I, Lecture Notes in Artificial Intelligence 4962, editor, *Knowledge-Based Intelligent Information and Engineering Systems, 11th International Conference, KES 2007*, pages 295–302, Vietri sul Mare, Italy, September 2007. Springer-Verlag, Ber.

[TTP+07] Carlos Toro, Maite Termenon, Jorge Posada, Joaquin Oyarzun, and Juanjo Falcon. Ontology supported adaptive user interfaces for structural CAD design. In *CIRP Conference on Digital Enterprise Technology (DET)*. Springer Berlin; Heidelberg, 2007.

[UK95] Mike Uschold and Martin King. Towards a methodology for building ontologies. In *In Workshop on Basic Ontological Issues in Knowledge Sharing, held in conjunction with IJCAI-95*, 1995.

[W3C97] W3C. RDF, 1997.

[W3C04] W3C. OWL Web Ontology Language Reference - W3C Recommendation, 10 February 2004, 2004.

[Wik07] Wikipedia. Reflexivity from a social point of view. Web, 2007.

[Wil29] John Wilson. The management of the maintenance department in industrial plants. Master's thesis, Massachusetts Institute of Technology. Dept. of Business and Engineering Administration, Available Online: http://dspace.mit.edu/handle/1721.1/16521, May 17, 1929.

[WKS03] J. Weidenhausen, C. Knoepfle, and D. Stricker. Lessons learned on the way to industrial augmented reality applications, a retrospective on ARVIKA. *Computers and Graphics*, 27(6):887–891, December 2003.

[WRA05] W. Weber, J. M. Rabaey, and E. Aarts. *Ambient Intelligence*. Springer, April 2005.

[YGB05] Michael Yudelson, Tatiana Gavrilova, and Peter Brusilovsky. Towards User Modeling Meta-ontology. In *Journal of User Modeling*, pages 448–452. 2005.

[Zha05] Zhijun Zhang. The ontology query languages for the semantiac web- a performance evaluation. Master's thesis, The University of Georgia, 2005.