RESEARCH REPORT

VTT-R-00031-20

Semantic representation of engineering knowledge, pre-study

Authors: Akhtar Zeb, Juha Kortelainen, Markus Ylikerälä

Confidentiality: Public

Report's title Semantic representation of engineering knowledge, pre-study	
Customer, contact person, address VTT Technical Research Centre of Finland Ltd P.O. Box 1000, FI-02044 VTT, Finland	**Order reference** –
Project name AI-assisted systems engineering, pre-study	**Project number/Short name** 124698/GG_PREAIASE
Author(s) Akhtar Zeb, Juha Kortelainen, Markus Ylikerälä	**Pages** 51/–
Keywords Semantic Web, ontology, linked data, engineering knowledge	**Report identification code** VTT-R-00031-20

Summary

Artificial intelligence, AI, is having a strong hype in research and in industry. The main factor for the hype is the fast progress in the development of data-based AI technologies, especially machine learning in all of its forms. The technologies and solutions have enabled fast algorithms to recognise features in data or to make quick decisions based on given inputs. The common feature in these applications is that either the approach requires a large amount of data or it can analyse and e.g. classify a large amount of data.

The need for increased computer intelligence also in engineering design is evident, but one narrow set of methods and technologies does not solve the challenges. The solution is to integrate different methods and technologies to fulfil the different kinds of needs. On one hand, we need fast algorithms e.g. to help the user in his/her tasks or to categorise large data sets, while/ on the other hand we also need to utilise the existing domain knowledge, especially in the design of complex products and systems.

The previous hype of artificial intelligence in 1990's and 2000's was emphasising knowledge representation, management and engineering, and one of the main outcomes of the hype has been the set of technologies for the Semantic Web. In this work, we revisit the selected technologies of the Semantic Web and study the state-of-the-art applications utilising them for representing engineering design data (Sections 1–6). In addition, some of the existing tools and systems for editing data models as well as for storing the knowledge data were studied and are presented in Section 7. To illustrate the overall approach of engineering knowledge representation, a small-scale case study was done and is presented in Section 8. A brief summary of the report is presented in Section 9.

Confidentiality	Public

Espoo 9.3.2020

Written by	**Reviewed by**	**Accepted by**
Akhtar Zeb, Research Scientist	Jari Lappalainen, Senior Scientist	Emmanuel Ory, Research Team Leader

VTT's contact address
VTT Technical Research Centre of Finland Ltd, P.O. Box 1000, FI-02044 VTT, Finland

Distribution (customer and VTT)
VTT Archive, 1 copy

Preface

This work is a pre-study for a new research theme at VTT, focusing on using artificial intelligence (AI) technologies in engineering and design. Within the theme, several important topics have been identified, including physics-based modelling and simulation, design and engineering data management, computing technologies, data analytics, machine learning, knowledge representation and engineering, and data integration, among others. When doing engineering design in industrial context, the main challenges can be compressed into three contradicting targets, i.e. to design better solutions, to do the design work with smaller resources, and to do it faster. To optimise this set of target, all the available possibilities have to be used, including the advances in AI technologies, but also the existing knowledge in archived materials, such as old designs and design artefacts. In this work, we revisit the Semantic Web technologies, and study the state-of-the-art of them and especially how they have been applied in engineering process. One of the main objectives of this work is to illustrate the potential of the technologies for engineering data management and the effort needed to develop them further. The project group would like to thank VTT Technical Research Centre of Finland Ltd for enabling and supporting this work.

Espoo 9.3.2020

Authors

Contents

List of Abbreviations

AEC-FM	Architecture, Engineering, Construction and Facility Management
AHP	Analytical Hierarchy Process
AI	Artificial Intelligence
AsD	Assembly Design
BDI	Belief-Desire-Intention
BEP	Building Energy Performance
BIM	Building Information Management
BIMSO	Building Information Management Shared Ontology
BLC	Building Life Cycle
BOM	Bill Of Materials
CAD	Computer Aided Design
CAE	Computer Aided Engineering
CFD	Computational Fluid Dynamics
CI	Critical Infrastructure
DAML	Digital Asset Modelling Language
DeMO	Discrete-event Modelling Ontology
DES	Discrete Event Simulation
DL	Description Logic
DocOnto	Document-based Ontology
ETO	Engineer-To-Order
F-logic	Frame logic
FOL	First-Order-Logic
IFC	Industry Foundation Classes
JADE	Java Agent Development Framework
KIF	Knowledge Interface Format
LIMS	Laboratory Information Management Systems
LOD	Linked Open Data
M&S	Modelling and Simulation
ODS	Ontology Driven Simulation
OIL	Ontology Interface Layer
OWL	Web Ontology Language
PFMEA	Potential Failure Modes and Effects Analysis
PLC	Programmable Logic Controller
PV	Photovoltaic
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
RS	Rule Selector

SN	Semantic Networks
SPCA	Short Paths Crossings Algorithm
SWRL	Semantic Web Rule Language
UML	Unified Modelling Language
W3C	World Wide Web Consortium
XML	Extensible Markup Language
XSD	XML Schema Definition

1 Introduction

Data, information and knowledge are representing increasingly important role in our society and industry, including engineering and design domain. The fast development in computer technology and science has brought us Big Data, Machine Learning and AI, among other technologies. In all these, data is having an important role. But data itself, without a useful meaning for its user, has little value and producing more data does not increase its value. This applies to engineering and design, where the advances in computing technology have enabled producing, managing and utilising large amount of data with computer simulation, data analytics and numerous digital engineering and design tools and systems. Processing data into higher value form usually means that the data is used for creating new information and further new knowledge. This created new information and knowledge has to be stored, managed, modified and utilised also in digital form, which requires dedicated technologies and solutions.

The previous hype of artificial intelligence in 1990's and 2000's was emphasising knowledge representation, management and engineering, and one of the main outcomes of the hype has been the set of technologies for the Semantic Web. The Semantic Web technology has been introduced to improve the process of information and knowledge representation, analysis, reasoning and utilisation. Key to this is the development of an ontology for representing knowledge of a domain.

In this work, we revisit the selected technologies of the Semantic Web and study the state-of-the-art applications utilising them for representing engineering design data (Sections 1–6). Section 2 introduces the Semantic Web technologies. Section 3 describes the drivers, methodologies and tools used for ontology development. A review of the state-of-the-art ontologies and the most active engineering domains utilising knowledge engineering, applications of semantic technologies in the architecture, engineering and construction (AEC) domain, and the role of ontologies in the modelling and simulation (M&S) domain are presented in Section 4, 5 and 6, respectively. In addition, some of the existing tools and systems for editing data models as well as for storing the knowledge data were studied and are presented in Section 7. To illustrate the overall approach of engineering knowledge representation, a small-scale case study was done and is presented in Section 8. A brief summary of the report is presented in Section 9.

2 Semantic Web Technology

Internet searches often overwhelm individuals and practitioners with millions of pages that they have to browse through in order to identify suitable innovations to use in their projects. Users are therefore unable to make informed choices and have to rely on specialists with experience on a limited range of innovations for advice. It has been widely acknowledged that the solution to this problem is the use of a machine-understandable language with rich semantics for some or all of the information on the Web.

This has led to the emergence of the Semantic Web, the next generation of the Web, which promises to considerably improve information representation, sharing, re-use and automated processing by software agents to make inferences. Key to this is the use of a common language or an ontology for representing knowledge from different sources to facilitate decision-making (Tah and Abanda 2011). According to the World Wide Web Consortium (W3C 2019), the goal of the Semantic Web is to allow data to be shared effectively by wider communities, and to be processed automatically by tools as well as manually.

Similarly, in the engineering environments the accuracy and efficiency of information exchange among various agents (product designers, manufactures, suppliers, etc.) improves by storing information once where it is generated and allowing access to the information over the Intranet/Internet. It is also beneficial to store project data in a format that allows machine processing of routine operations. Semantic Web technology uses a graph data structure for information modelling that facilitates integrating information from different sources and allows computers to access and process information distributed over the Internet (Niknam and Karshenas 2017).

The knowledge on the Semantic Web is presented in the form of semantic networks. A semantic network is a graph of the structure of meaning. Specifically, it is a graphical notation for representing knowledge in patterns of interconnected nodes and arcs. These nodes represent the concepts and the arcs describe the interrelationship between every two nodes. It provides a convenient approach to visualise a knowledge base (Park et al. 2008). The Description Language (DL) queries are used for searching information on the Semantic Web.

The point of the Semantic Web is not just to make applications smarter, but also to make data smarter. The data does not and should not reside in application-specific databases. The data become smarter through the use of higher semantics from technologies such as concept maps or ontologies (Kim, Manley, and Yang 2006). The Semantic Web technology allows anyone to express a piece of data about some entity in a way that can be combined with information that other sources provide.

In 2001, the W3C set up a standardisation working group to develop a standard for a Web Ontology Language (OWL) having recognised that an ontology-language standard is a prerequisite to developing the Semantic Web. This resulted in the development of the OWL ontology-language standard in 2004 exploiting earlier work on OIL, DAML and DAML+OIL languages. Table 1 shows some of the features of XML, RDF/RDFS and OWL languages. There are several other semantic languages, for instance, KIF, OntoLingua, OCML, FLogic, Loom, DublinCore, SHOE, XOL, OIL, DAML, DAML+OIL, DAML-L, C-OWL, UML, and so forth. Each of them is characterised by different reasoning capabilities, complexity, levels of difficulty in programming, and other features as listed in (Negri et al. 2016), and each of them answers in a different way to the requirements of the particular domain. Thus it is necessary to evaluate the available languages with respect to those requirements.

As the Semantic Web vision matures, further work will be required to develop Semantic Web-based browsers with user friendly interfaces to allow individuals and practitioners to be able to undertake queries and searches without the need to understand Description Language (DL) query constructs (Tah and Abanda 2011).

Table 1. Semantic languages and features (Negri et al. 2016)

XML	Extensible Markup Language (XML) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. Issues related to this language: (i) It lacks semantics: designed to describe the structure of a document, not the content (ii) ‘‘is-a’’ relationship does not exist (iii) Attributes are not local to an object but global to a document (iv) No notion of inheritance (v) It defines an order in which tags appear in a document; the order does not matter in an ontology (vi) Difficult for machines when new vocabulary is used: no difference between polysemous terms and no possibility to combine synonymous terms. Web language: Yes Reasoning support: No
RDF/RDFS	Resource Description Framework (RDF) provides a simple data model and the RDF Schema (RDFS) defines a simple ontology language with classes, sub-classes, properties, sub-properties, and domain and range

	restrictions in RDF for expressing metadata. It is used to describe information about Web resources, to make information machine processable and to provide automated processing of Web information by intelligent agents. RDF is not very expressive (it only represents concepts, concept taxonomies and binary relations). It was created by the World Wide Web to provide meaning to data. It can be linked to any Web resource: interoperability between applications that exchange machine-understandable information on the web (interoperability is an important advantage over XML, thanks to the semantics, because XML does not represent the meaning). It consists of independent objects that form object-attribute-value triples (representable with a directed graph data model with nodes and edges: nodes are subject and object while an edge is a predicate), where subjects, objects and predicates are identified by URIs (even if objects may also be literals). RDFS has been introduced as a layer on top of RDF as a set of ontological modelling primitives (classes and subclasses of resources, properties and relations): this allows to set a particular vocabulary for RDF data. With the structure of classes and subclasses, it allows users to publish ontologies on the Web. However, RDFS is not explicit (formal) enough and still does not provide exact semantics when it comes to representing complex constraints. Web language: Yes Reasoning support: Some inference engines mainly for the constraint checking
OWL	OWL is the de facto standard ontology language. It is compatible with SHOE and DAML+OIL and is an extension of RDFS, but has more power to express semantics. It includes classes and operations on classes such as conjunction and disjunction and existentially and universally quantifiable variables. One of the significant features of the OWL language is its ability to make equality claims; in fact, OWL introduces constructions to state equality between classes (owl:sameClassAs) and between properties (owl:samePropertiesAs). This enables mapping between different individual ontologies: in fact OWL provides built-in ontology mapping support, that is, a particular class or property in one ontology is the same as a class or property in another ontology (owl:sameClassAs, owl:samePropertyAs): the individuals therefore have the same ‘‘identity". It is characterised by logical inference and can derive knowledge. It has more powerful reasoning than RDF: RDF has only a propositional reasoning, OWL reasoning can be about whole documents. Drawbacks: (i) It has a very complex, not efficient reasoning (ii) It is not easy to us (iii) It is not intuitive (iv) It does not have built-in primitives for the (very important) part-whole relations (v) It cannot deal with the fact that the meaning of certain words is context dependent. For this reason, it comes in many flavours, of which the three main ones are: OWL Full, OWL DL, and OWL Lite. The selection criterion is to take the best for the system requirements. Web language: Yes Reasoning support: It allows complex reasoning about documents

3 Ontology

What is an ontology? How to develop an ontology? Why to develop an ontology? What are the benefits of using ontology? What methodologies and tools are available for ontology development? In this section, very brief answers to such questions are presented.

3.1 Definition of ontology

Ontology is originally a branch of philosophy about the basic characteristics of all reality in the world. Currently, it has become a popular topic in various communities, including artificial intelligence, knowledge engineering, and natural language processing. There are many definitions of ontology adopted for each community (Park et al. 2008).

Ontologies are explicit, formal specifications of terms in the domain and of the relations among them. Formal refers to the fact that the ontology should be machine-readable. Similarly, it is also stated that, an ontology is a formal explicit description of concepts in a domain of discourse (classes or concepts), properties of each concept describing various features and attributes of the concept (slots, roles or properties), and restrictions on slots (facets or role restrictions). An ontology together with a set of individual instances of classes constitutes a knowledge base. In reality, there is a fine line where the ontology ends and the knowledge base begins. In other words, ontologies are used to capture knowledge about some domain of interest. An ontology describes the concepts in the domain and the relationships that hold between those concepts.

Classes are the focus of most ontologies. Classes describe concepts in the domain. For example, a class of wines represents all wines. Specific wines are instances of this class. The *Bordeaux wine* in a glass is an instance of the class of Bordeaux *wines*. A class can have subclasses that represent concepts that are more specific than the superclass. For example, we can divide the class of all wines into red, white, and rose wines. Alternatively, we can divide a class of all wines into sparkling and non-sparkling wines (Noy and McGuinness 2001).

3.2 Developing an ontology

Ontologies are increasingly used in knowledge management systems, medical and bio-informatics and play a key role in the Semantic Web and grid computing. Engineering was among the earliest sectors to apply ontologies, and in this sector the approach is considered more mature than in others (Ma, Bal, and Issa 2014).

Ontologies are very beneficial for representing engineering knowledge. Engineers are dependent on accessing documents in order to fulfil various design and engineering tasks. In fact, today’s engineers simply do not make an effort to find engineering content beyond mere keyword searches. However, current information retrieval (IR) approaches either retrieve too much or irrelevant results for engineering (Li, Raskin, and Ramani 2007). In industry sectors, it was reported that design engineers spent 20% to 30% of their time retrieving and communicating information (Court, Ullman, and Culley 1998). “Delivering the right information to the right people at the right time” plays an important role in supporting engineers’ memory extension, knowledge sharing, design concept exploration, design reuse, and the learning process particularly of novice engineers (Ahmed and Wallace 2004).

Developing an ontology includes:

- Defining classes in the ontology
- Arranging the classes in a taxonomic (subclass–superclass) hierarchy
- Defining slots/properties and describing allowed values for these slots

- Filling in the values for slots for instances

3.3 Why to develop an ontology

An ontology defines a common vocabulary for researchers who need to share information in a domain. It includes machine-interpretable definitions of basic concepts in the domain and relations among them. Some of the reasons to develop an ontology are: (Noy and McGuinness 2001)

- To share common understanding of the structure of information among people or software agents
- To enable reuse of domain knowledge
- To make domain assumptions explicit
- To separate domain knowledge from the operational knowledge
- To analyse domain knowledge

There is no correct way to model a domain – there are always viable alternatives. The best solution usually depends on the application you have in mind and the extensions that you anticipate. Ontology development is necessarily an iterative process. It is a creative process and no two ontologies designed by different people would be the same. However, ontological commitment is important. It is the agreement by multiple parties (persons and software systems) to adopt a particular ontology when communicating about the domain of interest, even though they do not necessarily have the same experiences, theories, or prescriptions about that domain (Holsapple and Joshi 2002).

In the engineering domain, the processes of product design, manufacture and analysis completed with computer-aided systems are integrated together in today’s enterprise environment, like automobile manufacturing industry. Documents and drawings in different processes with different systems stored electronically should be unified for organisation and management in order to solve information isolation between systems. Documents in integrated environment come from different systems. Some are in the form of general documents like PDF, WORD and EXCEL; some are standard engineering documents like CAD drawings, computer aided processing planning (CAPP) sheets and bill of materials (BOM) sheets; some are custom documents like assembly information documents of automobile and clutch vibration analysis documents of automobile dynamics in automobile manufacturing. Additionally, there are non-text documents like images and videos. We collectively refer to all these documents that come from different sources, used in different cases with different standards and formats, as heterogeneous documents (Yao, Lin, and Dong 2009). Representing this huge amount of knowledge contained in the heterogeneous documents in the form of ontology would enhance accessibility, retrieval and re-use of knowledge.

Ontology can also be used in artificial intelligence, knowledge representation, inductive reasoning and a variety of problem solving techniques, as well as to support Semantic Web and systems integration (Pandit and Zhu 2007).

3.4 Benefits of ontology

Emerging computer network technologies enable an environment in which a product can be collaboratively and remotely developed rather than just locally developed. As product development becomes more knowledge-intensive and collaborative, research on knowledge retrieval, capture, accessibility, and reusability becomes increasingly important.

Heterogeneous tools and multiple designers are frequently involved in collaborative product development, and designers often use their own terms and definitions to represent a product design. Thus, to efficiently share design information among multiple designers, a designer's intentions should be persistently captured and the semantics of the designer's terms and intents should be interpreted in a consistent manner. For this purpose, a standardised data format is a prerequisite. Furthermore, the appropriate design knowledge should be provided during all design processes in order to make proper design decisions (Kim et al. 2006).

Ontologies in the Semantic Web can explicitly represent semantics and promote integrated and consistent access to data and services. Thus, if an ontology is used in a heterogeneous and distributed design collaboration, it will explicitly and persistently represent engineering relations that are imposed in the design. Design intent can be captured by reasoning, and, in turn, as reasoned facts, it can be propagated and shared with design collaborators.

3.5 Methodologies and tools for developing ontology

3.5.1 Methodologies

The field of ontologies is nowadays mature enough and many methodologies, tools and languages are already available. The future work in this field should be driven towards the creation of a common integrated workbench for ontology developers to facilitate ontology development, exchange, evaluation, evolution and management, to provide methodological support for these tasks, and translations to and from different ontology languages. This workbench should not be created from scratch, but instead integrating the technology components that are currently available.

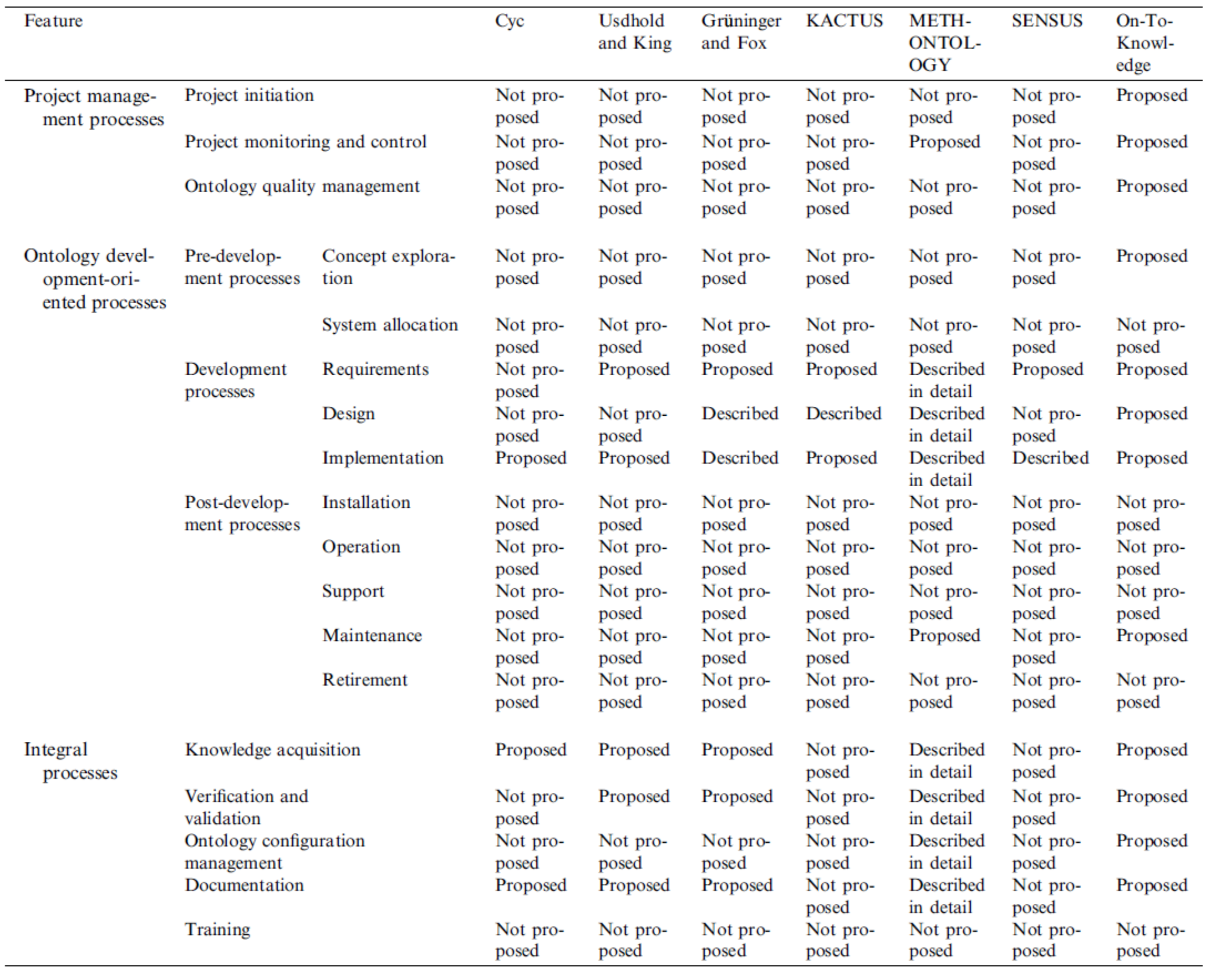
Table 2 shows a number of methodologies developed for building ontologies. Firstly, none of the approaches presented is fully mature if we compare them with software engineering and knowledge engineering methodologies. As summarised in Table 2, many key activities are not proposed by most of them. The most mature approach is METHONTOLOGY, which has been recommended by FIPA for the ontology construction task. Secondly, current proposals are not unified: each group applies its own approach. Consequently, great effort is required for creating a consensual methodology for ontology construction. Collaboration between different groups to unify their approaches seems the most reasonable way to achieve it (Corcho, Fernández-López, and Gómez-Pérez 2003).

3.5.2 Ontology tools

Tools that are used for constructing, editing, annotating and merging ontologies are called ontology tools. Without any kind of tool support, the implementation of ontologies in an ontology-language is a complex and time-consuming task. To ease this task, various ontology-building environments have been created by several research groups and software development organisations. The ontology tools have been classified into many types in (Abburu and Babu 2013), here we show the ontology development tools only.

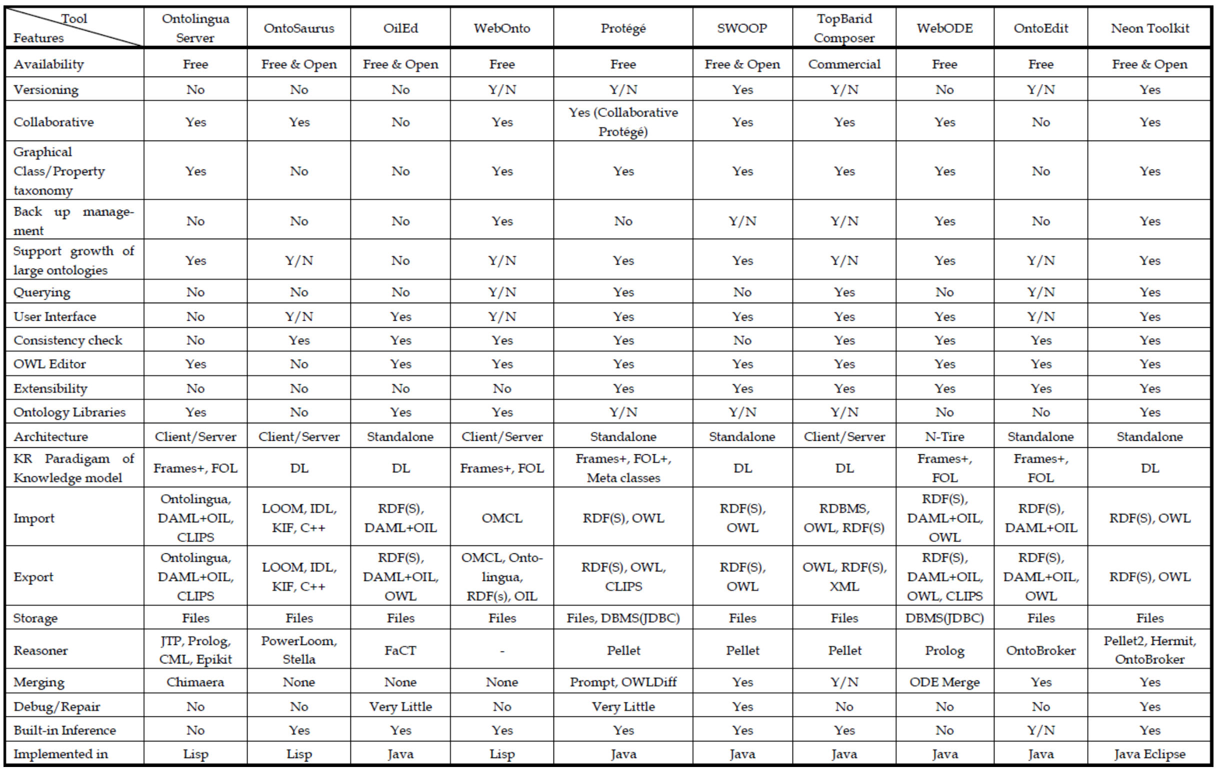
Table 3 shows various ontology development tools. Many of them are open source tools, except TopBraid Composer. Only the tools SWOOP and NeOn toolkit provide versioning features. Almost all the tools provide environment to build ontologies collaboratively except OilEd and On-toEdit. Only the tools WebOnto, WebODE and NeOn toolkit provide backup management functionalities. In short, the tools Protégé and NeOn toolkit provide most extensive functionalities for developing ontologies.

Table 2. A comparison of methodologies for building ontologies (Corcho et al. 2003)



Feature			Cyc	Usdhold and King	Grüninger and Fox	KACTUS	METH-ONTOL-OGY	SENSUS	On-To-Knowl-edge
Project management processes	Project initiation		Not proposed	Not proposed	Not proposed	Not proposed	Not proposed	Not proposed	Proposed
	Project monitoring and control		Not proposed	Not proposed	Not proposed	Not proposed	Proposed	Not proposed	Proposed
	Ontology quality management		Not proposed	Not proposed	Not proposed	Not proposed	Not proposed	Not proposed	Proposed
Ontology development-oriented processes	Pre-development processes	Concept exploration	Not proposed	Not proposed	Not proposed	Not proposed	Not proposed	Not proposed	Proposed
		System allocation	Not proposed	Not proposed	Not proposed	Not proposed	Not proposed	Not proposed	Not proposed
	Development processes	Requirements	Not proposed	Proposed	Proposed	Proposed	Described in detail	Proposed	Proposed
		Design	Not proposed	Not proposed	Described	Described	Described in detail	Not proposed	Proposed
		Implementation	Proposed	Proposed	Described	Proposed	Described in detail	Described	Proposed
	Post-development processes	Installation	Not proposed	Not proposed	Not proposed	Not proposed	Not proposed	Not proposed	Not proposed
		Operation	Not proposed	Not proposed	Not proposed	Not proposed	Not proposed	Not proposed	Not proposed
		Support	Not proposed	Not proposed	Not proposed	Not proposed	Not proposed	Not proposed	Not proposed
		Maintenance	Not proposed	Not proposed	Not proposed	Not proposed	Proposed	Not proposed	Proposed
		Retirement	Not proposed	Not proposed	Not proposed	Not proposed	Not proposed	Not proposed	Not proposed
Integral processes	Knowledge acquisition		Proposed	Proposed	Proposed	Not proposed	Described in detail	Not proposed	Proposed
	Verification and validation		Not proposed	Proposed	Proposed	Not proposed	Described in detail	Not proposed	Proposed
	Ontology configuration management		Not proposed	Not proposed	Not proposed	Not proposed	Described in detail	Not proposed	Proposed
	Documentation		Proposed	Proposed	Proposed	Not proposed	Described in detail	Not proposed	Proposed
	Training		Not proposed	Not proposed	Not proposed	Not proposed	Not proposed	Not proposed	Not proposed

Table 3. Comparison of Ontology Development tools (Abburu and Babu 2013)



Tool / Features	Ontolingua Server	OntoSaurus	OilEd	WebOnto	Protégé	SWOOP	TopBarid Composer	WebODE	OntoEdit	Neon Toolkit
Availability	Free	Free & Open	Free & Open	Free	Free	Free & Open	Commercial	Free	Free	Free & Open
Versioning	No	No	No	Y/N	Y/N	Yes	Y/N	No	Y/N	Yes
Collaborative	Yes	Yes	No	Yes	Yes (Collaborative Protégé)	Yes	Yes	Yes	No	Yes
Graphical Class/Property taxonomy	Yes	No	No	Yes	Yes	Yes	Yes	Yes	No	Yes
Back up management	No	No	No	Yes	No	Y/N	Y/N	Yes	No	Yes
Support growth of large ontologies	Yes	Y/N	No	Y/N	Yes	Yes	Y/N	Yes	Y/N	Yes
Querying	No	No	No	Y/N	Yes	No	Yes	No	Y/N	Yes
User Interface	No	Y/N	Yes	Y/N	Yes	Yes	Yes	Yes	Y/N	Yes
Consistency check	No	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes
OWL Editor	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Extensibility	No	No	No	No	Yes	Yes	Yes	Yes	Yes	Yes
Ontology Libraries	Yes	No	Yes	Yes	Y/N	Y/N	Y/N	No	No	Yes
Architecture	Client/Server	Client/Server	Standalone	Client/Server	Standalone	Standalone	Client/Server	N-Tire	Standalone	Standalone
KR Paradigam of Knowledge model	Frames+, FOL	DL	DL	Frames+, FOL	Frames+, FOL+, Meta classes	DL	DL	Frames+, FOL	Frames+, FOL	DL
Import	Ontolingua, DAML+OIL, CLIPS	LOOM, IDL, KIF, C++	RDF(S), DAML+OIL	OMCL	RDF(S), OWL	RDF(S), OWL	RDBMS, OWL, RDF(S)	RDF(S), DAML+OIL, OWL	RDF(S), DAML+OIL	RDF(S), OWL
Export	Ontolingua, DAML+OIL, CLIPS	LOOM, IDL, KIF, C++	RDF(S), DAML+OIL, OWL	OMCL, Onto-lingua, RDF(s), OIL	RDF(S), OWL, CLIPS	RDF(S), OWL	OWL, RDF(S), XML	RDF(S), DAML+OIL, OWL, CLIPS	RDF(S), DAML+OIL, OWL	RDF(S), OWL
Storage	Files	Files	Files	Files	Files, DBMS(JDBC)	Files	Files	DBMS(JDBC)	Files	Files
Reasoner	JTP, Prolog, CML, Epikit	PowerLoom, Stella	FaCT	-	Pellet	Pellet	Pellet	Prolog	OntoBroker	Pellet2, Hermit, OntoBroker
Merging	Chimaera	None	None	None	Prompt, OWLDiff	Yes	Y/N	ODE Merge	Yes	Yes
Debug/Repair	No	No	Very Little	No	Very Little	Yes	No	No	No	Yes
Built-in Inference	No	Yes	Yes	Yes	Yes	Yes	Yes	No	Y/N	Yes
Implemented in	Lisp	Lisp	Java	Lisp	Java	Java	Java	Java	Java	Java Eclipse

Note: “Yes” indicates a supported feature in the language, No indicates unsupported features, and Y/N indicates features that need further explanation.

4 Engineering domains utilising knowledge engineering

Ontologies have been applied in several engineering domains for different purposes. This section highlights some of the work that has been done previously for utilising ontologies in various industrial operations. For full details, the reader is advised to check the referenced literature.

4.1 Ontology for industrial process monitoring

A process plant is a very information rich domain. Tens of thousands of measurements track the system state and operation of the process equipment. Useful information is also stored in various IT systems, such as, process diaries, maintenance databases, measurement databases, and laboratory information management systems. As many of these systems may be products of different vendors, the variety of data formats and system interfaces used are often difficult to integrate, thus looking through all the relevant information, especially in a busy situation, is simply not possible. In addition, part of the information is free-form textual descriptions, which makes it even harder to search for the relevant information. As a result, process operators are left with an insufficient understanding of the overall situation, sometimes leading to confusion or even misjudged actions.

The study by (Pakonen et al. 2007) focused on the use of Semantic Web and information agent technologies in the context of industrial process monitoring. Monitoring of industrial processes aims at detecting disturbances as early as possible in order to maintain efficient process functioning and minimise losses in production. When dealing with highly automated processes, monitoring is the main content of work during normal plant operation. Typically, the responsibility of a process operator consists of selection and retrieval of relevant information, interpretation, and decision-making. Pakonen et al. developed a multi-agent system intended to support human users (processor operators) in tasks related to operational monitoring and maintenance of process plants, by providing them with autonomous, easily configurable monitoring services intended to extend their situation awareness. In the presented architecture, the process automation system has been augmented with an agent platform. The agents extract and refine information from the underlying control hardware (e.g. PLC, DCS, PC) and from different plant IT systems. All information extracted by the agents is mapped semantically to the common ontology.

For demonstration purposes, Pakonen et al. developed a process monitoring domain ontology in OWL using the Protégé ontology editor. The domain ontology integrates information from heterogeneous data sources, links data with other information, and enables complex queries. The domain ontology was split to sub-ontologies (maintenance ontology, physical equipment ontology and functional domain ontology). A tentative implementation of the agent architecture was built using publically available, open-source Java tools. A BDI-based (Belief-Desire-Intention) agent development tool called Jadex was selected as the agent platform. Jena toolkit was used for OWL processing. A browser-based user interface was created using the readily available Tomcat server interface for Jadex. In the demonstration, the operator is provided with an information search service, enabling him/her to look for interesting events occurred e.g. in the previous shifts. In a problem situation, the operator can also look for previous occurrences of similar problems, how they were dealt with, and what the results were. The agents then search for the events matching the user-defined criteria, and combine them with related measurements. The study demonstrates three test runs highlighting the importance of ontologies in process monitoring. The services aim to increase the overall situation awareness of the users by combining information from various heterogeneous data sources using a common ontology for data representation and query.

4.2 Engineering design knowledge of automatic assembly systems for manufacturing electronic components

The collaborative engineering design is a very complex process that involves extensive engineering knowledge from various disciplines at different design phases, covering mechanical, electrical, automation, kinetic, structural, dynamic, thermal, magnetic, and optical contexts. Additionally, these resources are often distributed geographically and represented in heterogeneous formats, thus making effective capture, retrieval, reuse, sharing and exchange of knowledge a critical issue in a distributed design environment.

The study by (Zhang and Yin 2008) illustrated the engineering design knowledge of automatic assembly systems for manufacturing electronic components, which contain a group of electro-mechanical components, in OWL format in order to facilitate semantic access and retrieval of electro-mechanical component information across different disciplines. The domain knowledge was built from interviews with domain experts and the ontology was developed using Protégé 2000. An ontology query language OWL-QL was used for semantic search that exploits the domain ontologies, semantic indexes, and semantic relationships built in the ontology models.

4.3 Integration and consolidation of distributed design data in chemical process engineering

During the design phase of a chemical plant, information is created by various software tools and stored in heterogeneous formats, such as technical documents, CAE databases, or simulation files. Eventually, these scattered information items need to be merged and consolidated. The study by (Wiesner, Morbach, and Marquardt 2011) presented a prototypical ontology-based software tool for the integration and consolidation of distributed design data in chemical process engineering. The deductive ontology language F-Logic and the related inference engine OntoBroker (Angele, Kifer, and Lausen 2009) was used. For the modelling of ontologies in F-Logic, the design environment OntoStudio (Weiten 2009) was chosen as an implementation basis. The core of their approach was an expressive knowledge base, which is based on the formal ontology OntoCAPE.

4.4 Knowledge sharing and reuse in PFMEA domain

Potential Failure Modes and Effects Analysis in Manufacturing and Assembly processes (PFMEA) is an important preventive method for quality assurance, and through it, the decisions based on the severity levels and probabilities of occurrences and detection of the failure modes can be planned and prioritised, seeking to improve the quality of the manufactured products. This activity generates a valuable source of knowledge about the manufacturing processes in the company. The study by (Mikos et al. 2011) described the development of a system for distributed knowledge sharing and reuse in the PFMEA domain using ontology-based knowledge retrieval approach. A prototype was implemented based on the Java Agent Development Framework (JADE). In the proposed architecture, the different knowledge bases can be distributed over the intranet/internet. In their work, the so-called METHONTOLOGY methodology proposed by (López et al. 1999) was adopted. The PFMEA-DL ontology was implemented through the OWL-DL and the Protégé-OWL Ontology Editor was used to construct and store the knowledge considered in the work. The RacerPro Server System (2008) was chosen as DL reasoning engine responsible for the inference service and knowledge retrieval, which is accessed by a TCP/IP communication port.

4.5 Ontological framework supporting decision-making in ETO products

Engineer-To-Order (ETO) is a type of manufacturing process for highly customised products, which are required to be designed and engineered in detail as per the specifications in the order placed by customers. One of the major problems associated with ETO products is their long lead-time due to communication delays among the agents. The study by (Pandit and Zhu 2007) proposed using ontology as a technical solution to integrate heterogeneous systems. An ontological framework was developed using OWL that supports the generation, analysis, sharing and reuse of domain knowledge as required by ETO business processes, thus providing information to support many information intensive business processes such as the evaluation of design alternatives. A case study (selection of transformer fulfilling a set of requirements) was presented to demonstrate the use of the ontology. The Analytical Hierarchy Process (AHP) and Expert Choice 11, a software tool by the Expert Choice (http://www.expertchoice.com/), were used for the case study.

4.6 Ontology-based assembly design and information sharing for collaborative product development

The study by (Kim et al. 2006) introduced a framework in which an assembly design (AsD) serves as a formal, explicit specification of assembly design making the assembly knowledge both machine-interpretable and sharable. The ontology model represents engineering, spatial, assembly, and joining relations of assembly in a way that promotes collaborative assembly information-sharing environments. The OWL (Web Ontology Language) and SWRL (Semantic Web Rule Language) were used to explicitly represent the implicit AsD constraints.

4.7 Engineering information retrieval from various documents

With the extensive use of computer-aided systems in product design, manufacture and analysis, a mass of documents is produced. These documents provide engineering information in integrated environment of enterprise for engineers, but maintained by different systems in different forms. Current information retrieval approaches usually lack semantic supports for different kinds of documents, which leads to insufficiency of content representation and misunderstanding of query intention. To effectively search engineering information in various documents and to consider semantic information in engineering domain, an ontology-based information retrieval framework was developed in the study by (Yao et al. 2009). As the centre of framework, ontologies were established to support document analysis and query processing through information representation in semantic level, and complete the mapping between user queries and document resources. The framework provides a unified platform for multi-source engineering information retrieval from various documents in integrated environment.

4.8 A bottom-up approach for building ontology from engineering documents

There have been many methodologies proposed and majority of them are using the top-down approaches, which do not maximise the benefits of bottom-up approaches. The study by (Park et al. 2008) proposed a systematic methodology to develop the ontology in a bottom-up style from engineering documents, called DocOnto (Document-based Ontology). The methodology is mainly composed of three phases such as defining ontology for terms in engineering documents, integrating the ontology with semantic networks for both a single document and a focused document group, and pruning the ontology for practical usage. In the approach, first-order logic (FOL) and semantic networks (SN) were used for formal and visual representation of ontology, and semantic mapping with similarity evaluation was used in integrating the ontology. The approach presented by Park et al. can be computerised for structured engineering documents.

4.9 Building information ontology

The Architecture, Engineering, Construction and Facility Management (AEC-FM) projects involve a large number of participants that must exchange information and combine their knowledge for successful completion of a project. Currently, most of the AEC-FM domains store their information about a project in text documents or use XML, relational, or object-oriented formats that make information integration difficult.

The study by (Niknam and Karshenas 2017) presented a shared ontology approach to semantically represent building information, which facilitates finding and integrating information distributed in several knowledge bases. The NeOn methodology was used as it provides flexibility for a variety of scenarios instead of prescribing a rigid workflow. The developed ontology was named as BIM Shared Ontology (BIMSO). The purpose of BIMSO was to provide a conceptual knowledge model for buildings that can be used by different building domains for developing domain ontologies. Every AEC-FM domain creates its own domain ontology by adding domain-related properties and relationships to BIMSO elements. They explained how building design ontology (BIMDO) can be built on top of BIMSO ontology. As a case study, they used the building of an Engineering Hall, which was designed using Autodesk Revit BIM platform. For validation purpose, SPARQL query language was used to retrieve knowledge base information.

4.10 Ontological representation of photovoltaic system information

A conceptual model for representing information about Photovoltaic (PV) system was developed in the study by (Tah and Abanda 2011). The main purpose of the ontology was to facilitate clients during the selection of PV systems for different applications. The development of the prototype ontology was facilitated by the Protégé-OWL editor. The study also discussed the ontology anomaly checking. Currently, there exist two major methods of performing anomalies checking of an ontology, i.e. manually and automatically. Automatic checking is achieved through the use of DL reasoners. Tah & Abanda adopted the automatic checking method for consistency checking of the PV ontology. This choice was guided by the availability of the reasoner FaCT++ plug-in incorporated in Protégé 4.0.2.

4.11 Formal ontologies for identifying and classifying the fundamental analysis modelling knowledge in manufacturing enterprises

The design of virtually all engineered products require the services of engineering analyses to predict the behaviour of the product and/or its manufacturing processes for evaluation and optimisation of the product design in terms of its intended function, reliability, quality, manufacturability, and so forth. The manufacturing enterprises rely heavily on engineering analyses to provide the information to inform design decisions as quickly and as cost effectively as possible. Considerable modelling knowledge has been invested in the development of these analysis models. In order to represent and operationalise this knowledge in ways that facilitate successful exchange, reuse, adaptation, or interoperation of analysis models, the study by (Grosse, Milton-Benoit, and Wileden 2005) proposed a set of formal ontologies for identifying and classifying the fundamental analysis modelling knowledge. These ontologies were then implemented into a Java-based object-oriented information structure, which was instantiated with a real-world industrial analysis model to form the basis for an engineering analysis modelling knowledge base system called ON-TEAM.

5 Semantic Web technologies in AEC domain

Projects in the domain of Architecture, Engineering and Construction (AEC) typically involve diverse parties, each bringing specific information into the project. The combination of this information plays a crucial role in the project success at the design phase. For instance, client information needs to be combined with the information of the architectural design firm, electrical engineering information needs to be combined with facility management information, and plumbing information needs to be combined with sensor information. Also after the construction phase, building information needs to be accessible for a range of diverse users, including the facility director, in-house machinery and systems, renovation specialists, technicians, and so forth. As a result, a well-functioning information flow throughout the complete Building Life Cycle (BLC) is crucial.

A number of strategies exist for sharing (building) information within the AEC domain for enabling an improved level of interoperability. Some of these strategies have been outlined in (Pauwels, De Meyer, and Van Campenhout 2010), (Pauwels et al. 2011) and (Törmä 2013). In short, a distinction can be made between the following three approaches:

- Sharing in the wild
- Centralised model
- Linked building data

Figure 1 shows the first two approaches for sharing information. On the left, all the information in a building project is considered crucial and the diverse people translate or convert it manually to their own model of the building project, which is a tedious task. On the right, only the centre model (BIM model) is updated by all parties. Problem is that not all information fits into the central model, whether this is through the IFC file format or any other industrial standard.

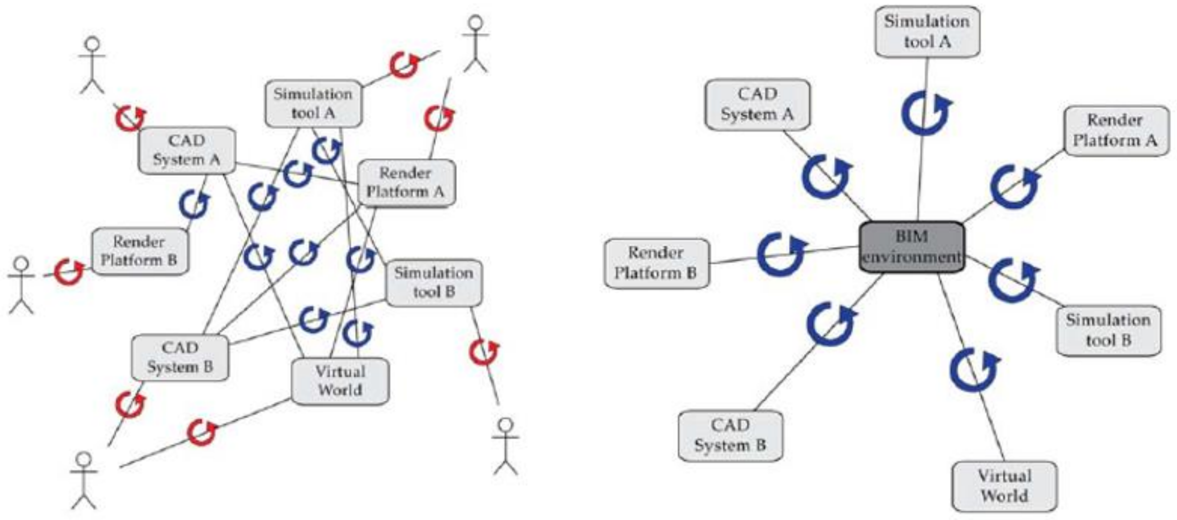


Figure 1: When sharing information "in the wild" (left), end users need to manually compare new information with information that they manage. In the centralised model strategy (right), information is stored in one central location, and other applications refer to that information only and store to that central model only. (Pauwels, Corry, and O'Donnell 2014)

Figure 2 shows information sharing following the third approach. In the "*linked building data"* approach, diverse information sources used within a building project are linked together as needed. A management system is set up for managing links between the diverse models. The issue with this approach is that linked building data consists mainly of pairwise links, resulting in loosely connected data resembling more to the "*sharing in the wild*" approach.

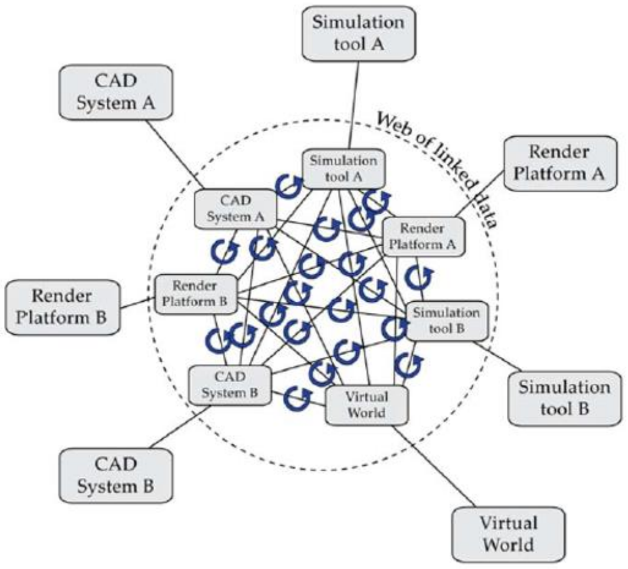


Figure 2: In a linked building data strategy, information keeps following the format that is required by the application that is using it (nodes outside dashed circle). Additionally, it is tightly linked together on data-level (nodes inside dashed circle), so that information in one format/application can be related to information in another format/application. (Pauwels et al. 2014)

The role of Semantic Web technologies in the AEC industry has been presented in a multitude of research articles. Here some of these articles covering various aspects of AEC domain are briefly introduced in the sub-sections 5.1 to 5.10, and the reader is advised to look for further details in the referenced articles.

5.1 Introduction to SimModel

The development of SimModel (an interoperable, structured and yet easily extensible data model) enables improved inter-disciplinary data exchange within the simulation domain. This model leverages original data (such as the building geometry as defined by an architect) in whole building energy simulation, computational fluid dynamics (CFD), fire and safety simulation and others. Re-use of geometric and other data from different models significantly reduces the overhead associated with the definition of input data and eliminates error-prone manual processes.

SimModel was developed to seamlessly import and export data relevant to EnergyPlus and other simulation software, and it is compatible with main BIM software used in the building industry (for instance, Autodesk Revit, ArchiCAD). Figure 3 shows interoperable data exchange enabled by SimModel (O’Donnell et al. 2013).

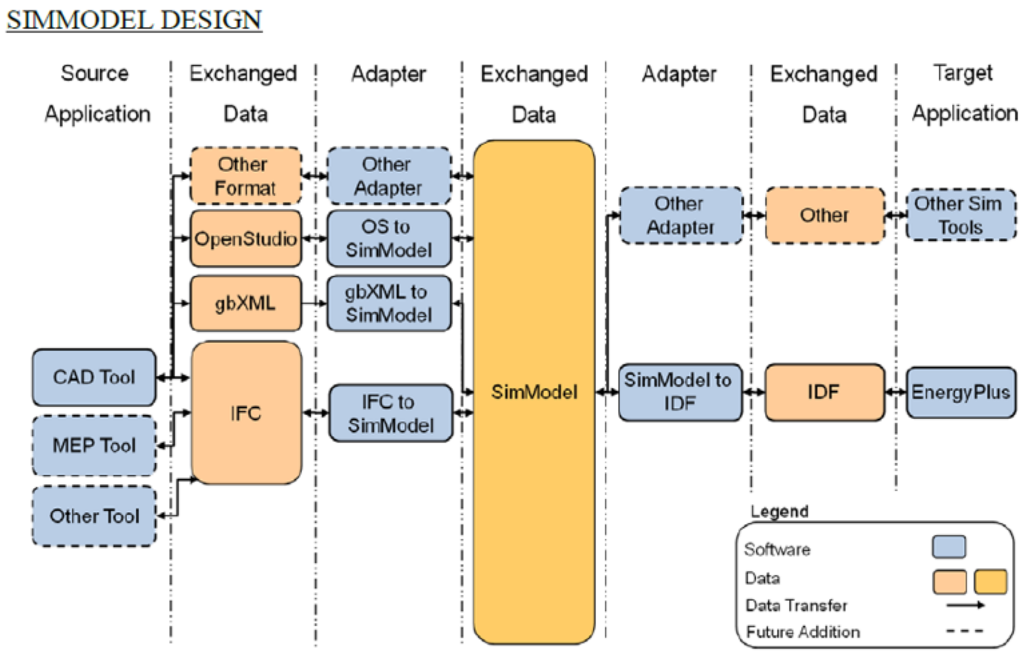


Figure 3: Interoperable data exchange enabled by SimModel. The solution enables re-use of original project data as contained in IFC based BIM (bi-directional mapping) and data from other sources (import only). (O’Donnell et al. 2013)

5.2 Converting SimModel XSD files to OWL ontology files

In order to be able to better integrate SimModel information with other building information, P. Pauwels et al. aimed at representing this information in RDF. A conversion service has been built that is able to parse the SimModel ontology in the form of XSD schemas and output a SimModel ontology in OWL. Once this OWL ontology is available, information can be represented in RDF graphs and the Semantic Web technologies can then be used for handling the represented information and, finally, the linked building data approach can be realised for Building Energy Performance (BEP) simulation data.

Simergy software

Data flow is possible to and from BIM models in IFC, DOE-2 software or tools that use the DOE-2 engine, EnergyPlus, and tools with gbXML export. These tools are typically used for BEP simulations. Data from any of these environments can be mapped to and from the SimModel data model using the Simergy software (LBNL 2013).

In the converter application, each of the five XSD files is parsed and converted into a corresponding OWL ontology file, while keeping track of the cross references. Each class or subtype is converted into an OWL class (*owl:Class*), referring to an upper class when required. For each class, the required properties are generated as *owl:DatatypeProperty* or *owl:ObjectProperty* declarations, resulting in a complete representation of the SimModel in five ontology files (Pauwels et al. 2014).

5.3 Converting SimModel information to RDF graphs

Making SimModel information available to Semantic Web technologies allows BEP information to link with information that is available from various other sources. Among these sources is the Linked Open Data (LOD) cloud, (Linkeddata.org 2014) and (Bizer, Heath, and Berners-Lee n.d.), which combines all kinds of information available world-wide (geographical, general knowledge, material properties, services, and so forth). Furthermore, the BEP simulation information in RDF graphs of SimModel can be linked to IFC building models that are converted through the IFC-to-RDF converter service (LBD 2013). Using the IFC & SimModel conversion services, a web of “linked building data” many thus be targeted, which combines diverse sources of information for particular buildings with the aim of improving information exchange throughout the building life cycle (Törmä 2013).

About the converter

In the converter application, the complete XML document is parsed and de-serialised into JAVA class instances and property instances. The JAVA classes and properties were previously generated during the conversion of the XSD schema into OWL ontology files. At the end of the serialisation process, a SimModel instance is available in-memory. These instances are then iterated again and converted into the corresponding RDF instances using JENA. The resulting JENA model can then be exported into files or triple stores as required (Pauwels and Van Deursen 2012).

5.4 Computing *diffs* resulting from the IFC-to-RDF graph conversions

BIM models can be exported into IFC format and converted into RDF graph. Links can then be generated between the URIs of entities across the models to support cross-model information access and change management. IFC models consist of anonymous objects without stable identities. The goal of the research (Oraskari and Törmä 2015) is to develop efficient algorithms for computing *diffs (sets of deletions and additions between versions)* during the successive versions of IFC-derived RDF graphs. The approach is to compute unique and stable identities for all blank nodes (anonymous objects with no GUID – globally unique identity) i.e. to assign the same identities to the same objects in different versions. As the main result, Oraskari & Törmä presented the Short Paths Crossings Algorithm (SPCA), which determines the identity of each blank node by computing a set of paths with limited lengths starting from the node.

5.5 Linking RDF graph of one ontology to the RDF graph of another ontology

Figure 4 is the RDF graph of the *Book Tower building*, which is obtained by converting the IFC model of the building using the IFC-to-RDF graph converter. (Pauwels et al. 2013) explains how to link RDF graph of a building (named as Book Tower) to different information available externally.

When linking the building to such extra information using the Semantic Web technologies, other information becomes readily available. For instance, when making the link between a building and a geographical location, the information about the particular location of the building becomes available as expert geographers described it. Also when linking building elements to material information, additional information becomes available about the particular material, as it was described by the people who made this material. Hence, an end user can relatively easily find out which construction techniques are used and how he/she should act in a proper way in order to maintain and renovate the building. Figure 5 shows linking RDF graph of Book Tower to the Geonames ontology (Pauwels et al. 2013).

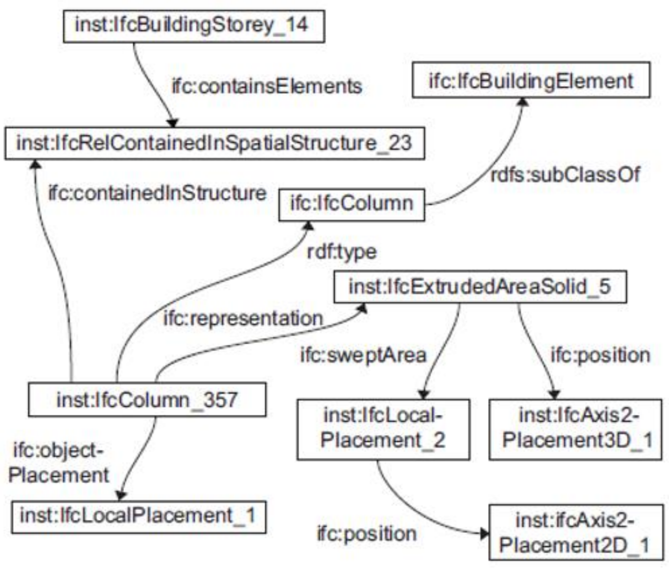


Figure 4: RDF graph displaying building information for one of the columns in the Book Tower according to the IFC ontology (Pauwels et al. 2013)

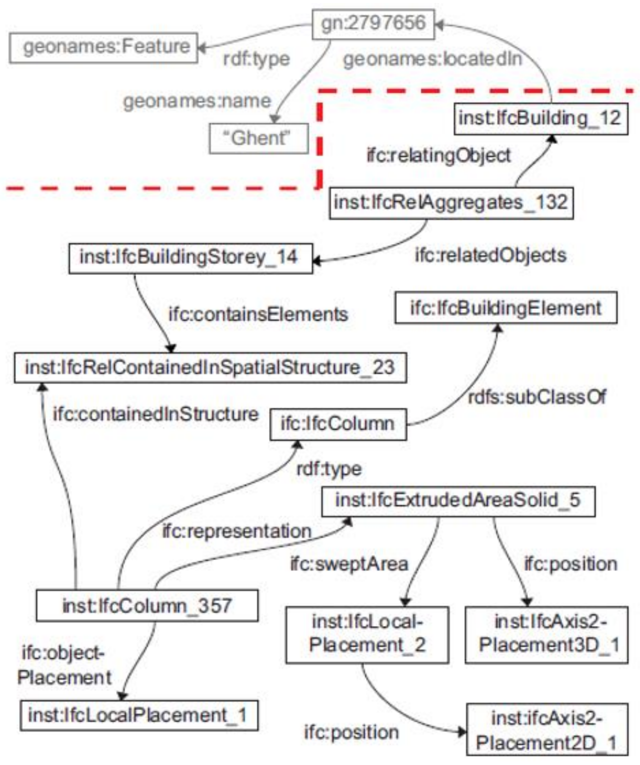


Figure 5: RDF graph displaying the link that is set up between instances in the IFC/RDF graph and instances in the RDF graph of the Geonames ontology (Pauwels et al. 2013)

5.6 Simplifying RDF graphs to address needs of specific industrial use cases

The production of several OWL ontologies allow to capture building data in RDF graphs. For example, an ifcOWL ontology allows capturing IFC data in a RDF graph. As the building data is now available in a semantic graph with an explicit formal basis, it can be restructured and simplified so that it more easily matches the different requirements associated with practical use case scenarios. The paper by (Pauwels and Roxin 2016) investigates several proposals and technological approaches to simplify ifcOWL building data, thus addressing the needs of specific industrial use cases. Pauwels & Roxin defined and exemplified four main approaches towards simplification of building models represented in ifcOWL graphs. These simplifications can be dynamically applied on existing ifcOWL graphs. Average simplification percentages can be obtained of 89.68% in terms of triple count and 91.58% in terms of file size.

5.7 Role of linked data and Semantic Web in building operation

The effective Decision Support Systems (DSS) for building service managers require adequate performance data from many building data silos in order to deliver a more complete view of building performance. Current performance analysis techniques tend to focus on a limited number of data sources, such as data measured in a Building Management System (temperature, humidity, CO_2), excluding a wealth of other data sources increasingly available in the modern building, including weather data, occupant feedback, mobile sensors and feedback systems, schedule information, and equipment usage information.

As part of a wider decision support framework for key building stakeholders, the paper by (Corry et al. 2013) presents a data driven approach to the structured performance assessment of buildings, utilising Semantic Web technologies and performance metrics. Taking an existing 14000m^2 naturally ventilated university building, the authors illustrate how diverse building data streams might be exposed and used to drive decision support for building operators, in the area of occupant satisfaction and performance optimisation.

5.8 Integration of serval ontologies

Semantic heterogeneity remains a problem when integrating data from various ontologies, which model the same information in different ways. Different ontologists (ontology designers) can produce different ontologies for a same knowledge domain. Thus, merely adopting ontologies, like just using XML, does not eliminate heterogeneity for good and thus elevates heterogeneity problems at a higher level. Semantic heterogeneity exists whenever there is more than one way to structure a body of data. Therefore, in order to address the problem of semantic interoperability by means of ontologies, the study by (Farias, Roxin, and Nicolle 2015) proposes a loosely coupled federated architecture (FOWLA) for OWL ontologies. This architecture is based on ontology alignments, logical rules and inference mechanisms.

5.9 Querying and reasoning over large scale building datasets

The architectural design and construction domains work on a daily basis with massive amount of data. Properly managing, exchanging and exploiting this data is an ever-ongoing challenge in this domain, and has resulted in large semantic RDF graphs that are to be combined with a significant number of other data sets (building product catalogues, regulation data, geometric point cloud data, simulation data, sensor data), thus making an already huge dataset even larger. Making these big data available at high performance rates, speeds, and into the correct (intuitive) formats is therefore an incredibly high challenge in this domain. Yet, hardly any benchmark is available for the industry that (1) gives an overview of this kind of data typically handled in this domain, and (2) that lists the query and reasoning performance results in handling these data.

The authors in (Pauwels et al. 2016) present a set of available sample data that explicates the scale of the situation, and perform a query and reasoning performance benchmark. This results not only in an initial set of quantitative performance results, but also in recommendations in implementing a web-based system relying heavily on large semantic data. As such, Pauwels et al. propose an initial benchmark through which new upcoming data management proposals in the architectural design and construction domains can be measured.

5.10 Minimising query time

Data interoperability represents a great challenge for today's enterprises. Indeed, they use various information systems, each relying on several different models for data representation. Ontologies and notably ontology matching have been recognised as interesting approaches for solving the data interoperability problem. The study by (Farias, Roxin, and Nicolle 2016) focused on improving the performance of queries addressed over ontology alignments expressed through SWRL rules. Indeed, when considering the context of executing queries over complex and numerous alignments, the number of SWRL rules highly affects the query execution time. Moreover, when hybrid or backward-chaining reasoning is applied, the query execution time may grow exponentially. Still, the reasoners involved deliver performant results (in terms of execution time) when applied over reduced and simpler rule sets. Based on this statement, and to address the issue of improving the query execution time, Farias et al. described a novel approach that allows, for a given query, to ignore unnecessary rules. The proposed Rule Selector (RS) is a middleware between the considered systems and the reasoner present on the triple store side. Through the benchmarks realised, Farias et al. proved that the presented approach allows considerably minimising query execution time.

6 Ontologies in Modelling and Simulation

A large number of software packages or simulation engines exist for the support of computer simulations in different domains (engineering, medicine, learning, etc.). These packages are application-oriented, designed for a specific use in a specific domain; hence, they apply diverse modelling approaches, different technologies, domain specific terminologies and store simulation models and results in a variety of formats. This is presenting a challenge for,

- Comparing simulation models and results
- Reusing and sharing existing models
- Querying and making inferences

This section presents some of the earlier work that has been done to tackle these problems.

6.1 Mapping of domain simulator’s ontology to an upper ontology

The modern society relies on a variety of Critical Infrastructures (CI) consisting of power system networks, water distribution, oil and natural gas systems, telecommunication networks and many others. Interdependency between these systems is high and failure in one of the systems may result in cascading failures spanning different infrastructures. The behaviour of each CI can be observed and analysed through the use of domain simulators, but this does not account for their interdependency. In order to explore the CI interdependencies, the domain simulators need to be integrated for collaboration.

The paper by (Grolinger et al. 2011) explored three different simulators: the EPANET water distribution simulator, the PSCAD power system simulator and the I2Sim infrastructure interdependency simulator. The core modelling ontology, as well as the initial ontology mapping between each simulator was created. The ontologies and their mapping will support collaboration of simulators by enabling exchange of information in a semantic manner.

The starting point in creating ontologies for the integration of simulators is to understand how each simulator models the world. To establish a relationship among the modelling ontologies of I2Sim, EPANET and PSCAD, the components of these simulators are grouped into five common entities as; ***cells, controls, channels, meters*** and ***others***, check the bottom row of each simulator in Figure 6. These entities become concepts of the upper ontology. Concepts from each simulator’s ontology were mapped to the upper ontology that serves as the mediator between the simulators ontologies, as shown in Figure 7.

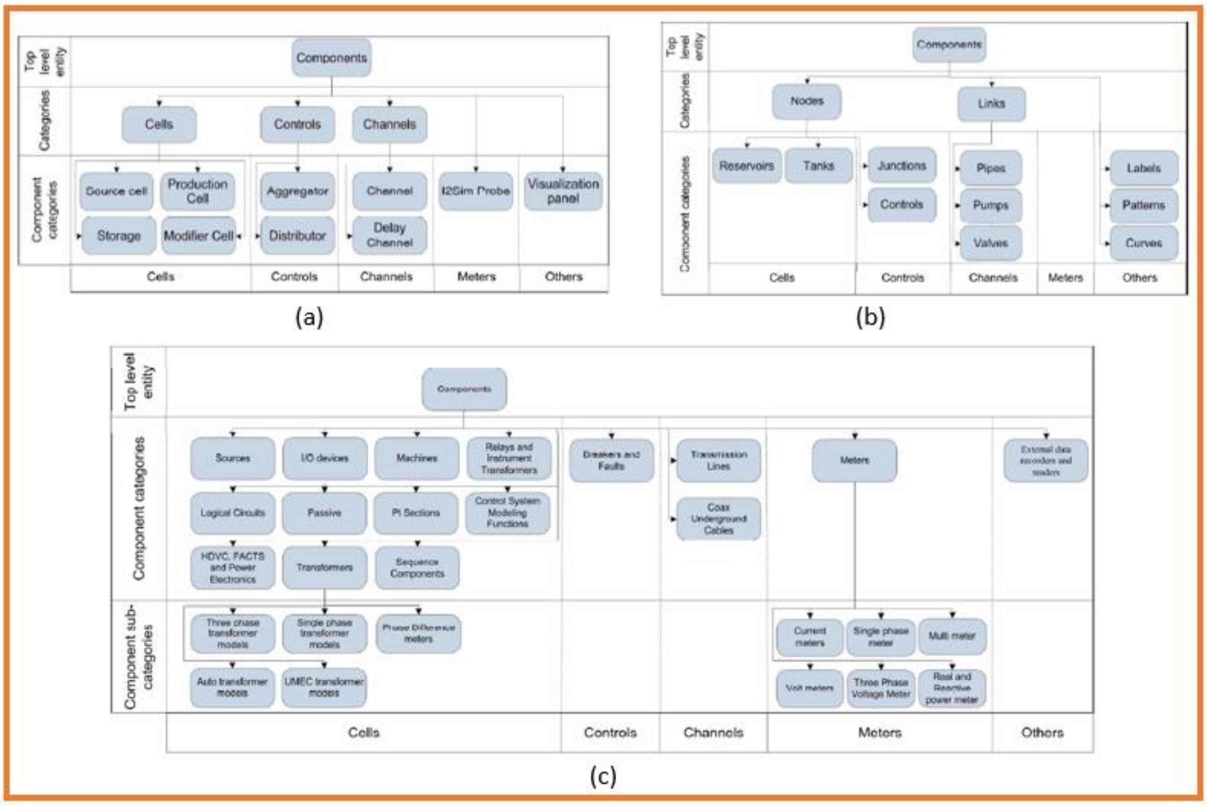


Figure 6: (a) I2Sim modelling ontology (b) EPANET modelling ontology (c) PSCAD modelling ontology. (Grolinger et al. 2011)

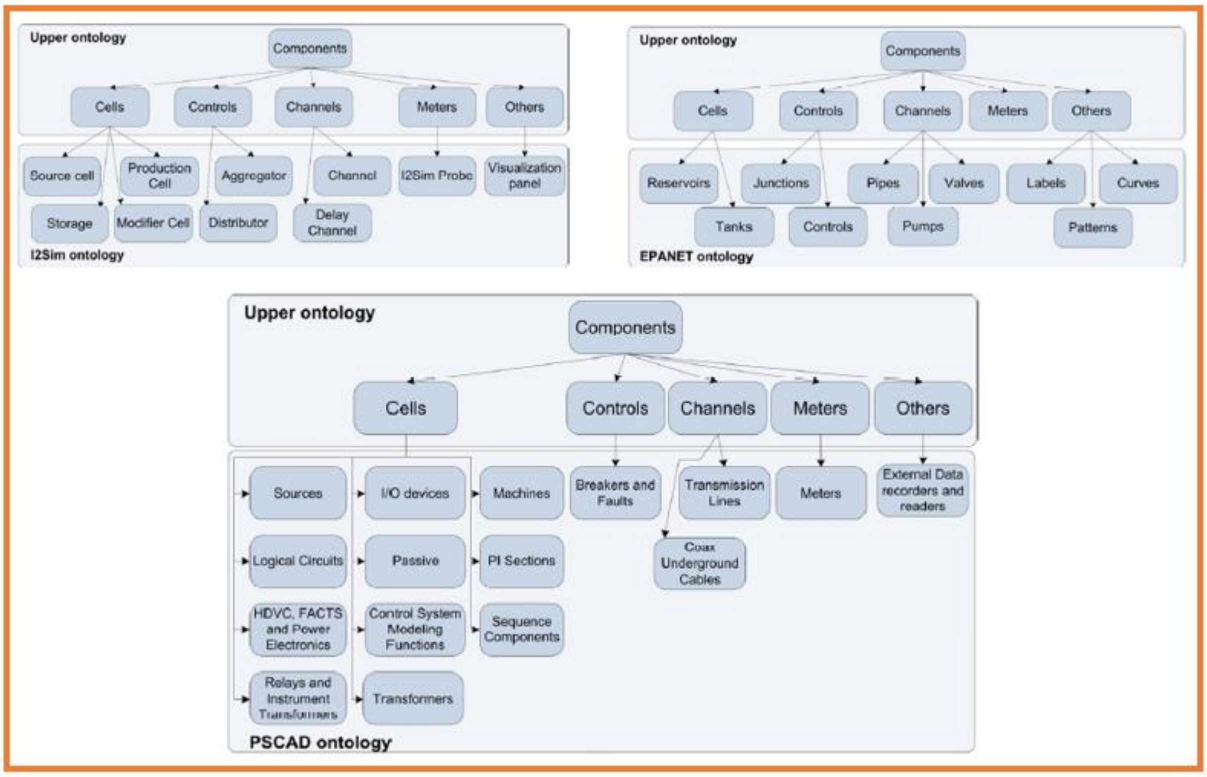


Figure 7: Concepts from each simulator's ontology mapped to the upper ontology (Grolinger et al. 2011)

6.2 Ontology-based representation of simulation models

The paper by (Grolinger et al. 2012) proposed the representation of domain simulation models as instances of Simulator's ontologies. By using the same formalism to represent various simulation models, Grolinger et al. placed them on the same platform, thus enabling a simplified comparison. The proposed approach used existing models in the simulation engine proprietary file formats as the foundation for the creation of its ontology-based representation. The ontology-based representation of simulation models has a layered architecture, as described in Figure 8.

- **Upper ontology layer:** contains generic concepts that are common for all simulation engines
- **Simulators' ontologies layer:** consists of ontologies that are specific to the actual simulator
- **Ontology-based simulation models layer:** contains ontology-based simulation models that are represented as instances of Simulators' ontologies
- **Rules layer:** is optional and contains a rule engine. This layer expresses design rules to which the simulation models should conform

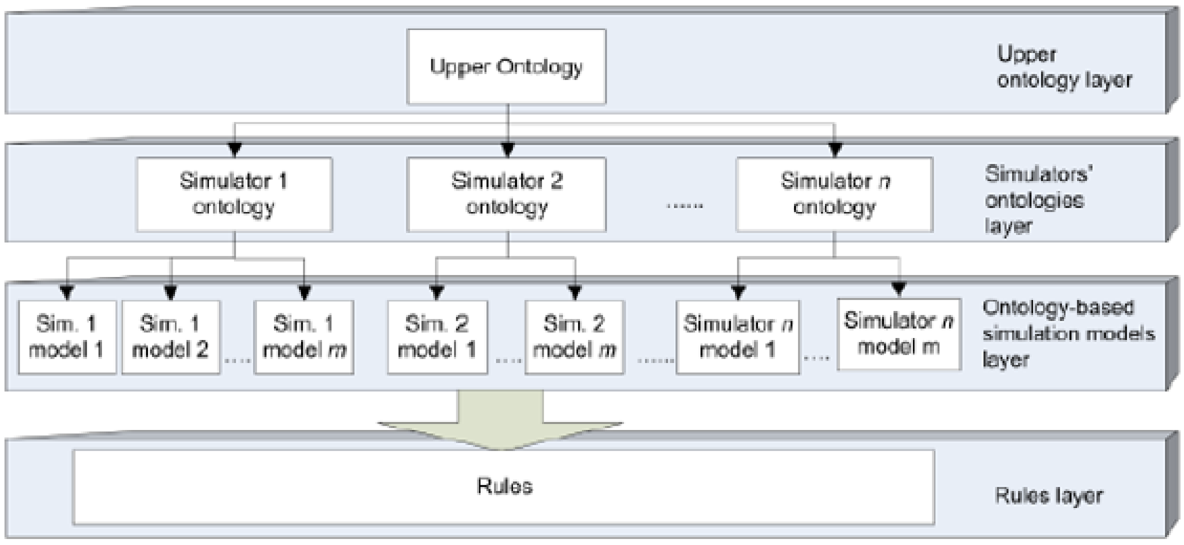


Figure 8: Architecture layers (Grolinger et al. 2012)

Defining relations between simulation model entities

In the ontology-based simulation model, entities are represented as instances of the Simulator's Ontology, while the relations among them are established by means of object properties. The description of the object properties is found in the upper ontology and the Simulators' Ontologies.

Figure 9 portrays the approach for the creation of an ontology-based simulation model representation. The Simulator's Ontology is simulator-specific, while the simulator's models are model-specific, as each model is stored in a separate file. The transformation engine consists of the following functions:

- **Ontology Reader:** responsible for acquiring information about simulator's classes and their properties

- **Simulation Model Reader:** has to be created for each simulator whose model requires transformation
- **Integrator:** uses the data received from the *Ontology* Reader and the *Simulation Model Reader*. Specifically, the *Integrator* receives info about the simulator's classes from the *Ontology Reader*. For each class, the *Integrator* obtains knowledge about its individuals from the *Simulation Model Reader*
- **Ontology Writer:** *Integrator* sends info about classes, individuals, data properties and object properties to the *Ontology Writer*, which writes an ontology-based simulation model representation

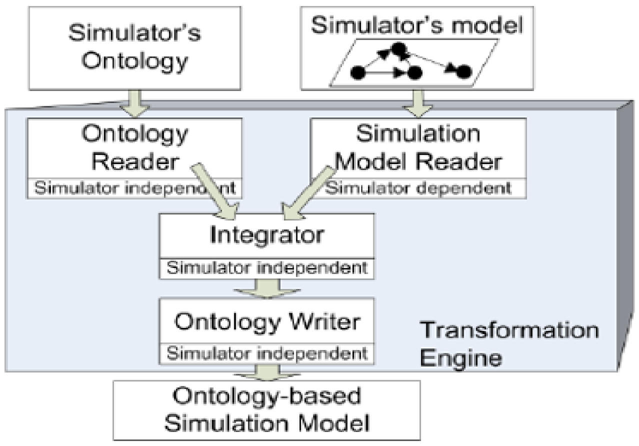


Figure 9: Ontology-based model creation from simulator's model (Grolinger et al. 2012)

As a case study, the proposed ontology-based representation of simulation models is evaluated using I2Sim infrastructure interdependencies simulator. Some of the benefits of ontology-based representation of simulation models are:

- Models from different simulation platforms are represented in a common manner
- Models can be queried using ontology querying languages
- Inferences can be performed using ontology reasoners

6.3 DeMO: An ontology for discrete-event M&S

Discrete-event Modelling Ontology (DeMO) is a general purposed Discrete Event Simulation (DES) ontology, which represents the domain of discrete-event modelling by describing the classic DES world views as well as many of the formalisms and modelling techniques that conform to the world views. DeMO might be used to support the increasing collaborative work among the M&S community.

The paper by (Silver et al. 2011) proposed a DeMO version of Ontology Driven Simulation (ODS) and discussed its practical implementation, which was realised in the development of the *DeMOforge tool*, as illustrated in Figure 10. DeMO may be used to support M&S, such as:

- Supporting model discovery via searches by modelling technique/formalism or domain ontology concept
- Supporting model reuse and component-based model development

- Providing formalised documentation of the model development process as by-product of ODS

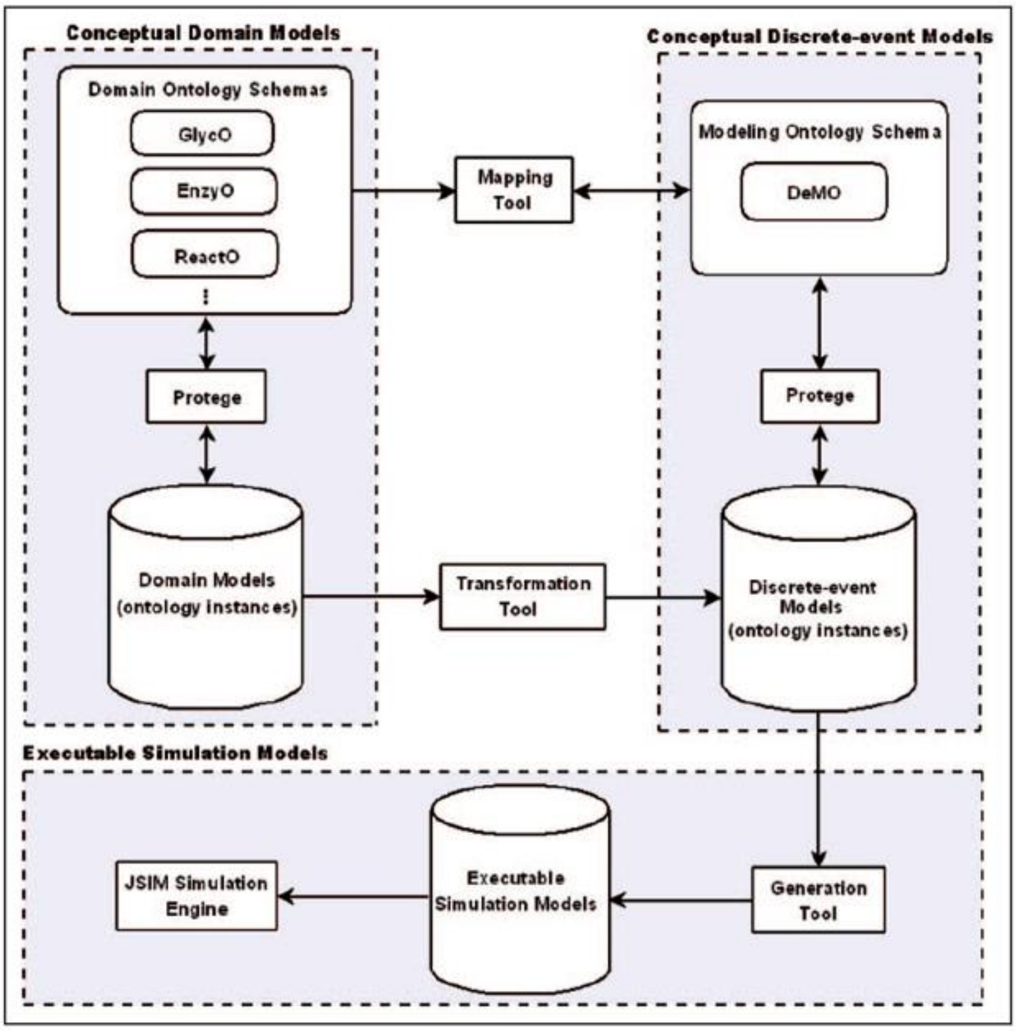


Figure 10: DeMOforge architecture (Silver et al. 2011)

6.4 From domain ontology to modelling ontology to executable ontology

When simulation models are developed using popular software packages, the reusability of the model is very limited for several reasons: (1) There is no formal way of specifying an agreed upon domain of discourse for the application domain of the process being modelled. (2) The same term may mean different things in different simulation software packages. (3) There is no commonly agreed format for representing and storing models.

The paper by (Silver, Hassan, and Miller 2007) proposed a tool suite, as shown in Figure 11. The *ODS Design Tool Suite* provides facilities for: (1) mapping concepts from domain ontologies to a modelling ontology in order to represent models as ontology instances, (2) translating ontology instances to an intermediate XML markup language, and (3) generating executable simulation models form markup language representations of models.

ODS takes advantage of this (ontology is machine readable) feature by using software tools to align knowledge resident in domain ontologies with knowledge resident in a modelling ontology in order to facilitate the creation of simulation models.

In ODS, a tool is used to map concepts from domain ontologies to concepts in a modelling ontology and then create instances of modelling ontology classes to represent a model. Once the ontology instances representing the model have been created, additional tools are used to translate the instances into an executable simulation model.

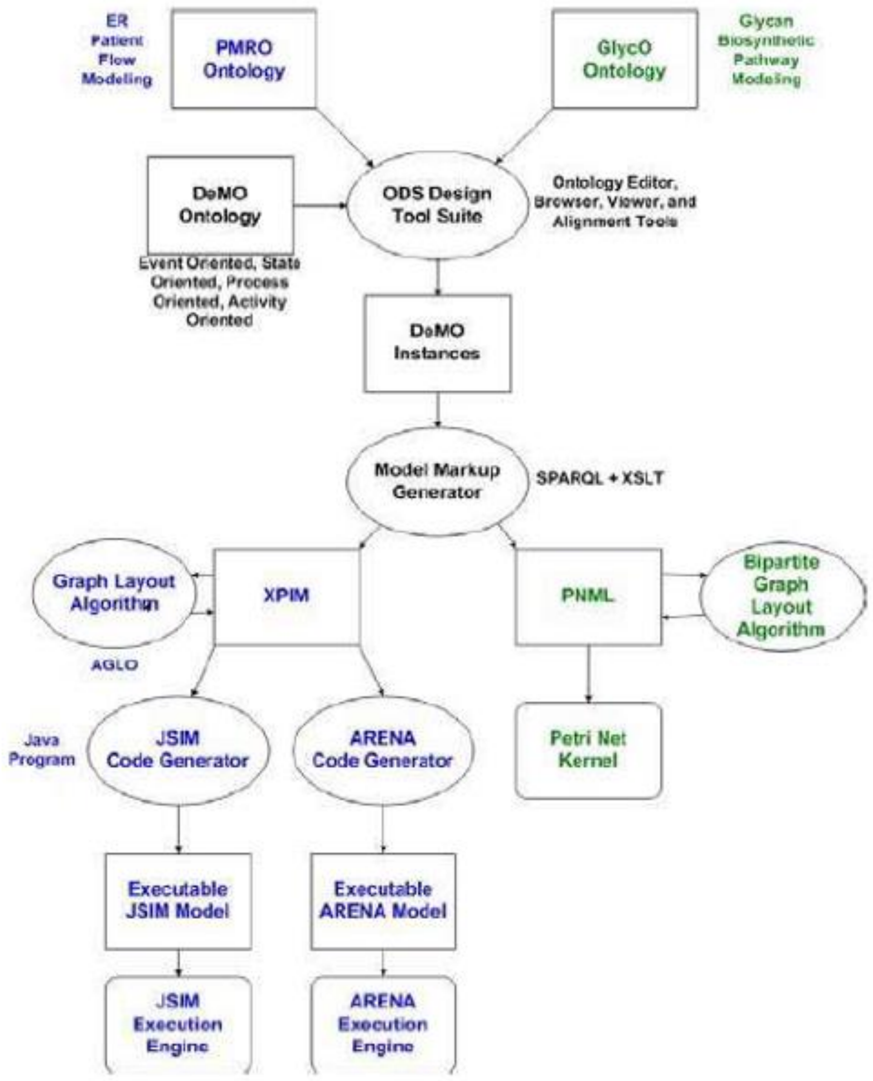


Figure 11: ODS Architecture (Silver et al. 2007)

7 Tools and systems for semantic engineering

Knowledge engineering and linked data are mature domains in research and development. There are a large set of core and supporting technologies and tools that are available, both open source and commercial ones. The development of the Semantic Web technologies and the standardisation done mainly by the World Wide Web Consortium (W3C) have progressed this development and enabled the rich offering of solutions. In this section, the state of the tool and solution offering is studied and some of these tools are introduced, tested and discussed.

7.1 Triplestore and Graph Database

A triple is a data entity composed of subject-predicate-object. A triplestore or RDF store is a purpose-built database for the storage and retrieval of triples through semantic queries. Some triplestores have been built as database engines from scratch, while others have been built on top of existing commercial relational database engines or NoSQL document-oriented database engines. Adding a name to the triple makes a "quad store" or named graph.

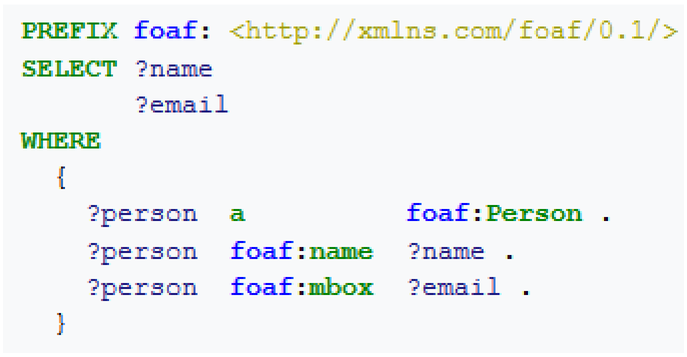
A graph database has a more generalised structure than a triplestore, using graph structures with nodes, edges, and properties to represent and store data. Graph databases might provide index-free adjacency, meaning every element contains a direct pointer to its adjacent elements, and no index lookups are necessary. General graph databases that can store any graph are distinct from specialised graph databases such as triplestores and network databases. (Wikipedia 2019)

7.2 Linked Data

Linked Data is structured data which is interlinked with other data so it becomes more useful through semantic queries. It builds upon standard Web technologies such as HTTP, RDF and URIs. Part of the vision of Linked Data is for the Internet to become a global database. Data linking is one of the main goals of the EU Open Data Portal[1], which makes available thousands of datasets for anyone to reuse and link.

7.3 SPARQL

SPARQL is an RDF query language that is able to retrieve and manipulate data stored in RDF format. SPARQL is a semantic query language for RDF databases, i.e. triplestores. SPARQL is an official W3C Recommendation (W3C 2013).



```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name
       ?email
WHERE
  {
    ?person  a          foaf:Person .
    ?person  foaf:name  ?name .
    ?person  foaf:mbox  ?email .
  }
```

Figure 12: An example of a SPARQL query.

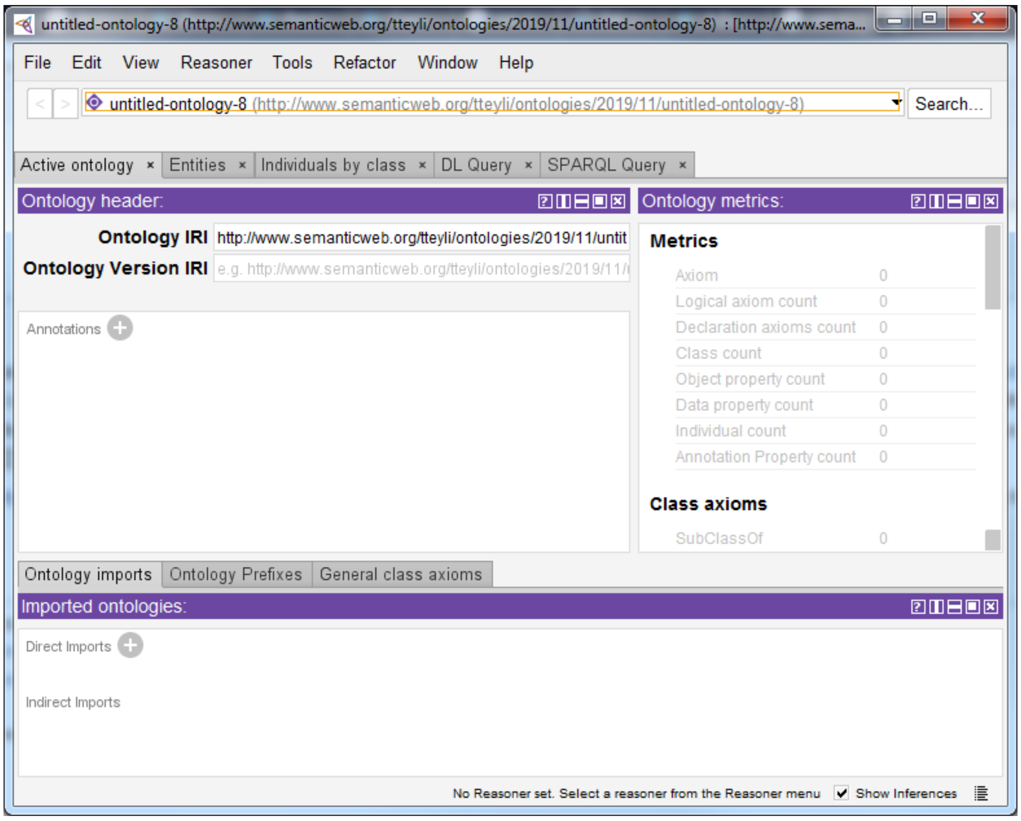
7.4 Protégé

Protégé is an open source ontology editor and a knowledge management system (Protégé 2019). This desktop application is written in Java programming language and uses the Swing widget toolkit for its Graphical User Interface (GUI) to define ontologies. Protégé is a framework for which various other projects suggest plugins to extend its functionality. Protégé is being developed at Stanford University and is made available under the BSD 2-clause license. Earlier versions of the tool were developed in collaboration with the University of Manchester. Protégé has been considered as the leading ontological engineering tool. The user interface of the Protégé tools is shown in Figure 13 and Figure 14.

In addition to the desktop version of Protégé, also a version that can be used through a web browser is available, namely WebProtégé which is an open source collaborative ontology development environment for the Web. Compared to the desktop version, the web version is, at

[1] EU Open Data Portal: https://data.europa.eu/euodp/en/home

least at the moment, more limited and lacking many functionalities found in the former. The user interface of the WebProtégé tool is shown in Figure 15.


untitled-ontology-8 (http://www.semanticweb.org/tteyli/ontologies/2019/11/untitled-ontology-8) : [http://www.sema...
File Edit View Reasoner Tools Refactor Window Help
Search...
Active ontology × Entities × Individuals by class × DL Query × SPARQL Query ×
Ontology header:
Ontology metrics:
Annotations
Ontology imports Ontology Prefixes General class axioms
Imported ontologies:
Direct Imports
Indirect Imports
Show Inferences

untitled-ontology-8 (http://www.semanticweb.org/tteyli/ontologies/2019/11/untitled-ontology-8)

Ontology IRI http://www.semanticweb.org/tteyli/ontologies/2019/11/untit

Ontology Version IRI e.g. http://www.semanticweb.org/tteyli/ontologies/2019/11/

Metrics

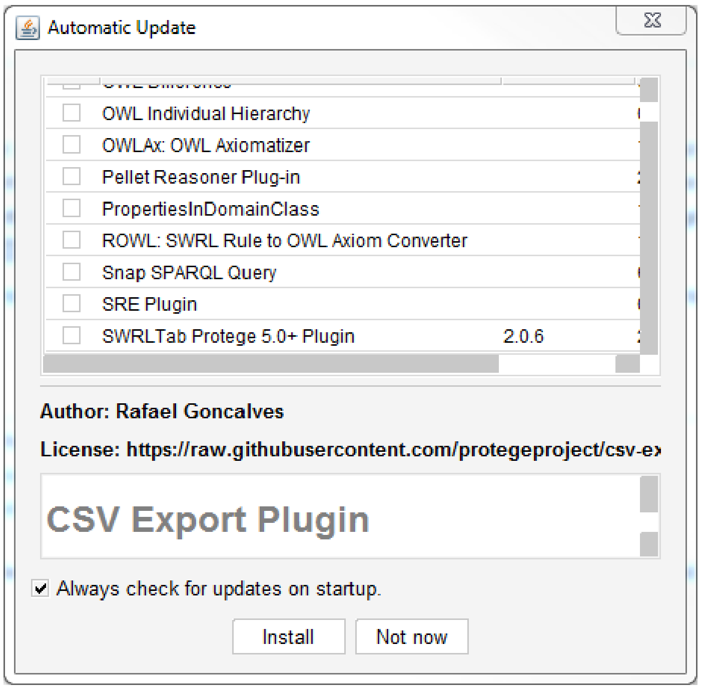
Axiom	0
Logical axiom count	0
Declaration axioms count	0
Class count	0
Object property count	0
Data property count	0
Individual count	0
Annotation Property count	0

Class axioms

SubClassOf	0

No Reasoner set. Select a reasoner from the Reasoner menu ☑ Show Inferences

Figure 13: The user interface of the Protégé tool.



☐	OWL Individual Hierarchy	
☐	OWLAx: OWL Axiomatizer	
☐	Pellet Reasoner Plug-in	
☐	PropertiesInDomainClass	
☐	ROWL: SWRL Rule to OWL Axiom Converter	
☐	Snap SPARQL Query	
☐	SRE Plugin	
☐	SWRLTab Protege 5.0+ Plugin	2.0.6

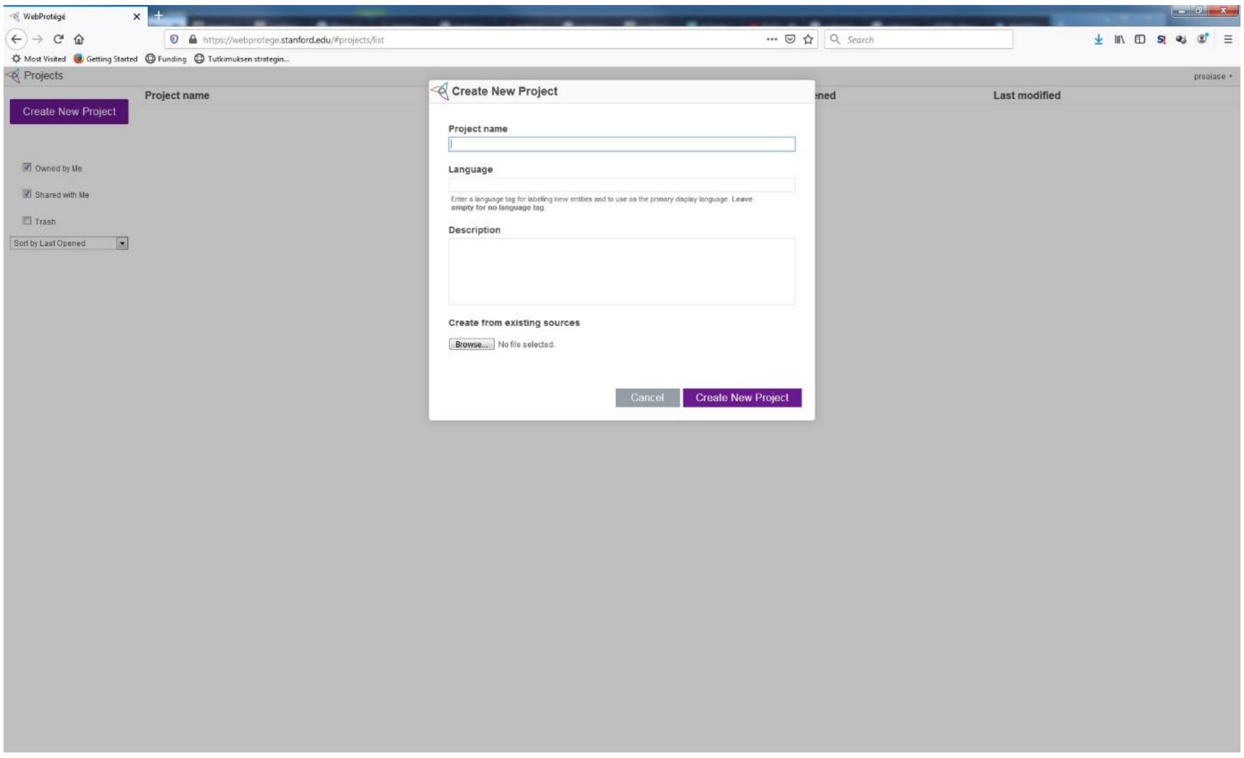
Author: Rafael Goncalves

License: https://raw.githubusercontent.com/protegeproject/csv-ex

CSV Export Plugin

☑ Always check for updates on startup.

Figure 14: Protégé is extensible with plugins.



Enter a language tag for labelling new entities and to use as the primary display language. Leave empty for no language tag.

Figure 15: The user interface of the WebProtégé tool.

7.5 Apache Jena

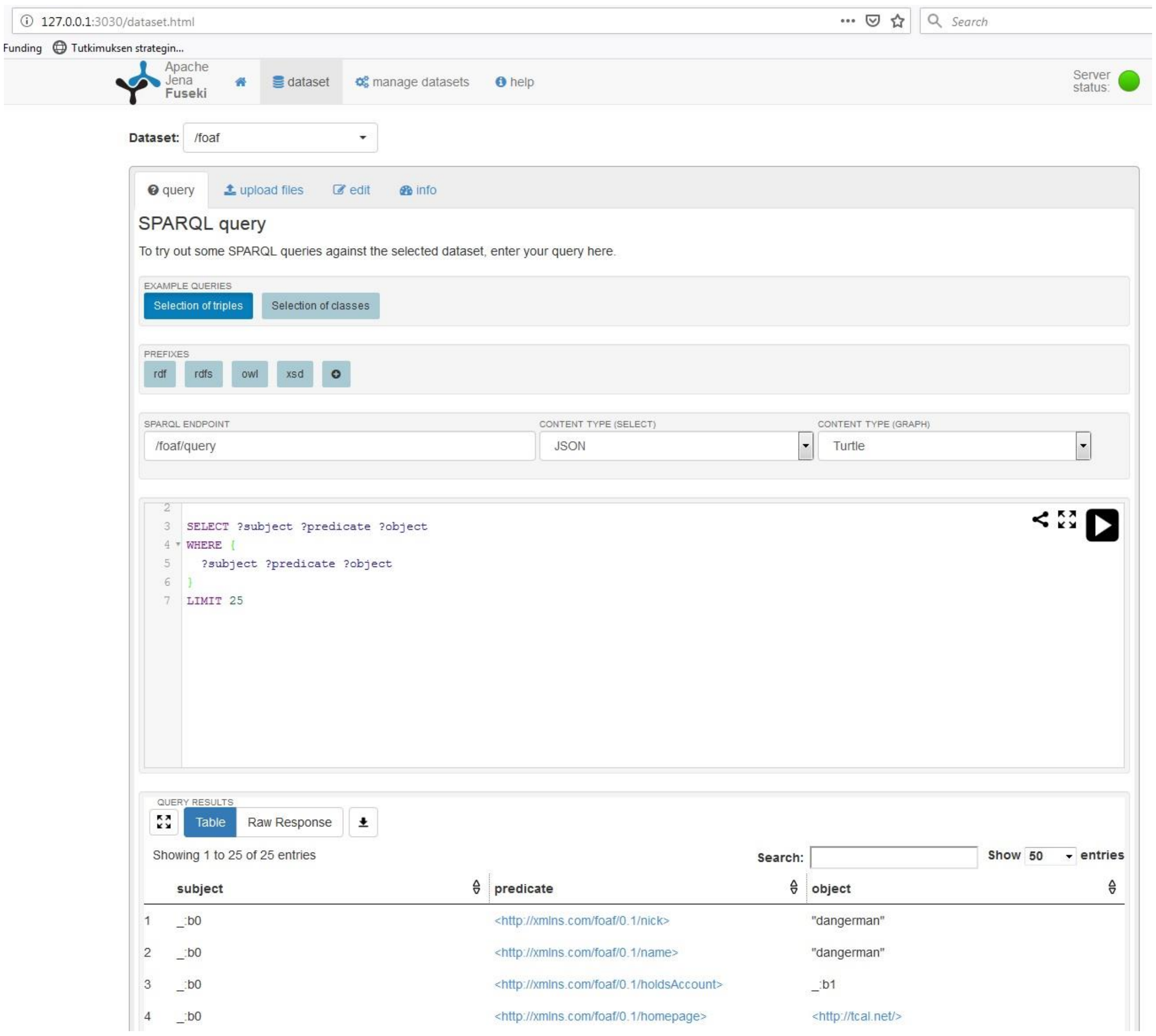
Apache Jena[2] is a free and open source Java framework for building Semantic Web and Linked Data applications. It supports RDF, OWL and provides Triple Store through Fuseki as an HTTP interface to RDF Data. Fuseki supports SPARQL for querying and updating and is developed as servlet. Fuseki can also be run as a stand-alone server. Jena was developed at HP Labs and then moved to Apache Software Foundation (ASF). The web browser user inferface of Apache Jena is shown in Figure 16.

7.6 Virtuoso Universal Server

Virtuoso Universal Server[3] is a middleware and database engine hybrid that in addition to RDF combines several other functionalities. There are both proprietary and community editions available. The free and open source edition of Virtuoso Universal Server is also known as OpenLink Virtuoso with GPLv2 license. The software has been developed by the OpenLink Software. The web browser user interface of Virtuoso is shown in Figure 17.

---

[2] Apache Jena website: https://jena.apache.org/
[3] The OpenLink Software website: https://virtuoso.openlinksw.com/



Dataset: /foaf

SPARQL query

To try out some SPARQL queries against the selected dataset, enter your query here.

EXAMPLE QUERIES

PREFIXES

SPARQL ENDPOINT: /foaf/query
CONTENT TYPE (SELECT): JSON
CONTENT TYPE (GRAPH): Turtle

```
SELECT ?subject ?predicate ?object
WHERE {
  ?subject ?predicate ?object
}
LIMIT 25
```

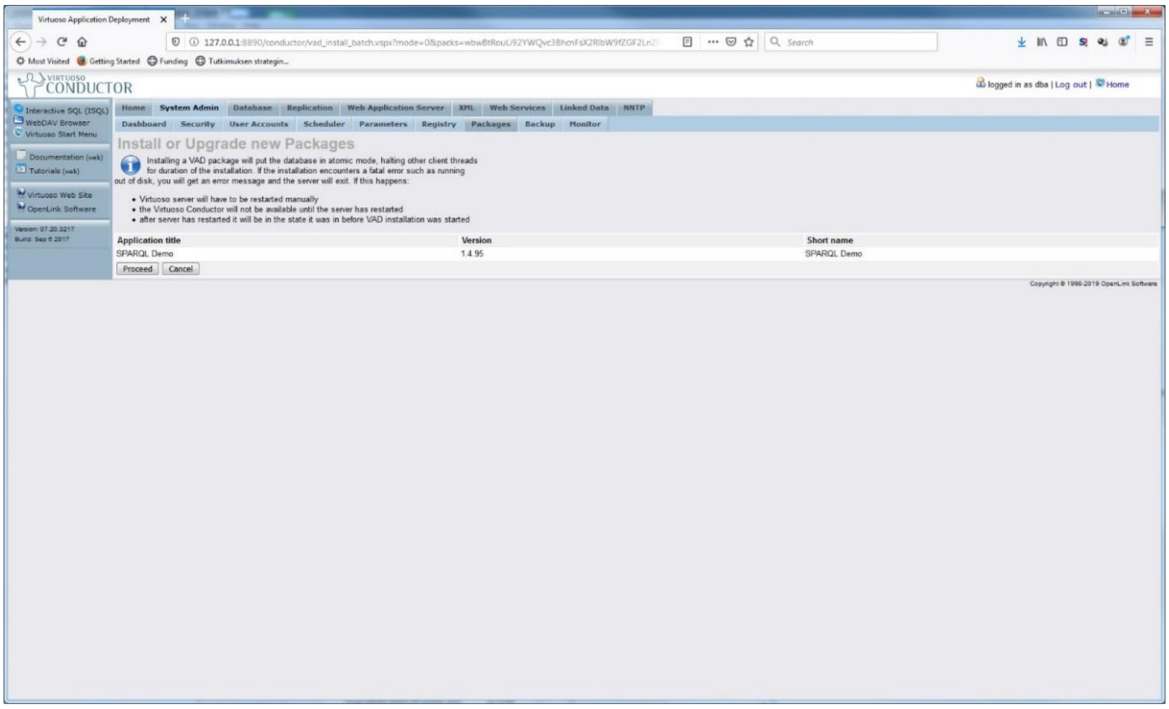
QUERY RESULTS

Showing 1 to 25 of 25 entries

Search: Show 50 entries

	subject	predicate	object
1	_:b0	<http://xmlns.com/foaf/0.1/nick>	"dangerman"
2	_:b0	<http://xmlns.com/foaf/0.1/name>	"dangerman"
3	_:b0	<http://xmlns.com/foaf/0.1/holdsAccount>	_:b1
4	_:b0	<http://xmlns.com/foaf/0.1/homepage>	<http://tcal.net/>

Figure 16: The web browser user interface of Jena with Fuseki and SPARQL.



Install or Upgrade new Packages

Installing a VAD package will put the database in atomic mode, halting other client threads for duration of the installation. If the installation encounters a fatal error such as running out of disk, you will get an error message and the server will exit. If this happens:

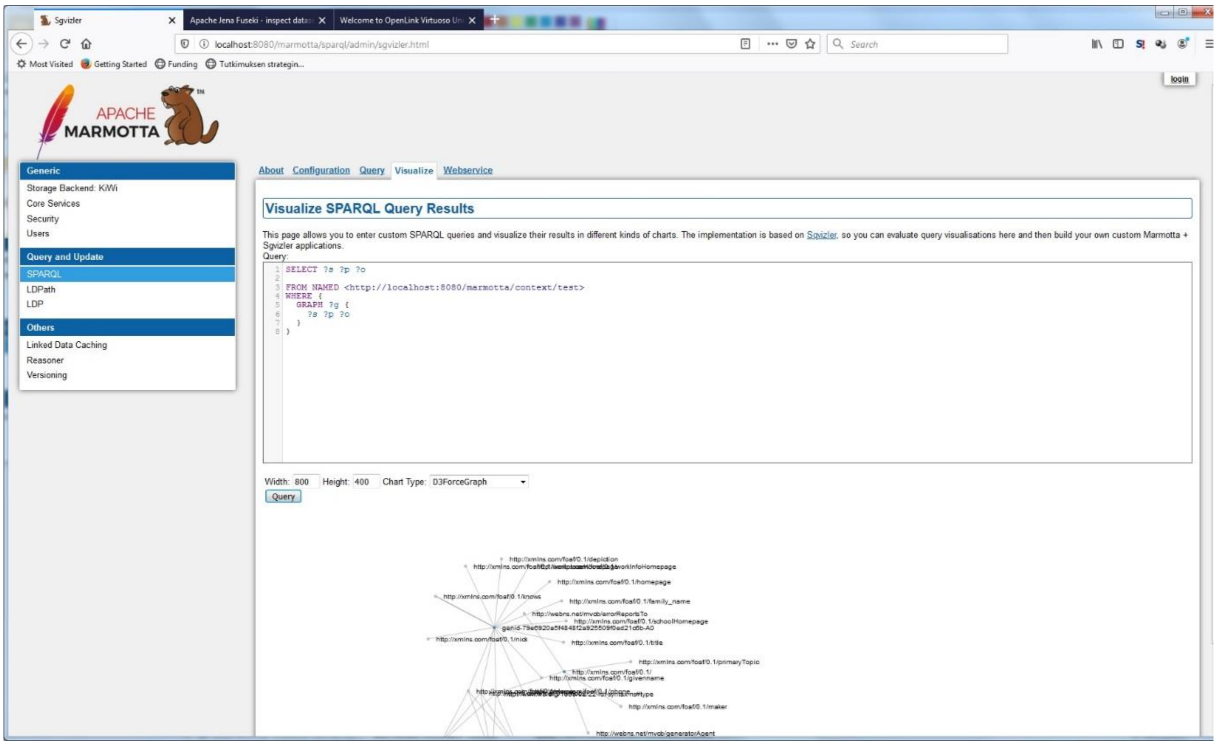
- Virtuoso server will have to be restarted manually
- the Virtuoso Conductor will not be available until the server has restarted
- after server has restarted it will be in the state it was in before VAD installation was started

Application title	Version	Short name
SPARQL Demo	1.4.95	SPARQL Demo

Figure 17: The web browser user interface of Virtuoso.

7.7 Apache Marmotta

Apache Marmotta[4] is a linked data platform that comprises several components. In its most basic configuration it is a Linked Data server. Marmotta is one of the reference projects early implementing the new Linked Data Platform recommendation that is being developed by W3C. Marmotta is licensed under the Apache License, Version 2.0. The web browser user interface of Apache Marmotta is shown in Figure 18.



Visualize SPARQL Query Results

This page allows you to enter custom SPARQL queries and visualize their results in different kinds of charts. The implementation is based on Sgvizler, so you can evaluate query visualisations here and then build your own custom Marmotta + Sgvizler applications.

Query:

```
SELECT ?s ?p ?o

FROM NAMED <http://localhost:8080/marmotta/context/test>
WHERE {
  GRAPH ?g {
    ?s ?p ?o
  }
}
```

Width: 800 Height: 400 Chart Type: D3ForceGraph

Figure 18: The web browser user interface of Marmotta with SOARQL and Graph Visualization.

7.8 Demonstrator

One element of this study was to test how to set up a common server for ontology development and the use of semantic data models. The goal was to have a common database for the ontologies as well as for the data models, In addition, ability to add, edit and modify the ontologies and data models in the common environment was required. A Windows Server 2016 system was available for this purpose. Although WebProtégé, Jena, Virtuoso and Marmotta were all considered and investigated, utilising the Protégé desktop version through a web browser was chosen after finding out that this should be also technically possible. Because Protégé is an open source tool, written in Java and numerous plugins are available, this seemed very feasible solution.

Apache Guacamole[5] is a remote desktop gateway solution. It supports standard protocols, such as Virtual Network Connection (VNC), Remote Desktop Protocol (RDP), and Secure Shell (SSH) protocol. No plugins or client software are required and desktop can be accessed once Guacamole is installed. Apache Guacamole can be deployed with Docker[6] technology.

[4] The Apache Marmotta website: https://marmotta.apache.org/
[5] The Apache Guacamole website: https://guacamole.apache.org/
[6] The Docker website: https://www.docker.com/

For running Apache Guacamole on a Windows host computer, virtualisation is needed for having Linux operating system were Guacamole could be installed. For this purpose Oracle VirtualBox[7] was chosen. But, it turned out that this could not be utilised on that particular server then. Instead Hyper-V[8] was available and it was finally configured to allow bi-directional traffic between the Windows server (host) and the Linux server (guest). Nevertheless still more configuration would had been needed for allowing traffic from internet to guest and the time table for this was fuzzy. At this very moment a reinstalled another Linux server was made available. Guacamole was installed there and it could be accessed through internet.

Having this Guacamole installation connect the Windows server with RDP was not possible because of network restrictions, so VNC server was installed to the Linux server for accessing Protégé desktop for demonstration with the web browser. Some planned things are connecting Protégé with Jena and tracking changes of the graph. Although some promising leads are found now, there is more research to be done.

8 Case study of semantic engineering data representation

Systems engineering is a methodology that combines a set of processes, methods, practices and concepts to manage the life cycle of products and systems. It includes engineering and design, but also project, procurement and other aspects that all together affect the success and efficiency of the product or system during its overall life cycle. One of the main aspects of systems engineering is to provide tools for managing complexity in the engineering process, complexity that is caused by the product or system itself, but also by the engineering process, organisation and all the other aspects.

8.1 About the case study

In this case study, the focus was to manage the information of the engineering design of a mileage marathon vehicle (see Figure 19). The engineering design, in this simplified case, contains the following areas:

- requirements engineering
- vehicle system architecture design
- aerodynamic design

For each of these engineering design areas, an ontology defining the common concepts (object types), relations (object properties) and data attributes (data properties) was designed. In addition, an ontology for multibody system simulation (simulation of the dynamics of a mechanical system) was available. For the design data in each of these areas, also a dedicated data model was created. The overall design model integrated the ontologies and the data models.

In systems engineering, requirements are the drivers and the constraints for the process. The requirements often come from several sources and can be categorised in several ways. In this case study, there were requirements set by the mileage marathon completion organisation, the Finnish Mileage Marathon Club ry[9], for the *Pisaralla pisimmälle -Marathon 2019* competition (Finnish Mileage Marathon Club r.y. 2019) – the authority requirements (Appendix A). In addition, the process produces internal requirements for clarifying the details of the design – the internal or design requirements.

[7] The VirtualBox website: https://www.virtualbox.org/
[8] The Hyper-V in Wikipedia page: https://en.wikipedia.org/wiki/Hyper-V
[9] The Finnish Mileage Marathon Club ry website: http://www.fmmc.fi/

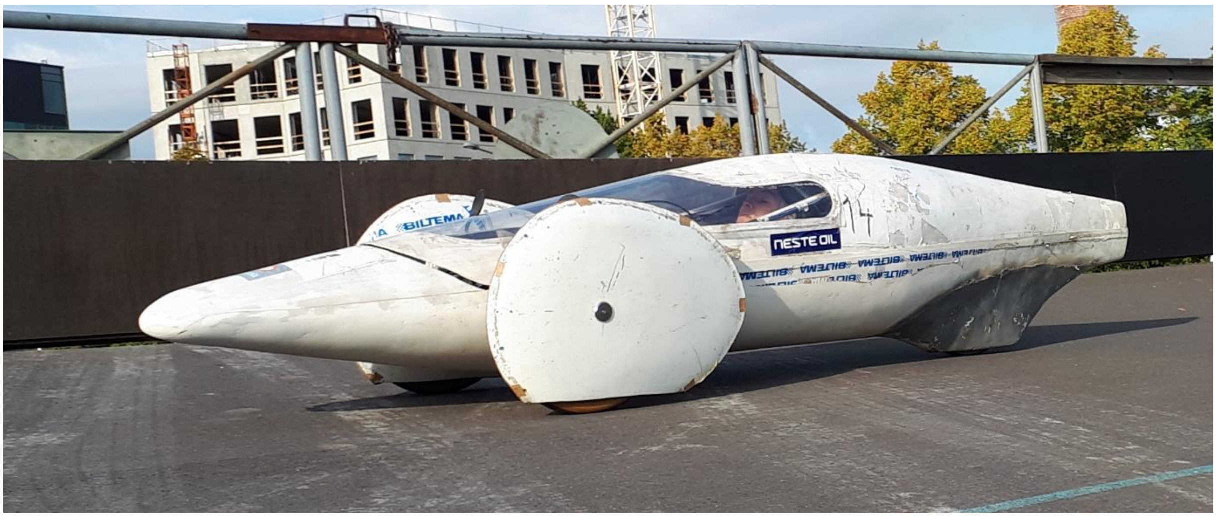


Figure 19: An example of a mileage marathon vehicle. Image by courtesy of Raimo Ruokonen.

In this case study, the particular engineering design work was demonstrated with the design of general level system architecture of the vehicle and a simple design of the CFD simulation. The system architecture is presented in Figure 20.

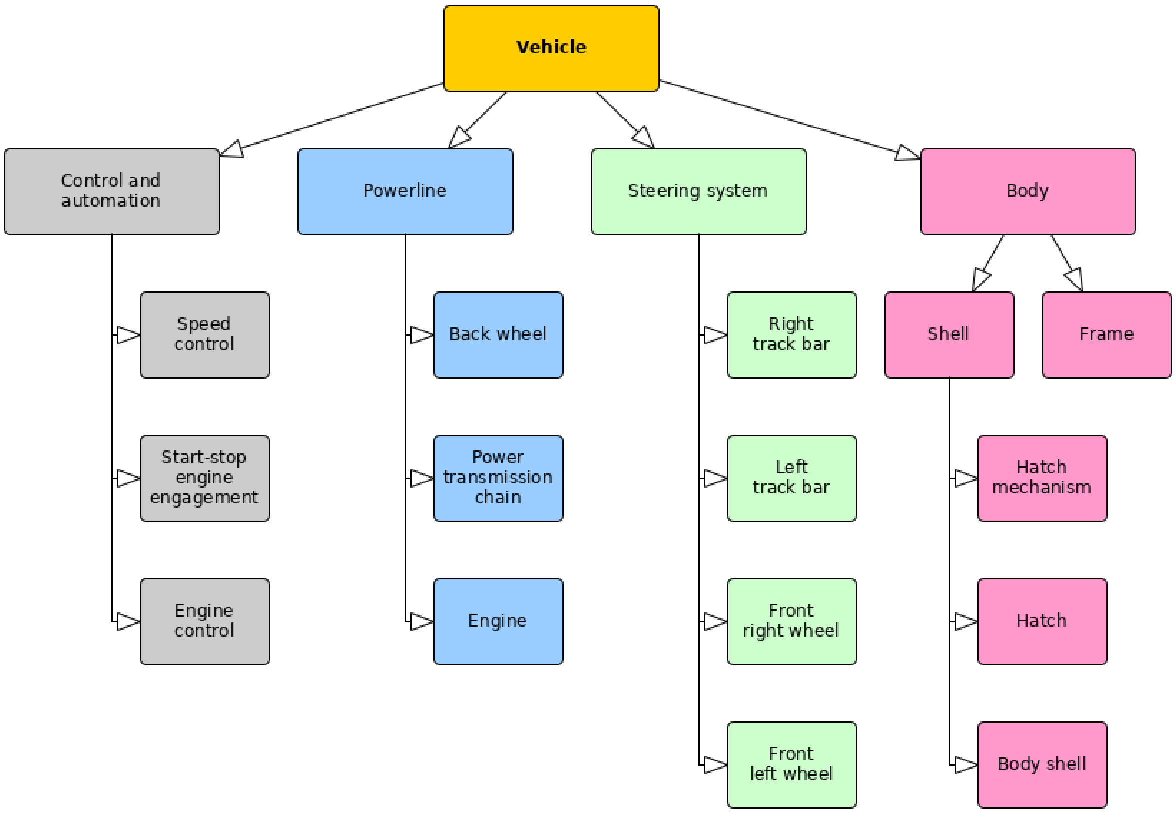


Figure 20: The system architecture design of the mileage marathon vehicle.

The aerodynamic design and the design for the CFD simulation were done in a very general level.

8.2 Ontologies

A modified ontology for requirements engineering was created based on the Semantic Web Ontology for Requirements Engineering, SWORE (Jens et al. 2018). The ontology provided object and relation types to map the design requirements, related documents and the design models, among other things. The object types of the requirements ontology are visualised in Figure 21.

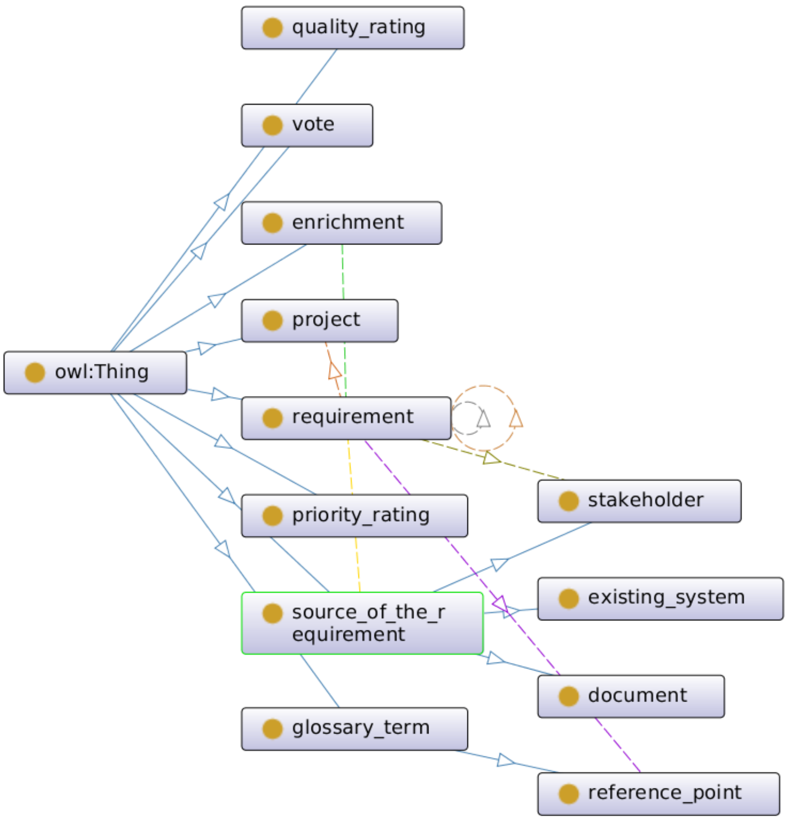


Figure 21: A simple requirements engineering ontology, based on the Semantic Web Ontology for Requirements Engineering, SWORE (Jens et al. 2018).

For designing the system architecture of the target, a simple system architecture design ontology was created. This very simple ontology contained only two object types, *system* and *sub-system*, and it is meant for simple system architecture structuring. The object types of the ontology are visualised in Figure 22.

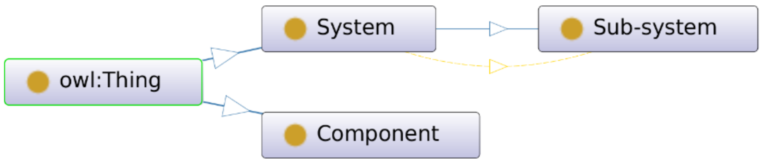


Figure 22: A simple system architecture design ontology.

A simple design ontology was created for aerodynamic design using CFD. The ontology contains the main computational elements needed in a CFD simulation. In addition, the ontology contains object types for related data, such as the geometry needed for meshing the flow domain and the design target for the CFD simulation. The object types of the ontology are visualised in Figure 23.

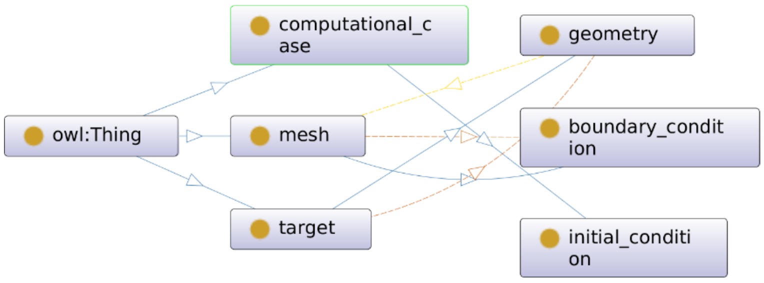


Figure 23: A simple aerodynamic design ontology.

In addition to the previously presented ontology, an additional design ontology was available for the simulation of the dynamics of a mechanical system using multibody system (MBS) simulation (Kortelainen 2011). The object types and their relations of the multibody system modelling and simulation ontology are visualised in Figure 24. The ontology was not used in the case study. The presence of the ontology demonstrates that there may be several design and engineering ontologies available, but only the necessary for the task at hand are used.

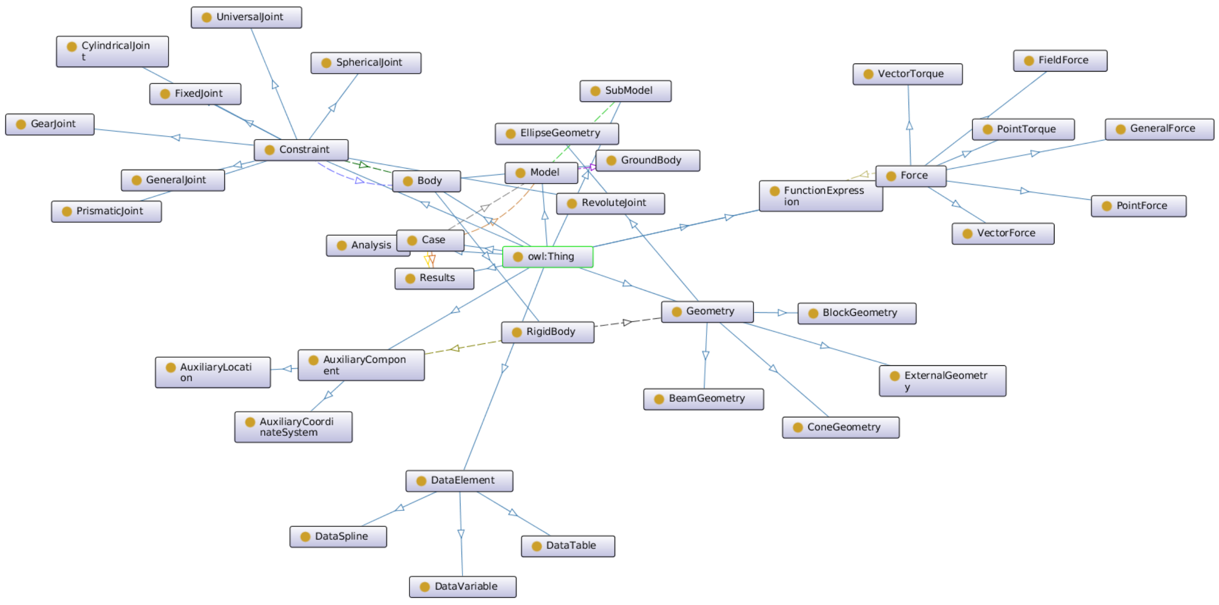


Figure 24: A MBS simulation ontology (Kortelainen 2011)

8.3 Data models

Working with one or couple domain ontologies at the time helps the engineer to focus on the tasks in hand, but keeping the track on relations to other design domains. In the design of a complex systems, the overall design model or design data may be very large and there may be numerous relations between different design domains. One example is the relations between the mechanical and the aerodynamic design of the target. The mechanical design is typically done by a mechanical engineer using computer-aided design (CAD) and other tools. The outcome of the design work is a CAD model. On the other hand, the aerodynamic design is typically done by an aerodynamics expert using CFD modelling and simulation tools. The starting point for both of these design domains in the process are the requirements. For the aerodynamic design, the shape, i.e. the outside geometry, of the target is needed for meshing the flow volume for the computation. This means that the aerodynamic engineering domain has a relation to the mechanical design domain via the geometry of the target. In the aerodynamic design work, the engineer does not need to know all the details, requirements and relations present in the mechanical engineering, but only those details that are affecting the aerodynamic design work.

In this case study, the design models for the different design domains, i.e. requirements engineering, system architecture design and aerodynamic design, were done using the corresponding ontologies for the design work. This meant defining the design objects with object types, defining their attributes and relations between the objects. The model of the authority and some design requirements and their relations are presented in Figure 25, including the ontology object types (nodes with a dark yellow circle). The model of the system architecture design is presented in Figure 26, including the ontology object types, and in Figure 27 is presented the model of the aerodynamic design for CFD simulation, also including the ontology object types.

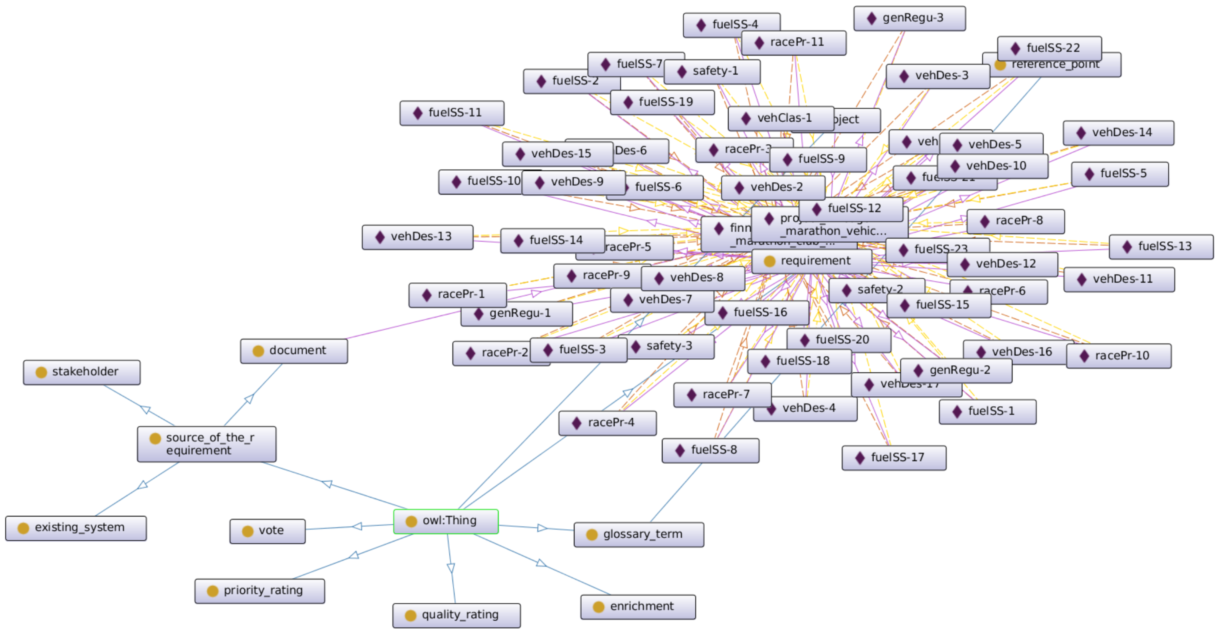


Figure 25: The requirements model for the target, including the requirements ontology object types.

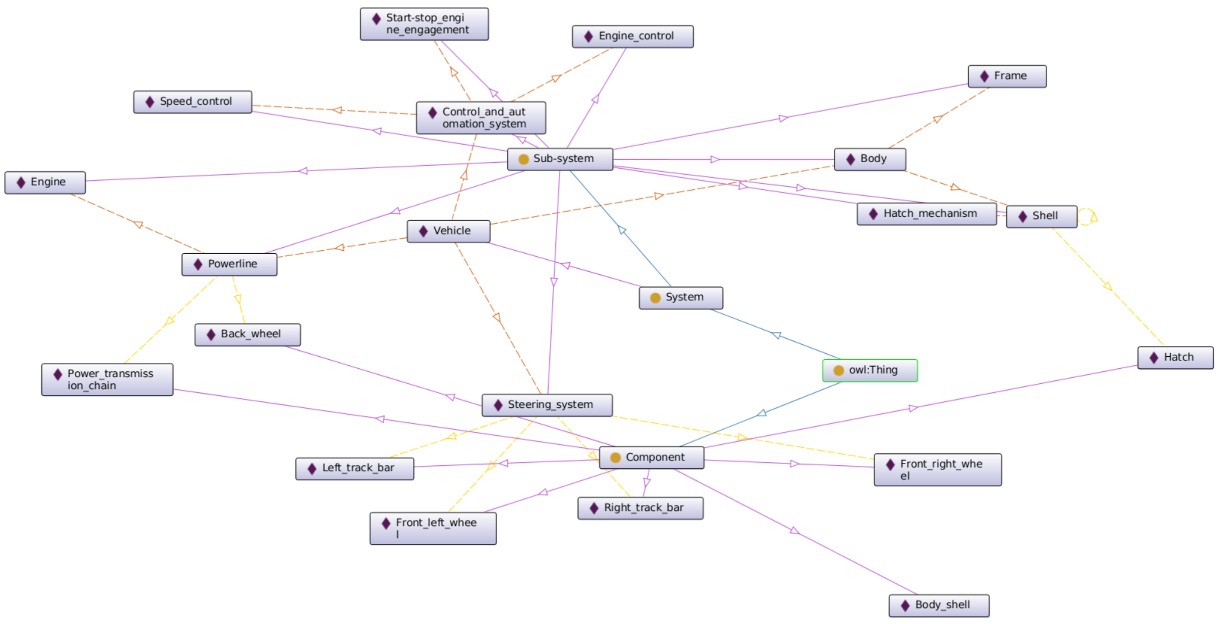


Figure 26: The system architecture design model of the target. With a very simple ontology and for a relatively simple system, the design model becomes already quite complex.

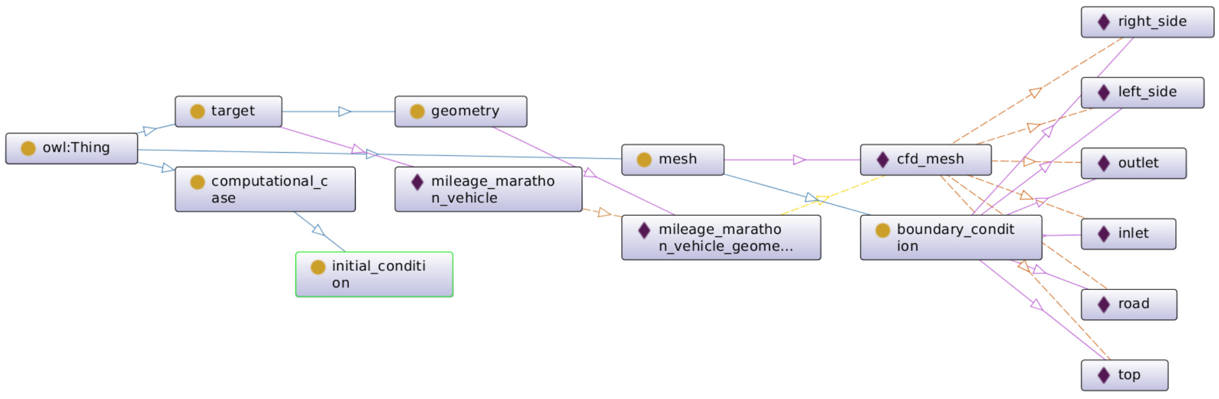


Figure 27: The aerodynamic design model for CFD simulation of the target.

8.4 Overall design model

The overall design model combines the domain design models. In our simple mileage marathon vehicle design case, the domain design models were the requirements model for the design, the system architecture design model, and the aerodynamic design model. In the overall design model, these design domain models are mapped and the constraints between the design domains are shown. Figure 28 shows the overall design model with relations to the requirements, and Figure 29 represents the design model with the requirements and relations to them. Even for a simple demonstration case, the data model with all the domains and objects included becomes complex and difficult to manage. The modular approach with domain ontologies and domain specific design models is thus relevant.

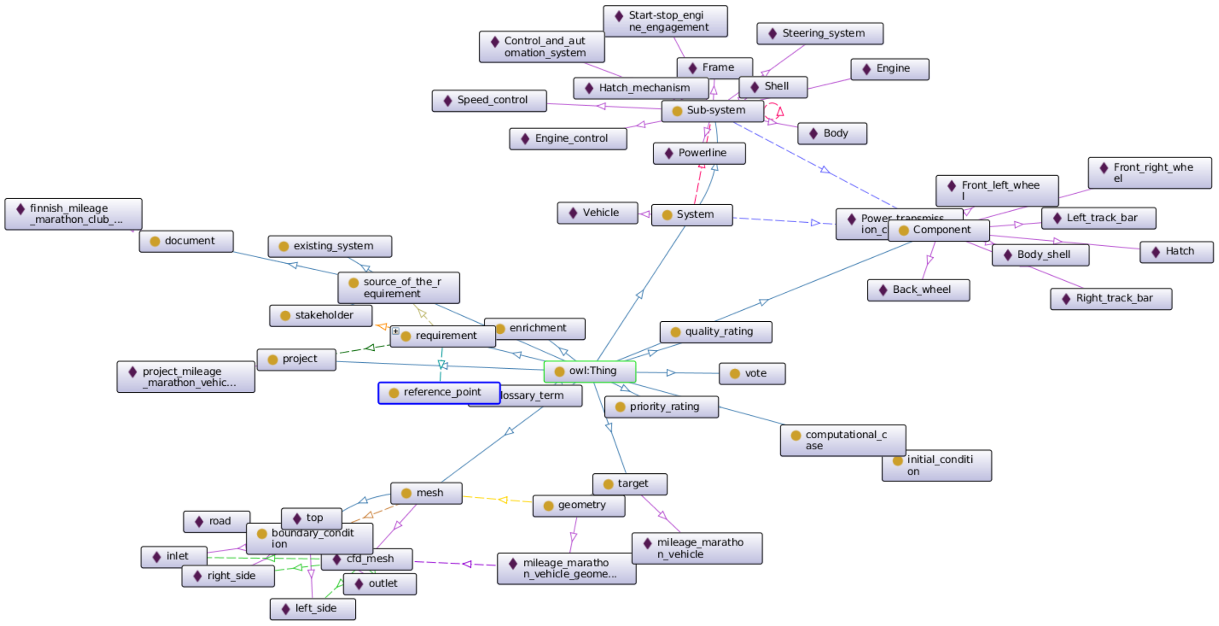


Figure 28: The overall design model without references to the requirements. Only some relations between different design domains are shown.

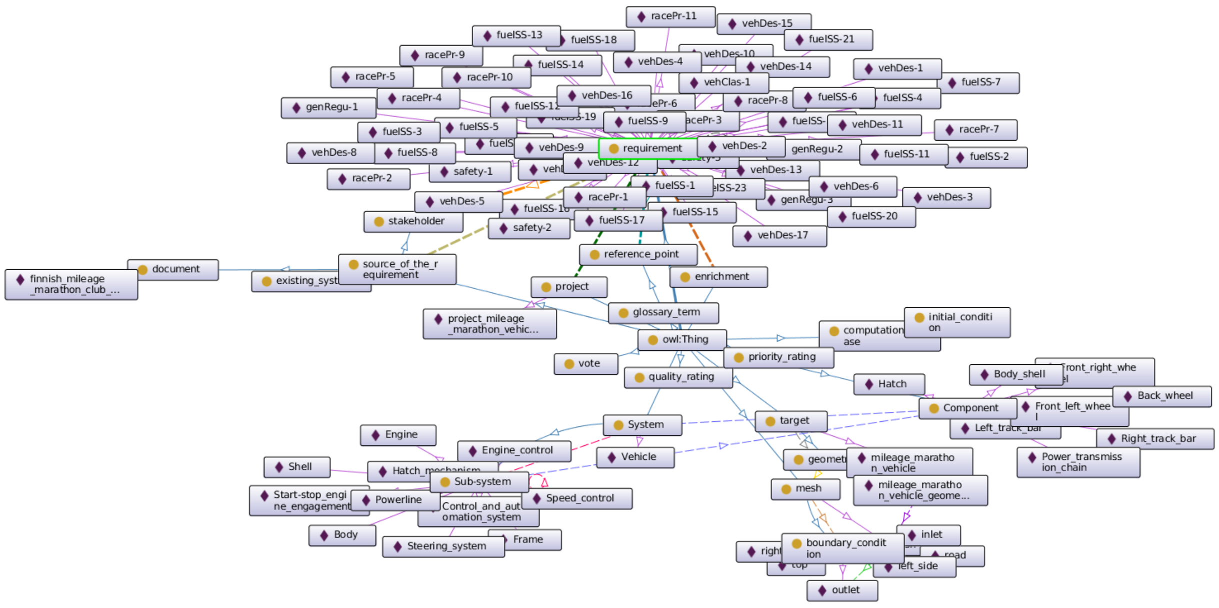


Figure 29: The overall design model with references to the requirements. Only some relations between different design domains are shown.

8.5 Lessons learned from the case study

The case study of the general design of a mileage marathon vehicle, focusing on the system architecture and aerodynamic design, was done with a general Protégé ontology editor. The editor is not designed for this kind of data modelling and especially not for engineering work. Still the case was able to illustrate the potential benefits of using a well-defined knowledge representation technology for domain knowledge description, working in limited engineering domains to enable the user from dealing with large amount and complex representation of data. One of the advantages in using the Semantic Web technologies, ontologies and data models is the ability to check the validity of the ontologies and data models based on them. This prevents the user for making errors in cases where there are explicit rules and constraints for knowledge representation. In addition, the approach enables the user to find implicit relations and correlations from the data by using the standardised technologies, i.e. SPARQL for doing data queries and SWRL for doing validity checking and implicit reasoning on the data.

The work with the Protégé editor showed that, for fluent working process, the engineering of ontologies and the application of the ontologies need to have dedicated and improved tools. The ability to change the domain ontologies when working with the design models increases the risk of mangling the ontology, which may be the basis for a very large and complex data model. This may lead to large additional work to fix the links between the domain ontology and the data model that is based on the ontology. In addition, the ontologies need to have proper version control and applying the ontologies must include an option to undo the changes and solve any issues that may come up in the modelling and editing process. It must be pointed out that Protégé is a general Semantic Web Ontology editor with extended features for some dedicated knowledge engineering needs. The editor is not specifically designed for engineering design work. In general, all the challenges in efficiency and fluency of using knowledge engineering tools and editors can be solved with proper design and implementation of the dedicated tools for the specific use.

A serious question that was left unanswered in the case study was the scalability of the technologies for very large and complex engineering design tasks. When designing e.g. a power plant, the size and the complexity of the design models is very large, and the importance and value of the design data is crucial for the users and their organisations. Especially if semantic reasoning plays an important role, the present technologies may require further development for guaranteeing fluent work flow together with safe data management.

9 Summary

With the extensive use of computer-aided systems in product design, manufacturing and analysis, a mass of documents is produced. These documents provide engineering information in integrated environment of enterprise for engineers, but they are maintained by different systems in different forms. Current information retrieval approaches usually lack semantic supports for different kinds of documents, which leads to insufficiency of content representation and misunderstanding of query intention. In the engineering environments the accuracy and efficiency of information exchange among various agents (product designers, manufactures, suppliers, etc.) improves by storing information once where it is generated and allowing access to the information over the intranet/internet. It is also beneficial to store project data in a format that is machine processable.

The Semantic Web technology promises to improve information representation, sharing, reuse and automated processing by software agents to make inferences. The Semantic Web technologies use a graph data structure for information modelling which facilitates integrating information from different sources and allows computers to access and process information distributed over the Internet. The point of the Semantic Web is not just to make applications smarter, but also to make data smarter. The data does not and should not reside in application-specific databases. The data become smarter through the use of higher semantics from technologies such as concept maps or ontologies.

Ontologies are used to capture knowledge about some domain of interest. An ontology describes the concepts in the domain and the relationship that hold between those concepts. Ontologies are very beneficial for representing engineering knowledge. Engineers are dependent on accessing documents in order to fulfil various design and engineering tasks. In industry sectors, it was reported that design engineers spent 20% to 30% of their time retrieving and communicating information (Court et al. 1998). “Delivering the right information to the right people at the right time” plays an important role in supporting engineers’ memory extension, knowledge sharing, design concept exploration, design reuse, and the learning process particularly of novice engineers (Ahmed and Wallace 2004). Thus ontologies are developed: (1)

to share common understanding of the structure of information among people or software agents, (2) to enable reuse of domain knowledge, (3) to make domain assumptions explicit, (4) to separate domain knowledge from the operational knowledge, and (5) to analyse domain knowledge.

A number of ontologies have been developed for the representation of engineering knowledge of various engineering domains. There are several approaches and tools available for building ontologies, however, a standard approach is still missing and different applications require different tools for constructing ontologies. In addition, depending on the necessary level of expressive power and computational complexity, a proper ontology language could be selected for different knowledge representation tasks. Furthermore, in the near future Semantic Web-based browsers with user-friendly interfaces would be needed to allow individuals and practitioners to be able to undertake queries and searches without the need to understand Description Language query constructs.

A study about existing tools and systems for linked and semantic data was done. These are available both as open source as well as commercial, for both research and for commercial and industrial scale. Based on the study, the field of linked and semantic data tools seems to be maturing and valid options for tools and systems are available.

To concretise the use of linked and semantic data, and the application of the Semantic Web technologies, a small-scale case study was conducted. The case study was about data modelling for the requirements engineering, and system architecture and aerodynamic design of a mileage marathon vehicle. The study included the design of simple domain ontologies and the application for these ontologies for a simplified design process. The study showed that the approach is potential for managing design process complexity, when several design domains are involved and the domains share common data. The study also showed that the engineering tools need to be improved from the used general ontology editor to guarantee safe and reliable use of data.

References

Abburu, Sunitha and GS Babu. 2013. “Survey on Ontology Construction Tools.” *Ijser.Org* 4(6):1748–52.

Ahmed, S. and K. M. Wallace. 2004. *Identifying and Supporting the Knowledge Needs of Novice Designers within the Aerospace Industry*. Vol. 15.

Angele, Jürgen, Michael Kifer, and Georg Lausen. 2009. “Ontologies in F-Logic.” Pp. 45–70 in *Handbook on Ontologies*.

Bizer, C., T. Heath, and T. Berners-Lee. n.d. “Linked Data - The Story So Far.” *International Journal on Semantic Web and Information Systems* 5(3):1–22.

Corcho, Oscar, Mariano Fernández-López, and Asunción Gómez-Pérez. 2003. “Methodologies, Tools and Languages for Building Ontologies. Where Is Their Meeting Point?” *Data and Knowledge Engineering* 46(1):41–64.

Corry, Edward, Daniel Coakley, James O’Donnell, and Marcus M. Keane. 2013. “The Role of Linked Data and the Semantic Web in Building Operation.” *NUIG-UL Alliance - Engineering, Informatics & Science Research Day* 70.

Court, A. W., David G. Ullman, and Stephen J. Culley. 1998. “A Comparison between the Provision of Information to Engineering Designers in the UK and the USA.” *International Journal of Information Management* 18(6):409–25.

Farias, Tarcisio Mendes de, Ana Roxin, and Christophe Nicolle. 2016. “SWRL Rule-Selection Methodology for Ontology Interoperability.” *Data and Knowledge Engineering* 105:53–72.

Farias, Tarision M., Ana Roxin, and Christophe Nicolle. 2015. “FOWLA, A Federated Architecture for Ontologies.” *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 9202(August).

Finnish Mileage Marathon Club r.y. 2019. “REGULATIONS TEK - PISARALLA PISIMMÄLLE® 2019.” Retrieved (http://www.fmmc.fi/wp-content/uploads/2019/08/PIS-2019-ENG.pdf).

Grolinger, Katarina, Miriam A. M. Capretz, José R. Marti, Krishan D. Srivastava, Katarina Grolinger, and Miriam A. M. Capretz. 2012. “Ontology – Based Representation of Simulation Models.” *The Twenty-Fourth International Conference on Software Engineering and Knowledge Engineering (SEKE)* 432–37.

Grolinger, Katarina, Miriam A. M. Capretz, Adam Shypanski, and Gagandeep S. Gill. 2011. “Federated Critical Infrastructure Simulators: Towards Ontologies for Support of Collaboration.” *Canadian Conference on Electrical and Computer Engineering* 001503–6.

Grosse, Ian R., John M. Milton-Benoit, and Jack C. Wileden. 2005. “Ontologies for Supporting Engineering Analysis Models.” *Artificial Intelligence for Engineering Design, Analysis and Manufacturing: AIEDAM* 19(1):1–18.

Holsapple, Clyde W. and K. D. Joshi. 2002. “A C OLLABORATIVE A PPROACH to ONTOLOGY DESIGN.” 45(2).

Jens, Lehmann, Lauenroth Kim, Heim Philipp, Lohmann Sebastian, and Riecher Thomas. 2018. “Semantic Web Ontology for Requirements Engineering (SWORE).” Retrieved (http://ns.softwiki.de/req/2/index-en.html).

Kim, Kyoung Yun, David G. Manley, and Hyungjeong Yang. 2006. “Ontology-Based Assembly Design and Information Sharing for Collaborative Product Development.”

CAD Computer Aided Design 38(12):1233–50.

Kortelainen, Juha. 2011. *Semantic Data Model for Multibody System Modelling*.

LBD. 2013. “Tool: IFC-to-RDF Conversion Tool.” Retrieved (http://linkedbuildingdata.net/tools/tool-ifc-to-rdf-conversion-tool/).

LBNL. 2013. “Simergy: Simergy Homepage.” Retrieved (https://simergy-beta.lbl.gov/).

Li, Zhanjun, Victor Raskin, and Karthik Ramani. 2007. “A Methodology of Engineering Ontology Development for Information Retrieval.” *Proceedings of ICED 2007, the 16th International Conference on Engineering Design* DS 42(August):1–12.

Linkeddata.org. 2014. “Linked Data - Connect Distributed Data across the Web.” Retrieved (http://linkeddata.org/).

López, MF, A. Gómez-Pérez, AP Sierra, and JP Sierra. 1999. “Building a Chemical Ontology Using METHONTOLOGY and the Ontology Design Environment.” *IEEE Intelligent Systems* 4:37–46.

Ma, Xiao, Jay Bal, and Ahmad Issa. 2014. “A Fast and Economic Ontology Engineering Approach towards Improving Capability Matching: Application to an Online Engineering Collaborative Platform.” *Computers in Industry* 65(9):1264–75.

Mikos, Walter L., João C. E. Ferreira, Paulo E. A. Botura, and Leandro S. Freitas. 2011. “A System for Distributed Sharing and Reuse of Design and Manufacturing Knowledge in the PFMEA Domain Using a Description Logics-Based Ontology.” *Journal of Manufacturing Systems* 30(3):133–43.

Negri, Elisa, Luca Fumagalli, Marco Garetti, and Letizia Tanca. 2016. “Requirements and Languages for the Semantic Representation of Manufacturing Systems.” *Computers in Industry* 81:55–66.

Niknam, Mehrdad and Saeed Karshenas. 2017. “A Shared Ontology Approach to Semantic Representation of BIM Data.” *Automation in Construction* 80:22–36.

Noy, Natalya F. and Deborah L. McGuinness. 2001. “Ontology Development 101: A Guide to Creating Your First Ontology.” *Stanford Knowledge Systems Laboratory* 25.

O’Donnell, James, Richard See, Cody Rose, Tobias Maile, Vladimir Bazjanac, and Philip Haves. 2013. “SIMMODEL: A DOMAIN DATA MODEL FOR WHOLE BUILDING ENERGY SIMULATION.”

Oraskari, Jyrki and Seppo Törmä. 2015. “RDF-Based Signature Algorithms for Computing Differences of IFC Models.” *Automation in Construction* 57:213–21.

Pakonen, Antti, Teemu Tommila, Teppo Pirttioja, and Ilkka Seilonen. 2007. “OWL Based Information Agent Services for Process Monitoring.” *IEEE International Conference on Emerging Technologies and Factory Automation, ETFA* 9–16.

Pandit, Aarti and Yimin Zhu. 2007. “An Ontology-Based Approach to Support Decision-Making for the Design of ETO (Engineer-To-Order) Products.” *Automation in Construction* 16(6):759–70.

Park, J. M., J. H. Nam, Q. P. Hu, and H. W. Suh. 2008. “Product Ontology Construction from Engineering Documents.” *ICSMA 2008 - International Conference on Smart Manufacturing Application* 305–10.

Pauwels, P., E. Corry, and J. O’Donnell. 2014. “Representing SimModel in the Web Ontology Language.” *Computing in Civil and Building Engineering* 955–1865.

Pauwels, Pieter, Rens Bod, Danilo Di Mascio, and Ronald De Meyer. 2013. “Integrating

Building Information Modelling and Semantic Web Technologies for the Management of Built Heritage Information.” *Proceedings of the DigitalHeritage 2013 - Federating the 19th Int’l VSMM, 10th Eurographics GCH, and 2nd UNESCO Memory of the World Conferences, Plus Special Sessions FromCAA, Arqueologica 2.0 et Al.* 1:481–88.

Pauwels, Pieter and D. Van Deursen. 2012. “IFC-to- RDF: Adaptation, Aggregation and Enrichment.” Pp. 1–3 in *IN Workshop Report for the 1st International Workshop on Linked Data in Architecture and Construction*.

Pauwels, Pieter, D. Van Deursen, Jos De Roo, T. Van Ackere, Ronald De Meyer, R. Van de Walle, and J. Van Campenhout. 2011. “Three-Dimentional Information Exchange over the Semantic Web for the Domain of Architecture, Egnineering, and Construction.” *AI Edam* 317–32.

Pauwels, Pieter, Tarcisio Mendes De Farias, Chi Zhang, Ana Roxin, Jakob Beetz, Jos De Roo, and Christophe Nicolle. 2016. “Querying and Reasoning over Large Scale Building Data Sets: An Outline of a Performance Benchmark.” *Proceedings of the ACM SIGMOD International Conference on Management of Data*.

Pauwels, Pieter, Ronald De Meyer, and J. Van Campenhout. 2010. “Interoperability for the Design and Construction Industry through Semantic Web Technology.” Pp. 143–58 in *5th International Conference on Semantic and Digital Media Technologies*.

Pauwels, Pieter and Ana Roxin. 2016. “SimpleBIM: From Full IfcOWL Graphs to Simplified Building Graphs.” *EWork and EBusiness in Architecture, Engineering and Construction - Proceedings of the 11th European Conference on Product and Process Modelling, ECPPM 2016* 11–18.

Protégé. 2019. “Products.” Retrieved (https://protege.stanford.edu/products.php).

Silver, Gregory A., Osama Al-Haj Hassan, and John A. Miller. 2007. “From Domain Ontologies to Modelling to Executable Simulation Models.”

Silver, Gregory A., John A. Miller, Maria Hybinette, Gregory Baramidze, and William S. York. 2011. “DeMO: An Ontology for Discrete-Event Modeling and Simulation.” *Simulation* 87(9):747–73.

Tah, Joseph H. M. and Henry F. Abanda. 2011. “Sustainable Building Technology Knowledge Representation: Using Semantic Web Techniques.” *Advanced Engineering Informatics* 25(3):547–58.

Törmä, S. 2013. “Semantic Linking of Building Information Models.” in *7th IEEE International Conference on Semantic Computing*.

W3C. 2004. “OWL Web Ontology Language Overview.” Retrieved (https://www.w3.org/TR/owl-features/).

W3C. 2013. “SPARQL 1.1 Query Language.” Retrieved (https://www.w3.org/TR/sparql11-query/).

W3C. 2019. “Semantic Web.” Retrieved (https://www.w3.org/standards/semanticweb/).

Weiten, Moritz. 2009. “OntoSTUDIO® as a Ontology Engineering Environment.” Pp. 51–60 in *Semantic Knowledge Management: Integrating Ontology Management, Knowledge Discovery, and Human Language Technologies*. Springer.

Wiesner, Andreas, Jan Morbach, and Wolfgang Marquardt. 2011. “Information Integration in Chemical Process Engineering Based on Semantic Technologies.” *Computers and Chemical Engineering* 35(4):692–708.

Wikipedia. 2019. “Triplestore.” Retrieved (https://en.wikipedia.org/wiki/Triplestore).

Yao, Yuangang, Lanfen Lin, and Jinxiang Dong. 2009. “Research on Ontology-Based Multi-Source Engineering Information Retrieval in Integrated Environment of Enterprise.” *Proceedings - 2009 International Conference on Interoperability for Enterprise Software and Applications, IESA 2009* 277–82.

Zhang, W. Y. and J. W. Yin. 2008. “Exploring Semantic Web Technologies for Ontology-Based Modeling in Collaborative Engineering Design.” *International Journal of Advanced Manufacturing Technology* 36(9–10):833–43.

Appendix A: The authority requirements for a mileage marathon competition vehicle

Table 4: The authority requirements for a mileage marathon vehicle, set by the Finnish Mileage Marathon Club ry (Finnish Mileage Marathon Club r.y. 2019)

Name	Document Object Class	Description
genRegu-1	General Regulations	1 PRINCIPLE OF RACE: The principle of the race is to use as little fuel as possible during the race on a given distance to be covered within a given time according to these regulations.
genRegu-2	General Regulations	2 SCRUTINISING: On the day of the race no vehicle will be admitted onto the racetrack before the Technical Stewards, appointed by the organisers, have approved the design, construction, road worthiness, braking efficiency, safety and compliance with these regulations, especially those relating to propulsion and fuel supply system. The Technical Stewards will be considered as being de facto judges of all these aspects, and their decision is final without appeal. This approval will not prejudge the results of the subsequent inspections that the Technical Stewards may make at any time during the competition. The Technical Stewards have the right to seal any part or component of the vehicle if they consider it as necessary. The Jury reserves the right to claim any parts of the vehicle for inspection. It should be able to make this inspection within 30 minutes after the claim. The Jury should make such a claim within 30 minutes after the end of the race. Only the persons appointed by the Jury, including representatives of competitor in question, are permitted to be present at the inspection. If needed the parts may be dismantled. The Jury reserves the right to penalties, up to disqualification, if regulations have been violated. In cases where regulations have not been violated the representatives of the Jury are bound to confidentiality. If any technical changes are made to the vehicle after scrutinising, the changes must be scrutinised before competing.
genRegu-3	General Regulations	3 METHOD OF PROPULSION: A heat engine using only fuels mentioned in article 6. of these regulations must solely produce the propulsion. The type or design of the engine will not be subject to any restrictions. If any kind of stored energy (electric, pneumatic, etc.) is used for other purposes than for self-starter, ignition system, measuring and control instrument circuits and injector nozzle, the competitor must prove the Technical Stewards that this energy is replaced during the race by the engine. However use of stored electrical energy is tolerated on the engine lubricant circulation when starter motor is running. Pressurising the fuel with air is also allowed on conditions mentioned in article 8.
vehDes-1	Vehicle Design	4 GENERAL CONSTRCUTION AND SAFETY OF VEHICLE (GCASOV)
vehDes-2	Vehicle Design	4.1 GCASOV: The structure of the vehicle must be safe to its driver and surroundings. The vehicle must not have any sharp edges of prominent parts that may be of danger to others. The vehicle must have 3 or 4 carrying wheels, which, in normal running conditions, must all be in continuous contact with the track. The maximum height of the vehicle, measured at the highest point of the vehicle, is 1.25 times the track of the two outermost wheels, which must be at least 50 cm and at most 110 cm. The wheelbase of the fore- and rearmost axles must be at least 100 cm.
vehDes-3	Vehicle Design	4.2 GCASOV: The vehicle must have at least two brakes or fully independent braking systems, so that a failure in one of the systems does not prevent the other from operating. However these braking systems may effect on the same wheel if it is the centremost wheel of the 3-wheeled vehicle. The brakes will be submitted to the Technical Stewards for approval. The driver must be able to operate the brakes without loosing the steerability of the vehicle. The efficiency of the brakes will be tested using a test-bench where the vehicle is inclined to a 20% slope. The brake being controlled may not slip during the test. Each brake will be tested separately.
vehDes-4	Vehicle Design	4.3 GCASOV: A self-starter may be used during the race on condition that it can only operate when the ignition and fuel supply systems are operating normally. It must be demonstrated that the starter does not provide the vehicle with any propulsive force. The vehicle must have a red light indicating the operation of the starter. The brightness of the light should be equal to car brake light and it must be seen to both sides and back of the vehicle. This indicator light must be connected directly to terminals of an electric self-starter. See example on Regulation Appendixes.
vehDes-5	Vehicle Design	4.4 GCASOV: The vehicle must have a clutch system so that it can be immobilised on the start line just before the start of the attempt and then make a standing start without any outside assistance.

vehDes-6	Vehicle Design	4.5 GCASOV: The driving compartment must be designed in order to enable outside assistance to easily extract the driver from the vehicle. Vehicles must be equipped with a cockpit opening large enough for the driver to get easily out of the vehicle by one's own means. This opening can be fully or partially closed by a hinged, removable or folding element on condition that an opening mechanism can be easily actuated both from the inside and the outside without the use of any tool, and that the outside opening system is clearly indicated. It is forbidden to fasten or consolidate the fastening of the cockpit cover with tape. Both sides of the cockpit must provide a protection for the driver against possible lateral shocks. A prone 'head-first' driving position is prohibited.
vehDes-7	Vehicle Design	4.6 GCASOV: In the normal driving position, the driver must have adequate direct 180-degree horizontal angle of vision to the front. This visibility must be achieved without aid of any optical devices. The vehicle must be equipped with driving mirrors providing rear view on both sides. The correct visibility of these mirrors will be submitted to the Technical Stewards for approval. The recommended minimum size of the mirrors is 25 cm2 each.
vehDes-8	Vehicle Design	4.7 GCASOV: There must be a fireproof separation or bulkhead between driving and engine compartments in such a way that the driver would not have a direct contact with the possible fire. Only control and measuring circuits and electric wires may pass through this separation. This separation does not have to be fixed.
vehDes-9	Vehicle Design	4.8 GCASOV: The front and rearwheels may be steered. However the organisers would like to draw the attention of the participants to the fact that steerable rearwheels might have a negative influence on the vehicle's stability.
vehDes-10	Vehicle Design	4.9 GCASOV: All intentional changes to the aerodynamic form of the vehicle during the attempt are prohibited.
vehDes-11	Vehicle Design	4.10 GCASOV: The vehicle must be equipped with an effective horn. Use of an automobile type horn is recommended.
vehDes-12	Vehicle Design	4.11 GCASOV: The maximum permitted sound level is 100 dB(A) measured on soft ground at 50 cm from the side of the vehicle's exhaust outlet.
vehDes-13	Vehicle Design	4.12 GCASOV: The wheels inside the bodywork must be made inaccessible to the driver by a partition.
vehDes-14	Vehicle Design	4.13 GCASOV: The vehicle must be fitted with an efficient rollbar, the transversal size of which must be larger than the height and breadth of the drivers allowed to drive the vehicle. This rollbar must be able to stand a 70 kg static force applied in its centre without bending. To improve the driver's safety in case of a collision, it is recommended that the carrying chassis of the vehicle extends in front of the driver's feet or there is a rigid protective device fixed firmly to the carrying chassis of the vehicle. It is also recommended that there are no rigid structures above the driver's legs.
vehDes-15	Vehicle Design	4.14 GCASOV: Driver's seat must be equipped with an efficient safety belt with a buckle specifically designed for this purpose. The belt must have at least 3-point structure and it must be firmly attached to carrying chassis or rollbar, not to removable parts of bodywork. The belt must have 2 shoulder harnesses and it must have a symmetrical structure in relation to the driver. 5-point belt is recommended. A diagonal 3-point structure usually used on road cars is not accepted. Please see Regulation Appendixes for further info.
vehDes-16	Vehicle Design	4.15 GCASOV: There must be a switch or similar device fitted on the outside surface of the vehicle in order to turn off the engine. This device must be marked with a sticker specified on Regulation Appendixes.
vehDes-17	Vehicle Design	4.16 GCASOV: Each vehicle must be equipped with a fire-extinguishing device. Acceptable devices are either an extinguishing blanket having a minimum size of 90 x 120 cm or a fire extinguisher having a minimum capacity of 1 kg.
vehClas-1	Vehicle Classes	5 OPEN CLASS: The competition has only open class. Basic class has been discontinued since 2012.
fuelSS-1	Fuel Supply System	6 FUEL: The only fuels that may be used are those specified in he invitation of the event. These fuels will be supplied by the organisers. The competitors can procure the quantities of fuel required for practising and the race from the officials responsible for measuring the fuel consumption. This fuel must be used alone with no additive; only the power produced in the engine by its combustion with air can be used for propulsion, with the exception of factors considered natural, such as wind and gradient. No other product liable to be used as fuel should be transported on board the vehicle. Water injection is also permitted.
fuelSS-2	Fuel Supply System	7 GENERAL REGULATIONS OF FUEL SYSTEM (GROFS)
fuelSS-3	Fuel Supply System	7.1 GROFS: The fuel supply system must be translucent and designed in such a way that it can be fully drained and filled again before the attempt up to a given mark. After a top-up, after the attempt, it should provide an exact indication of the volume of fuel consumed. The competitors are recommended to carefully avoid any increase in the temperature of the circuit, which would lead to the formation of vapour bubbles. Conversely, cooling of the fuel below the ambient temperature is not permitted.
fuelSS-4	Fuel Supply System	7.2 GROFS: Competitor must use a fuel tank specified in the invitation of the event.

fuelSS-5	Fuel Supply System	7.3 GROFS: All fuel and pressure circuits, including the pressure reservoir, must be of translucent and semi-rigid or rigid materials. All pipes of the system must be made of the non-coloured polyamide-tube used for pneumatic assemblies.
fuelSS-6	Fuel Supply System	7.4 GROFS: There must not be any kind of valve, non-return valve, gauge, etc., fitted to the fuel pipe between the tank and the fuel distributor (carburettor / injector nozzle / pressure-pump). As an exception to this a translucent non-coloured fuel filter, and for a diesel engine a switch-off valve, are permitted.
fuelSS-7	Fuel Supply System	7.5 GROFS: Design of fuel system, including fuel distributor, must be such that possible vapour bubbles can be easily noticed and removed. Inside diameter of fuel pipes after tank must be at least 4 mm for easiness to remove possible bubbles.
fuelSS-8	Fuel Supply System	7.6 GROFS: The fuel system must not, even partially, be situated in driving compartment. The whole fuel supply system must be in a ventilated compartment behind a fire-proof separation or bulkhead and inaccessible and unalterable by the driver except for the control circuits.
fuelSS-9	Fuel Supply System	7.7 GROFS: If the fuel system incorporates any mechanism or device regulating the fuel flow (e.g. float- or diaphragm-chamber), there must be a provision for testing its operation by the ability to draw off some of the fuel inside that system. There must, therefore, be a drain tap or similar device acceptable to the Technical Stewards. When drawing off the fuel, the fuel level in the tank must drop. When the fuel drawn off is returned to the tank, the fuel level should rise back to the original level.
fuelSS-10	Fuel Supply System	7.8 GROFS: Only non-pressurised fuel system is permitted for carburettor-engines.
fuelSS-11	Fuel Supply System	7.9 GROFS: Recycling engine blowby gas back to the engine is prohibited during the race.
fuelSS-12	Fuel Supply System	8 PRESSURISED FUEL SYSTEM (PFS)
fuelSS-13	Fuel Supply System	8.1 PFS: When pressurised fuel system is used, the maximum pressure allowed, including the pressure reservoir, is 5 bars and the vehicle must be fitted with a pressure meter, which constantly shows the pressure in the system. Normal running pressure must be clearly indicated on the meter. The pressure must not significantly change during the attempt. The fuel tank must be at atmospheric pressure when the measurements of the fuel level are made.
fuelSS-14	Fuel Supply System	8.2 PFS: The pressure system must have a coupling for a reference pressure meter of the organiser. Accepted coupling is a normal car tyre valve, which has a thread and needle compatible to TR412 valve (Ø 7.7 mm).
fuelSS-15	Fuel Supply System	8.3 PFS: If a pressure regulator is used in the system, the pressure meter and the coupling mentioned in articles 8.1 and 8.2. must be on both sides of the regulator.
fuelSS-16	Fuel Supply System	8.4 PFS: In pressurised fuel systems the maximum capacity of the tank is 100 ml.
fuelSS-17	Fuel Supply System	8.5 PFS: The pressurised air circuit must be equipped with a safety valve set to 5 bars maximum.
fuelSS-18	Fuel Supply System	8.6 PFS: Pressurisation of the system, including the pressure reservoir, will take place at the start top-up by the means of a hand-pump.
fuelSS-19	Fuel Supply System	PRESSURE-PUMP (PP)
fuelSS-20	Fuel Supply System	9.1 PP: There is no pressure limit for using a pressure pump. If the pump produces over 10 bar pressure the fuel pipes between the pump and the injection nozzle must be metallic.
fuelSS-21	Fuel Supply System	9.2 PP: The pipe between the pump and the injector nozzle must not be fitted with any kind of a valve, non-return valve, coupling, tap etc.
fuelSS-22	Fuel Supply System	9.3 PP: The fuel system before the pressure-pump must be non-pressurised.
fuelSS-23	Fuel Supply System	9.4 PP: Operating power for a pressure-pump must be taken from the engine. N.B.! For example an electric pump is permitted on conditions mentioned in article 3.
safety-1	Safety	10 DRIVING: No vehicle should be moved or driven on the track against the driving direction. Competitors are requested to leave room for those wishing to pass them. All passing must be done with utmost care. The competitor driving ahead is allowed to choose one's driveline freely as long as it does not deliberately interfere or danger other competitors. Slip streaming of other competitors is prohibited.
safety-2	Safety	11 DRIVER'S SAFETY: On the racetrack, during both the testing and the race, the drivers must wear protective helmets, which will be submitted to the Technical Stewards for approval. The helmet must be approved to traffic-use in EU. Cyclist-helmets are forbidden. It should always be possible for the drivers to get out of their vehicles or for rescuers to remove them in case of accident or emergency. Drivers may not wear synthetic clothes.

safety-3	Safety	12 ACCESS TO RACETRACK: Throughout the race, all members of the teams must keep off the track, with the exception of the drivers on or in their vehicles. No vehicle nor any member of the team personnel should enter the track to provide assistance without special permission from the organisers. In case of failure or accident, the driver must remove his/her vehicle from the track. If he/ she no longer wishes to continue, he/ she must wait for the assistance until helpers are given permission to go to the track.
racePr-1	Race Procedure	13 DISTANCE AND SPEED OF COMPETITION: The competitors must complete a given distance at a minimum average speed of approximately 25 km/h. The distance and maximum time are specified in the Invitation. Switching off the engine and rolling on neutral are allowed during the race. In case of delay considered excessive by the Timekeepers, who will in this respect to be considered as de facto judges, the competitor concerned will not be timed and should give way to those awaiting their turn, thus renouncing any priority over them.
racePr-2	Race Procedure	14 START OF RACE: The vehicles must be stopped on the start line and make a standing start engine running with no outside assistance. The vehicle on the start line must give way to those already taking their laps on the track. The competitors will wait until the start line is clear in order to get into place in their turn, but the starting orders can be drawn by lot if the Timekeepers so decide.
racePr-3	Race Procedure	15 INCIDENTS DURING RACE: The driver will be required to indicate to the Timekeepers or the Technical Stewards any movement, made or attempted, by means other than the vehicle’s own motive power, and the lap will not be taken into account. If this type of incident is not indicated the driver will be automatically excluded. Nevertheless, if repairs can be made on the spot and if the vehicle has not advanced, the laps need not be invalidated. The competitors will be solely responsible for submitting themselves to all the aforesaid obligations to the Timekeepers and Technical Stewards.
racePr-4	Race Procedure	16 MINIMUM WEIGHT OF DRIVER: The minimum weight of the driver wearing his/ her racing clothes is 45 kg. Ballast will be fitted in the vehicle in case the minimum weight is not met. The weight of the driver as well as the use of the ballast are surveyed during the competition by random checks. At the finish-line a 1 kg tolerance to the weight of the driver is accepted.
racePr-5	Race Procedure	17 RESULTS AND MEASUREMENT (RAM)
racePr-6	Race Procedure	17.1 RAM: Before the start of the attempt the Fuel Measurers will top-up the tanks. After the attempt competitors must not carry out any work on their vehicles before having permission by Technical Stewards or Fuel Measurers. The Measurers will measure the fuel consumed during the attempt either by volume or by mass. The completed race laps will only be taken into consideration for the results if they have been completed in the time allowed during the hours in which the track is officially opened to the competitors and if the vehicle’s fuel consumption is officially measured and the sheet duly signed. The competitors will be entitled to obtain confirmation from the official Timekeeper that they have duly completed the race in the regulation time before having their fuel consumption measured. The organisers reserve the right to define the procedures by which the consumed fuel volume is measured and if required, corrected depending on temperature variations and to measure the total volume of fuel contained in the fuel supply system.
racePr-7	Race Procedure	17.2 RAM: All fuel quantities will be normalised to following physical values: - Gasoline density 0.750 kg/l, caloric value 43.5 MJ/kg, temperature 15oC - Diesel: density 0.835 kg/l, caloric value 43.5 MJ/kg, temperature 15oC
racePr-8	Race Procedure	17.3 RAM: Provisional results will be displayed during the competition and will indicate the competitors energy consumption calculated as km/kWh as well as fuel consumption in l/100km and km/l. The order will be based on the energy consumption. The units in fuel consumption will be based on units shown on article 17.2 (diesel ≠ gasoline).
racePr-9	Race Procedure	18 COMPLAINTS: Complaints will only be accepted from the team managers or the drivers and will be received by the Technical Stewards or the Jury. According to their subject, these complaints should be made within the following times: - Vehicles : within ten minutes following the end of the race. - Conduct of competitors and drivers : within ten minutes following the end of the race. - Results : within 30 minutes after the result in question is displayed.
racePr-10	Race Procedure	19 DISPUTES: In case of dispute the decision of the Jury will be binding and without appeal.
racePr-11	Race Procedure	20 RIGHTS OF ORGANISERS: The organisers reserve the right: - To modify, postpone, or cancel the race in the case of unforeseen circumstances, notably on mete-orological grounds. No indemnification will be paid. - To exclude, disqualify or penalise any competitor who, according to the Jury’s judgement, may have been assisted thus violating these regulations, may have impeded other competitors, may have strayed from the normal racetrack or have acted in such a way as to provide a wrong idea of the results, in particular insofar as concerns the fuel consumption or propulsion.