MASTER

The added value of an owl concept library in the development stage of the systems engineering process

a research of implementing and adapting an OWL concept library to standardize and improve the development stage of the Systems Engineering process

van Thiel, B.J.A.

Award date:
2014

THE ADDED VALUE OF AN OWL CONCEPT LIBRARY IN THE DEVELOPMENT STAGE OF THE SYSTEMS ENGINEERING PROCESS

A research of implementing and adapting an OWL concept library to standardize and improve the development stage of the Systems Engineering process

EINDHOVEN UNIVERSITY OF TECHNOLOGY
ARCADIS

August 18, 2014
: Final version

Colophon

Report : The added value of an OWL concept library in the development stage of the Systems Engineering process
Status : 2.0
Date : August 18, 2014
Place : Amersfoort

Author : B.J.A. (Bob) van Thiel
Student number : 0780165
E-mail : b.j.a.v.thiel@student.tue.nl
University : Eindhoven University of Technology
Master track : Construction, Management and Engineering

Chair : Design Systems

IN COLLABORATION WITH
Company : ARCADIS
Division : Mobility
Advice group : Road, Traffic and Information Management

GRADUATION COMMITTEE

Chairman
Prof. Dr. Ir. B. (Bauke) de Vries
b.d.vries@tue.nl
Faculty of Built Environment
Eindhoven University of Technology

Supervisor
Dr. Dipl. –Ing. J. (Jakob) Beetz
j.beetz@tue.nl
Faculty of Built Environment
Eindhoven University of Technology

First extern supervisor
Ir. E.C.M. (Ewoud) de Borst
ewoud.deborst@arcadis.nl
Advisor Systems Engineering
ARCADIS

Second extern supervisor
Ir. R.J. (Ronald) van Schie
ronald.vanschie@arcadis.nl
Team leader Integral Design
ARCADIS

Preface

I am happy to present this report to you, which is the result of my 6 month graduation research carried out in collaboration with Eindhoven University of Technology and ARCADIS. This report is my final piece before completing my time as a student. The 3 years I spend studying at the university in Eindhoven have quickly passed, but they were very important for my personal development.

It was not easy to fulfil my research objective and complete this report. In January 2014 I did not know which out of all the different knowledge areas I had to focus on. The knowledge areas are as best described by the keywords of this report; semantic web, ontology, RDF(S) and OWL, concept library and systems engineering. During the process it became more clear, week by week. The main focus of my research was to fill in the knowledge gaps between the keywords. That is the reason of my front page. For me it is a remarkable sketch of my graduation period (thanks to Daan Oostinga).

Like I said, it was not easy. But I have no regrets about the difficulty of my topic. All the experiences and knowledge I gained in these months are the best foundation for my next step.

I never could produce this level of report without the help of some people. During moments when I lost my 'way' these persons helped me to find my way back. A special thanks to my three closed supervisors, Ewoud de Borst, Ronald van Schie (ARCADIS) and Jakob Beetz (Eindhoven University of Technology). Each meeting, Ewoud had always the right questions according to my research. Not to answer but to think about. Ronald kept me on the way by focussing on the link between science and practise. I appreciate the time, care and interest both my daily supervisors had in both the subject and in my graduation process. Jakob gave me useful information and advice according to the semantic web. With his expertise I could reach the needed depth for my research. Furthermore, I want to thank Bram Mommers (ARCADIS), who helped me to start up this research at ARCADIS. All employees of ARCADIS, that I had to pleasure to talk and interview. Daan Oostinga (director Semmtech), who helped me to understand the links between all important components of my research. Chi Zhang (PhD student Eindhoven University of Technology), with brainstorming about the case study.

Last but not least, a big thanks to my family and friends. The support of my parents were essential for me. Not only during my graduation research, but during my whole 'student career'. Together with my (international) friends they made my student life unforgettable!

Bob van Thiel

Beuningen, August 2014

Summary (English)

Currently, information systems have started to use semantic web ontologies, namely expressed by means of OWL, to store and manage domain knowledge. This research connects an information system, called concept library, to the Dutch construction sector. Store and manage domain knowledge is one of the focus points of the BIR[1]. However, nowadays there are more than one concept library in development. These libraries occur on different levels; organisational, national and international. The semantic web offers open standards (one of them is OWL) which make it possible that concept libraries (including semantics) can be interpreted by the computer. The magnifying modelling capabilities of OWL in comparison to RDF(S) give us the possibility to make the digital information semantically richer. Are the modelling capabilities of OWL in a concept library of added value during the construction process? More specific, during the development stage of the Systems Engineering (SE) process?

The development stage of the SE process is an interesting focus area for research, because new concepts are introduced, applied and transferred to next SE stages. Literature and interviews highlighted the importance of decomposing a system (or read project) into sub-systems (functional breakdown structure and object breakdown structure) and verifying these sub-systems. A case study is conducted from these important point of views. Demonstrations illustrate the support of OWL in these two cases; setting up a so-called 'mereology' and verifying concepts. The first demonstration illustrates that a decomposition structure of a functional and object breakdown structure can be modelled in OWL. This mereology occurs besides a taxonomy as hierarchical structure within an ontology. This decomposition structure can be queried by using a SPARQL-DL engine. The second demonstration tests the consistency of the ontology. This test is the result of the consternation that a full verification is not possible. This is the result of the fact that the semantic web is based on Open World Assumption. A higher consistency check can be reached by using OWL in combination with SPARQL-DL and SWRL. It is important to categorize the system requirements according to the project. Generic and semi-generic requirements are useful to model within the ontology. This meets one of the main objectives of the concept library; (re)using of generic concepts.

This research focusses on the extension of the modelling capabilities of OWL in comparison to RDF(S). RDF(S) is able to model taxonomies, but the representation power of OWL is higher. Concept libraries, in this case within the domain of the construction industry, are able to express more than only a taxonomy. To use the development stage of the SE process as focus point, we conclude that these libraries can be modelled semantically richer and have added value on the process.

[1] Construction Information Board.

Samenvatting (Dutch)

Dit onderzoek sluit aan op het feit dat de Nederlandse bouwsector een probleem heeft met eenduidige vastlegging van digitale informatie (o.a. een speerpunt van de BIR[2]). Conceptenbibliotheken staan in de kinderschoenen, maar moeten dit probleem gaan oplossen. Echter, er zijn meerdere conceptenbibliotheken in ontwikkeling, zowel op organisatorisch, nationaal en internationaal niveau, en er zijn meerdere modelleringstalen beschikbaar om deze bibliotheken op te bouwen. Het semantisch web biedt open standaarden (o.a. in de titel genoemde OWL) die bibliotheken voorzien van de nodige semantiek, wat door de computer geïnterpreteerd kan worden. Dit resulteert in een eenduidige vastlegging van sector gerelateerde concepten en hun onderlinge relatie. De vergrotende modelleringscapaciteit van OWL ten opzichte van RDF(S) biedt de mogelijkheid om informatie semantisch rijker te maken. Zullen de modelleringscapaciteiten van OWL binnen een conceptbibliotheek een toegevoegde waarde leveren tijdens het bouwproces? Meer specifieker, de ontwerpfase van het Systems Engineering (SE) proces?

De ontwerpfase van het SE proces is interessant in combinatie met de modelleringstaal OWL, omdat in deze fase nieuwe concepten worden bedacht, toegepast en mee worden genomen naar volgende SE fases. Literatuur en interviews binnen dit onderzoek geven weer dat het de-componeren van het systeem (lees ook project) in meerdere subsystemen en componenten (ontwerpen van functie- en objectenboom) en het verifiëren daarop de belangrijkste aandachtspunten zijn. Een case studie is uitgevoerd vanuit deze twee punten, waarbij getest wordt of OWL hieraan ondersteuning kan bieden vanuit een conceptenbibliotheek. De eerste demonstratie laat zien dat een decompositiestructuur van een functie- en objectenboom gemodelleerd kan worden in OWL. Deze zogenaamde ‘mereologie’ treedt naast een taxonomie als tweede ‘hiërarchische structuur’ op in de bibliotheek. Aangeraden wordt om deze mereologie met behulp van een SPARQL-DL op te vragen binnen de bibliotheek. De tweede demonstratie test de ontologie op een zo hoog mogelijke consistentie. Dit naar aanleiding van het feit dat testen op volledige verificatie niet mogelijk is, omdat het semantisch web gebaseerd is op ‘Open World Assumption’. Een hogere consistentie kan bereikt worden waarbij OWL gecombineerd wordt met SPARQL-DL en SWRL. Het is wel van belang dat de systeemeisen conform het project gecategoriseerd worden en uiteindelijk niet allemaal worden gemodelleerd in de ontologie. Generieke en herbruikbare concepten zijn hier van belang.

Dit onderzoek laat zien wat de extensie is van het modelleren in OWL ten opzichte van RDF(S), welk met name gericht is op het modelleren van taxonomies. De ontwerpfase van het SE proces is als uitgangspunt gebruikt, maar mogen het vooral duidelijk zijn dat toekomstige conceptenbibliotheken semantisch rijker gemaakt kunnen worden.

[2] Bouw Informatie Raad.

Content

List of figures

List of tables

PART A: Problem definition and research methodology

1 Introduction

1.1 INFORMATION MANAGEMENT AS A PRIMARY PROCESS

According to Forrest Research statistics, more than 80% of the data in the Internet and Intranet is in unstructured form, such as documents, emails, expert statements, drawings, photographs and videos (Yang & Tan, 2011) (Brown, 2013). Unstructured data is a generic label for describing any corporate information that is not in a database (Brown, 2013). Unstructured data can be textual (e.g. email messages, word documents) or non-textual (e.g. JPEG images, MP3 audio files). There is no exception for the construction industry; the vast majority of files produced in the whole lifecycle of a construction project contains a large amount of unstructured data, such as contracts, memos, cost budgets, purchase orders, drawings, design changes, project schedules, and so on. The only way to interpret its content or check the quality of this unstructured information depends on heavy manual work; computers cannot understand and deal with unstructured information automatically (Ram, et al., 2011). As a result, the traditional way of handling/ processing information and communication cannot satisfy the needs of a modern construction project (e.g. DBFM(O)[3] and PPP[4] contracts).

Modern construction projects include approaches such as Systems Engineering (hereafter: SE) and Building Information Modelling (hereafter: BIM). Both provide a model-based approach for managing unstructured data. It is both a method for technical developments of a system and for the management of the development process (Parnell, et al., 2011). SE thinks in systems and works through a systematic approach. Where SE focuses on risk management, BIM focuses on design integration, by using geometrical information and virtual models. The combination of SE and BIM aims to control complex projects in the construction industry and to communicate with stakeholders in a transparent, traceable and unambiguous way (van Ruijven, 2014). However, problems occur when unambiguous information is used and exchanged. Stakeholders do not use the same descriptions, definitions and interpretations, which results in miscommunication.

To make unambiguous communication possible between all stakeholders, a common language is required. Concept libraries in the construction industry (e.g. CB-NL[5], OTL[6] and

[3] Design, Build, Finance, Maintain (and Operate).

[4] Public Private Partnership.

[5] The Dutch concept library for the built environment. The CB-NL is a library or collection of concepts for the building knowledge domain.

ETIM[7]) purpose a common language. These libraries are formed from unambiguous glossaries and object libraries. Concept libraries include concepts that are 'typical things' in the domain of the construction industry. Bridge, for example, which is a physical object, a harbour, which is a spatial object, or 'connecting', which is a functional characteristic. By selecting a concept and knowing its classification, the frequency in which miscommunications occur will be reduced (van Ruijven, 2014). A concept library also provides the multiple reuse of the concepts, which are single recorded. (re)Using of unambiguous information prevents duplication (iNFRANEA, 2013). A concept library should reduce failure costs, development costs and transaction costs in construction projects (Adriaanse, et al., 2012).

Currently, the domain construction industry already has several concept libraries. For example the CB-NL. It is still in a developmental phase and is intended to function as the switchboard between all other concept libraries in the Dutch construction industry. Another example is the OTL, which is the concept library of the biggest infrastructure client in the Netherlands. To explain and define concepts and their relationships, concepts have to be modelled in a certain way. These models are used to organize human thought in the form of explanations (Allemang & Hendler, 2011). The CB-NL and OTL are both being modelled in OWL[8], a Semantic Web language (section 2.2). According to the CB-NL Modelling Guide (2013) OWL is the key result of the so-called 'Semantic Web Activity'. The Semantic Web provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries (Hawke, et al., 2013). The OWL Recommendation is now in version 2.0[9] (OWL 2 – latest version), which increases the number of capabilities of OWL 1.0 with a number of new modelling constructs, but does not change the fundamental principles of how OWL works (Allemang & Hendler, 2011). At present, the goal of flexible, granular and cascading concept libraries for the construction industry is still in one of its early conceptual stages. The detailed specification, prototypical implementation, evaluation and testing of some of these approaches are currently carried out (Beetz, 2014). In addition to Beetz (2014), for a non-expert in ontological engineering to be able to use an information system, domain-dependent business logic is needed. This research aims to acquire that logic.

1.2 MAIN FOCUS

A great need to formalize new design guidelines that concern concept libraries implemented in modern construction projects exists. This report concerns the theory of OWL and its implementation on top of current concept libraries used in the construction industry. OWL semantics is extended here to search for a higher potential of using concept libraries related to modern processes such as SE and Building (or 'infrastructure') Information Modelling. This report focuses on the SE process, or more specifically, the development stage of the SE lifecycle process. All systems that have been made during this stage have the most profound impact on project outcomes. The development stage has the highest potential to improve

[6] Object Type Library. The OTL is an intern library or collection of objects conducted by Rijkswaterstaat.

[7] Europees Technisch Informatie Model. A library or collection of electrical and mechanical products.

[8] Web Ontology Language. OWL is a W3C (footnote 8) standard to exchange and define web ontologies.

[9] OWL 2 is introduced by W3C as open standard in 2009.

the result of the entire project. The possible relation between an OWL concept library and the development stage of the SE process in the construction industry is explored and defined in this report.

Why OWL?

- Allow for a more expressive vocabulary to satisfy user requirements more comprehensively whilst retaining good computational properties;
- Supports important concept libraries in the construction industry (CB-NL and OTL);
- An W3C[10] open standard. Has become the de-facto standard for specifying ontologies.

1.3 MOTIVATION

Nowadays, the scope of a construction project is becoming more extensive , for example due to the more intensive involvement of multiple disciplines. DBFM and PPP contracts result in a higher complexity, because of an increasing number of actors, disciplines and requirements. This causes a substantial increase in the effort required to manage information (figure 1). It will be problematic to provide correct and unambiguous information. This should lead to less efficient and adequate work by the parties involved in a project (van Ruijven, 2014).

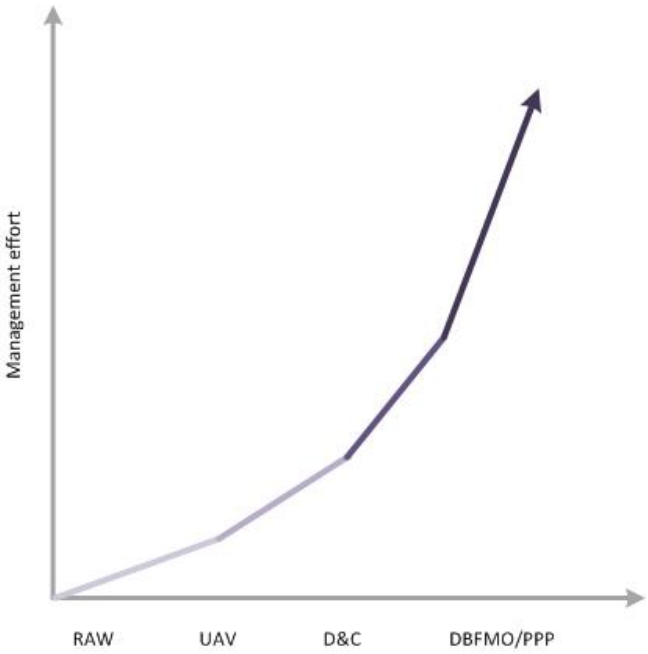


Figure 1: Relation between management effort for controlling information and complexity of contracts (van Ruijven, 2014)

Written in section 1.1, the use of SE and BIM have increased significantly during recent years. BIM is one of the most promising (technological) developments in the architecture, engineering and construction industry (Eastman, et al., 2008). Related Information and Communication Technology (ICT) ensures that computers are able to manage and transfer digital information. More than 25% of failure costs of a construction project are due to inadequate exchange of information (Busker, 2011).

Currently, Information Management is becoming a primary process in the construction industry. However, the absence of a working concept library can result in loss of confidence and the user's credibility. Nowadays, concept libraries, e.g. the CB-NL, are seen as a complex theoretical matter.

[10] The World Wide Web Consortium (W3C) is an international community that develops open standards to ensure the long-term growth of the Web.

1.4 RESEARCH OBJECTIVES

Indicated in the previous subsection, the successful implementation of SE is of great importance to the success of upcoming complex construction projects. The quantities of electronic data within construction projects are increasing, of which the largest part is non unambiguous data. This report proposes a methodology for the development stage of the SE process by evolving the expressiveness of concept libraries modelled in the OWL 2 language. Therefore, this report focusses on the interpretation, implementation and application of an OWL concept library in relation to SE. The explicit aims are:

- Determine whether OWL is useful for supporting aspects in the development stage of the SE process (according to a design task of a client);
- Formalize a model in an OWL ontology tool for implantation into the development stage of the SE process. This model should help the reader understand the most important aspects of knowledge representation from the point of view of the SE approach;
- Determine whether a concept library including OWL is of added value to the development stage of the SE process in comparison to the current SE process.

1.5 RESEARCH QUESTIONS

Based on the focus area and the research objectives, this section outlines the research questions of the proposed research. The main research question is:

> What is the added value of OWL constructs in a concept library, in relation to the development stage of the Systems Engineering process?

The main research question can be divided into a number of sub-questions. The sub-questions are based on the knowledge gaps that where identified in section 1.1 and 1.2.

- What is a Semantic Web?
- Why apply the Semantic Web?
- What are the characteristics of OWL?
- Which other semantic languages occur within the Semantic Web?
- What is a concept library?
- What is Systems Engineering?
- What are the characteristics of a development stage in the Systems Engineering process?

The results of all formulated sub-questions will not answer the main research question. These sub-questions function as knowledge baseline and input for the case study related research questions, which are formulated in chapter 6 – 'Practical framework'.

1.6 STRATEGY AND OUTLINE OF THE REPORT

Figure 2 illustrates the schematic overview of the proposed research approach. Within the figure this research is divided into four parts. Below, the strategy of this research is combined with the outline of this report.

Part A covers the problem definition and research methodology (chapter 1). The problem definition is the input for defining the research methodology, which includes the research objectives and questions.

Part B covers the desk research. The desk research is divided in a literature study, interviews and conclusions. The literature includes three chapters (chapter 2, 3 and 4), namely 'the Semantic Web', 'concept library as an ontology' and 'Systems Engineering', respectively. The literature is based on the sub research questions defined in part A. The interviews provide a better vision on the application of SE and concept libraries in the construction industry nowadays. The interviews are helpful specifying the focus area. The relationship between the different literature chapters and interviews are valuable input for part C. To conclude an overall conclusion chapter 5 is included.

Part C receives input of the answers on the sub-questions and conclusions drawn from part B. Chapter 6 – 'Practical framework' functions as a baseline in this report. New research sub-questions and scope will be set up to specify the case study, covered by chapter 7. The expectations of OWL, defined in the sub-questions, are tested in two demonstration sessions in relation to a realistic project. A software tool is used which is able to support the development of ontologies. A clear focus of current ontology management tools is to support the development of ontologies with a wide range of editing features (Waterfeld, et al., 2008). The whole case study will be conducted from the perspective of and future benefits of SE, more specifically, the development stage of the SE process.

The validation of the case study is monitored on different manners. The validation is indicated by using multiple sources for collecting data; project documentation, a software tool and interviews with involved actors. In addition, the validation is also indicated because of the reporting proof and assertions. The approach of the case study connects the literature study and gives this research a scientific contribution.

Part D: According to the case study-related conclusions in part C, final conclusions and recommendations are provided in chapter 8.

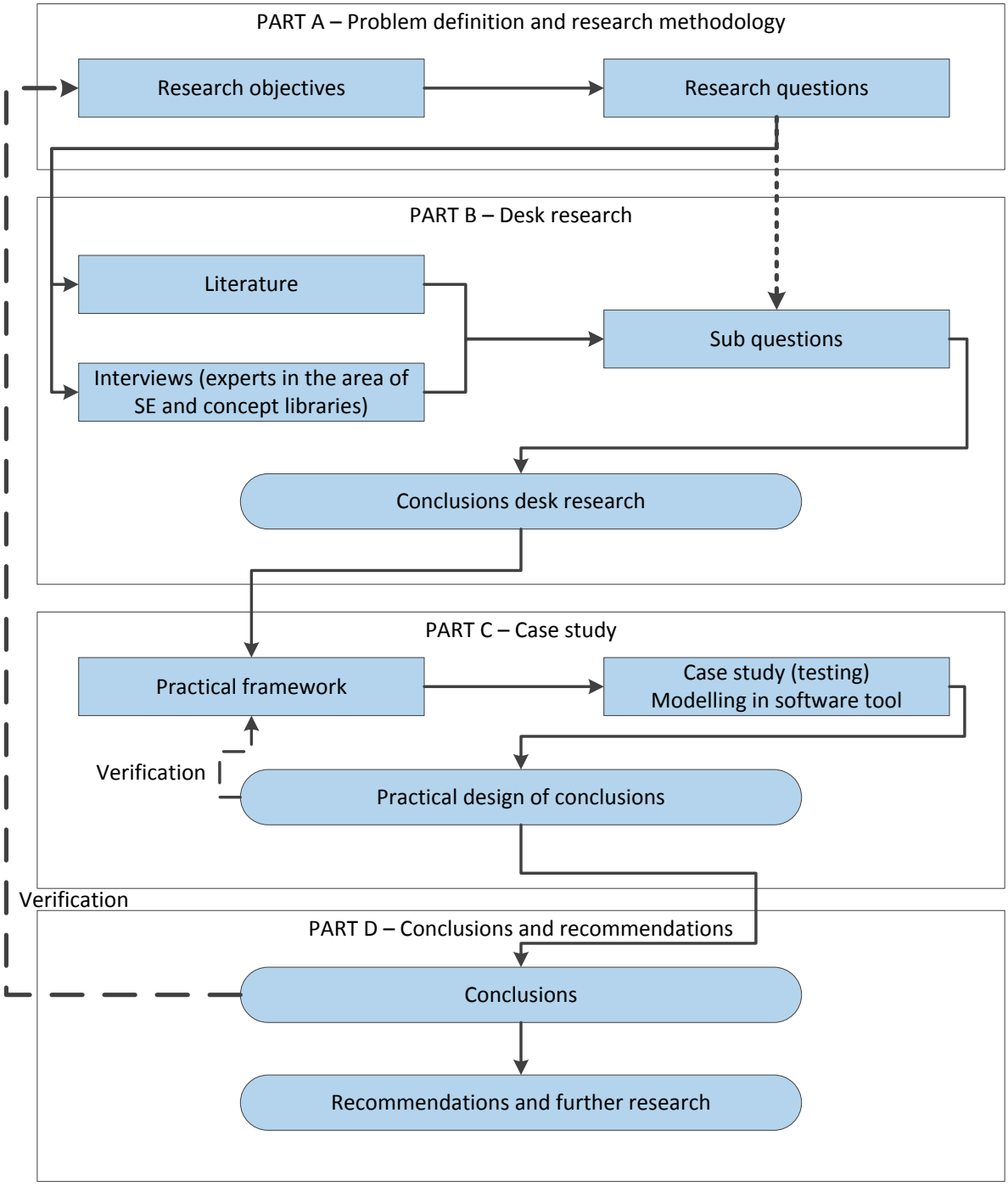


Figure 2: Research model (based on Verschuren & Doorewaard, 2007)

PART B: Desk research

2 The Semantic Web

This chapter explains the meaning and underlying definitions of the Semantic Web. The next research sub-questions are answered in this chapter:

- What is a Semantic Web?
- Why apply the Semantic Web?
- What are the characteristics of OWL?
- Which other semantic languages occur within the Semantic Web?

2.1 THE SEMANTIC WEB

The World Wide Web (WWW, or simply Web) is a collection of interlinked documents, which might be written using the (X)HTML[11] format, accessible over a standardized protocol (e.g. HTTP), and each document is identified by a Uniform Resource Identifier (URI) (de Bruijn, et al., 2008). URIs follow the very simple but effective identification principle, whereby global naming leads to a global network (Fensel & Hench, 2008). HTML has been, and still is, the dominant format for the representation of Web documents. Recently, more and more formats for the representation of data on the Web have arisen. An example is XML[12]. XML has widely been adopted as a format for exchanging structured data over the Web. In fact, it forms the basis for the formats typically used in the context of Web services as well as the Semantic Web (de Bruijn, et al., 2008).

A major drawback of the use of XML as a data model is that XML documents do not convey the meaning of the data contained in the document. Schema language such as XML schema allow constraining the format, but not the meaning of XML data (de Bruijn, et al., 2008). The key to enabling machine-to-machine communication is semantics. Semantics, in this context, is not limited to its definition in the field of linguistics (referring to the study of meaning of linguistic expressions) nor to its prior understanding in computer science (e.g. referring to the study of the formal semantics of a programming language); it is an extension of both. Semantics is machine-usable content, and the Semantic Web is the next generation of the World Wide Web which includes (and of course utilizes) such content (Fensel & Hench, 2008). The Semantic Web allows the representation and exchange of information in a meaningful way, facilitating automated processing of descriptions on the Web (de Bruijn, et al., 2008). The Semantic Web express links between information recourses on the web and

[11] HyperText Markup Language. A format for representing documents and their inter-linkage on the Web.

[12] eXtensible Markup Language is a language for the representation of (semi-)structured data.

connect information recourses to formal terminologies. These connective structures are called ontologies (Fensel, 2003). Ontologies are formalized vocabularies of terms, often covering a specific domain and shared by a community of users (W3C OWL Working Group, 2012). Sharing common understanding of the structure of information among people or software agents is one of the more common goals in developing ontologies (Musen, 1992) (Gruber, 1993). The vision of the Semantic Web: an organized worldwide system where information flows from one place to another in a smooth but orderly way (Allemang & Hendler, 2011).

> The challenge for the design of the Semantic Web is not to make a web infrastructure that is as smart as possible; it is to make an infrastructure that is most appropriate to the job of integrating information on the Web (Allemang & Hendler, 2011).

An ontology is the backbone of the Semantic Web (de Bruijn, et al., 2008). An ontology can also be defined as a Semantic Model. They allow the machine understanding of information through the links between information recourses and the terms in the ontologies (figure 3) (de Bruijn, et al., 2008). Employing URIs, ontologies – which are explicit specification of concepts in a given domain using vocabularies and relations between them (Gruber, 1993) – can be mixed, matched and merged. More about the definition of ontologies in section 2.3. Language for creating links between recourses and annotating recourses with connections to ontologies on the Semantic Web is RDF (subsection 2.2.1) (de Bruijn, et al., 2008). The Semantic Web is often mistaken for an effort to make everyone agree on a single ontology - but that just is not the way the Web works. The Semantic Web is not about getting everyone to agree, but rather about coping in a world where not everyone will agree, and achieving some degree of interoperability nevertheless. There will always be multiple ontologies, just as there will always be multiple web pages on any given topic. The Web is innovative because it allows all these multiple viewpoints to coexist (Allemang & Hendler, 2011).

The AAA-slogan

In a web of documents, the AAA-slogan[13] means that anyone can write a page saying whatever they please, and publish it to the Web infrastructure. In the case of the Semantic Web, it means that our data infrastructure has to allow any individual to express a piece of data about some entity in a way that can be combined with information from other sources. This requirement sets some of the foundation for the design of RDF (subsection 2.2.1) (Allemang & Hendler, 2011).

[13] Anyone can say Anything about Any topic.

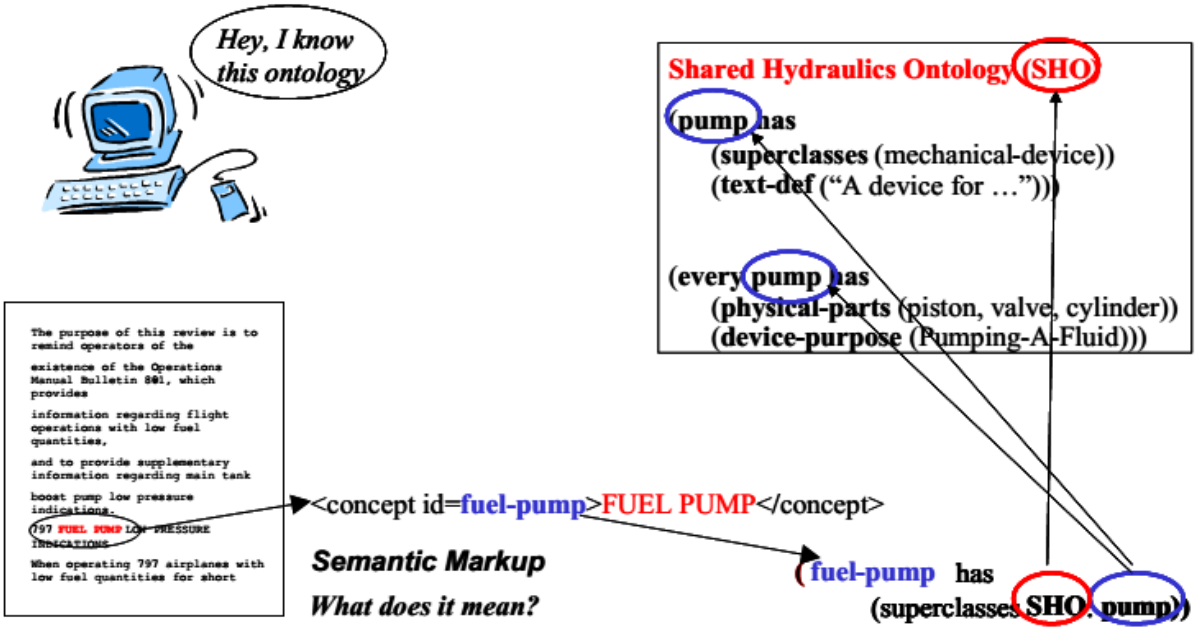


Figure 3: Formal semantics for machine processing (Uschold, 2001)

2.2 STANDARDIZED SEMANTIC WEB LANGUAGES

Described in the previous subsection, a semantic language create links between information recourses and the terms in the ontologies. This language is RDF, described in subsection 2.2.1. RDF, RDFS (subsection 2.2.2), and OWL (subsection 2.2.3) are the basic representation languages of the Semantic Web (with RDF serving as the foundation) (Allemang & Hendler, 2011). The latter is an extension of the former (de Bruijn, et al., 2008). All languages provided in this subsection are W3C standards.

2.2.1 RDF

The Resource Description Framework (RDF) is the first language developed especially for the Semantic Web. RDF is, first and foremost, a system for modelling data. It gives up in compactness what it gains in flexibility. Every relationship between any two data elements is explicitly represented, allowing for a very simple model of merging data (Allemang & Hendler, 2011). RDF uses XML for its serialization. In fact, RDF/XML is the standard syntax for RDF (de Bruijn, et al., 2008). RDF is built on top of XML. RDF does not extend XML, but XML can be used for writing down and exchanging RDF statements. RDF/XML, as the XML-based serialization of RDF is called, can be seen as an XML language (de Bruijn, et al., 2008).

RDF uses a basic building block called 'triple'. The name triple refers to the three values of the basic building block: subject – predicate – object. An example is illustrated in figure 4. The linked structure of the triple forms a directed, labelled graph, where the edges represent the named link between two resources, represented by the graph nodes. Finally, RDF has a notion of blank node, which is essentially a node that does not have a name (de Bruijn, et al., 2008).

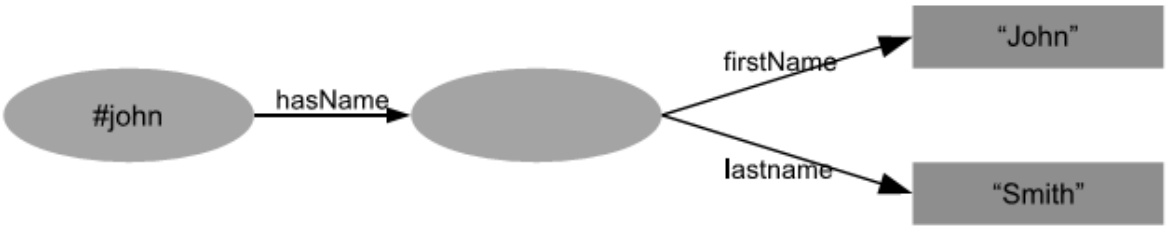


Figure 4: Example of a RDF graph. The triple form of an RDF statement: John (subject) – has last name (predicate) – Smith (object) (de Bruijn, et al., 2008)

2.2.2 RDFS

RDF itself simply creates a graph structure to represent data. RDFS provides some guidelines about how to use this graph structure in a disciplined way. It provides a way to talk about the vocabulary that will be used in an RDF graph (Allemang & Hendler, 2011). RDFS can in fact be seen as an extension of RDF with a vocabulary for defining classes, class hierarchies, properties (binary relations), property hierarchies, and property restrictions (de Bruijn, et al., 2008).

The terms 'domain' and 'range' play a role in RDFS. One benefit of the RDF property-centric approach is that it allows anyone to extend the description of existing resources, one of the architectural principles of the Web (Brickley & Guha, 2014). The *rdfs:domain* is an instance of *rdf:Property* that is used to state that any resource that has a given property is an instance of one or more classes.

The *rdfs:range* of *rdfs:domain* is the class *rdfs:Class*. This states that any resource that is the value of an *rdfs:domain* property is an instance of *rdfs:Class* (Brickley & Guha, 2014). The basic facilities provided by *rdfs:domain* and *rdfs:range* do not provide any direct way to indicate property restrictions that are local to a class. Although, it is possible to combine use *rdfs:domain* and *rdfs:range* with sub-property hierarchies (section 2.3), direct support for such declarations are provided by richer Web Ontology languages such as OWL (Brickley & Guha, 2014).

Besides *rdfs:domain* and *rdfs:range*, RDFS has some other important properties. *rdfs:subClassOf* and *rdfs:subPropertyOf* are both an instance of *rdf:property. rdfs:subClassOf* is used to state that all the instances of one class are instances of another. *rdfs:subPropertyOf* is used to state that all resources related by one property are also related by another (Brickley & Guha, 2014). These two properties are illustrated in figure 5.This figure illustrates the relation between the two instances Mary and Jack, property *rdfs:subClassOf* and some classes. In addition, all RDF(S) classes and properties (RDF vocabulary) are attached in appendix A.

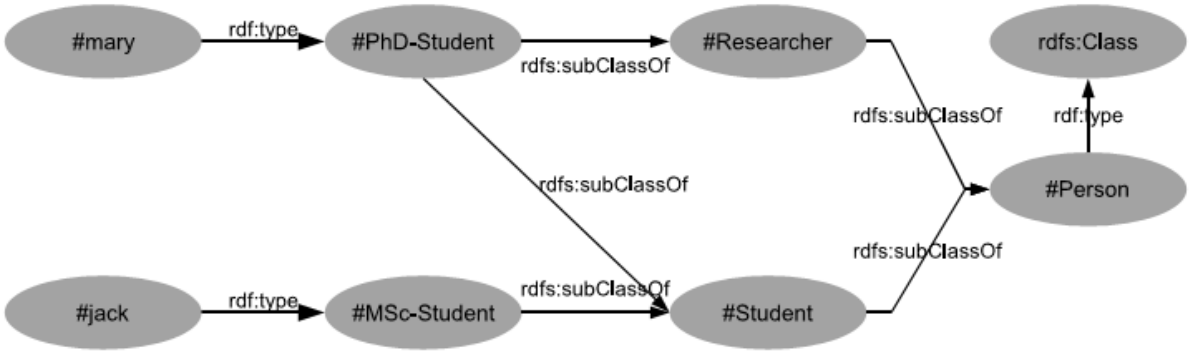


Figure 5: RDFS ontology of persons (de Bruijn, et al., 2008)

2.2.3 OWL

The Web Ontology Language OWL is an expressive ontology language that extends RDFS (de Bruijn, et al., 2008). The OWL Recommendation is now in version 2.0 (OWL 2), which extends the capabilities of OWL 1.0 with a number of new modelling constructs, but does not change the fundamental principles of how OWL Works (Allemang & Hendler, 2011). Like OWL 1, OWL 2 is designed to facilitate ontology development and sharing via the Web, with the ultimate goal of making Web content more accessible to machines (W3C OWL Working Group, 2012). Figure 6 illustrates an overview of the OWL language. At the top are various concrete syntaxes that can be used to serialize and exchange ontologies. At the bottom are the two semantic specifications that define the meaning of OWL ontologies (W3C OWL Working Group, 2012).

Written above, OWL 2 extends OWL 1. This report uses the term 'OWL' to intend all related OWL constructs (OWL 1 and OWL 2).

> OWL is a web language: based on RDF(S); and an ontology language: based on logic (Knublauch, 2004).

OWL use cases can be divided in at least two different user groups:

- OWL used as data exchange language (define interfaces of services and agents);
- OWL used for terminologies or knowledge models.
  (Knublauch, 2004)

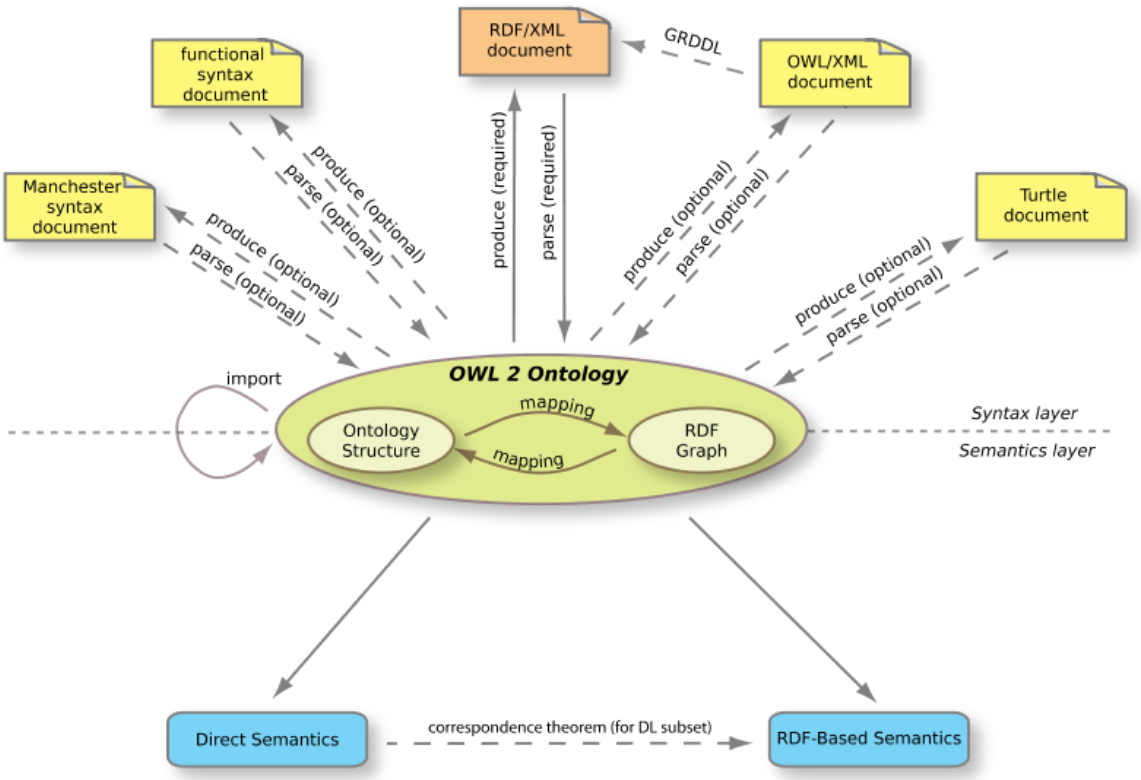


Figure 6: The structure of OWL 2 (W3C OWL Working Group, 2012)

OWL ontologies can be used along with information written in RDF, and OWL ontologies themselves are primarily exchanged as RDF documents. Figure 7 illustrates a RDF-based Semantics. In addition to the RDF vocabulary (appendix A), the OWL vocabulary is attached in appendix B.

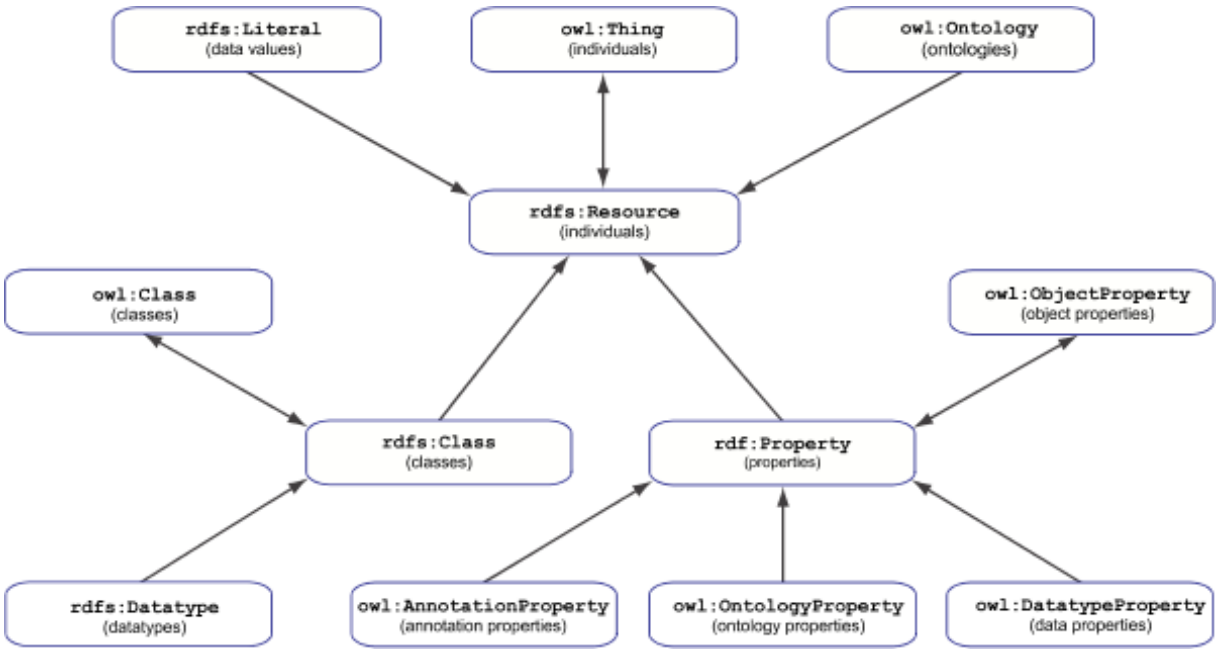


Figure 7: Parts hierarchy of the OWL 2 RDF-based Semantics (Carroll, et al., 2012)

What we see above is how to define simple classes and relationships between them in RDFS and OWL. In addition to these definitions OWL has also single constructs which enhance the representation power of OWL. These OWL constructs are:

- Restrictions;
  - Value constraints
  - Cardinality constraints
- Intersections;
- Unions;
- Complements.

Restrictions

Restrictions provide a concise way to describe a class of individuals in terms of the properties we know that describe the individuals themselves (Allemang & Hendler, 2011). An *owl:Restriction* is a special kind of class (i.e., *owl:Restriction* is an *rdfs:subClassOf owl:Class*). Note that this restriction class has no specific name associated with it (Allemang & Hendler, 2011). OWL provides six kind of property restrictions, divided over two constraints, value constraints and cardinality constraints. The three value constraints are described below.

owl:someValuesFrom[14] is used to produce a restriction of the form 'All individuals for which at least one value of the property X comes from class Y'.
owl:allValuesFrom[15] is used to produce a restriction class of the form 'the individuals for which all values of the property X come from class Y'.

owl:hasValue is used to produce a restriction whose description is of the form 'All individuals that have the value Z for the property X'. Formally, the hasValue restriction is just a special case of the someValuesFrom restriction (Allemang & Hendler, 2011).

> Membership in a restriction class must satisfy the conditions specified by the kind of restriction (*owl:someValuesFrom, owl:allValuesFrom* or *owl:hasValue*, as well as the onProperty specification (Allemang & Hendler, 2011).

OWL also includes restrictions that refer to cardinalities – that is, referring to the number of distinct values for a particular property some individual has (Allemang & Hendler, 2011). A cardinality restriction allows much finer-grained way to define classes in compare with the other restrictions defined before. An example of a restriction that defines the class of things that have exactly nine individuals, at least nine individuals and at most nine individuals, respectively:

[a owl:Restriction; *owl:onProperty :hasXXX;* *owl:cardinality 9]*	*[a owl:Restriction;* *owl:onProperty :hasXXX;* *owl:minCardinality 9]*	*[a owl:Restriction;* *owl:onProperty :hasXXX;* *owl:maxCardinality 9]*[16]

[14] Also called existential quantifiers. Existential quantification are words like 'some' or 'one' (*owl:hasValue*).

[15] Also called universal quantifiers.

Intersections

The owl:intersectionOf property links a class to a list of class descriptions. An owl:intersectionOf statement describes a class for which the class extension contains precisely those individuals that are members of the class extension of all class descriptions in the list (Bechhofer, et al., 2004).

Unions

The *owl:unionOf* property links a class to a list of class descriptions. An *owl:unionOf* statement describes an anonymous class for which the class extension contains those individuals that occur in at least one of the class extensions of the class descriptions in the list (Bechhofer, et al., 2004).

Complements

An *owl:complementOf* property links a class to precisely one class description. An *owl:complementOf* statement describes a class for which the class extension contains exactly those individuals that do not belong to the class extension of the class description that is the object of the statement. *owl:complementOf* is analogous to logical negation: the class extension consists of those individuals that are 'not' members of the class extension of the complement class (Bechhofer, et al., 2004).

Intersections, unions, and complements[17] can be used to combine restrictions together or to combine the classes we use to define restrictions (Allemang & Hendler, 2011).

OWL sublanguages

The W3C-endorsed OWL specification includes the definition of three variants of OWL, with different levels of modelling capabilities/ expressiveness. These are OWL Lite, OWL DL[18] and OWL Full (ordered by increasing expressiveness). Each of these sublanguages is a syntactic extension of its simpler predecessor. OWL-DL is so named due to its correspondence with description logic. Description logic is a family of formal knowledge representation languages.

- Every legal OWL Lite ontology is a legal OWL DL ontology;
- Every legal OWL DL ontology is a legal OWL Full ontology;
- Every valid OWL Lite conclusion is a valid OWL DL conclusion;
- Every valid OWL DL conclusion is a valid OWL Full conclusion.
  (Smith, et al., 2004)

[16] Examples written in the Manchester OWL Syntax. This syntax, developed by the CO-ODE project at The University of Manchester, is a syntax designed for writing OWL class expressions.

[17] Also called Boolean Class Constructors.

[18] OWL Description Logics.

2.2.4 SPARQL

A representation of data is useless without some means of accessing that data. The standard way to access RDF data uses a query language called SPARQL[19]. The queried RDF graph can be created from one kind of data or merged from many; in either case, SPARQL is the way to query it (Allemang & Hendler, 2011). SPARQL is a RDF-based query language. RDF-based query languages are based on the notion of RDF triple patterns and their semantics is based on matching triples with RDF graphs. It is harder to provide a semantics for these QLs under OWL-DL semantics because RDF representation mixes the syntax of the language with its assertions. The triple patterns in a query do not necessarily map to well-formed OWL-DL constructs (Sirin & Parsia, 2007). Besides RDF-based query language Semantic Web ontologies can also be classified under another category: DL-based query language. DL-based query languages have well-defined semantics based on the OWL DL model theory (Sirin & Parsia, 2007).

SPARQL-OWL versus SPARQL DL

SPARQL-OWL and SPARQL DL are two different query languages each with its own characteristics (table 1). The semantics of SPARQL-DL is very similar to the semantics of OWL-DL. We specify the conditions under which an interpretation satisfies a query atom in much the same way that satisfaction is defined for OWL-DL axioms (Sirin & Parsia, 2007).

SPARQL-OWL	**SPARQL DL**
Queries are very powerful – variables can occur within complex class expressions and can additionally bind to class or property names apart from individuals and literals	Variables occur in places such that queries are mapped to standard reasoning tasks e.g. subclass retrieval
Does not allow for proper non-distinguished variables	Allows non-distinguished variables

Table 1: SPARQL-OWL versus SPARQL DL (Kollia, et al., 2011)

2.2.5 RIF AND SWRL

The Rule Interchange Format (RIF) was designed as an interchange format for exchanging rules between rule systems, such as those that implement Sematic Web Rule Language (SWRL). SWRL in an initiative of Rule Markup[20] (RuleML). RuleML builds a hierarchy of rule sublanguages upon XML, RDF, and OWL, e.g., SWRL (O'Connor, 2009). The proposal extends the set of OWL axioms to include Horn-like rules. It thus enables Horn-like rules to be combined with an OWL knowledge base (Horrocks, et al., 2004). It is one of the simpler rule languages and most SWRL features are covered by RIF-BLD[21] with the exception of

[19] SPARQL Protocol And RDF Query Language. This query language (version 1.0) was introduced by W3C in 2008.

[20] RuleML Inc. is a non-profit organization that drives the specification of standard semantic-technology & business rules, coordinates rule research and development, and holds international meetings.

[21] RIF Basic Logic Dialect.

"different-from", thus it should be possible to exchange most SWRL rules via RIF (Feigenbaum, et al., 2013).

All rules of SWRL are expressed in terms of OWL concepts (classes, properties, individuals) (O'Connor, 2009).

Atoms in these rules can be of the form C(x), P(x,y), sameAs(x,y) or differentFrom(x,y), where C is an OWL description, P is an OWL property, and x,y are either variables, OWL individuals or OWL data values. It is easy to see that OWL-DL becomes undecidable when extended in this way as rules can be used to simulate role value maps (Horrocks, et al., 2004).

SWRL semantics:

- Based on OWL-DL (illustrated in figure 8);
- Has a formal semantics;
- Complements OWL and fully semantically compatible;
- More expressive yet at expense of decidability;
- Use OWL if extra expressiveness not required (possible exception: querying).
  (O'Connor, 2009)

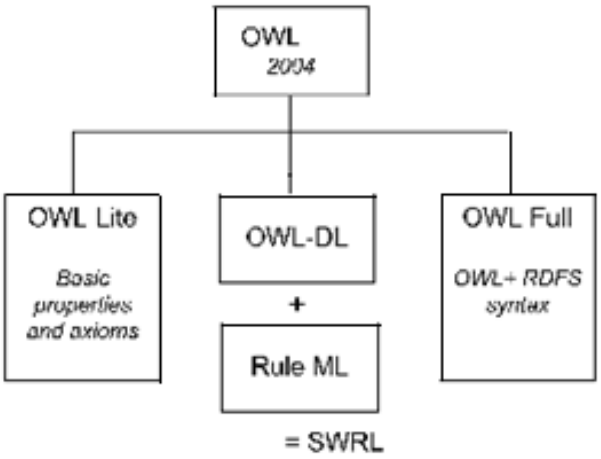


Figure 8: Standardized OWL languages (Fortineau, et al., 2012)

3 Concept library as an ontology

Currently, the worldwide construction industry has several concept libraries, developed and/ or maintained by companies and organisations. Concept libraries like the CB-NL and OTL are modelled in a Semantic Web language, or ontology language. In this chapter the definition of a concept library is defined in terms of the Semantic Web. The next research sub-question is addressed:

- What is a concept library?

3.1 DEFINITION OF THE TERM CONCEPT

A concept library includes concepts. These concepts are the content of the library. Concepts, also known as classes, are used in a broad sense. They can be abstract or concrete, elementary or composite, real or fict(it)ious. In short, a concept can be anything about which something is said, and, therefore, could also be the description of a task, function, action, strategy, reasoning process, etc. (Corcho & Gomez-Perez, 2000). The difficulties with the linguistic reading have led to the crystallization of a second, engineering reading of 'concept', a reading best exemplified in the use of the term 'conceptual model' (Smith, 2004). The term 'concept' itself, on the engineering reading, refers to entities that are created by modellers. Concepts are creatures of the computational realm which exist (in some sense hard to explain) through their representations in software, in UML[22] diagrams, XML representations, in systems of axioms, or what one will (Smith,

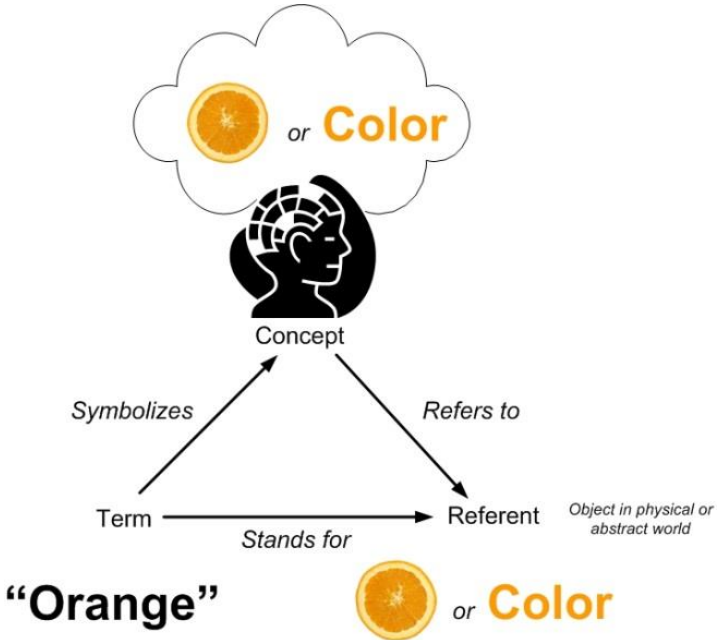


Figure 9: A model of how linguistic symbols stand for objects they represent, which in turn provide an index to concepts in our mind (Ogden & Richards, 1923)

[22] Unified Modelling Language. UML is a modelling language to make object-orientated analysis and designs for information systems.

2004). Figure 9 illustrates the meaning of an object and a concept. The term 'orange' can be the fruit or the colour. In our mind we are making a representation of it. This area of study is known as semantics.

3.2 ONTOLOGY CHARACTERIZATION

The key to utilizing the knowledge of an application domain is identifying the basic vocabulary consisting of terms or concepts of interest to a typical user in the application domain and the interrelationships among the concepts in the ontology (Kashyap, et al., 2008).

> Concepts form the subject-matter of ontology (Smith, 2004).

An ontology may be defined as the specification of a representational vocabulary for a shared domain of discourse which may include definitions of classes, relationships, functions and other objects (Gruber, 1993) (Noy & McGuinness, 2001). However, there exist a variety of standards that are being currently used for enabling high quality search, categorization and interoperation of information across multiple information systems. These standards do not have the formal sophistication and logical underpinnings of an ontology, but are currently in extensive use and provide an extensive collection of concepts and relationships that can serve as a substrate for rich metadata descriptions, and thus can be viewed as 'ontology like' artefacts. Figure 10 presents a characterization of these artefacts from an informatics perspective (Kashyap, et al., 2008).

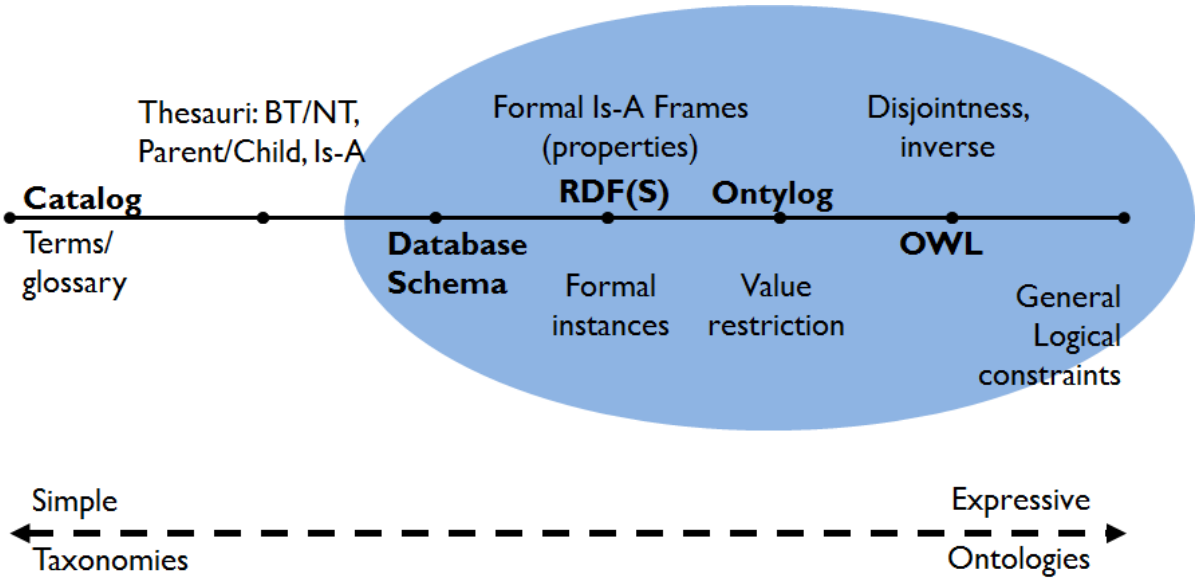


Figure 10: Ontology characterization (Kashyap, et al., 2008)

An informal categorization of ontology-like artefacts illustrated in figure 10 has the following categories, based on Kashyap, et al. (2008):

- Term Lists: These are collections of terms or tokens that play the role of standardized dictionaries in an organisation;

- Thesauri/ Taxonomies: these are collections of concepts in which the concepts are organized according to interrelationships with each other. The relationships represented in taxonomies and thesauri are typically limited to synonyms and hierarchical relationships such as 'broader than/ narrower than', 'parent/ child' and informal 'is-a';

Thesaurus: Terms are ordered, e.g., alphabetically, and concepts may be described by one or more synonymous terms. Taxonomy/ classification: Concepts are arranged using generic (is_a) relationships (Kashyap, et al., 2008).

- Database Schemas: These are collections of tables and columns that are used to implement underlying information models represented in entity-relationship and UML models;
- RDF and XML Schemas: These are collections of concepts that have complex interrelationships and nesting within each other. Concepts and relationships in RDF Schemas are instantiated in RDF based on a graph-based data model with reification, whereas XML Schema documents are instantiated in XML using a tree-based data model;
- OWL Ontologies: Ontologies represented using the Web Ontology Language (OWL) capture specialized constraints and axioms. OWL-DL, a variant of OWL, is equivalent in expressiveness to Descriptions Logics. OWL reasoners can use these constraints and axioms to infer subsumption and equivalence relationships between two concepts and mutual contradictions between concepts if they exist. The interesting property of OWL-DL is that it is a highly expressive and tractable subset of first-order logic for a large number of important use cases;

A semantic reasoner, reasoning engine, rules engine, or simply a reasoner, is a piece of software able to infer logical consequences from a set of asserted facts or axioms. Protégé 4.3.0 uses a semantic reasoner (chapter 6).

- Logical First Order Theories: A first-order logic language, though more expressive than OWL-DL, is for the most part intractable and are not implemented in practical information systems;
- Higher-Order Logics: The goal of an ontology with higher-order logics is to capture common sense knowledge and reason with it to a limited extent. Such ontologies are not considered practical, given the state of the art today.

Why would someone want to develop an ontology? Some of the reasons are:

- 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.
  (Noy & McGuinness, 2001)

In practical terms, developing an ontology includes:

- Defining classes in the ontology;
- Arranging the classes in a taxonomic (subclass–superclass) hierarchy;
- Defining slots and describing allowed values for these slots;
- Filling in the values for slots for instances.

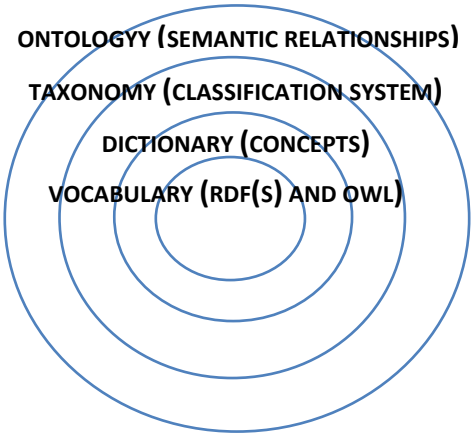


Figure 11: From vocabulary to ontology (based on Baggen, et al., 2013)

3.3 STANDARDS FOR CONCEPT LIBRARIES

In recent years some standards are set up to develop a concept library. ISO[23] 16354 – 'Guidelines for knowledge libraries and object libraries' is based on NTA 8611:2008[24]. On both a national and international level knowledge libraries exist or are being developed (ISO/IEC 16354, 2013). Furthermore, ISO 19150-2 – 'Geographic information – ontology – part 2: Rules for developing ontologies in the Web Ontology Language (OWL)' is under construction.

> Knowledge libraries are databases that contain modelled knowledge about kinds of things (ISO/IEC 16354, 2013).

Knowledge libraries are intended to support business processes concerning any kind of products during their lifetime, for example to support their design, procurement, construction, operation or maintenance. There is an increasing awareness of the high potential value of knowledge libraries and of the drawbacks of the inconsistencies and lack of interoperability between different knowledge libraries. The objective of ISO 16354 is to categorize knowledge libraries and object libraries and to provide recommendations for the creation of such libraries. Libraries that are compliant with the guidelines of this standard may be more easily linked to, or integrated with other libraries (ISO/IEC 16354, 2013).

Mentioned above, not only the term 'concept library' is used. The title of ISO 16354 refers to the term 'object library'. The difference between 'object' and 'concept' are explained in section 3.1. In addition, table 2 illustrates the definitions of these different libraries.

Concept library	Object library	Product library
Generic descriptions	Specific project-related descriptions	Specific supplier-related descriptions of standard products
Dictionary	'BIM'	Catalogue

Table 2: Definitions of different kind of libraries (based on Zwakhals, 2014)

[23] International Organisation for Standardization.

[24] Dutch Technical Agreement (NTA 86611:2008 EN) is a standard to set guidelines for object libraries in the Netherlands.

4 Systems Engineering

This chapter starts with a short elaboration on the definition of SE. The intention of this research is not to come up with a better or more accurate description of the SE methods. Therefore the elaboration on SE is kept short meaning to provide a basic understanding of its intended goals, the generally used SE process and its related activities. To obtain more insight into the goals and activities relations can be made with the aim of concept libraries modelled according to Semantic Web standards. In this chapter the next research sub-questions are addressed:

- What is Systems Engineering?
- What are the characteristics of a development stage in the Systems Engineering process?

4.1 SYSTEMS ENGINEERING IN GENERAL

According to INCOSE[25] (consistent to ISO 15288:2002[26]), Systems Engineering is an interdisciplinary approach and means to enable the realization of successful systems. It focuses on defining customer needs and required functionality early in the development cycle, documenting requirements, and then proceeding with design synthesis and system validation while considering the complete problem. Systems Engineering considers both the business and the technical needs of all customers with the goal of providing a quality product that meets the user needs (Haskins, 2006).

A system is an integrated composition of persons, products and processes which achieve a specific goal by features need (Department of Defense, 2001). A system passes a life cycle (figure 13) and is based on 'system thinking'. According to SE, system thinking offers a structure of systems wherein a project can be developed imitable and demonstrably, realized and maintained (Werkgroep Leidraad SE, 2009). Be aware, within SE a system is always a part of bigger picture, called 'system of systems'. Figure 12 illustrates a system of systems. The system-of-interest is a part of the scope of this total system. It is dependent of the project level.

[25] International Council on Systems Engineering. INCOSE is a not-for-profit membership organization founded in 1990. Their mission is to share, promote and advance the best of systems engineering from across the globe for the benefit of humanity and the planet.

[26] ISO 15288:2002(E) – Systems Engineering – system life cycle processes. An international standard that is a generic process description.

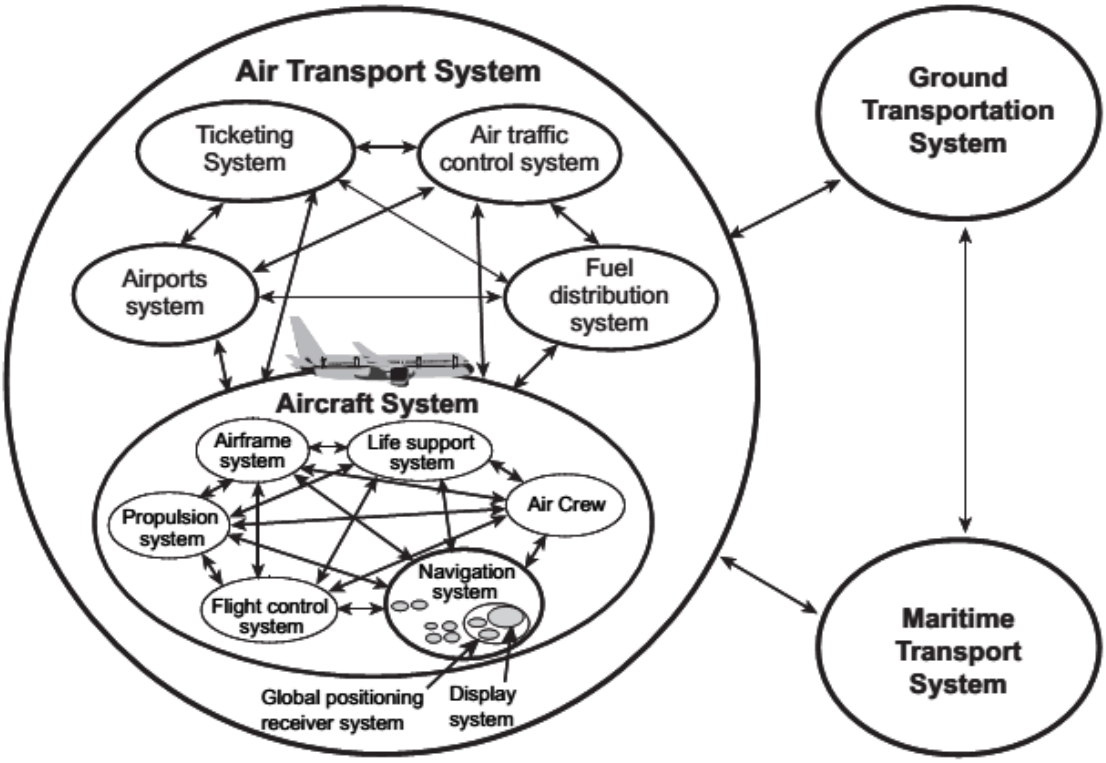


Figure 12: Example of the multitude of perceivable systems-of-interest in an aircraft and its environment of operation within a Transport System-of-Systems (Haskins, 2006)

The aim of SE is to reduce these failure costs by creating an appreciate system on an effective and transparent manner (Werkgroep Leidraad SE, 2009). This aim is intended for the Dutch construction industry.

- Effectiveness: meet the needs of the customer, within the social responsible costs;
- Efficiency: reduce failure costs on an efficient manner and a better use of available recourses;
- Transparency: deliver customer wishes on a demonstrable and controlled way. (Werkgroep Leidraad SE, 2009)

Life cycle stages

A system-of-interest passes a life cycle. A life cycle model for a system identifies the major stages that the system goes through, from its inception to its retirement. The life cycle stages of the SE approach are illustrated in table 3. Six stages are defined, based on INCOSE and ISO 15288. Noted that stages can overlap and the production, utilization and support stage run in parallel, illustrated in figure 13.

> The purpose in defining the system life cycle is to establish a framework for meeting the stakeholders' needs in an orderly and efficient manner. This is usually done by defining life cycle stages, and using decision gates to determine readiness to move from one stage to the next (Haskins, 2006).

Purpose concept stage: The concept stage is executed to assess new business opportunities and to develop preliminary system requirements and a feasible design solution (Haskins, 2006).

Purpose development stage: The development stage is executed to develop a system-of-interest that meets acquirer requirements and can be produced, tested (verified), evaluated, operated, supported, and retired (Haskins, 2006).

Verification and validation are independent procedures that are used together for checking that a product, service, or system meets requirements and specifications and that it fulfils its intended purpose.

> Verification: The evaluation of whether or not a product, service, or system complies with a regulation, requirement, specification, or imposed condition. It is often an internal process. Contrast with validation (IEEE, 2011).

> Validation: The assurance that a product, service, or system meets the needs of the customer and other identified stakeholders. It often involves acceptance and suitability with external customers. Contrast with verification (IEEE, 2011).

Purpose production stage: The production stage is executed to produce or manufacture the product, to test (verify) the product, and to produce related supporting and enabling systems as needed (Haskins, 2006).

Purpose utilization stage: The utilization stage is executed to operate the product, to deliver services within intended environments and to ensure continued operational effectiveness (Haskins, 2006).

Purpose support stage: The support stage is executed to provide logistics, maintenance, and support services that enable continued system-of-interest operation and a sustainable service (Haskins, 2006).

Purpose retirement stage: The retirement stage is executed to provide for the removal of a system-of-interest and related operational and support services, and to operate and support the retirement system itself (Haskins, 2006).

LIFE CYCLE STAGES	PURPOSE	DECISION GATES
CONCEPT	*Identify stakeholders' needs* *Explore concepts* *Propose viable solutions*	*Decision Options* *– Execute next stage* *– Continue this stage* *– Go to a preceding stage* *– Hold project activity* *– Terminate project*
DEVELOPMENT	*Refine system requirements* *Create solution description* *Build system* *Verify and validate system*	
PRODUCTION	Produce systems Inspect and test [verify]	
UTILIZATION	*Operate system to satisfy users' needs*	
SUPPORT	*Provide sustained system capability*	
RETIREMENT	*Store, archive, or dispose of the system*	

Table 3: Life cycle stages, their proposes and decision gates option (Haskins, 2006)

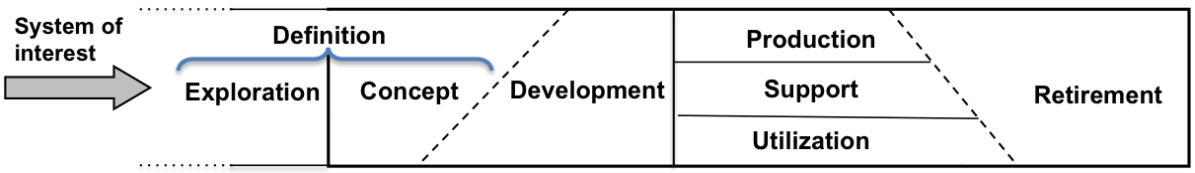


Figure 13: A generic life cycle model (Lawson, 2010)

In the construction industry life cycle stages are defined differently by parties. This is the result of making their own description and interpretation of the SE approach, applicable in their own organisation. Besides different usable names in the life cycle stages, the life cycle models of SE are also different to each other. Figure 14 is an example of a primary model.

> There is no single 'one-size-fits-all' system life cycle model that can provide specific guidance for all project situations.

Readers are referred to Haskins (2006) and NASA (2012) for more detailed information about the SE life cycle stages.

4.2 DEVELOPMENT STAGE OF THE SYSTEMS ENGINEERING LIFECYCLE

According to Haskins (2006) the development stage is divided in four purposes: refine system requirements, create solution description, build system and verify and validate system. This stage is a part/ section of the SE approach. The Vee model (figure 14) is used to visualize the system engineering focus, particularly during the concept and development stages. The Vee highlights the need to define verification plans during requirements development, the need for continuous validation with the stakeholders, and the importance of continuous risk and opportunity assessment (Haskins, 2006).

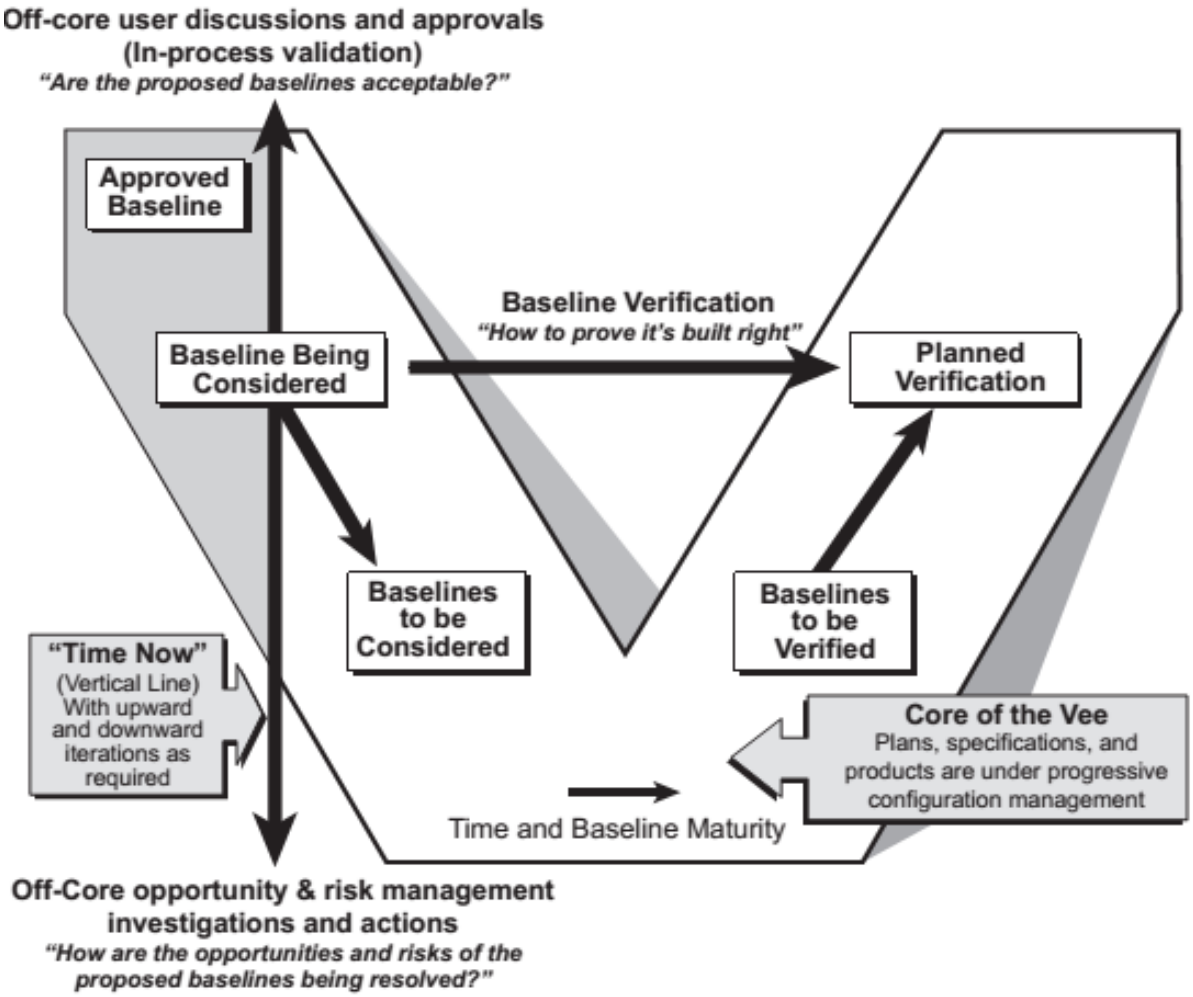


Figure 14: Vee model (Haskins, 2006)

The development stage begins with sufficiently detailed technical refinement of the system requirements and the design solution and transforms these into one or more feasible products that enable a service during the utilization stage. The system-of-interest may be a prototype in this stage (ISO/IEC 15288, 2002). System requirements are the translation of the client's needs which have to comply with the concept of SMART[27]. The SMART principle is a management pedagogues jargon for simple and unambiguous drafting and checking of objectives (Roozenburg & Eekels, 1998).

> The development stage is executed to develop a system-of-interest that meets acquirer requirements and can be produced, tested, evaluated, operated, supported and retired (ISO/IEC 15288, 2002).

System development is a top-down/ decomposition process. The choice of decomposition is based on the risks of the project. If the total system and its relations with the environment are too complex (or too abstract) to manage decomposition is used to engineering on a more detailed level (or sub-system). The system will be specified till the level where the risks

[27] Specific, Measurable, Assignable, Realistic and Time-related.

are acceptable (van Netten, 2005). The decomposition process translates system requirements into detailed functional design criteria. The functional design criteria include:

- The functions which have to fulfil by the object. The functions of a higher level in the decomposition structure can be allocated to the regarding object;
- The performances of the identified functions, by answering questions like: who, what, where, when, how many/ much, which, etcetera.
  (van Netten, 2005)

> Breaking a large complex problem (project) into smaller pieces (decomposition) makes them easier to solve since the problems are smaller and specialists can focus on their functional areas (Graf, 2008).

The result of the process is a defined functional architecture with allocated system requirements that are traceable to each system function. Traceable means that each requirement converts to the upper or lower requirements in the decomposition structure. The traceability is based on the verification process (van Netten, 2005). The Vee model is useful in defining the start, stop and activities appropriate to life cycle stages (Haskins, 2006). Besides the top – down structure the development stage contains also an iteratively process. The decomposition process can be applied concurrently, iteratively and recursively to a system throughout the entire life cycle (ISO/IEC 26702, 2007). SE assumes an iterative specification process whereby functions, requirements and solutions be developed in conjunction (Werkgroep Leidraad SE, 2013).

> The complexity and dynamics of the client requirements ask for a iterative method (Werkgroep Leidraad SE, 2013).

Requirements, functions and solutions are illustrated in decomposition structures, also described as breakdown structures.

- The Requirements Breakdown Structure (RBS) is a hierarchy of requirements of a system (Werkgroep Leidraad SE, 2009);
- The Functional Breakdown Structure (FBS) is a structured, modular breakdown of every function that must be addressed to perform a generic mission, or any subset of that mission (De Hoff, et al., 2009);
- The System Breakdown Structure (SBS) is a hierarchy of elements, related life cycle processes, and personnel used to assign development teams, conduct technical reviews, and to partition out the assigned work and associated resource allocations to each of the tasks necessary to accomplish the objectives of the project (IEEE, 2005).

Related to the system requirements (RBS) of the upper level of detail a design of the system is made. These choices of the design have some functionalities (FBS) which have to be fulfilled by the system (SBS). Between these breakdown structures an iterative development occurs, illustrated by the requirements loop and design loop in figure 15. This development approach is used when the requirements are unclear from the beginning or the customer wishes to hold the system-of-interest open to the possibilities of inserting new technology (Haskins, 2006). The iterative process repeats specification on more detail levels when

complex systems occur (Werkgroep Leidraad SE, 2013). During the iteratively process there is continuous verification of the design if it meets the system requirements. Validation will be done to check if the system meets the client requirements (output concept stage). Figure 15 illustrates the iterative and verification process in the development stage.

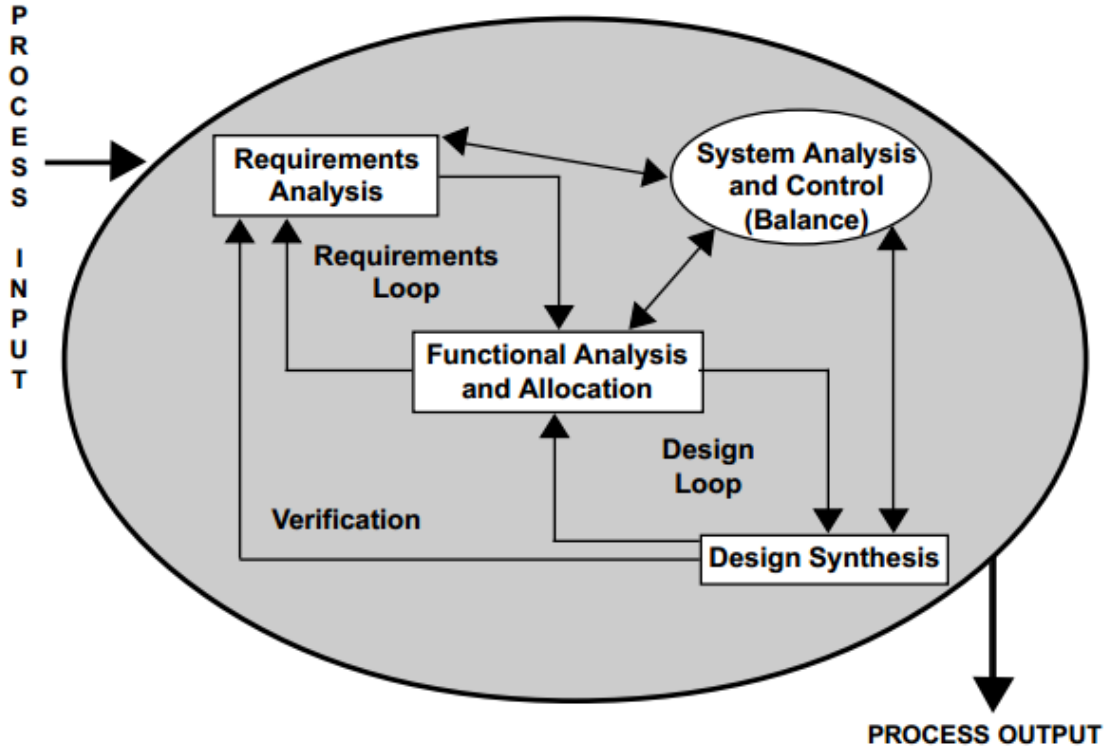


Figure 15: Development stage of the SE process. Including the iterative process and verification (Department of Defense, 2001)

5 Conclusion on the theory

This chapter provides an overall conclusion on the literature treated in the three previous chapters. The research sub-questions of section 1.5 are answered more explicitly and potential bridges between the three literature chapters are made.

Currently, the worldwide construction industry has several concept libraries. In this case a concept library can be defined as an ontology (chapter 3), which provides concepts and semantic relations between these concepts. In computer science, the term semantics refers to the meaning of languages (section 2.2). The term is applied to concept libraries specifically designed and used for representing information content. These concept libraries are developed and/ or maintained by companies and organisations. They are individual (e.g. OTL) and national (e.g. CB-NL) initiatives. There will always be multiple ontologies, as there will always be multiple web pages on any given topic (Allemang & Hendler, 2011).

Common structure and vocabulary is required to provide semantic relations between concepts within and between concept libraries. Open and worldwide standards according to the Semantic Web (W3C initiative) support these communications. One of the great strengths of the Semantic Web is that information that has been specified by one person in one context can be reused either by that person or by others in different contexts (Allemang & Hendler, 2011). RDF (subsection 2.2.1) provides a common structure of data and RDFS (subsection 2.2.2) and OWL (subsection 2.2.3) provide a common vocabulary of data. OWL builds upon RDFS by adding more advanced constructs to describe semantics of RDF statements. When restrictions (*owl:Restrictions*) are used in combination with the constructs of RDFS (especially *rdfs:subPropertyOf* and *rdfs:subClassOf*), and when they are cascaded with one another (restrictions referring to other restrictions), they can be used to model complex relationships between OWL concepts (properties, classes and individuals). The advantage of modelling relationships in this way (over informal specification) is that interactions of multiple specifications can be understood and even processed automatically (Allemang & Hendler, 2011). This results in that concept libraries which are based on Semantic Web standards, can be made richer in semantics in addition to current taxonomic structures, which are included in ISO 16354 and NTA 8611 (section 3.3).

As already described, in the construction industry a concept library refers to an ontology. According to Kashyap, et al. (2008), an ontology has several categories. A common ontology category is a thesauri/ taxonomy. These are collections of concepts in which the concepts are organized according to interrelationships with each other. The relationships represented in taxonomies and thesauri are typically limited to synonyms and hierarchical relationships such as ‘broader than/ narrower than’, ‘parent/ child’ and informal ‘is-a’ (Kashyap, et al.,

2008). RDFS is an example hereof, in which a hierarchy of classes and properties is created. More expressive ontology categories are provided, and are illustrated in figure 10 (section 3.2). OWL is able to express a more powerful ontology, based on description logic (sublanguage OWL-DL). According to Knublauch (2004), OWL can be divided in two different cases for usage:

- OWL used as data exchange language (defines interfaces of services and agents);
- OWL used for terminologies or knowledge models (OWL-DL).

Intuitively, a good ontology is one which corresponds to reality as it exists beyond our concepts (Smith, 2004).

Besides OWL, two other components of the Semantic Web occur on the same layer of the Semantic Web layer cake in figure 16. This layer is described as the middle (blue) layer. It contains technologies standardized by W3C to enable building Semantic Web applications. SPARQL (subsection 2.2.4) and SWRL (subsection 2.2.5) are also part of this middle layer. SPARQL is required for accessing the data. SPARQL is a RDF-based query language. The triple patterns in a query do not necessarily map to well-formed OWL-DL constructs (Sirin & Parsia, 2007). However, SPARQL-DL (a query category) makes it possible to express languages like OWL-DL. DL-based query languages have well-defined semantics based on the DL model theory of OWL (Sirin & Parsia, 2007). SWRL is based on a combination of the OWL-DL and OWL Lite sublanguages (Horrocks, et al., 2004). All rules are expressed in terms of OWL concepts (O'Connor, 2009). The basic assumption is that we define links, extensions of OWL interpretations that also map variables to elements of the domain (Horrocks, et al., 2004). In other words, SWRL is more valuable to express data.

- SPARQL is a RDF-based query language;
- SWRL is an OWL-DL-based rule language.

The development stage of the SE process is a design phase. On the basis of stakeholders' requirements a system-of-interest is built. The system-of-interest is decomposed (FBS and SBS) to a more detailed level (or sub-system) to reduce the complexity and improve the risk management of the project. This is illustrated in figure 17. In the context of SE, complexity incorporates not only engineering systems, but also the logical human organization of data, which we also should understand as structured data.

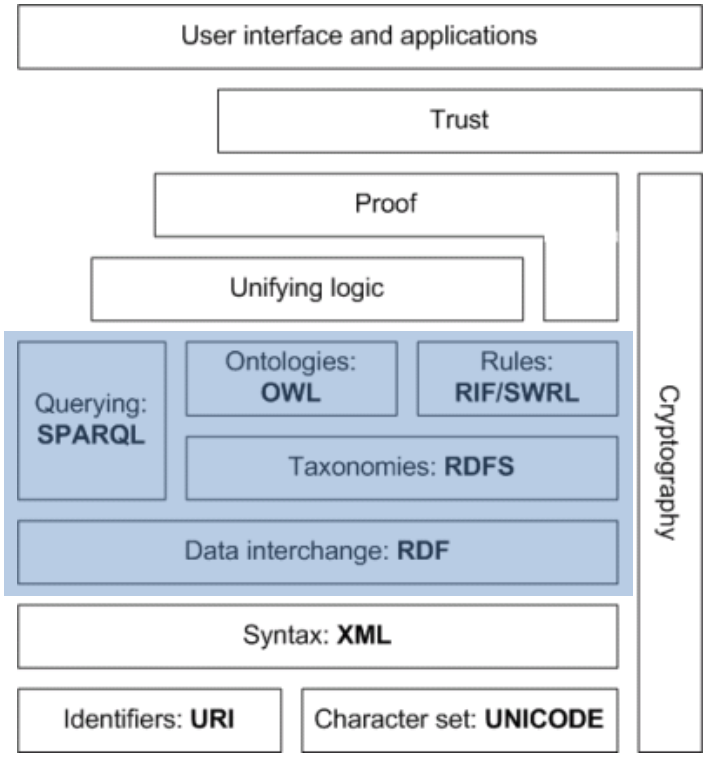


Figure 16: Semantic Web Layer Cake (Berners-Lee, 2006)

Designing a system-of-interest based on the stakeholders' requirements requires defined concepts. Defined concepts in the development stage are required because:

- An effort can to classify an example with a primitive concept;
- Whether the expected result is not reached should be explained/ assessed.

In addition to the literature in part B, some interviews were conducted. Five of the seven interviewees[28] (appendix C) expressed a significant opinion, namely that designing a SBS seems difficult. These five experts argue that making a decomposition structure is difficult because of the conflicts come up between actors concerning concepts, which results in a time-consuming process. Furthermore, as described in chapter 4, verification and validation are important aspects of the development stage. Concepts of the FBS and SBS are verified to the systems requirements and validated to the stakeholder requirements of a project.

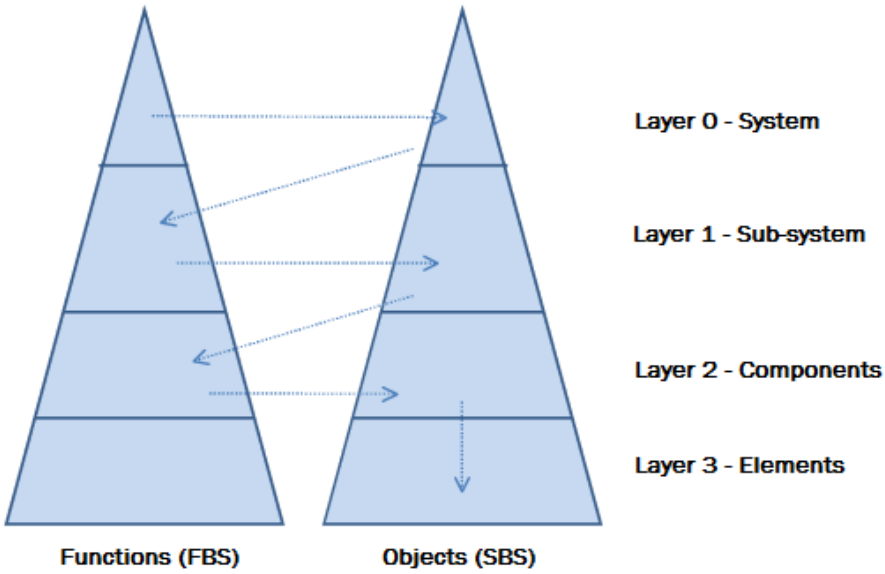


Figure 17: Decomposition process within the development stage (Coinsweb, 2008)

[28] The interviewees are selected on: (1) having expertise/ knowledge about the approach of SE in the construction industry and/ or (2) expertise/ knowledge about concept libraries applied in the construction industry (e.g. CB-NL, OTL).

PART C: Case study

6 Practical framework

As described in section 1.5, the main research question is: 'What is the added value of an OWL concept library in relation to the development stage of the SE process?' The previous chapters explore and define information that is relevant to answer this main research question. This chapter covers a baseline in this report. It describes the essential traits of chapter 5, so that the reader gets acquainted to the more specific research sub-questions and scope, which are addressed in sections 6.2 and 6.4.

6.1 FOCUS AREA CASE STUDY

The essential traits which are discussed in the case study are:

- **Decomposition of a system-of-interest**
  The choice for decomposition is based on the high importance and attention by the interviewees. The term 'mereology' is used instead of the term 'decomposition structure' in the further chapters and sections of this research. Mereology is the theory of part-hood relations: of the relations of part to whole and the relations of part to part within a whole (Mesarović, et al., 1970);
- **Verification of an object (illustrated in figure 15 – section 4.2)**
  Firstly, according to Knublauch (2004), one of the two cases in which OWL is used is to refer to terminologies or knowledge models (in terms of this report known as concept libraries). Secondly, according to Allemang & Hendler (2011), the advantage of modelling relationships in OWL (over informal specification) is that interactions of multiple specifications can be understood and even processed automatically. These assertions are the basis for this focus area. The modelling capabilities of OWL is tested to find whether it is able to verify objects (corresponds to system requirements) automatically within a concept library. In addition to test the modelling capabilities of OWL, Semantic Web standards SPARQL and SWRL are included in the case study. Any advantages and disadvantages of these standards are explored and compared to OWL.

6.2 RESEARCH QUESTIONS CASE STUDY

Based on the previous subsection, the next research sub-questions for the case study are addressed:

Demonstration I: Decomposition of a system-of-interest

- Is OWL able to model a mereology in an ontology?

Demonstration II: Verification of an object within the system-of-interest

- Is OWL-DL able to verify an object automatically?
- Does SPARQL-DL provide any added value on OWL-DL according to verify an object?
- Does SWRL provide any added value on OWL-DL according to verify an object?

6.3 SCHEME CASE STUDY

The theoretical scheme of this case study is illustrated in figure 18. This figure illustrates the development stage of the SE process including the two focus areas (demonstration I and II) defined in section 6.1.

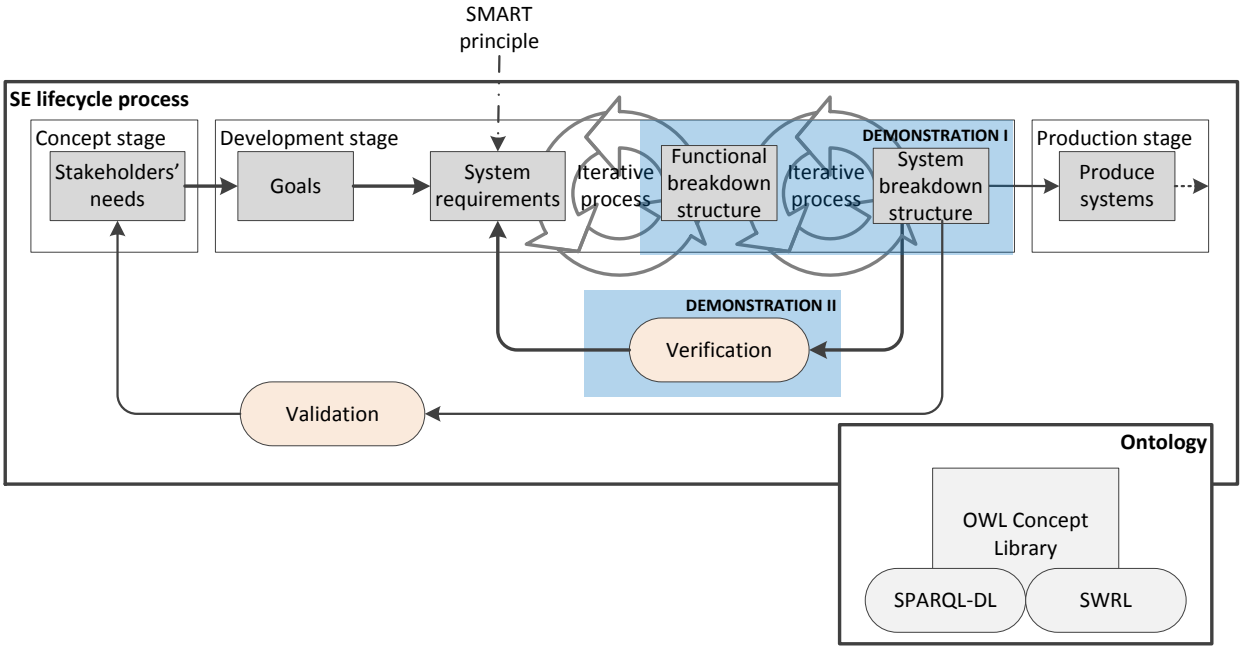


Figure 18: Theoretical scheme case study (based on Haskins, 2006 & Department of Defence, 2001)

6.4 SCOPE CASE STUDY

In addition to the sub-questions defined in section 6.2, these technologies and tools/ methods below, have applied in the case study.

Protégé

A clear focus of current ontology management tools is to support the development of ontologies with a wide range of editing features (Waterfeld, et al., 2008). Protégé is created by the Stanford Medical Informatics group at Stanford University. Protégé is a Java-based open source standalone application and is to be installed and run a local computer. It enables users to load and save OWL and RDF ontologies, edit and visualize classes, properties and SWRL rules (Horrocks, et al., 2004), define logical class characteristics as OWL expressions and edit OWL individuals (Waterfeld, et al., 2008). Protégé version 4.3.0 (latest version) is used.

OWL-DL

The OWL language provides three increasingly expressive sublanguages designed to use by specific communities of implementers and users (Smith, et al., 2004). Any RDF graph is typically OWL Full, unless it has been restricted (via its RDF Schema) to the other dialects (Kashyap, et al., 2008). Described in chapter 5, OWL exists of two different user groups; OWL used as data exchange language and OWL used for terminologies or knowledge models. OWL-DL is the subset of OWL (full) that is optimized for reasoning and knowledge modelling. The choice of using OWL-DL is based on the fact that the SE related research question is a knowledge modelling objective. The interesting property of OWL-DL is that it is a highly expressive and tractable subset of first-order logic for a large number of important use cases (Kashyap, et al., 2008). Description Logics are a decidable fragment of First Order Logic and are therefore capable to automated reasoning. It is therefore possible to automatically compute the classification hierarchy and check for inconsistencies in an ontology that conforms to OWL-DL (Horridge, et al., 2004). In terms of clarity the term 'OWL' is used instead of 'OWL-DL' in the next chapters.

SPARQL-DL

The semantics of SPARQL-DL are very similar to the semantics of OWL-DL. They specify the conditions under which an interpretation satisfies a query atom in much the same way in which satisfaction is defined for OWL-DL axioms (Sirin & Parsia, 2007). Protégé has a built-in function for SPARQL-DL, called DL Query.

Pellet reasoner

The DIG[29] interface is a standard interface/protocol that was introduced to provide a common interface to DL reasoners. In this way, applications, such as Protégé, can access several reasoners by using a common interface (Protege, 2008). Pellet is a complete OWL-DL consistency checker and a very incomplete OWL-Full consistency checker. It is also an OWL syntax checker.

> An OWL consistency checker takes a document as input, and returns one word being consistent, inconsistent, or unknown (Sirin & Parsia, 2007).

Turtle

A more compact serialization of RDF, called Turtle (figure 6), is used for the syntax examples in this report. Turtle combines the apparent display of triples from N-Triples with the terseness of qnames. N-Triples is a triple-based syntax format. Qualified names (qnames) were introduced by 'XML Namespaces'. They were defined for element and attribute names and provided a mechanism for concisely identifying a URI/ localname pair.

```
:Tunnel        rdf:type owl:Class ;
               rdfs:subClassOf :Engineering_Structure .
```

Listing 1: Example of the Turtle syntax expressing the fact that a class 'tunnel' from the local default namespace is a specialization of the class 'Engineering structure'.

[29] Domain Information Groper.

7 Case study

This chapter covers the case study. Firstly, a project is described as input for conducting the case study. A small example is given of a project-related taxonomy in OWL, modelled in Protégé. Secondly, the two research parts are divided in two separate subsections. The first subsection elaborates on the conducted case study 'decomposition of a system-of-interest'. The second subsection focuses on the 'verification of a concept'. These two subsections include several 'mock-up' examples (in Turtle syntax) as support on the textual information. Figures to illustrate the use cases conducted in Protégé are attached in appendix D.

7.1 DESCRIPTION OF THE PROJECT

According to the research methodology, one project is highlighted as depth study for conducting the case study. The infrastructural project 'Heiloo A9' is selected as case study in this report. The decision is made in consultation with some experts of the company and based on an already completed development stage of the SE process of the project. Project Heiloo A9 includes the next major objective: Improvement of accessibility, traffic safety and liveability.

Figure 19 illustrates the project-related ontology, set up in Protégé. The built-in taxonomy illustrates some project-related concepts. It is based on the classification structure of 'Semantic Concepts Library'. Semantic Concepts Library is a subset of BuildingSMART Data Dictionary (bSDD)[30], but its content is more strictly organized. Semantic Concepts provides input for the CB-NL and is development model for both ISO 12006-3 – 'Building construction: Organization of information about construction work'; and NTA 8611. Concepts from the FBS and SBS of the Heiloo A9 project are intertwined in the taxonomy. The

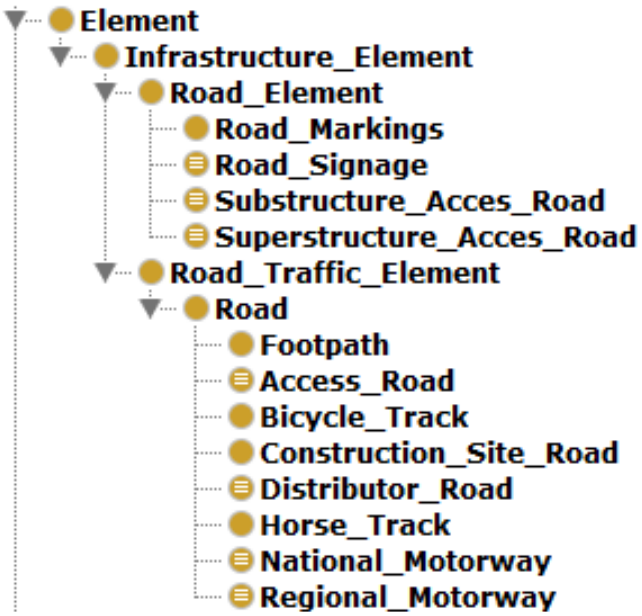


Figure 19: Part of the taxonomy of the project (Protégé 4.3.0)

[30] Open standard of the buildingSMART Alliance.

concept 'Road' has a central role. Road concepts which occur in the case study are based on 'Handboek wegontwerp 2013', a publication of the CROW[31].

7.2 DEMONSTRATION I: DECOMPOSITION OF A SYSTEM-OF-INTEREST

In this subsection the next research sub-question is addressed:

- Is OWL able to model a mereology in an ontology?

Described in subsection 2.2.2 and appendix A, RDFS provides the property *rdfs:subClassOf*. This property is used in ontologies to define subclasses (and their inverse 'super classes'). According to Kashyap et al. (2008) a taxonomy includes concepts which are arranged using generic 'is a' relationships. OWL provides a built-in initiative for this. Protégé characterizes this built-in function by illustrating 'Sub Class Of' in the description window of each class and property. This corresponds with the syntax below.

```
:Road   rdf:type owl:Class ;
        rdfs:subClassOf :Road_Traffic_Element .
```

Listing 2: Taxonomic structure of class 'Road'.

A mereology has a different hierarchy structure than a taxonomy. A mereology includes another relationship. Where a taxonomy using generic 'is a' relationships a mereology uses a 'part – whole' relationship. OWL does not provide a built-in initiative of a 'part – whole' relationship (as it does for the 'is a' relationship). This requires sufficient expressive power to capture this hierarchical structure. The next 'part – whole' structures are based on the W3C document of Rector & Welty (2005):

- **Transitive relations**
  An important requirement for the basic relation 'object A – part of – object B' (RDF triple) that it is transitive, i.e. if A is part of B, and B is part of C, then A is part of C. Axiom *owl:transitiveProperty* provides this relation. In other words, C is a 'part' of the 'whole' A.
- **Parts an direct parts**
  Because a mereology could be large, a division is made of the 'part of' relation. 'Direct part of' is used as sub property of 'part of'. In this way A is even like B a part of C. But B distinguish itself from A because of its direct relation with C, direct part of. Property 'part of' keeps the axiom *owl:transitiveProperty* and 'direct part of' not. Otherwise A would be a direct part of C. Sub property 'component of' can also be used instead of 'direct part of'.
- **Inverse relations**
  'part-of' is an *owl:inverse* of 'has-part'. This is the same for 'direct part of' and 'has direct part'.
- **Restrictions**
  Express the 'part – whole' relations amongst individuals using *owl:hasValue* with 'direct part of' and express the 'part – whole' relations amongst classes using *owl:someValuesFrom* with 'direct part of'.

[31] Centrum voor Regelgeving en Onderzoek in de Grond-, Water- en Wegenbouw en de Verkeerstechniek.

OWL-DL does not allow datatype properties to be transitive or have inverse properties.

```
:Traffic_Road_System    rdf:type owl:Class ;
                        rdfs:subClassOf :System ,
                                        [ rdf:type owl:Restriction ;
                                        owl:onProperty :part of ;
                                        owl:someValuesFrom :Access_System
                                        ] .
```

Listing 3: Example of a restriction on class 'Traffic road system'.

However, the mereology of the development stage is different to the theory of Rector & Welty (2005). According to the literature of SE, two mereologies have to be defined, namely the FBS and SBS. Figure 15 (chapter 4) illustrated the design loop between these two mereologies. These are not built separately, but in relationship with each other. This reason aims to another approach than the 'part – whole' relationship. According to Van Netten (2005) the system requirements are translated into functional design criteria. Next step is to relate the function to a function fulfiller, known as 'object'. From this point the decomposition of both mereologies has the structure such as illustrated in figure 17 (chapter 5). Instead of *owl:ObjectProperty* 'part of' *owl:ObjectProperty* 'function of' is defined in the ontology. A part of this decomposition approach is attached in appendix D.

'function of' includes the following syntax:

```
:functionOf     rdf:type owl:ObjectProperty ,
                         owl:TransitiveProperty ;

                owl:inverseOf :hasFunction ;
                owl:equivalentProperty :hasObject ;
                rdfs:range owl:Thing ;
                rdfs:domain owl:Thing .
```

Listing 4: Modelling capabilities of OWL-DL for a mereology.

Each function has a function fulfiller (an object). This object occurs on the same level.

```
:Accessing      rdf:type owl:Class ;
                owl:equivalentClass [   rdf:type owl:Restriction ;
                                        owl:onProperty :functionOf ;
                                        owl:someValuesFrom :Access_System
                                        ] ;
                Rdfs:subClassOf : Function .
```

Listing 5: Example of an equivalent class; Accessing – Access system.

Each object has a function which occurs on the direct lower level. An example is given of function 'accessing road traffic' and object 'traffic road system' below'. This example is part of a bigger part of the FBS and SBS modelled in OWL, attached in appendix D.

```
:Accessing_Road_Traffic  rdf:type owl:Class ;
                         owl:equivalentClass [   rdf:type owl:Restriction ;
                                                 owl:onProperty :functionOf;
                                                 owl:someValuesFrom :Traffic_Road_System
                                                 ] ;
                         rdfs:subClassOf :Accessing ,
                                       [         rdf:type owl:Restricton ;
                                                 Owl:onProperty :functionOf ;
                                                 owl:someValuesFrom :Access_System
                                                 ] .
```

Listing 6: Example of a mereology based on the project.

7.3 DEMONSTRATION II: VERIFICATION OF AN OBJECT

The next research sub-questions are addressed in this subsection:

- Is OWL-DL able to verify an object automatically?
- Does SPARQL-DL provide any added value according to verify an object?
- Does SWRL provide any added value according to verify an object?

Firstly, a list of different types of project-related system requirements are set up. System requirements tend to be more situation-based compared to the more general explicit requirements (client needs – concept stage of SE). The type of specifications for a system are based on CROW (2011) and defined in table 4. Secondly, in subsection 7.3.1 the modelling capabilities of OWL-DL is tested to verify an object to these several requirements. Furthermore, the modelling capabilities of SPARQL-DL and SWRL is also tested in this case in, respectively subsection 7.3.2 and subsection 7.3.3.

Types of specifications	Definition	Example (project related)
Specification	A specification may refer to an explicit set of requirements to be satisfied by a material, design, product or service (ASTM International, 2014).	
▪ Requirements	In product development and process optimization, a requirement is a singular documented physical and functional need that a particular design, product or process must be able to perform (ASTM International, 2014).	
– Aspect requirements	Aspect requirements defines specific requirements of the system, which has an indirect contribution to the primary function (CROW, 2011).	Engineering structures (e.g. viaducts) have to be safe for traffic and environment during the exploitation phase.
– Extern interface requirements	External Interface requirements state the required characteristics at a point or region of connection of the system to the outside world (Halligan, 2013).	Integration of a noise barrier in the surrounding area of the project.
– Functional requirements	A functional requirement defines a function of a system or its component (CROW, 2011).	Function of a road is accommodating road traffic.
– Intern interface requirements	Intern interface requirements are design requirements (Halligan, 2013).	The primary watercourse has to have a minimal width of 7,30 meter.
▪ Precondition	A precondition is a condition that must always be true just prior to the execution of a project (CROW, 2011).	Traffic roundabout has to be designed according to the CROW directive 'eenheid in rotondes', edition 126.

Table 4: Types of specifications (based on CROW, 2011)

7.3.1 OWL-DL

According to Allemang & Hendler (2011), the advantage of modelling relationships in OWL is that interactions of multiple specifications can be understood and even processed automatically. In the context of verification, a demonstration is given how an object corresponds to several specifications, which are listed in table 4.

Requirements can be characterized by *owl:Restriction*. A restriction has to be on values of properties, which are always an *owl:ObjectProperty* or *owl:DatatypeProperty*. This is defined by *owl:onProperty*. This means that a certain property is specified, which is used by a restriction. A restriction can be a part of a class or an individual. Figure 20 illustrates a model how an OWL restriction (specified on project-related concepts) is applied.

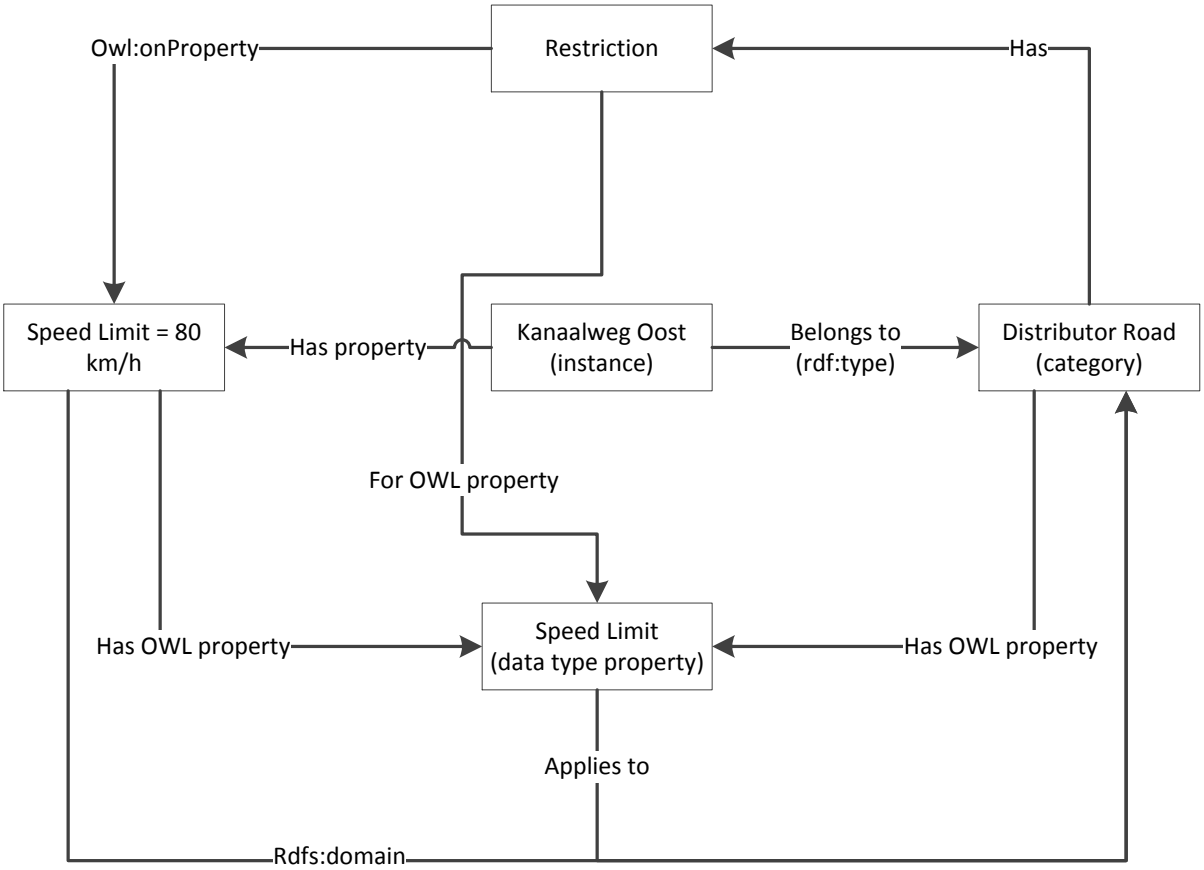


Figure 20: Model of an *owl:Restriction*

The example in figure 20 illustrates a precondition. Preconditions which are related to datatype properties are easy to model in an ontology. Making the datatype property 'speed limit' functional it will enhance the consistency check. A functional property is a property that can have only one (unique) value y for each instance x, i.e. there cannot be two distinct values y1 and y2 such that the pairs (x,y1) and (x,y2) are both instances of this property. Both object properties and datatype properties can be declared as 'functional' (Bechhofer, et al., 2004).

```
:hasSpeedLimit rdf:type owl:DatatypeProperty ,
                        owl:FunctionProperty ;

               rdfs:domain :Road ;

               rdfs:range xsd:integer .
```

Listing 7: Datatype property 'has speed limit' as functional property.

Example: owl:FunctionalProperty

This example is the result when property 'speed limit' has the same syntax as described above. 'Kanaalweg Oost' is characterized by speed limit = 80 km/h (according to the system requirements). The category 'distributor road' has the same restriction. Using the reasoner 'Kanaalweg Oost' will be the inference of 'distributor road'. If in the next case 'Kanaalweg Oost' is characterized by type 'access road' and speed limit = 80 km/h, the instance is inconsistent. This is the result of:

- Acces road has a speed limit of 60 km/h;
- Distributor road has a speed limit of 80 km/h.

The reasoner illustrates a reasoner error with the message: *'Kanaalweg Oost' has more than one value for the functional property*. However, if other instances are also inconsistent, the ontology defines this problem as one reasoning error. This fact relates to Sirin & Parsia (2007); an OWL-DL consistency checker takes a document as input, and returns one word being consistent, inconsistent, or unknown.

In contrast to the inconsistency check in the last example, an example is given of the consistency of the ontology. However, in terms of verification the consistency check is limited. This is proved in the following example.

Example: a consistent ontology

The instance 'Kooibrug' is a viaduct. More specifically, a road traffic viaduct. In this case Kooibrug is part of the class 'road traffic viaduct'. In addition to this class, another class is linked to Kooibrug, namely 'noise barrier'.

```
:Kooibrug      rdf:type owl:NameIndividual ,
                        [ rdf:type owl:Restriction ;
                        owl:onProperty :is_engineering_structure ;
                        owl:onClass :Road_Traffic_Viaduct ;
                        owl:qualifiedCardinality "1"^^xsd:nonNegativeInteger
                        ] ,

                        [ rdf:type owl:Restriction ;
                        owl:onProperty :is_engineering_structure ;
                        owl:someValuesFrom :Noise_Barrier
                        ] .
```

Listing 8: Class 'Kooibrug' is part of two classes, namely 'Road traffic viaduct' and 'Noise barrier'.

Using Pellet this results into a consistent ontology. However, we know that a viaduct is a different engineering structure than a noise barrier. OWL-DL is able to solve this problem by

using *owl:disjointWith* (see example below). This results in an inconsistency after reasoning. However, a lot of manual work is required to prevent this problem, because each object in the concept library differs from each other.

```
:Noise_Barrier   rdf:type owl:Class ;
                 rdfs:subClassOf :Engineering_Structure ,
                                 [ rdf:type owl:Restriction ;
                                 owl:onProperty :hasFunction ;
                                 owl:someValuesFrom :Reducing_Noise
                                 ] ;

                 owl:disjointWith :Road_Traffic_Viaduct ,
                                  :Tunnel .
```

Listing 9: Example of 'disjoint with' modelling capability.

Last, functional requirements and preconditions cover the most system requirements of the project. These specifications are generic and reusable. Intern interface requirements are in most cases project design requirements, which are project-specific. OWL-DL is able to provide these types of specifications, but in not every case it meets the aim of the concept library; reusing of generic concepts.

7.3.2 SPARQL-DL

As explained in the previous subsection, the modelling capabilities of OWL-DL allows to test the consistency of an ontology. SPARQL-DL provides a query language on top of OWL-DL. It is a query engine which is a subset of SPARQL. Protégé provides a built-in function for SPARQL-DL. The user-friendly manner to choose related classes (right column in figure 22) to your query is of added value in comparison to SPARQL.

7.3.3 SWRL

According to O'Conner (2009), SWRL is more expressive than OWL-DL. If a higher expressiveness is not required, O'Conner (2009) suggests to use OWL-DL. SWRL is tested in terms of improving the reasoning of the ontology.

As indicated in section 6.3, Protégé supports both SWRL and the built-in reasoner Pellet. Pellet also supports SWRL. Firstly, a rule is illustrated of the restriction 'speed limit', expressed in SWRL. This rule corresponds to figure 20.

```
Road(?x), hasSpeedLimit(?x, ?y), equal(?y, 80) -> Distributor_Road(?x)
```

Listing 10: Example of SWRL.

To find added value of SWRL, an expression of a relationship which is not expressible in OWL-DL has to be found in the context of the case study. OWL-DL provides the construct *owl:FunctionalProperty*, which results in a clear consistency check (example subsection 7.3.1). With respect to reasoning, the outcome will be consistent, inconsistent or unknown. In case of this project, SRWL provides an alternative, what can be a better solution. In comparison to OWL-DL, SWRL is able to infer new facts.

A new fact can be realized by saying: any value which is 'not equal' to the value of a certain property is 'false'. According to the several types of roads, this results in the following rules (figure 21):

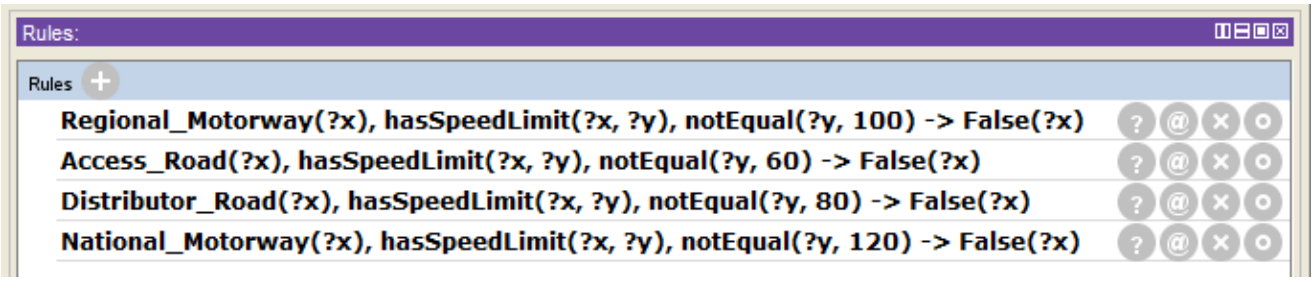


Figure 21: Rules (SWRL) expressed for different road types (Protégé 4.3.0)

The 'notEqual' construct is only provided in SWRL. Removing the *owl:FunctionalProperty* for datatype property 'has speed limit' and add these rules, inconsistent instances will be part of the new class 'False', as illustrated in figure 22. This new class gives an overview of all inconsistent concepts by using the SPARQL-DL engine (also called DL query). The example of figure 22 illustrates instance 'Kapellaan' as part of class 'False', because Kapellaan has type 'distributor road' and restriction on property of speed limit, value 60. This results in inconsistency. Because property 'speed limit' is in this case not a functional property, reasoning results in an unknown result.

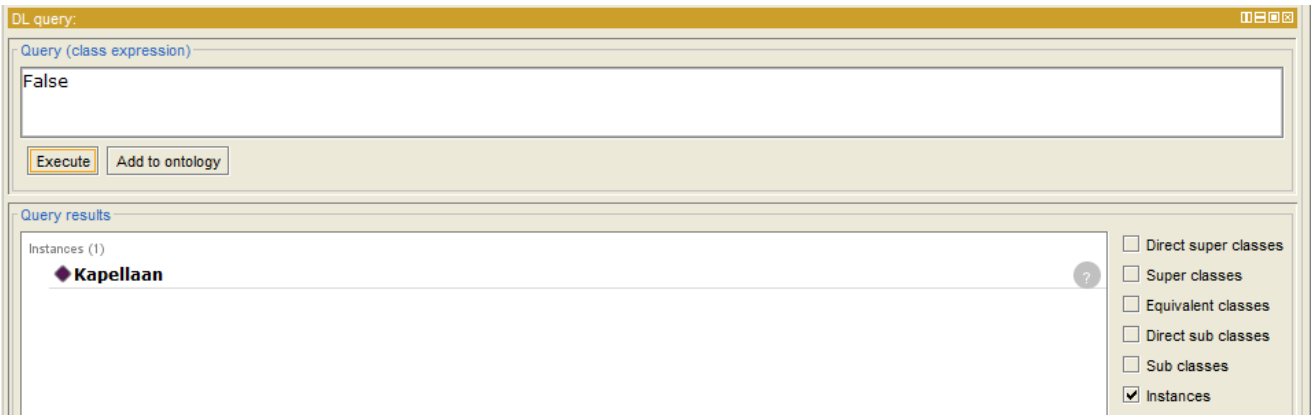


Figure 22: Outcome(s) of query class 'False' in DL query (Protégé 4.3.0)

7.4 CONCLUSIONS

7.4.1 DEMONSTRATION I: DECOMPOSITION OF A SYSTEM-OF-INTEREST

It is possible to implement a mereology in an ontology. OWL does not provide a built-in initiative, but it has the modelling capabilities to set up a mereology. The combination of the following OWL constructs provide a FBS and SBS in the ontology:

- *owl:functionOf* as sub property of *owl:ObjectProperty*;
- *owl:directFunctionOf* as sub property of *owl:functionOf*;
- *owlFunctionOf* as *owl:inverse* of *owl:hasFunction*. Same for their sub property;
- *owl:TransitiveProperty*; only for 'direct function of' and its inverse;

- *owl:someValueOf* (existential quantification).

Optional:

- *owl:hasObject* as *owl:inverse* of *owl:functionOf*;
- *owl:objectOf* as *owl:inverse* of *owl:hasFunction*.

The mereology occurs on the background in the ontology. This hierarchical structure only emerge by querying the ontology. As demonstrated in subsection 7.3.2, SPARQL-DL is of added value by providing the possibility of querying the 'function of' property. SPARQL-DL is built on the expressiveness of OWL-DL. In case of an FBS and SBS modelled in OWL, SPARQL-DL can have an added value to the modeller/ user. SPARQL-DL is able to illustrate a better overview of possible solutions during decomposing the system-of-interest. Two examples are illustrated in figure 23 and 24. Figure 23 illustrates the possibility to search for an equivalent class. This is necessary in the case of designing a FBS and SBS on the same time, also illustrated in appendix D. Furthermore, using SPARQL-DL prevents the modelling structure of 'direct function of' (*owl:directFunctionOf*). It already provides a built-in function for this direct relationship property.

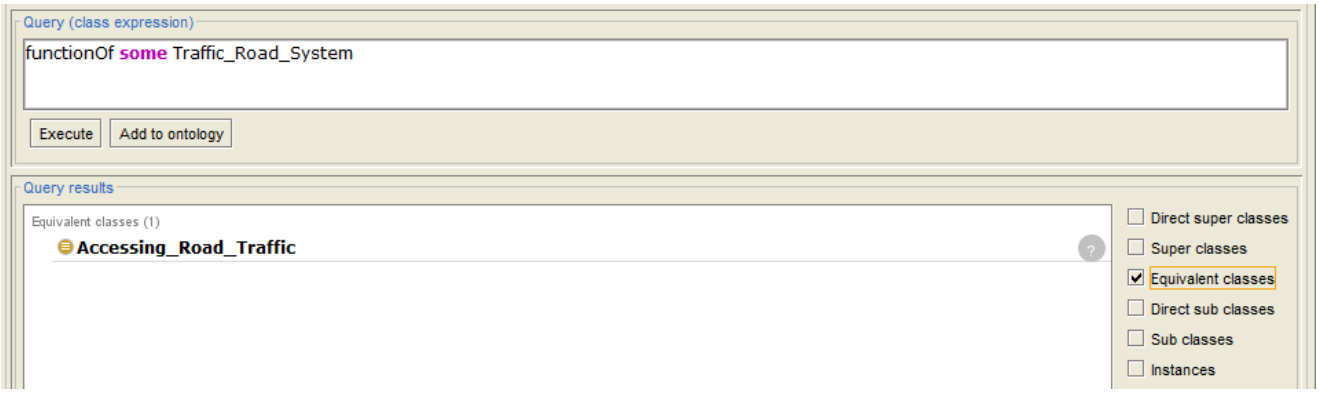


Figure 23: SPARQL-DL example 1: equivalent classes (Protégé 4.3.0)

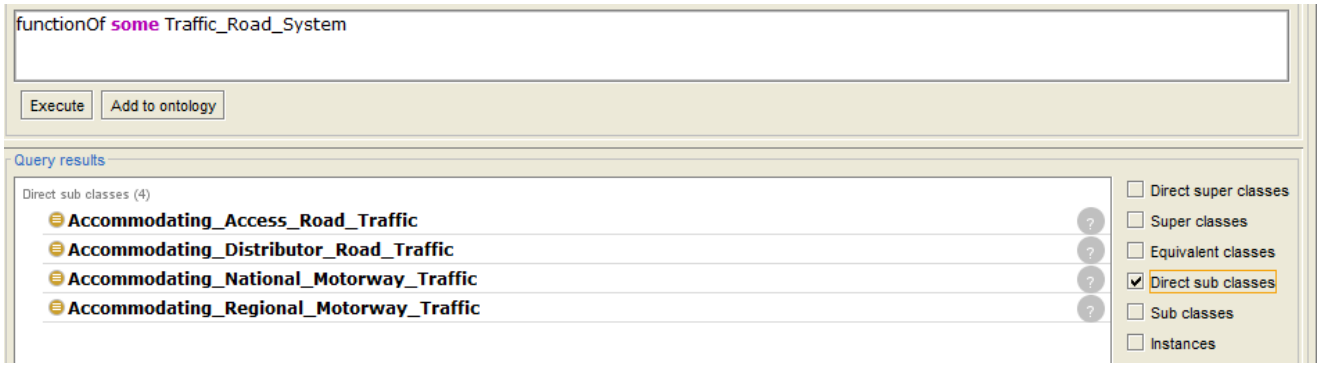


Figure 24: SPARQL-DL example 2: direct sub classes (Protégé 4.3.0)

7.4.2 DEMONSTRATION II: VERIFICATION OF AN OBJECT

The main objective of this demonstration is to verify objects to the system requirements. However, OWL-DL provides not full verification but reasoning of information within the ontology. This is the result of the fact that OWL is an open standard of the Semantic Web and the Semantic Web is based on the Open World Assumption (OWA). When information is missing information is not incorrect information. An useful component of the Semantic Web is the possibility to infer new information. OWA provides reusable and extendable information. This meets the one of the aims of a concept library, namely the possibility to extend the concept library with new concepts. Closed World Assumption (CWA) is the opposite of OWA. Table 5 explains the differences between these two assumptions.

Open World Assumption	**Closed World Assumption**
Negation as contradiction	Negation as failure
Anything might be true unless it can be proven false	Anything that cannot be found is false
Reasoning about any world consistent with the model	Reasoning about this world
Reasoning answer: I do not know/ undefined	Reasoning answer: No/ false

Table 5: OWA versus CWA (based on Drummond and Shearer, 2006)

Verification is replaced by checking an ontology on consistency, because of the fact that the Semantic Web is an OWA. OWL-DL has two major drawbacks in the context of reasoning (using Pellet – explained in section 6.4).

- Inconsistency of the ontology results into just one error. After this result the ontology can be checked on inconsistent concepts. These concepts are not summarized in one specific class or engine;
- The consistency check is limited. If we want to check whether the project data conform to an ontology, we can only check consistency of the data conform the ontology. OWL-DL cannot specify syntactic constraints (more than one word requirements) and cannot constrain the existence of some written information. To check this, the project data is put together with the ontology in a reasoner. The outcome will just inconsistent , consistent or unknown.

If we want to check whether the project data conform to an ontology, we can only check consistency of the data conform the ontology. OWL-DL cannot specify constraints and cannot constrain the existence of some written information. To check this, we just throw the project data together with the ontology in a reasoner. The reasoner reacts with the conclusion: consistent, inconsistent or unknown.

SWRL is better to express deductive knowledge (top – down logic) within ontologies. This knowledge is not necessary in this project, because it includes general statements which can be modelled by OWL-DL. However, SWRL is able to improve the consistency check by reasoning new facts, as illustrated in subsection 7.3.3. To reach an high consistency check, the next steps has to be taken:

- *owl:ObjectProperty* and *owl:DatatypeProperty* are not an *owl:FunctionalProperty*;
- All rules refer to every value which is not equal to the value of a system requirement;
- All rules refer to class 'False' or 'Inconsistent'.

The ontology is not able to check all system requirements. As mentioned in subsection 7.3.1, there is a difference between the several specifications. Table 6 distinguishes these specifications into three categories. System requirements of category 1 can be added in the ontology. System requirements of category 2 has to be decided by the organisation itself, which depends on the capability of reusing. System requirements of category 3 are not recommended to add in the ontology.

Category 1: generic specifications	**Category 2: semi-generic specifications**	**Category 3: project-related specifications**
Functional requirements	Extern interface requirements	Intern interface requirements
Preconditions	Aspect requirements	

Table 6: Division of the types of specifications

PART D: Conclusions and recommendations

8 Conclusions and recommendations

8.1 CONCLUSIONS

This report concerns the theory of OWL and its implementation on top of the current concept libraries which occur in the Dutch construction industry. This section provides an answer on the next main research question: what is the added value of OWL constructs in a concept library, in relation to the development stage of the Systems Engineering process? Not only OWL, but also modelling languages SPARQL and SWRL are part of this report to answer the main research question.

This report pays attention to upcoming concept libraries modelled in OWL. The modelling capability of OWL does not provide only taxonomic structures, but also a representation power to add restrictions. This power enhances the expressiveness of the modelling capabilities. OWL allows us to take advantage of the semantic relationships you put into your concept library, indicated as an ontology in this report. The following modelling capabilities of OWL give you the possibility to gain added value of an concept library in relation to the SE process, more specifically, the development stage of the SE process. The development stage is a design phase and is based on requirements from the client (and other stakeholders). Each project has its own set of stakeholder requirements. Nowadays, set up mereologies as FBS and SBS are a time-consuming processes. This is also the fact for verifying concepts, divided in objects (SBS) and functions (FBS).

- The first demonstration of the case study illustrates the possibility of modelling a mereology in OWL. However, this meronomic structure (appendix D) is different in comparison to the 'part – whole' relation according to Rector & Welty[32] (2005), Motschnig-Pitrik & Kaasbøll (1999) and Baggen, et al. (2013). Their approach does not assemble one of the sub-aims of the development stage; define and specify the functional boundary and performance. This will specify what the system should be able to do (FBS) (Haskins, 2006). The mereology in demonstration I provides a 'function – object – function' relation and the other addressed sources provide a 'object – object' structure;
- As described above, OWL has the modelling capability for modelling a mereology. However, the development stage is a design stage. In this case the mereology has to function as a support tool. Opportunities of concepts during setting up a mereology can be queried if needed. This in terms of one or more synonyms or the right definition;

[32] W3C publication: Simple part – whole relations in OWL ontologies.

- SPARQL-DL functions on top of OWL-DL. To use the mereology as support tool in the development stage the SPARQL-DL engine is a good option to use. You can query the ontology for possibilities ('function of' property);
- OWL provides not a full automated verification within an ontology. This is the result of OWA. Instead of full verification the use of automated reasoners to prove certain semantic relationships and thus verify in a semi-automated way is the best alternative. Nowadays full verification is only possible in a CWA, which not meets the aim of the concept library;

The difference between full automated and semi-automated verification is: explanation of reasoning (Ahrendt, et al., 2010).

- The consistency check can be enhanced by using SWRL. Using SWRL is an alternative for the standard reasoning by using Pellet. SWRL provides one class including all inconsistent individuals. This is the result of the SWRL function 'notEqual', which is a new modelling construct on top of OWL-DL;
- The consistency check is not possible for all system requirements, which is the result of OWA. Table 6 illustrates the division of different kind of requirements. Category 1 requirements can be added to the concept library and category 2 requirements has to be decided by the organisation.

According to Kublauch (2004) OWL has two different cases (chapter 5). One of them is to use OWL for terminologies or knowledge models. This case is highlighted in this research. This report supports for future more reliable developments, and might be soon applied in concept libraries on organisational level in the construction industry. Concept libraries are in a development phase and have their focus especially on generating concepts and mapping, but this research is of added value to explain what are the next possibilities for extending concept libraries. Figure 25 illustrates a knowledge representation of machine understanding information. At the moment we are in a concept learning OWL phase. The figure illustrates the more expressive possibilities of semantics. According Van Ruijven (2014) the SE process has three problems; linguistic disruption, organisational disruption and methodological disruption. Main objective of a concept library is to remedy the linguistic disruption. On the same time, this research covers also the organisational disruption. This is the result of the representation power of OWL (and SPARQL-DL and SWRL). This report just explains a small extension on the nowadays concept libraries, but it is a first step to extend these kind of libraries in the construction industry. The link between concept library and the semantic web makes u able to do more than only modelling a dictionary.

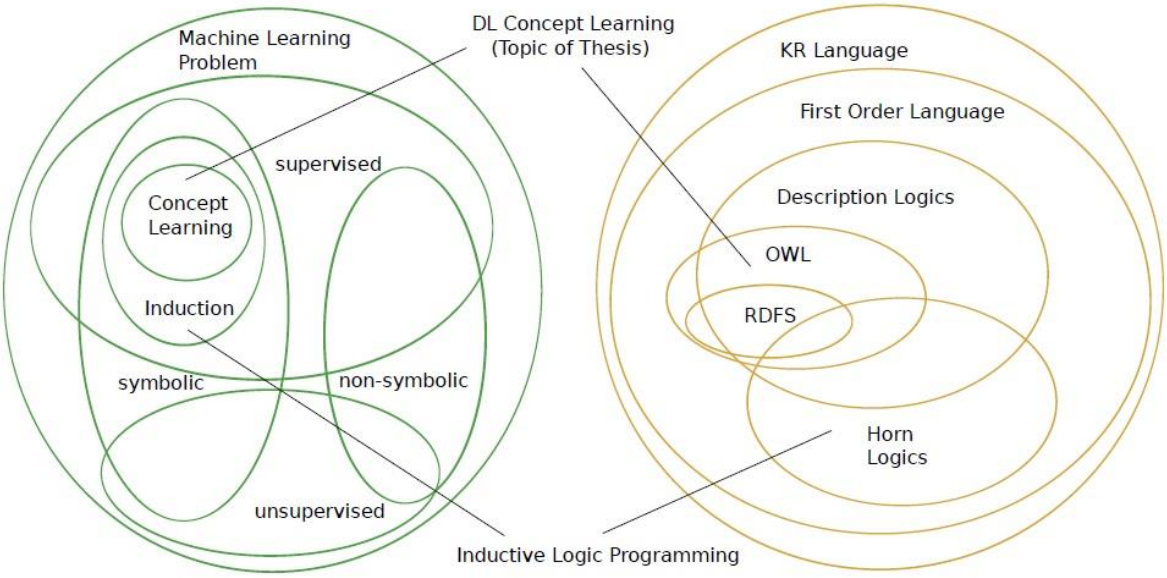


Figure 25: Overview of knowledge representation languages and machine learning problems to illustrate how this thesis integrates into existing research areas (Lehmann, 2010).

8.2 RECOMMENDATIONS

The world of ontology applications itself is changing in a way that is putting more pressure on ontology language implementations and editing tools to handle new tasks.

- Using OWL and other W3C open standards results into the use of the RDF structure, called triples. If all organisations in the Dutch construction industry use the W3C open standard to model their concept library, each organisation uses the same structure. In terms of modelling their ontology in the same structure, transformation applications as COINS[33] are not necessary needed. Triples can be exchanged without transformation. Point of notice is that this consternation is done from the point of view of this research. Officially COINS is an open BIM-standard and deals also with e.g. drawings;
- In case of this research the results are of added value for (OWL) concept libraries on organisational level. A mereology is part of the development stage of the SE process, which is a designing stage. Each organisation within the construction industry has its own manner of designing. Implement a second hierarchical structure (besides taxonomy a mereology) in a national concept library (CB-NL) will be twice more time to build up. That is a time-consuming process plus a more difficult process to agree on. Furthermore, a mereology on national level reduces the solution opportunities for the designer. This is contrary to the aim of the development stage of SE;
- As described SPARQL-DL engine has a lot of benefits for both demonstrations of the case study. Recommended is to use this engine in the user-interface. The user-interface is the other side of this research. Evidence is illustrated in the case study which is conducted in Protégé. A translation has to be made to the user-interface and finally the acceptation of the (final) users;

33

- The system requirements of the case study were abstract. This is the result of an infrastructure project. A building project could have more specific requirements, for example a wall (dimensions, materials, functions). The full potential of SWRL is not covered in the case study. It has to be used for complex business scenarios, which is not the case in this research. However, demonstration II explained an added value of using SWRL for reaching a higher consistency.

8.3 FURTHER RESEARCH

Figure 25 illustrates the knowledge representation languages. The full expressiveness of OWL is a step further than nowadays technologies as XML and RDF(S). However, it is not the last step. Ontologies can be made more and more expressive. This report explains that a concept library within the construction industry is not able to provide only a taxonomy, but also another hierarchical structure and even checking on consistency. This small piece of evidence is a part of a big research according to the semantic web.

- The semantic web is an OWA. CWA meets the aim of verification of a database. Further research has to be done if a combination between OWA and CWA is possible. This is a first step in the direction of automatic verification;
- The application of OWL constructs provides a mereology (according to the aims of the development stage) within a concept library. SKOS[34] (also a W3C standardization) provides a another vocabulary for representation of ontologies. Its main objective is to enable easy publication. SKOS has 'softer semantics'. Important point to know, the difference between SKOS and OWL is not in the process, but choices for expressing the result (output) of the process (Bechhofer & Miles, 2008).

[34] Simple Knowledge Organization System. Published by W3C as open standard in 2009.

9 References

Adriaanse, A. et al., 2012. *CB-NL Hoofdrapportage Pilotfase,* Gouda: CB-NL.

Ahrendt, W., Beckert, B., Giese, M. & Rummer, P., 2010. *Practical Aspects of Automated Deduction for Program Verification,* Koblenz: University of Koblenz.

Allemang, D. & Hendler, J., 2011. *Semantic Web for the Working Ontologist; Effective Modeling in RDFS and OWL.* 2nd red. sl:Morgan Kauffman.

ASTM International, 2014. *Form and Style of Standards, ASTM Blue Book.* West Conshohocken: ASTM International.

Baggen, M. et al., 2013. *Nederlandse Conceptenbibliotheek voor de gebouwde omgeving (CB-NL); Modeling Guide,* Amersfoort: CB-NL.

Bechhofer, S. & Miles, A., 2008. *Using OWL and SKOS,* Manchester and Oxford: W3C.

Bechhofer, S. et al., 2004. *OWL Web Ontology Language; Reference.* [Online]
Available at: http://www.w3.org/TR/owl-ref/#
[Geopend 11 May 2014].

Beetz, J., 2014. *A scalable network of concept libraries using distributed graph databases,* Eindhoven: Eindhoven University of Technology.

Berners-Lee, T., 2006. *Artificial Intelligence and the Semantic Web,* sl: Association for the Advancement of Artificial Intelligence.

Brickley, D. & Guha, R., 2014. *RDF Schema 1.1.* [Online]
Available at: http://www.w3.org/TR/rdf-schema/
[Geopend 7 May 2014].

Brown, S., 2013. *Order from Chaos,* Australia: TechEd.

Busker, H., 2011. *Faalkosten in de GWW sector dalen licht,* Rotterdam: USP Marketing Consultancy BV.

Carroll, J., Herman, I. & Patel-Schneider, P. F., 2012. *OWL 2 Web Ontology Language; RDF-Based Semantics (Second Edition).* [Online]
Available at: http://www.w3.org/TR/2012/REC-owl2-rdf-based-semantics-20121211/
[Geopend 7 May 2014].

Coinsweb, 2008. *Voorbeeld 'Ongelijkvloerse Kruising Van Harinxmakanaal'.* [Online]
Available at:
http://www.coinsweb.nl/wiki/index.php/Voorbeeld_%E2%80%98Ongelijkvloerse_Kruising_Van_Harinxmakanaal%E2%80%99
[Geopend 14 June 2014].

Corcho, O. & Gomez-Perez, A., 2000. A Roadmap to Ontology Specification Languages, in Rose Dieng and Olivier Corby (eds.). *Knowledge Engineering and Knowledge Management; Methods, Models and Tools,* pp. 80-96.

CROW, 2011. *Handboek specificeren.* Rotterdam: CROW.

de Bruijn, J. et al., 2008. *Modeling Semantic Web Services; The Web Service Modeling Language.* 1st red. sl:Springer.

De Hoff, B., Levack, D. J. & Rhodes, R. E., 2009. *The Functional Breakdown Structure (FBS) and Its Relationship to Life Cycle Cost.* Denver, Colorado, 45th AIAA/ASME/SAE/ASEE Joint Propulsion Conference.

Department of Defense, 2001. *Systems Engineering Fundamentals.* Virginia: Defense aquisition university press.

Drummond, N. & Shearer, R., 2006. *The Open World Assumption,* Manchester: The University of Manchester.

Eastman, C., Teicholz, P., Sacks, R. & Liston, K., 2008. *BIM Handbook: A Guide to Building Information Modeling for Owners, Managers, Designers, Engineers and Contractors.* New York: John Wiley and Sons.

Feigenbaum, L. et al., 2013. *RIF FAQ.* [Online]
Available at: http://www.w3.org/2005/rules/wiki/RIF_FAQ
[Geopend 7 June 2014].

Fensel, D., 2003. *Ontologies: Silver Bullet for Knowledge Management and Electronic Commerce.* 2nd edition red. Berlin: Springer-Verlag.

Fensel, D. & Hench, G., 2008. From Web to Semantic Web. In: D. Fensel, M. Kerrigan & M. Zaremba, red. *Implementing Semantic Web Services; The SESA Framework.* Innsbruck: Springer, pp. 3-25.

Fortineau, V., Paviot, T., Louis-Sidney, L. & Lamouri, S., 2012. *SWRL as a rule language for ontology-based models in power plant design,* Paris, France: LCPI/Arts.

Graf, P., 2008. *Space Systems Engineering: System hierarchy and Work Breakdown Structure Module,* Boulder, Colorado: NASA.

Gruber, T., 1993. A translation approach to portable ontology specifications. *International Journal of Knowledge Acquisition for Knowledge-Based Systems,* 5(2).

Halligan, R., 2013. *Project Performance International; Types of Requirements.* [Online]
Available at: http://www.ppi-int.com/systems-engineering/types-of-requirements.php
[Geopend 24 June 2014].

Haskins, C., 2006. *Systems Engineering Handbook; a guide for system life cycle processes and activities.* 3.0 red. sl:INCOSE.

Hawke, S., Herman, I., Archer, P. & Prud'hommeaux, E., 2013. *Semantic Web Activity.* [Online]
Available at: http://www.w3.org/2001/sw/
[Geopend 29 May 2014].

Horridge, M. et al., 2004. *A practical guide to building OWL ontologies using the Protege-OWL plugin and CO-ODE tools,* Manchester: The University of Manchester.

Horrocks, I. et al., 2004. *A Semantic Web Rule Language - Combining OWL and RuleML.* [Online]
Available at: http://www.w3.org/Submission/SWRL/
[Geopend 4 June 2014].

Horrocks, I. et al., 2004. *SWRL: A Semantic Web Rule Language combining OWL and RuleML,* sl: W3C Member Submission.

IEEE, 2005. *IEEE Standard for Application and Management of the Systems Engineering Process,* Washington, DC, USA: Institute of Electrical and Electronics Engineers.

IEEE, 2011. *IEEE Guide - adoption of the project management institute (PMI) standard. A guide to the project management body of knowledge (PMBOK guide).* 4th edition red. sl:Institute of Electrical and Electronics Engineers.

iNFRANEA, 2013. *iBIM – the iNFRANEA BIM Concept,* Antwerpen: iNFRANEA.

ISO/IEC 15288, 2002. *ISO/IEC 15288(E) - Systems engineering - System life cycle processes,* Geneva: ISO/IEC.

ISO/IEC 16354, 2013. *ISO/IEC 16354:2013-3(E) - Guidelines for knowledge libraries and object libraries,* Geneva: ISO/IEC.

ISO/IEC 26702, 2007. *ISO/IEC 26702; Systems engineering - Application and management of the systems engineering process,* Delft: Nederlandse Normalisatie-instituut.

Kashyap, V., Bussler, C. & Moran, M., 2008. *The Semantic Web; Semantics for Data and Services on the Web.* Manchester: Springer.

Knublauch, H., 2004. *Editing OWL Ontologies with Protege,* Stanford: Stanford University.

Kollia, I., Glimm, B. & Horrocks, I., 2011. *Answering queries over OWL ontologies with SPARQL,* San Francisco, California: OWLED Steering Committee.

Lawson, H., 2010. *A journey through the systems landscape,* London: College Publications.

Lehmann, J., 2010. *Learning OWL Class Expressions,* Leipzig: University of Leipzig.

Mesarović, M. D., Macko, D. & Takahara, Y., 1970. *Theory of Hierarchical, Multilevel, Systems.* New York: Academic Press.

Motschnig-Pitrik, R. & Kaasboll, J., 1999. Part-Whole Relationship Categories and Their Application in Object-Oriented Analysis. *IEEE Transactions on knowledge and data engineering,* 11(5), pp. 779-797.

Musen, M., 1992. Dimensions of knowledge sharing and reuse. *Computers and Biomedical Research,* Issue 25, pp. 435-467.

NASA, 2012. *EE11 MSFC Technical standard; Project management and Systems Engineering handbook.* B red. Alabama: NASA.

Noy, N. F. & McGuinness, D. L., 2001. *Ontology Development 101: A Guide to Creating Your First Ontology,* Stanford: Stanford University.

O'Connor, M., 2009. *The Semantic Web Rule Language,* Stanford: Stanford University.

Ogden, C. & Richards, I., 1923. he meaning of meaning. A study of the influence of language upon thought and of the science of symbolism. *Supplementary essays by Malinowski and Crookshank.*

Parnell, G., Driscoll, P. & Henderson, D., 2011. *Decision making in Systems Engineering and Management,* Hoboken, New Jersey: John Wiley and Sons.

Protege, 2008. *Using DL reasoners in Protege-OWL.* [Online]
Available at: http://protegewiki.stanford.edu/wiki/Using_Reasoners
[Geopend 4 June 2014].

Ram, D., James, R. & Jan, V., 2011. Information mining - Reflections on recent advancements and the road aheadin data, text and media mining. *Decision support system,* Issue 51, pp. 727-731.

Rector, A. & Welty, C., 2005. *Simple part-whole relations in OWL Ontologies.* [Online]
Available at: http://www.w3.org/2001/sw/BestPractices/OEP/SimplePartWhole/
[Geopend 9 June 2014].

Roozenburg, N. & Eekels, J., 1998. *Productontwerpen, structuur en methoden.* Utrecht: Lemma.

Sirin, E. & Parsia, B., 2007. *SPARQL-DL: SPARQL Query for OWL-DL.* Innsbruck, OWLED.

Smith, B., 2004. *Beyond Concepts: Ontology as Reality Representation.* Saarbrucken, Germany, Institute for Formal Ontology and Medical Information Science, Saarland University.

Smith, M. K., Welty, C. & McGuinness, D. L., 2004. *OWL Web Ontology Language Guide.* [Online]

Available at: http://www.w3.org/TR/owl-guide/
[Geopend 15 May 2014].

Uschold, M., 2001. *Where are the semantics in the Semantic Web?,* Seattle, USA: The Boeing Company.

van Netten, J., 2005. *Handreiking Functioneel Specificeren.* 1.0 red. sl:ExpertiseCentrum Opdrachtgeverschap.

van Ruijven, L., 2014. *White paper: BIM (Bouw Informatie Model); de rol van informatie in (complexe) projecten,* Rotterdam: Croon Elektrotechniek B.V..

W3C OWL Working Group, 2012. *OWL 2 Web Ontology Language; Document Overview (Second Edition).* [Online]
Available at: http://www.w3.org/TR/owl-overview/
[Geopend 7 May 2014].

W3C, 2012. *W3C Standards.* [Online]
Available at: http://www.w3.org/standards/
[Geopend 12 May 2014].

Waterfeld, W., Weiten, M. & Haase, P., 2008. Ontology Management Infrastructures. In: M. Hepp, A. de Moor, P. de Leenheer & Y. Sure, red. *Ontology Management; Semantic Web, Semantic Web Services, and Business Applications.* Darmstadt: Springer, pp. 59-87.

Werkgroep Leidraad SE, 2009. *Leidraad voor Systems Engineering binnen de GWW-sector,* Den Haag: MVA Communicatie.

Werkgroep Leidraad SE, 2013. *Leidraad voor Systems Engineering binnen de GWW-sector; De samenhang centraal,* Den Haag: MVA Communicatie.

Yang, F. & Tan, Y., 2011. A method of object-based deduplication. *Journal of Networks,* Issue 6, pp. 1705-1712.

Zwakhals, P., 2014. *Een standaard voor objectenbibliotheken in de installatiemarkt ,* Zoetermeer: Uneto-Vni.

10 Appendices

Appendix A – RDF(S) vocabulary

RDF classes **Source:** Brickley & Guha (2014)

Class name	Comment
rdfs:Resource	The class resource, everything.
rdfs:Literal	The class of literal values, e.g. textual strings and integers.
rdf:langString	The class of language-tagged string literal values.
rdf:HTML	The class of HTML literal values.
rdf:XMLLiteral	The class of XML literal values.
rdfs:Class	The class of classes.
rdf:Property	The class of RDF properties.
rdfs:Datatype	The class of RDF datatypes.
rdf:Statement	The class of RDF statements.
rdf:Bag	The class of unordered containers.
rdf:Seq	The class of ordered containers.
rdf:Alt	The class of containers of alternatives.
rdfs:Container	The class of RDF containers.
rdfs:ContainerMembershipProperty	The class of container membership properties, rdf:_1, rdf:_2, ..., all of which are sub-properties of 'member'.
rdf:List	The class of RDF Lists.

RDF properties

Property name	Comment	Domain	Range
rdf:type	The subject is an instance of a class.	rdfs:Resource	rdfs:Class
rdfs:subClassOf	The subject is a subclass of a class.	rdfs:Class	rdfs:Class
rdfs:subPropertyOf	The subject is a subproperty of a property.	rdf:Property	rdf:Property
rdfs:domain	A domain of the subject property.	rdf:Property	rdfs:Class
rdfs:range	A range of the subject property.	rdf:Property	rdfs:Class
rdfs:label	A human-readable name for the subject.	rdfs:Resource	rdfs:Literal
rdfs:comment	A description of the subject resource.	rdfs:Resource	rdfs:Literal
rdfs:member	A member of the subject resource.	rdfs:Resource	rdfs:Resource
rdf:first	The first item in the subject RDF list.	rdf:List	rdfs:Resource
rdf:rest	The rest of the subject RDF list after the first item.	rdf:List	rdf:List
rdfs:seeAlso	Further information about the subject resource.	rdfs:Resource	rdfs:Resource
rdfs:isDefinedBy	The definition of the subject resource.	rdfs:Resource	rdfs:Resource
rdf:value	Idiomatic property used for structured values.	rdfs:Resource	rdfs:Resource
rdf:subject	The subject of the subject RDF statement.	rdf:Statement	rdfs:Resource
rdf:predicate	The predicate of the subject RDF statement.	rdf:Statement	rdfs:Resource
rdf:object	The object of the subject RDF statement.	rdf:Statement	rdfs:Resource

Appendix B – OWL vocabulary

Source: Bechhofer et al. (2014)

OWL classes

owl:AllDifferent
owl:AnnotationProperty
owl:Class
owl:DataRange
owl:DatatypeProperty
owl:DeprecatedClass
owl:DeprecatedProperty
owl:FunctionalProperty
owl:InverseFunctionalProperty
owl:Nothing
owl:ObjectProperty
owl:Ontology
owl:OntologyProperty
owl:Restriction
owl:SymmetricProperty
owl:Thing
owl:TransitiveProperty

OWL properties

owl:allValuesFrom
owl:backwardCompatibleWith
owl:cardinality
owl:complementOf
owl:differentFrom
owl:disjointWith
owl:distinctMembers
owl:equivalentClass
owl:equivalentProperty
owl:hasValue
owl:imports
owl:incompatibleWith
owl:intersectionOf
owl:inverseOf
owl:maxCardinality
owl:minCardinality
owl:oneOf
owl:onProperty
owl:priorVersion
owl:sameAs
owl:someValuesFrom
owl:unionOf
owl:versionInfo

Appendix C – Interviews

Interviewprotocol betreft de ontwikkeling van de CB-NL
Gericht op personen die voorheen al hebben deelgenomen aan de GWW content groep van de CB-NL organisatie

In het kader van de afstudeeropdracht: The added value of an OWL concept library in the development stage of the Systems Engineering process.

Bob van Thiel
In samenwerking met de Technische Universiteit van Eindhoven en ARCADIS.

Datum: *03-03-2014*

Locatie: -

Informatie respondent

Naam: *Anton Beks*

Functie: *Hoofd wegontwerp*

Bedrijf: *Heijmans*

Als afstudeerder binnen ARCADIS doe ik onderzoek naar de implementatie van een OWL conceptenbibliotheek in de ontwikkelingsfase (volgens de Systems Engineering levenscyclus van INCOSE) van het Systems Engineering (SE) proces. Aan de hand van een case studie wil ik aandachtspunten binnen deze fase combineren met een OWL conceptenbibliotheek om te onderzoeken of deze bibliotheek een toegevoegde waarde kan leveren. Naast het bestuderen van literatuur zou ik graag een kort(e) interview/ vragenlijst willen afnemen om de belangrijke aandachtspunten en raakvlakken tussen SE en de CB-NL in kaart te brengen.

Om geen misverstanden te ondervinden heeft dit interview betrekking op de volgende scope:

- De SE methodiek (ISO 15288);
    - Het specificeren (top-down structuur) en structuren van informatie (op basis van de klanteisen);
    - Verificatieloop ontwerp met klant- en systeemeisen;
    - Toewijzen systeemeisen aan functioneel ontwerp;
    - Definiëren interne en externe raakvlakken functioneel ontwerp;
    - Verifiëren en valideren van het systeem.

De onderstaande vragen zijn voortgekomen uit de bestudeerde literatuur over SE en de CB-NL en interne gesprekken. Deze informatie dient als basis voor dit interview.

1. In hoeverre bent u tot op heden nog betrokken bij de ontwikkelingen van de CB-NL?

Op dit moment ben ik lid van de testgroep voor de CB-NL GWW. Daarnaast ben ik als gebruikersdeskundige betrokken bij de gesprekken van CB-NL met ICT-vendors.

2. Als conclusie heeft de experimentgroep geformuleerd dat er duidelijk behoefte is aan een sector brede conceptenbibliotheek. Waarop is deze conclusie gebaseerd?

Als belangrijk onderdeel van de life cycle is ontwerp en realisatie spil in het proces. Men krijgt vanuit de initiatief fase gegevens aangeleverd en moet data opleveren na realisatie voor de beheers- en onderhoudsfase. Ook tussen het ontwerp- en de realisatiefase is er overdracht van data en kennis. Hiervoor is het belangrijk om één taal te spreken. De conceptenbibliotheek moet hierin voorzien. Daarnaast werken in de ontwerp en de realisatiefase vaak meerdere partijen samen. Ook dan is een eenduidige taal van groot belang.

3. Tijdens de eerste bijeenkomst is er bij de eerste onderzoeksvraag (opstellen objectenboom) vastgelopen. Wat was hiervan de oorzaak?

De semantische bibliotheek die op dat moment voorhanden was bleek sterk vervuild en zeer onvolledig. Hierdoor ontstond een discussie binnen de experimentgroep. Tevens bleek dat het hiermee niet mogelijk was een goede objectenboom op te stellen (te veel omissies).

4a. Vindt u de viertal onderzoeksvragen (zie hieronder) van het vorige experiment nog steeds als ''hoofdonderzoeksvragen'' gelden voor het volgend experiment?

- Kan met de objectenbibliotheek een SE model gemaakt worden (objectenboom)?
- Kan de planning gekoppeld worden aan de objectenboom (BIM) die is gebaseerd op de objectenbibliotheek?
- Kan de SSK raming opgesteld worden op basis van deze objectenboom?
- Kan het verificatieproces doorlopen worden met een objectenboom (BIM) die is gebaseerd op de objectenbibliotheek?

Ja, deze onderzoeksvragen zijn nog steeds actueel.

4b. Wat vindt u naast of in plaats van de genoemde viertal onderzoeksvragen uit het vorig experiment belangrijke aandachtspunten binnen de ontwerp- en realisatiefase om de CB-NL toe te passen/ te testen? Graag toelichten waarom.

Een zo volledig mogelijke objectenbibliotheek om een goede objectenboom te kunnen maken waarbij het voor iedereen duidelijk is hoe deze objecten zijn gedefinieerd. Daarnaast moet het mogelijk zijn om bedrijfs- of project specifieke objecten te kunnen definiëren in een eigen omgeving, maar die wel dezelfde opbouw hebben als die in de CB-NL. Ook uiteraard weer met als doel om data en informatie eenduidig uit te kunnen wisselen.

5. Verwacht u (nog steeds) een toegevoegde waarde van de CB-NL met name in de ontwerp- en realisatie fase van het SE proces?

Zoals bij vraag 2 aangeven is het van belang dat alle betrokken partijen in de life cycle van een project op een
eenduidige wijze met elkaar data uit kunnen wissel en met elkaar kunnen communiceren. Een door de gehele branche gedragen conceptenbibliotheek is daarbij van het grootste belang.

Interviewprotocol betreft de ontwikkeling van de CB-NL
Gericht op personen die voorheen al hebben deelgenomen aan de GWW content groep van de CB-NL organisatie

In het kader van de afstudeeropdracht: The added value of an OWL concept library in the development stage of the Systems Engineering process.

Bob van Thiel
In samenwerking met de Technische Universiteit van Eindhoven en ARCADIS.

Datum: *06-03-2014*

Locatie: -

Informatie respondent

Naam: *Kenzo Oijevaar*

Functie: *Systems Engineer en risicomanager*

Bedrijf: *Heijmans*

Als afstudeerder binnen ARCADIS doe ik onderzoek naar de implementatie van een OWL conceptenbibliotheek in de ontwikkelingsfase (volgens de Systems Engineering levenscyclus van INCOSE) van het Systems Engineering (SE) proces. Aan de hand van een case studie wil ik aandachtspunten binnen deze fase combineren met een OWL conceptenbibliotheek om te onderzoeken of deze bibliotheek een toegevoegde waarde kan leveren. Naast het bestuderen van literatuur zou ik graag een kort(e) interview/ vragenlijst willen afnemen om de belangrijke aandachtspunten en raakvlakken tussen SE en de CB-NL in kaart te brengen.

Om geen misverstanden te ondervinden heeft dit interview betrekking op de volgende scope:

- De SE methodiek (ISO 15288);
    - Het specificeren (top-down structuur) en structuren van informatie (op basis van de klanteisen);
    - Verificatieloop ontwerp met klant- en systeemeisen;
    - Toewijzen systeemeisen aan functioneel ontwerp;
    - Definiëren interne en externe raakvlakken functioneel ontwerp;
    - Verifiëren en valideren van het systeem.

De onderstaande vragen zijn voortgekomen uit de bestudeerde literatuur over SE en de CB-NL en interne gesprekken. Deze informatie dient als basis voor dit interview.

1. In hoeverre bent u tot op heden nog betrokken bij de ontwikkelingen van de CB-NL?

   Ik maak nu deel uit van de Contentgroep GWW.

2. Als conclusie heeft de experimentgroep geformuleerd dat er duidelijk behoefte is aan een sector brede conceptenbibliotheek. Waarop is deze conclusie gebaseerd?

-

3. Tijdens de eerste bijeenkomst is er bij de eerste onderzoeksvraag (opstellen objectenboom) vastgelopen. Wat was hiervan de oorzaak?

-

4a. Vindt u de viertal onderzoeksvragen (zie hieronder) van het vorige experiment nog steeds als ''hoofdonderzoeksvragen'' gelden voor het volgend experiment?

- Kan met de objectenbibliotheek een SE model gemaakt worden (objectenboom)?
- Kan de planning gekoppeld worden aan de objectenboom (BIM) die is gebaseerd op de objectenbibliotheek?
- Kan de SSK raming opgesteld worden op basis van deze objectenboom?
- Kan het verificatieproces doorlopen worden met een objectenboom (BIM) die is gebaseerd op de objectenbibliotheek?

Geen van de vraagstellingen kan kloppen aangezien we het niet hebben over objectenboom, maar over een concepten bibliotheek.

4b. Wat vindt u naast of in plaats van de genoemde viertal onderzoeksvragen uit het vorig experiment belangrijke aandachtspunten binnen de ontwerp- en realisatiefase om de CB-NL toe te passen/ te testen? Graag toelichten waarom.

1. *Hoe gaat de CB-NL straks in de praktijk contractueel door aannemers worden gebruikt? Rijkswaterstaat heeft aangegeven altijd de OTL voor te zullen schrijven en niet de CB-NL. Hierdoor is het een vraag hoe en of een aannemer rekening moet houden met de CB-NL.*
2. *Snappen de gebruikers (zoals aannemers) van de CB-NL uit het Midden – en Kleinbedrijf hoe ze de CB-NL moeten gaan gebruiken? Op dit moment lijkt de CB-NL nog erg theoretisch te zijn voor veel mensen. Is de CB-NL in staat om haar boodschap ook 'eenvoudig' uit te leggen, of blijft het een abstract verhaal zodat het midden – en kleinbedrijf niet zal (kunnen) aanhaken?*
3. *Hoe kan een aannemer straks de gegevens onttrekken aan de CB-NL en hoe kan zij die gegevens daarna eenvoudig intern gebruiken? Puur de website zorgt nog niet voor gemakkelijk gebruik intern. Intern (bij bijvoorbeeld projecten) wil je een lijstje hebben met termen. Nu lijkt het nog lastig te zijn om die gegevens eruit te halen en te gebruiken. Als iemand bijvoorbeeld gewoon een simpele Excel wil gebruiken, kan dat dan, of is er altijd specialistische tooling nodig (bijv. via webservices)? Niet elke aannemer (zeker vanuit het midden – en kleinbedrijf) beschikt wellicht over geavanceerde tooling. Dus hoe moeten zij er dan mee om gaan?*
4. *Hoe gaat de beheerorganisatie straks feitelijk werken? Gedurende de eerste fase zal de markt een bijdrage leveren, maar hoe ziet dit er op termijn uit en hoe blijft daarbij de aansluiting met de markt gewaarborgd? Er bestaat een kans dat de*

beheerorganisatie 'ver' van de markt af staat zodat men de CB-NL als een theoretisch framework gaat zien en niet als een praktisch inzetbare woordenlijst.

5. Verwacht u (nog steeds) een toegevoegde waarde van de CB-NL met name in de ontwerp- en realisatie fase van het SE proces?

   Ik ben overtuigd van de toegevoegde waarde als de CB-NL ook daadwerkelijk hèt woordenboek wordt voor de Nederlandse bouwwereld. Als de CB-NL niet als centrale en contractuele plek wordt gezien, maar mensen moeten nog steeds naar de afzonderlijke bronnen toe gaan (zoals de OTL van RWS), dan heb ik daar mijn vraagtekens bij.

Interviewprotocol betreft de ontwikkeling van de CB-NL
Gericht op personen die voorheen al hebben deelgenomen aan de GWW content groep van de CB-NL organisatie

In het kader van de afstudeeropdracht: The added value of an OWL concept library in the development stage of the Systems Engineering process.

Bob van Thiel
In samenwerking met de Technische Universiteit van Eindhoven en ARCADIS.

Datum: *06-03-2014*

Locatie: -

Informatie respondent

Naam: *Renzo van Rijswijk*

Functie: *Manager R&D*

Bedrijf: *Strukton*

Als afstudeerder binnen ARCADIS doe ik onderzoek naar de implementatie van een OWL conceptenbibliotheek in de ontwikkelingsfase (volgens de Systems Engineering levenscyclus van INCOSE) van het Systems Engineering (SE) proces. Aan de hand van een case studie wil ik aandachtspunten binnen deze fase combineren met een OWL conceptenbibliotheek om te onderzoeken of deze bibliotheek een toegevoegde waarde kan leveren. Naast het bestuderen van literatuur zou ik graag een kort(e) interview/ vragenlijst willen afnemen om de belangrijke aandachtspunten en raakvlakken tussen SE en de CB-NL in kaart te brengen.

Om geen misverstanden te ondervinden heeft dit interview betrekking op de volgende scope:

- De SE methodiek (ISO 15288);
    - Het specificeren (top-down structuur) en structuren van informatie (op basis van de klanteisen);
    - Verificatieloop ontwerp met klant- en systeemeisen;
    - Toewijzen systeemeisen aan functioneel ontwerp;
    - Definiëren interne en externe raakvlakken functioneel ontwerp;
    - Verifiëren en valideren van het systeem.

De onderstaande vragen zijn voortgekomen uit de bestudeerde literatuur over SE en de CB-NL en interne gesprekken. Deze informatie dient als basis voor dit interview.

1. In hoeverre bent u tot op heden nog betrokken bij de ontwikkelingen van de CB-NL?

   Ik ben betrokken in de GWW werkgroep als Reviewer

2. Als conclusie heeft de experimentgroep geformuleerd dat er duidelijk behoefte is aan een sector brede conceptenbibliotheek. Waarop is deze conclusie gebaseerd?

De wens om informatie aan elkaar door te kunnen geven (de BIM-zaagtand). Om dat te kunnen doen zijn afspraken nodig over wat je nu precies bedoeld. Om te voorkomen dat we voor elke opdrachtgever een andere bibliotheek moeten hanteren is het goed dat de integratie wordt gezocht tussen RWS, RGD en Prorail en de disciplines GWW, B&U en Installaties en Omgeving.

3. Tijdens de eerste bijeenkomst is er bij de eerste onderzoeksvraag (opstellen objectenboom) vastgelopen. Wat was hiervan de oorzaak?

Probleem is met name definitiekwestie en daarnaast de verschillende standpunten waarmee naar deze materie gekeken wordt. Daarbij speelt ook de valkuil voor teveel theoretisch geneuzel, waardoor je niet van de kant af komt en in cirkeltjes blijft discussiëren.

4a. Vindt u de viertal onderzoeksvragen (zie hieronder) van het vorige experiment nog steeds als ''hoofdonderzoeksvragen'' gelden voor het volgend experiment?

- Kan met de objectenbibliotheek een SE model gemaakt worden (objectenboom)?
- Kan de planning gekoppeld worden aan de objectenboom (BIM) die is gebaseerd op de objectenbibliotheek?
- Kan de SSK raming opgesteld worden op basis van deze objectenboom?
- Kan het verificatieproces doorlopen worden met een objectenboom (BIM) die is gebaseerd op de objectenbibliotheek?

Ja, mits er eerst een duidelijke definitie wordt gesteld voor wat een object, een activiteit, een werkpakket, een objectenboom, een objecttypenboom, een WBS, een SBS is…zonder dat heeft het geen zin.

En nee, de SE-leidraad geeft daar geen oplossing in. Die blijft in te abstracte termen definities opstellen, waardoor iedereen weer zijn eigen interpretatie er op los kan laten

4b. Wat vindt u naast of in plaats van de genoemde viertal onderzoeksvragen uit het vorig experiment belangrijke aandachtspunten binnen de ontwerp- en realisatiefase om de CB-NL toe te passen/ te testen? Graag toelichten waarom.

Uit eigen onderzoek is gebleken dat we niet zonder ruimtes kunnen in dit verhaal. Dit moet echt ingebracht worden willen we de boel kunnen oplossen.

5. Verwacht u (nog steeds) een toegevoegde waarde van de CB-NL met name in de ontwerp- en realisatie fase van het SE proces?

Ja.

Interviewprotocol betreft de ontwikkeling van de CB-NL
Gericht op personen die voorheen al hebben deelgenomen aan de GWW content groep van de CB-NL organisatie

In het kader van de afstudeeropdracht: The added value of an OWL concept library in the development stage of the Systems Engineering process.

Bob van Thiel
In samenwerking met de Technische Universiteit van Eindhoven en ARCADIS.

Datum: *05-03-2014*

Locatie: -

Informatie respondent

Naam: *Sander van Nederveen*

Functie: *Assistent professor BIM integraal ontwerp*

Bedrijf: *Technische Universiteit Delft*

Als afstudeerder binnen ARCADIS doe ik onderzoek naar de implementatie van een OWL conceptenbibliotheek in de ontwikkelingsfase (volgens de Systems Engineering levenscyclus van INCOSE) van het Systems Engineering (SE) proces. Aan de hand van een case studie wil ik aandachtspunten binnen deze fase combineren met een OWL conceptenbibliotheek om te onderzoeken of deze bibliotheek een toegevoegde waarde kan leveren. Naast het bestuderen van literatuur zou ik graag een kort(e) interview/ vragenlijst willen afnemen om de belangrijke aandachtspunten en raakvlakken tussen SE en de CB-NL in kaart te brengen.

Om geen misverstanden te ondervinden heeft dit interview betrekking op de volgende scope:

- De SE methodiek (ISO 15288);
    - Het specificeren (top-down structuur) en structuren van informatie (op basis van de klanteisen);
    - Verificatieloop ontwerp met klant- en systeemeisen;
    - Toewijzen systeemeisen aan functioneel ontwerp;
    - Definiëren interne en externe raakvlakken functioneel ontwerp;
    - Verifiëren en valideren van het systeem.

De onderstaande vragen zijn voortgekomen uit de bestudeerde literatuur over SE en de CB-NL en interne gesprekken. Deze informatie dient als basis voor dit interview.

1. In hoeverre bent u tot op heden nog betrokken bij de ontwikkelingen van de CB-NL?

   Zijdelings via project v-con.

2. Als conclusie heeft de experimentgroep geformuleerd dat er duidelijk behoefte is aan een sector brede conceptenbibliotheek. Waarop is deze conclusie gebaseerd?

Ik denk vooral op basis van het enthousiasme van de deelnemers.

3. Tijdens de eerste bijeenkomst is er bij de eerste onderzoeksvraag (opstellen objectenboom) vastgelopen. Wat was hiervan de oorzaak?

Ik denk verschil in definities van de objectenboom, en/of verschillende ideeën over het omgaan met functies.

4a. Vindt u de viertal onderzoeksvragen (zie hieronder) van het vorige experiment nog steeds als ''hoofdonderzoeksvragen'' gelden voor het volgend experiment?

- Kan met de objectenbibliotheek een SE model gemaakt worden (objectenboom)?
- Kan de planning gekoppeld worden aan de objectenboom (BIM) die is gebaseerd op de objectenbibliotheek?
- Kan de SSK raming opgesteld worden op basis van deze objectenboom?
- Kan het verificatieproces doorlopen worden met een objectenboom (BIM) die is gebaseerd op de objectenbibliotheek?

Ja

4b. Wat vindt u naast of in plaats van de genoemde viertal onderzoeksvragen uit het vorig experiment belangrijke aandachtspunten binnen de ontwerp- en realisatiefase om de CB-NL toe te passen/ te testen? Graag toelichten waarom.

Toegevoegde waarde van bibliotheken, rol van standaards.

Heel belangrijk om echt te gaan bouwen en testen en niet te blijven hangen in modellen en documenten.

5. Verwacht u (nog steeds) een toegevoegde waarde van de CB-NL met name in de ontwerp- en realisatie fase van het SE proces?

Ja, maar er moet meer tijd voor uitgetrokken worden.

Er werd eerder beloofd dat er binnen x maanden een draaiend systeem zou zijn, dat was niet realistisch en dat leidt tot verlies van geloofwaardigheid en krediet. Het is lastige materie en daar moet je helder over zijn.

Interviewprotocol betreft de ontwikkeling van de CB-NL
Gericht op personen die werkzaam zijn bij ARCADIS en bekend en/ of werkzaam zijn met Systems Engineering

In het kader van de afstudeeropdracht: The added value of an OWL concept library in the development stage of the Systems Engineering process.

Bob van Thiel
In samenwerking met de Technische Universiteit van Eindhoven en ARCADIS.

Datum: *03-03-2014*

Locatie: *ARCADIS Amersfoort*

Informatie respondent

Naam: *Dick Grotholt*

Functie: *Adviseur integraal ontwerp*

Als afstudeerder binnen ARCADIS doe ik onderzoek naar de implementatie van een OWL conceptenbibliotheek in de ontwikkelingsfase (volgens de Systems Engineering levenscyclus van INCOSE) van het Systems Engineering (SE) proces. Aan de hand van een case studie wil ik aandachtspunten binnen deze fase combineren met een OWL conceptenbibliotheek om te onderzoeken of deze bibliotheek een toegevoegde waarde kan leveren. Naast het bestuderen van literatuur zou ik graag een kort(e) interview/ vragenlijst willen afnemen om de belangrijke aandachtspunten en raakvlakken tussen SE en de CB-NL in kaart te brengen.

Om geen misverstanden te ondervinden heeft dit interview betrekking op de volgende scope:

- De SE methodiek (ISO 15288);
    - Het specificeren (top-down structuur) en structuren van informatie (op basis van de klanteisen);
    - Verificatieloop ontwerp met klant- en systeemeisen;
    - Toewijzen systeemeisen aan functioneel ontwerp;
    - Definiëren interne en externe raakvlakken functioneel ontwerp;
    - Verifiëren en valideren van het systeem.

De onderstaande vragen zijn voortgekomen uit de bestudeerde literatuur over SE en de CB-NL en interne gesprekken. Deze informatie dient als basis voor dit interview.

1a. Welke processen binnen de systeem ontwikkelingsfase ondervinden problemen en/ of knelpunten met het uitwisselen van digitale informatie en/ of communicatie?

Een groot probleem is denk ik om aan het begin van een proces om alle impliciete informatie als expliciete informatie te verwerken in SE input.

Lange discussies doen zich voor om een objectenboom op te bouwen. Een goede structuur vinden van de objectenboom is lastig vanwege het complexe project (in dit geval sprekend over project De Zuidas). Tijdrovend in de opstartfase van een project.

1b. Zou de CB-NL met haar eenduidige beschrijving van concepten een rol kunnen betekenen om deze problemen en/ of knelpunten te verbeteren?

Bij het maken van een objectenboom zou de CB-NL van toegevoegde waarde zijn als het voor gestandaardiseerde afspraken (i.p.v. bilaterale afspraken) kan zorgen hoe objecten zich verhouden tot elkaar. Dit zou theoretische discussies (met name op fundamenteel niveau) over de opbouw van een objectenboom bij elk project kunnen wegnemen.

2. Naast vragen 1a en 1b, wat vindt u belangrijke aandachtspunten/ processen binnen de systeem ontwikkelingsfase om de CB-NL toe te passen/ testen? Graag toelichten waarom.

Aandachtspunt is, vanuit een voorbeeld verteld, dat sommige basisspecificaties niet op elkaar aansluiten. Met een Rijkswaterstaat project sloten de basisspecificaties van een tunnel niet aan op die van een weg.

3. Verwacht u een toegevoegde waarde van de CB-NL, met name in de systeem ontwikkelingsfase van het SE proces? Waarom wel/ niet?

De grootste winst verwacht ik om gestandaardiseerde objecten te definiëren en makkelijker een objectenboom op te bouwen met de CB-NL.

Verder kan de CB-NL hopelijk ook voor een benchmark zorgen. Dat projecten met name op een hogere structuur met elkaar vergeleken kunnen worden.

4. Helpt de CB-NL bij de verdere standaardisering en automatisering van SE processen? (Bijvoorbeeld: is validatie mogelijk op basis van het semantisch web van CB-NL?)

-

5. Overige punten.

Binnen de werkgroep GWW waar ik een aantal sessies heb meegedraaid was er bij iedereen veel onduidelijkheid hoe het precies in elkaar stak en hoe het proces moest gaan verlopen.

Ik kom bij de CB-NL niet tegen dat grenzen/ scope worden aangegeven van een bepaald object. Het object ''weg'' kan bij iedere partij uiteindelijk hetzelfde gedefinieerd zijn, maar bij mij is ''weg'' het asfalt inclusief de berm en bij de andere partij alleen het asfalt.

Interviewprotocol betreft de ontwikkeling van de CB-NL
Gericht op personen die werkzaam zijn bij ARCADIS en bekend en/ of werkzaam zijn met Systems Engineering

In het kader van de afstudeeropdracht: The added value of an OWL concept library in the development stage of the Systems Engineering process.

Bob van Thiel
In samenwerking met de Technische Universiteit van Eindhoven en ARCADIS.

Datum: *03-03-2014*

Locatie: *ARCADIS Amersfoort*

Informatie respondent

Naam: *Ivo Walta*

Functie: *Adviseur SE*

Als afstudeerder binnen ARCADIS doe ik onderzoek naar de implementatie van een OWL conceptenbibliotheek in de ontwikkelingsfase (volgens de Systems Engineering levenscyclus van INCOSE) van het Systems Engineering (SE) proces. Aan de hand van een case studie wil ik aandachtspunten binnen deze fase combineren met een OWL conceptenbibliotheek om te onderzoeken of deze bibliotheek een toegevoegde waarde kan leveren. Naast het bestuderen van literatuur zou ik graag een kort(e) interview/ vragenlijst willen afnemen om de belangrijke aandachtspunten en raakvlakken tussen SE en de CB-NL in kaart te brengen.

Om geen misverstanden te ondervinden heeft dit interview betrekking op de volgende scope:

- De SE methodiek (ISO 15288);
    - Het specificeren (top-down structuur) en structuren van informatie (op basis van de klanteisen);
    - Verificatieloop ontwerp met klant- en systeemeisen;
    - Toewijzen systeemeisen aan functioneel ontwerp;
    - Definiëren interne en externe raakvlakken functioneel ontwerp;
    - Verifiëren en valideren van het systeem.

De onderstaande vragen zijn voortgekomen uit de bestudeerde literatuur over SE en de CB-NL en interne gesprekken. Deze informatie dient als basis voor dit interview.

1a. Welke processen binnen de systeem ontwikkelingsfase ondervinden problemen en/ of knelpunten met het uitwisselen van digitale informatie en/ of communicatie?

Niet zozeer problemen, maar een bibliotheek kan van toegevoegde waarde zijn om projecten op te zetten. Vandaar dat momenteel de bibliotheek van SEm wordt ontwikkeld.

1b. Zou de CB-NL met haar eenduidige beschrijving van concepten een rol kunnen betekenen om deze problemen en/ of knelpunten te verbeteren?

Een gevalideerde taxonomie-laag zou een mooie ambitie zijn. Dit leidt tot een set gestandaardiseerde keuzes.
Echter, de vraag is: wie bepaalt de inhoud van deze taxonomie in CB-NL? En waarop is deze bepaling
gebaseerd? Deze spelregels waren voor mij onduidelijk, waardoor ik ben afgehaakt bij de testgroep GWW in
2012.

2. Naast vragen 1a en 1b, wat vindt u belangrijke aandachtspunten/ processen binnen de systeem ontwikkelingsfase om de CB-NL toe te passen/ testen? Graag toelichten waarom.

Kan ik de taxonomie binnen CB-NL gebruiken om te de-componeren? Wanneer neem ik iets over vanuit de
CB-NL en wanneer hang ik mijn eigen project specifieke functies en objecten aan die van de CB-NL?
Als wij binnen ARCADIS een SEm bibliotheek hebben, moet deze ergens aansluiten op de CB-NL. waar ligt
deze verhouding en waar en hoe sluiten wij als ARCADIS aan?

Voor mij is een onderzoeksvraag al van belang of de CB-NL haar ambities kan waarmaken. Als hierop al een gedeeltelijk antwoord op gevonden kan worden hebben wij hier allemaal wat aan.

We kunnen een experiment uitvoeren met de content van de CB-NL in onze eigen SEm. Wat belangrijk is is om de stappen die zijn genomen te vergelijken met de ambities van de CB-NL. Dit om de grijze vlakken in kaart te brengen. Lopen de ambities van de CB-NL parallel aan de ontwikkelingen?

Aandachtspunt bij een case studie kan zijn om het uit te voeren met menselijke interventie (actieve handeling/ ingreep om problemen op te lossen) en dat te vergelijken waar CB-NL eigenlijk een geautomatiseerde rol zou moet spelen. Wat moet er dan nog door de CB-NL aan gedaan worden en aandacht aan besteed worden? Hier heeft de CB-NL ook wat aan.

3. Verwacht u een toegevoegde waarde van de CB-NL, met name in de systeem ontwikkelingsfase van het SE proces? Waarom wel/ niet?

De CB-NL kan zeker van toegevoegde waarde zijn. Alleen het moet duidelijk zijn hoe. Momenteel zie ik geen
spelregels. Is de ambitie van de CB-NL om een generieke taxonomie te leveren, dan ben ik al blij. Als ik maar
weet hoe ik hiermee om moet gaan. Of zal het leveren van een generieke taxonomie niet het doel zijn van de
CB-NL?

Het proces zal ook sneller moeten verlopen.

4. Helpt de CB-NL bij de verdere standaardisering en automatisering van SE processen? (Bijvoorbeeld: is validatie mogelijk op basis van het semantisch web van CB-NL?)

   Het uiteindelijke doel moet het automatisch valideren zijn. Het proces verloopt dan sneller. Echter, dit
   kunnen wij nog lang niet realiseren met CB-NL.

   Iedereen snapt de ambitie alleen de weg ernaartoe, die is nog onduidelijk.

5. Overige punten.

   Kees Woestenenk heeft Semantic Concepts opgericht. Is het wel het juiste om hiermee verder te gaan als je als ambitie hebt om een gemeenschappelijke taal te ontwikkelen? Breid je nu niet uit op zijn visie? Dit is positief kritisch bekeken, want ik sta wel open voor het principe.

   Blijf niet blind zoeken naar de afspraken binnen de CB-NL op een hoger niveau, die vind je niet.

   Een conclusie van de literatuurstudie kan zijn hoeveel en waar open einden er zitten m.b.t. de CB-NL.

Interviewprotocol betreft de ontwikkeling van de CB-NL
Gericht op personen die werkzaam zijn bij ARCADIS en bekend en/ of werkzaam zijn met Systems Engineering

In het kader van de afstudeeropdracht: The added value of an OWL concept library in the development stage of the Systems Engineering process.

Bob van Thiel
In samenwerking met de Technische Universiteit van Eindhoven en ARCADIS.

Datum: *06-03-2014*

Locatie: *ARCADIS Amersfoort*

Informatie respondent

Naam: *Rik Opgenoort*

Functie: *Informatie manager*

Als afstudeerder binnen ARCADIS doe ik onderzoek naar de implementatie van een OWL conceptenbibliotheek in de ontwikkelingsfase (volgens de Systems Engineering levenscyclus van INCOSE) van het Systems Engineering (SE) proces. Aan de hand van een case studie wil ik aandachtspunten binnen deze fase combineren met een OWL conceptenbibliotheek om te onderzoeken of deze bibliotheek een toegevoegde waarde kan leveren. Naast het bestuderen van literatuur zou ik graag een kort(e) interview/ vragenlijst willen afnemen om de belangrijke aandachtspunten en raakvlakken tussen SE en de CB-NL in kaart te brengen.

Om geen misverstanden te ondervinden heeft dit interview betrekking op de volgende scope:

- De SE methodiek (ISO 15288);
    - Het specificeren (top-down structuur) en structuren van informatie (op basis van de klanteisen);
    - Verificatieloop ontwerp met klant- en systeemeisen;
    - Toewijzen systeemeisen aan functioneel ontwerp;
    - Definiëren interne en externe raakvlakken functioneel ontwerp;
    - Verifiëren en valideren van het systeem.

De onderstaande vragen zijn voortgekomen uit de bestudeerde literatuur over SE en de CB-NL en interne gesprekken. Deze informatie dient als basis voor dit interview.

1a. Welke processen binnen de systeem ontwikkelingsfase ondervinden problemen en/ of knelpunten met het uitwisselen van digitale informatie en/ of communicatie?

Het opbouwen van een objectenboom kost veel moeite en tijd, omdat iedere betrokkene een eigen visie op de objectenboom met haar objecten heeft. Definities komen niet overeen. Dit komt zowel extern (met de opdrachtgever) als intern (binnen ARCADIS) voor.

1b. Zou de CB-NL met haar eenduidige beschrijving van concepten een rol kunnen betekenen om deze problemen en/ of knelpunten te verbeteren?

Ja, als de CB-NL regels opstelt op een hoog niveau (abstract niveau) van objecten, kunnen deze discussies niet meer voorkomen.

2. Naast vragen 1a en 1b, wat vindt u belangrijke aandachtspunten/ processen binnen de systeem ontwikkelingsfase om de CB-NL toe te passen/ testen? Graag toelichten waarom.

Naast het realiseren van een objectenboom is het valideren van mijn eigen object met die van de CB-NL erg belangrijk. De objectenboom valideren aan de hand van de taxonomie (theorie van overerving) in de CB-NL. Hoe verloopt deze validatie?

3. Verwacht u een toegevoegde waarde van de CB-NL, met name in de systeem ontwikkelingsfase van het SE proces? Waarom wel/ niet?

Ja, een topstructuur creëren voor alle bronnen is van toegevoegde waarde. CB-NL zou ook regels moeten opstellen voor de scope van objecten. Of de CB-NL inderdaad deze scope/ geografische ruimte meeneemt is nog de vraag. Hopelijk komen hier regels voor.

4. Helpt de CB-NL bij de verdere standaardisering en automatisering van SE processen?

Bijvoorbeeld: is validatie mogelijk op basis van het semantisch web van CB-NL?

Het helpt bij het sneller doorlopen van het SE proces en op termijn met het valideren van de objecten die worden opgebouwd.

5. Overige punten.

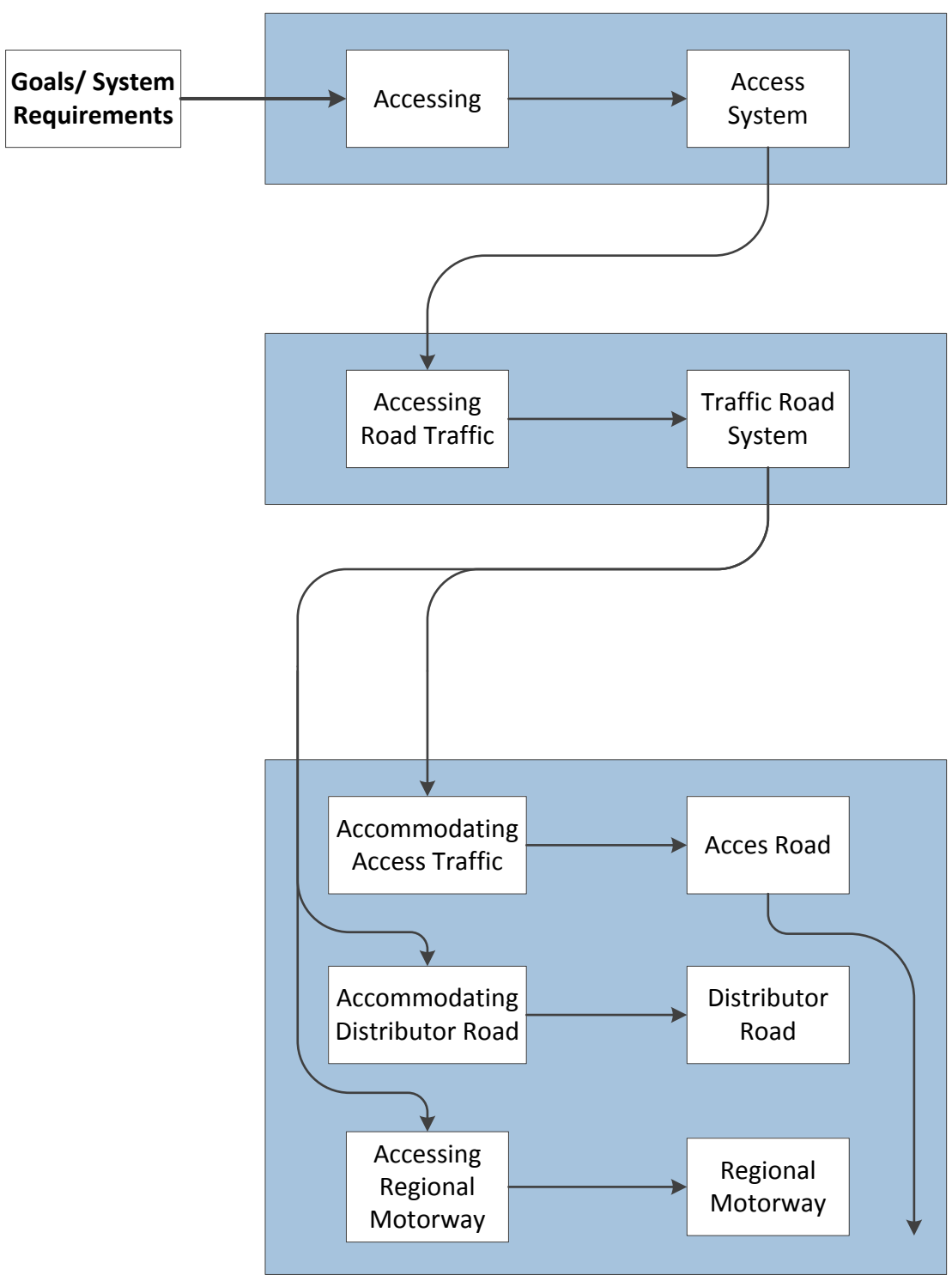
OTL (waar ik aan meewerk) is opgebouwd in Relatics, net als SEm.

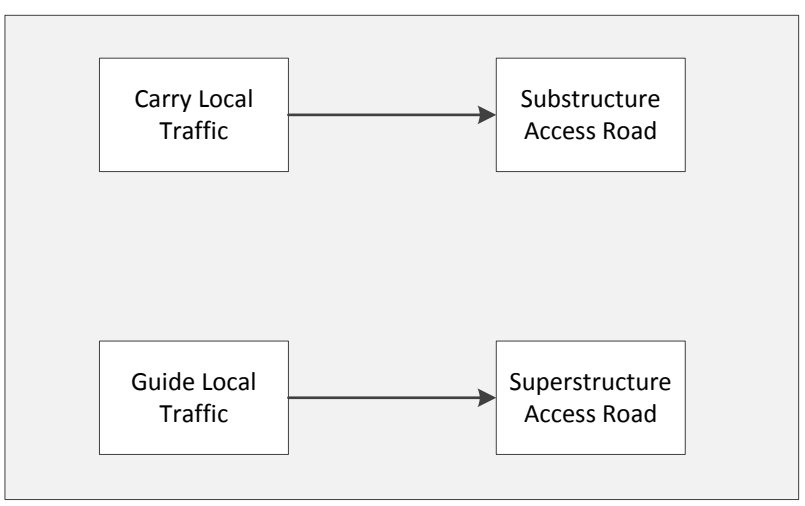
Software interfaces worden ontweken met de CB-NL. Er wordt makkelijk overheen gestapt, terwijl dit belangrijke aspecten zijn om verder te ontwikkelen.

Systemenboom linken met een andere systemenboom levert problemen op. Iedereen heeft zijn of haar eigen kijk erop. Een classificatiesysteem als CB-NL kan hierbij goed helpen, omdat iedereen op dit systeem zijn of haar eigen boom gaat baseren.

Appendix D – Scheme decomposition

Functional Breakdown Structure	System Breakdown Structure	Classes (Turtle syntax)	Decomposition level





1

```
:Accessing rdf:type owl:Class ;

        owl:equivalentClass [ rdf:type owl:Restriction ;
                              owl:onProperty :functionOf ;
                              owl:someValuesFrom :Access_System
                            ] ;

        rdfs:subClassOf :Function .
```

2

```
:Accessing_Road_Traffic rdf:type owl:Class ;

        owl:equivalentClass [ rdf:type owl:Restriction ;
                              owl:onProperty :functionOf ;
                              owl:someValuesFrom :Traffic_Road_System
                            ] ;

        rdfs:subClassOf :Accessing ,
                        [ rdf:type owl:Restriction ;
                          owl:onProperty :functionOf ;
                          owl:someValuesFrom :Access_System
                        ] .
```

3

```
:Accommodating_Access_Road_Traffic rdf:type owl:Class ;

        owl:equivalentClass [ rdf:type owl:Restriction ;
                              owl:onProperty :functionOf ;
                              owl:someValuesFrom :Access_Road
                            ] ;

        rdfs:subClassOf :Accommodating_Road_Traffic ,
                        [ rdf:type owl:Restriction ;
                          owl:onProperty :functionOf ;
                          owl:someValuesFrom :Traffic_Road_System
                        ] .
```

```
:Accommodating_Distributor_Road_Traffic rdf:type owl:Class ;

        owl:equivalentClass [ rdf:type owl:Restriction ;
                              owl:onProperty :functionOf ;
                              owl:someValuesFrom :Distributor_Road
                            ] ;

        rdfs:subClassOf :Accommodating_Road_Traffic ,
                        [ rdf:type owl:Restriction ;
                          owl:onProperty :functionOf ;
                          owl:someValuesFrom :Traffic_Road_System
                        ] .
```

```
:Accommodating_Regional_Motorway_Traffic rdf:type owl:Class ;

        owl:equivalentClass [ rdf:type owl:Restriction ;
                              owl:onProperty :functionOf ;
                              owl:someValuesFrom :Regional_Motorway
                            ] ;

        rdfs:subClassOf :Accommodating_Road_Traffic ,
                        [ rdf:type owl:Restriction ;
                          owl:onProperty :functionOf ;
                          owl:someValuesFrom :Traffic_Road_System
                        ] .
```

4

```
:Carry_Local_Traffic rdf:type owl:Class ;

        owl:equivalentClass [ rdf:type owl:Restriction ;
                              owl:onProperty :functionOf ;
                              owl:someValuesFrom :Substructure_Acces_Road
                            ] ;

        rdfs:subClassOf :Function ,
                        [ rdf:type owl:Restriction ;
                          owl:onProperty :functionOf ;
                          owl:someValuesFrom :Access_Road
                        ] .
```

```
:Guide_Local_Traffic rdf:type owl:Class ;

        owl:equivalentClass [ rdf:type owl:Restriction ;
                              owl:onProperty :functionOf ;
                              owl:someValuesFrom :Superstructure_Acces_Road
                            ] ;

        rdfs:subClassOf :Function ,
                        [ rdf:type owl:Restriction ;
                          owl:onProperty :functionOf ;
                          owl:someValuesFrom :Access_Road
                        ] .
```

Appendix E - Script

```
@prefix : <http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix xml: <http://www.w3.org/XML/1998/namespace> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@base <http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2> .

<http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2> rdf:type owl:Ontology .


#################################################################
#
#    Object Properties
#
#################################################################


###  http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#functionOf

:functionOf rdf:type owl:ObjectProperty ,
                owl:TransitiveProperty ;

        owl:inverseOf :hasFunction ;

        owl:equivalentProperty :hasObject ;

        rdfs:range owl:Thing ;

        rdfs:domain owl:Thing .



###  http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#hasFunction

:hasFunction rdf:type owl:ObjectProperty ,
                owl:TransitiveProperty ;

         owl:equivalentProperty :objectOf ;
```

```
        rdfs:domain owl:Thing ;

        rdfs:range owl:Thing .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#hasObject

:hasObject rdf:type owl:ObjectProperty ,
              owl:TransitiveProperty ;

       rdfs:range owl:Thing ;

       rdfs:domain owl:Thing .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#is_engineering_structure

:is_engineering_structure rdf:type owl:FunctionalProperty ,
                          owl:ObjectProperty ;

                  rdfs:range :Engineering_Structure ;

                  rdfs:domain owl:Thing .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#objectOf

:objectOf rdf:type owl:ObjectProperty ,
             owl:TransitiveProperty ;

      owl:inverseOf :hasObject ;

      rdfs:domain owl:Thing ;

      rdfs:range owl:Thing .


#################################################################
```

```
#
#    Data properties
#
#################################################################


###  http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-
2#hasMarking

:hasMarking rdf:type owl:DatatypeProperty ;

          rdfs:domain :Road ;

          rdfs:range xsd:boolean .



###  http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-
2#hasSpeedLimit

:hasSpeedLimit rdf:type owl:DatatypeProperty ,
                    owl:FunctionalProperty ;

            rdfs:domain :Road ;

            rdfs:range xsd:integer .



###  http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-
2#hasWidth

:hasWidth rdf:type owl:DatatypeProperty ;

        rdfs:range xsd:integer ;

        rdfs:domain owl:Thing .




#################################################################
#
#    Classes
#
#################################################################
```

```
### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Access_Road

:Access_Road rdf:type owl:Class ;

        owl:equivalentClass [ rdf:type owl:Restriction ;
                              owl:onProperty :hasSpeedLimit ;
                              owl:hasValue 60
                            ] ;

        rdfs:subClassOf :Road ,
                        [ rdf:type owl:Restriction ;
                          owl:onProperty :hasFunction ;
                          owl:someValuesFrom :Accommodating_Access_Road_Traffic
                        ] .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Access_System

:Access_System rdf:type owl:Class ;

        owl:equivalentClass [ rdf:type owl:Restriction ;
                              owl:onProperty :hasFunction ;
                              owl:someValuesFrom :Accessing
                            ] ;

        rdfs:subClassOf :System .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Accessing

:Accessing rdf:type owl:Class ;

        owl:equivalentClass [ rdf:type owl:Restriction ;
                              owl:onProperty :functionOf ;
                              owl:someValuesFrom :Access_System
                            ] ;

        rdfs:subClassOf :Function .
```

```
### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Accessing_Road_Traffic

:Accessing_Road_Traffic rdf:type owl:Class ;

        owl:equivalentClass [ rdf:type owl:Restriction ;
                  owl:onProperty :functionOf ;
                  owl:someValuesFrom :Traffic_Road_System
                ] ;

        rdfs:subClassOf :Accessing ,
                [ rdf:type owl:Restriction ;
                  owl:onProperty :functionOf ;
                  owl:someValuesFrom :Access_System
                ] .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Accommodating

:Accommodating rdf:type owl:Class ;

      rdfs:subClassOf :Function .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Accommodating_Access_Road_Traffic

:Accommodating_Access_Road_Traffic rdf:type owl:Class ;

            owl:equivalentClass [ rdf:type owl:Restriction ;
                      owl:onProperty :functionOf ;
                      owl:someValuesFrom :Access_Road
                    ] ;

            rdfs:subClassOf :Accommodating_Road_Traffic ,
                    [ rdf:type owl:Restriction ;
                      owl:onProperty :functionOf ;
                      owl:someValuesFrom :Traffic_Road_System
                    ] .
```

```
### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Accommodating_Distributor_Road_Traffic

:Accommodating_Distributor_Road_Traffic rdf:type owl:Class ;

                    owl:equivalentClass [ rdf:type owl:Restriction ;
                                owl:onProperty :functionOf ;
                                owl:someValuesFrom :Distributor_Road
                              ] ;

                    rdfs:subClassOf :Accommodating_Road_Traffic ,
                            [ rdf:type owl:Restriction ;
                              owl:onProperty :functionOf ;
                              owl:someValuesFrom :Traffic_Road_System
                            ] .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Accommodating_National_Motorway_Traffic

:Accommodating_National_Motorway_Traffic rdf:type owl:Class ;

                    owl:equivalentClass [ rdf:type owl:Restriction ;
                                owl:onProperty :functionOf ;
                                owl:someValuesFrom :National_Motorway
                              ] ;

                    rdfs:subClassOf :Accommodating_Road_Traffic ,
                            [ rdf:type owl:Restriction ;
                              owl:onProperty :functionOf ;
                              owl:someValuesFrom :Traffic_Road_System
                            ] .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Accommodating_Regional_Motorway_Traffic

:Accommodating_Regional_Motorway_Traffic rdf:type owl:Class ;

                    owl:equivalentClass [ rdf:type owl:Restriction ;
                                owl:onProperty :functionOf ;
                                owl:someValuesFrom :Regional_Motorway
                              ] ;

                    rdfs:subClassOf :Accommodating_Road_Traffic ,
```

```
[ rdf:type owl:Restriction ;
  owl:onProperty :functionOf ;
  owl:someValuesFrom :Traffic_Road_System
] .
```

http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Accommodating_Road_Traffic

```
:Accommodating_Road_Traffic rdf:type owl:Class ;

        rdfs:subClassOf :Accommodating .
```

http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Activity

```
:Activity rdf:type owl:Class .
```

http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Aquaduct

```
:Aquaduct rdf:type owl:Class ;

    rdfs:subClassOf :Engineering_Structure .
```

http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Bicycle_Track

```
:Bicycle_Track rdf:type owl:Class ;

        rdfs:subClassOf :Road .
```

http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Bridging_System

```
:Bridging_System rdf:type owl:Class ;

        rdfs:subClassOf :System .
```

```
### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Carry_Local_Traffic

:Carry_Local_Traffic rdf:type owl:Class ;

        owl:equivalentClass [ rdf:type owl:Restriction ;
                  owl:onProperty :functionOf ;
                  owl:someValuesFrom :Substructure_Acces_Road
                ] ;

        rdfs:subClassOf :Carry_Road_Traffic ,
                [ rdf:type owl:Restriction ;
                  owl:onProperty :functionOf ;
                  owl:someValuesFrom :Access_Road
                ] .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Carry_Road_Traffic

:Carry_Road_Traffic rdf:type owl:Class ;

        rdfs:subClassOf :Function .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Civil_Engineering_Work

:Civil_Engineering_Work rdf:type owl:Class .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Construction_Site_Road

:Construction_Site_Road rdf:type owl:Class ;

          rdfs:subClassOf :Road .
```

```
### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Crossing_System

:Crossing_System rdf:type owl:Class ;

        owl:equivalentClass [ rdf:type owl:Restriction ;
                              owl:onProperty :hasObject ;
                              owl:someValuesFrom :Viaduct
                            ] ;

        rdfs:subClassOf :System .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Distributor_Road

:Distributor_Road rdf:type owl:Class ;

        owl:equivalentClass [ rdf:type owl:Restriction ;
                              owl:onProperty :hasSpeedLimit ;
                              owl:hasValue 80
                            ] ;

        rdfs:subClassOf :Road ,
                        [ rdf:type owl:Restriction ;
                          owl:onProperty :hasFunction ;
                          owl:someValuesFrom :Accommodating_Distributor_Road_Traffic
                        ] .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Element

:Element rdf:type owl:Class .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Engineering_Structure

:Engineering_Structure rdf:type owl:Class ;

            rdfs:subClassOf :Civil_Engineering_Work .
```

http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#False

:False rdf:type owl:Class .

http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Fauna_Passage

:Fauna_Passage rdf:type owl:Class ;

rdfs:subClassOf :Engineering_Structure ;

owl:disjointWith :Tunnel .

http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Fly-over

:Fly-over rdf:type owl:Class ;

rdfs:subClassOf :Viaduct .

http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Footpath

:Footpath rdf:type owl:Class ;

rdfs:subClassOf :Road .

http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Function

:Function rdf:type owl:Class ;

rdfs:subClassOf :Activity .

```
### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Gas_Station

:Gas_Station rdf:type owl:Class ;

        rdfs:subClassOf [ rdf:type owl:Restriction ;
                owl:onProperty :hasFunction ;
                owl:someValuesFrom :Threading
              ] .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Guide_Local_Traffic

:Guide_Local_Traffic rdf:type owl:Class ;

            owl:equivalentClass [ rdf:type owl:Restriction ;
                    owl:onProperty :functionOf ;
                    owl:someValuesFrom :Superstructure_Acces_Road
                  ] ;

            rdfs:subClassOf :Function ,
                [ rdf:type owl:Restriction ;
                  owl:onProperty :functionOf ;
                  owl:someValuesFrom :Access_Road
                ] .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Guide_Road_Traffic

:Guide_Road_Traffic rdf:type owl:Class ;

            owl:equivalentClass [ rdf:type owl:Restriction ;
                    owl:onProperty :functionOf ;
                    owl:someValuesFrom :Road_Markings
                  ] ;

            rdfs:subClassOf :Function ,
                [ rdf:type owl:Restriction ;
                  owl:onProperty :functionOf ;
                  owl:someValuesFrom :Information_System
                ] .
```

```
### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Horse_Track

:Horse_Track rdf:type owl:Class ;

        rdfs:subClassOf :Road .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Information_System

:Information_System rdf:type owl:Class ;

            rdfs:subClassOf :System ,
                    [ rdf:type owl:Restriction ;
                      owl:onProperty :hasFunction ;
                      owl:someValuesFrom :Informing
                    ] .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Informing

:Informing rdf:type owl:Class ;

       rdfs:subClassOf :Function .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Infrastructure_Element

:Infrastructure_Element rdf:type owl:Class ;

                rdfs:subClassOf :Element .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Integrating

:Integrating rdf:type owl:Class ;

        owl:equivalentClass [ rdf:type owl:Restriction ;
```

```
                    owl:onProperty :functionOf ;
                    owl:someValuesFrom :Integration_System
                  ] ;

        rdfs:subClassOf :Function ,
                [ rdf:type owl:Restriction ;
                  owl:onProperty :functionOf ;
                  owl:someValuesFrom :Access_System
                ] .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Integration_System

:Integration_System rdf:type owl:Class ;

            owl:equivalentClass [ rdf:type owl:Restriction ;
                                  owl:onProperty :hasFunction ;
                                  owl:someValuesFrom :Integrating
                                ] ;

            rdfs:subClassOf :System .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#National_Motorway

:National_Motorway rdf:type owl:Class ;

            owl:equivalentClass [ rdf:type owl:Restriction ;
                                  owl:onProperty :hasSpeedLimit ;
                                  owl:hasValue 120
                                ] ;

            rdfs:subClassOf :Road ,
                    [ rdf:type owl:Restriction ;
                      owl:onProperty :hasFunction ;
                      owl:someValuesFrom :Accommodating_National_Motorway_Traffic
                    ] .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Noise_Barrier
```

```
:Noise_Barrier rdf:type owl:Class ;

        rdfs:subClassOf :Engineering_Structure ,
                [ rdf:type owl:Restriction ;
                  owl:onProperty :hasFunction ;
                  owl:someValuesFrom :Reducing_Noise
                ] ;

        owl:disjointWith :Road_Traffic_Viaduct ,
                 :Tunnel .



### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Outside_the_Urban_Area

:Outside_the_Urban_Area rdf:type owl:Class ;

            owl:equivalentClass [ rdf:type owl:Restriction ;
                        owl:onProperty :hasSpeedLimit ;
                        owl:someValuesFrom [ rdf:type rdfs:Datatype ;
                                    owl:onDatatype xsd:integer ;
                                    owl:withRestrictions ( [ xsd:minExclusive 50
                                                ]
                                              )
                                  ]
                      ] ;

            rdfs:subClassOf :Space .



### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Rail_Bar_Structure

:Rail_Bar_Structure rdf:type owl:Class ;

            rdfs:subClassOf :Traffic_Connection .



### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Railway_Viaduct

:Railway_Viaduct rdf:type owl:Class ;

            rdfs:subClassOf :Viaduct .
```

```
### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Reducing_Noise

:Reducing_Noise rdf:type owl:Class ;

        owl:equivalentClass [ rdf:type owl:Restriction ;
                              owl:onProperty :functionOf ;
                              owl:someValuesFrom :Noise_Barrier
                            ] ;

        rdfs:subClassOf :Function ,
                        [ rdf:type owl:Restriction ;
                          owl:onProperty :functionOf ;
                          owl:someValuesFrom :Integration_System
                        ] .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Regional_Motorway

:Regional_Motorway rdf:type owl:Class ;

        owl:equivalentClass [ rdf:type owl:Restriction ;
                              owl:onProperty :hasSpeedLimit ;
                              owl:hasValue 100
                            ] ;

        rdfs:subClassOf :Road ,
                        [ rdf:type owl:Restriction ;
                          owl:onProperty :hasFunction ;
                          owl:someValuesFrom :Accommodating_Regional_Motorway_Traffic
                        ] .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Road

:Road rdf:type owl:Class ;

    rdfs:subClassOf :Road_Traffic_Element .
```

```
### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Road_Element

:Road_Element rdf:type owl:Class ;

        rdfs:subClassOf :Infrastructure_Element .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Road_Markings

:Road_Markings rdf:type owl:Class ;

        rdfs:subClassOf :Road_Element ,
                [ rdf:type owl:Restriction ;
                  owl:onProperty :hasFunction ;
                  owl:someValuesFrom :Guide_Local_Traffic
                ] .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Road_Signage

:Road_Signage rdf:type owl:Class ;

        owl:equivalentClass [ rdf:type owl:Restriction ;
                  owl:onProperty :hasFunction ;
                  owl:someValuesFrom :Guide_Road_Traffic
                ] ;

        rdfs:subClassOf :Road_Element ,
                [ rdf:type owl:Restriction ;
                  owl:onProperty :hasFunction ;
                  owl:someValuesFrom :Informing
                ] .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Road_Traffic_Element

:Road_Traffic_Element rdf:type owl:Class ;

            rdfs:subClassOf :Infrastructure_Element .
```

```
### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Road_Traffic_Viaduct

:Road_Traffic_Viaduct rdf:type owl:Class ;

        rdfs:subClassOf :Viaduct .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Space

:Space rdf:type owl:Class .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Substructure_Acces_Road

:Substructure_Acces_Road rdf:type owl:Class ;

        owl:equivalentClass [ rdf:type owl:Restriction ;
                              owl:onProperty :hasFunction ;
                              owl:someValuesFrom :Carry_Local_Traffic
                            ] ;

        rdfs:subClassOf :Road_Element .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Superstructure_Acces_Road

:Superstructure_Acces_Road rdf:type owl:Class ;

        owl:equivalentClass [ rdf:type owl:Restriction ;
                              owl:onProperty :hasFunction ;
                              owl:someValuesFrom :Guide_Local_Traffic
                            ] ;

        rdfs:subClassOf :Road_Element .
```

```
### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#System

:System rdf:type owl:Class .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Threading

:Threading rdf:type owl:Class ;

        rdfs:subClassOf :Function ,
                [ rdf:type owl:Restriction ;
                  owl:onProperty :functionOf ;
                  owl:someValuesFrom :Integration_System
                ] .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Traffic_Connection

:Traffic_Connection rdf:type owl:Class .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Traffic_Road_System

:Traffic_Road_System rdf:type owl:Class ;

                owl:equivalentClass [ rdf:type owl:Restriction ;
                                      owl:onProperty :hasFunction ;
                                      owl:someValuesFrom :Accessing_Road_Traffic
                                    ] ;

                rdfs:subClassOf :System .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Tunnel

:Tunnel rdf:type owl:Class ;

      rdfs:subClassOf :Engineering_Structure .
```

```
### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Viaduct

:Viaduct rdf:type owl:Class ;

    rdfs:subClassOf :Engineering_Structure ,
                    :Traffic_Connection .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Within_the_Urban_Area

:Within_the_Urban_Area rdf:type owl:Class ;

            owl:equivalentClass [ rdf:type owl:Restriction ;
                        owl:onProperty :hasSpeedLimit ;
                        owl:someValuesFrom [ rdf:type rdfs:Datatype ;
                                    owl:onDatatype xsd:integer ;
                                    owl:withRestrictions ( [ xsd:maxInclusive 50
                                                ]
                                               )
                                  ]
                      ] ;

            rdfs:subClassOf :Space .


#################################################################
#
#   Individuals
#
#################################################################


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#A9

:A9 rdf:type owl:NamedIndividual ;

   :hasSpeedLimit 120 .
```

```
### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Kanaalweg_Oost

:Kanaalweg_Oost rdf:type owl:NamedIndividual ;

        :hasSpeedLimit 80 .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Kapellaan

:Kapellaan rdf:type owl:NamedIndividual ;

      :hasSpeedLimit 60 .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Kooibrug

:Kooibrug rdf:type owl:NamedIndividual ,
            [ rdf:type owl:Restriction ;
              owl:onProperty :is_engineering_structure ;
              owl:onClass :Road_Traffic_Viaduct ;
              owl:qualifiedCardinality "1"^^xsd:nonNegativeInteger
            ] ,
            [ rdf:type owl:Restriction ;
              owl:onProperty :is_engineering_structure ;
              owl:someValuesFrom :Noise_Barrier
            ] .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#N100

:N100 rdf:type owl:NamedIndividual ;

    :hasSpeedLimit 100 .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-2#Test_Decomposition_1
```

```
:Test_Decomposition_1 rdf:type :Accommodating_Access_Road_Traffic ,
                        owl:NamedIndividual ;

                :functionOf :Test_Decomposition_2 .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-
2#Test_Decomposition_2

:Test_Decomposition_2 rdf:type owl:NamedIndividual .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-
2#Test_SWRL_1

:Test_SWRL_1 rdf:type :Tunnel ,
                owl:NamedIndividual .


### http://www.semanticweb.org/bobvanthiel/ontologies/2014/5/untitled-ontology-
2#Test_SWRL_2

:Test_SWRL_2 rdf:type owl:NamedIndividual ;

          :hasSpeedLimit 100 .


#################################################################
#
#   Rules
#
#################################################################


<urn:swrl#x> rdf:type <http://www.w3.org/2003/11/swrl#Variable> .
<urn:swrl#y> rdf:type <http://www.w3.org/2003/11/swrl#Variable> .
[ rdf:type <http://www.w3.org/2003/11/swrl#Imp> ;
  <http://www.w3.org/2003/11/swrl#head> [ rdf:type
<http://www.w3.org/2003/11/swrl#AtomList> ;
                              rdf:rest rdf:nil ;
```

```
                    rdf:first [ rdf:type <http://www.w3.org/2003/11/swrl#ClassAtom> ;
                            <http://www.w3.org/2003/11/swrl#classPredicate> :False ;
                            <http://www.w3.org/2003/11/swrl#argument1> <urn:swrl#x>
                          ]
                  ] ;
  <http://www.w3.org/2003/11/swrl#body> [ rdf:type
<http://www.w3.org/2003/11/swrl#AtomList> ;
                    rdf:rest [ rdf:type <http://www.w3.org/2003/11/swrl#AtomList> ;
                            rdf:first [ rdf:type
<http://www.w3.org/2003/11/swrl#DatavaluedPropertyAtom> ;
                                    <http://www.w3.org/2003/11/swrl#propertyPredicate>
:hasSpeedLimit ;
                                    <http://www.w3.org/2003/11/swrl#argument1>
<urn:swrl#x> ;
                                    <http://www.w3.org/2003/11/swrl#argument2>
<urn:swrl#y>
                                  ] ;
                            rdf:rest [ rdf:type
<http://www.w3.org/2003/11/swrl#AtomList> ;
                                    rdf:rest rdf:nil ;
                                    rdf:first [ rdf:type
<http://www.w3.org/2003/11/swrl#BuiltinAtom> ;
                                            <http://www.w3.org/2003/11/swrl#builtin>
<http://www.w3.org/2003/11/swrlb#notEqual> ;
                                            <http://www.w3.org/2003/11/swrl#arguments> [
rdf:type rdf:List ;
                                                    rdf:first <urn:swrl#y> ;
                                                    rdf:rest [ rdf:type rdf:List
;
                                                            rdf:first 60 ;
                                                            rdf:rest rdf:nil
                                                          ]
                                                  ]
                                          ]
                                  ]
                          ] ;
                    rdf:first [ rdf:type <http://www.w3.org/2003/11/swrl#ClassAtom> ;
                            <http://www.w3.org/2003/11/swrl#classPredicate>
:Access_Road ;
                            <http://www.w3.org/2003/11/swrl#argument1> <urn:swrl#x>
                          ]
                  ]
] .
[ rdf:type <http://www.w3.org/2003/11/swrl#Imp> ;
  <http://www.w3.org/2003/11/swrl#body> [ rdf:type
<http://www.w3.org/2003/11/swrl#AtomList> ;
                    rdf:rest [ rdf:type <http://www.w3.org/2003/11/swrl#AtomList> ;
```

```
                                    rdf:first [ rdf:type
<http://www.w3.org/2003/11/swrl#DatavaluedPropertyAtom> ;
                                          <http://www.w3.org/2003/11/swrl#propertyPredicate>
:hasSpeedLimit ;
                                          <http://www.w3.org/2003/11/swrl#argument1>
<urn:swrl#x> ;
                                          <http://www.w3.org/2003/11/swrl#argument2>
<urn:swrl#y>
                                        ] ;
                                    rdf:rest [ rdf:type
<http://www.w3.org/2003/11/swrl#AtomList> ;
                                          rdf:rest rdf:nil ;
                                          rdf:first [ rdf:type
<http://www.w3.org/2003/11/swrl#BuiltinAtom> ;
                                                <http://www.w3.org/2003/11/swrl#builtin>
<http://www.w3.org/2003/11/swrlb#notEqual> ;
                                                <http://www.w3.org/2003/11/swrl#arguments> [
rdf:type rdf:List ;
                                                                          rdf:first <urn:swrl#y> ;
                                                                          rdf:rest [ rdf:type rdf:List
;
                                                                                rdf:first 80 ;
                                                                                rdf:rest rdf:nil
                                                                              ]
                                                                        ]
                                                ]
                                          ]
                                    ] ;
                              rdf:first [ rdf:type <http://www.w3.org/2003/11/swrl#ClassAtom> ;
                                    <http://www.w3.org/2003/11/swrl#classPredicate>
:Distributor_Road ;
                                    <http://www.w3.org/2003/11/swrl#argument1> <urn:swrl#x>
                                  ]
                            ] ;
  <http://www.w3.org/2003/11/swrl#head> [ rdf:type
<http://www.w3.org/2003/11/swrl#AtomList> ;
                              rdf:rest rdf:nil ;
                              rdf:first [ rdf:type <http://www.w3.org/2003/11/swrl#ClassAtom> ;
                                    <http://www.w3.org/2003/11/swrl#classPredicate> :False ;
                                    <http://www.w3.org/2003/11/swrl#argument1> <urn:swrl#x>
                                  ]
                            ]
] .
[ rdf:type <http://www.w3.org/2003/11/swrl#Imp> ;
  <http://www.w3.org/2003/11/swrl#body> [ rdf:type
<http://www.w3.org/2003/11/swrl#AtomList> ;
                              rdf:rest [ rdf:type <http://www.w3.org/2003/11/swrl#AtomList> ;
```

```
                                    rdf:rest [ rdf:type
<http://www.w3.org/2003/11/swrl#AtomList> ;
                                          rdf:rest rdf:nil ;
                                          rdf:first [ rdf:type
<http://www.w3.org/2003/11/swrl#BuiltinAtom> ;
                                                <http://www.w3.org/2003/11/swrl#builtin>
<http://www.w3.org/2003/11/swrlb#notEqual> ;
                                                <http://www.w3.org/2003/11/swrl#arguments> [
rdf:type rdf:List ;
                                                                          rdf:first <urn:swrl#y> ;
                                                                          rdf:rest [ rdf:type rdf:List
;
                                                                                  rdf:first 100 ;
                                                                                  rdf:rest rdf:nil
                                                                                ]
                                                                        ]
                                                ]
                                          ] ;
                                    rdf:first [ rdf:type
<http://www.w3.org/2003/11/swrl#DatavaluedPropertyAtom> ;
                                           <http://www.w3.org/2003/11/swrl#propertyPredicate>
:hasSpeedLimit ;
                                           <http://www.w3.org/2003/11/swrl#argument1>
<urn:swrl#x> ;
                                           <http://www.w3.org/2003/11/swrl#argument2>
<urn:swrl#y>
                                          ]
                                     ] ;
                              rdf:first [ rdf:type <http://www.w3.org/2003/11/swrl#ClassAtom> ;
                                       <http://www.w3.org/2003/11/swrl#classPredicate>
:Regional_Motorway ;
                                       <http://www.w3.org/2003/11/swrl#argument1> <urn:swrl#x>
                                      ]
                            ] ;
  <http://www.w3.org/2003/11/swrl#head> [ rdf:type
<http://www.w3.org/2003/11/swrl#AtomList> ;
                              rdf:rest rdf:nil ;
                              rdf:first [ rdf:type <http://www.w3.org/2003/11/swrl#ClassAtom> ;
                                       <http://www.w3.org/2003/11/swrl#classPredicate> :False ;
                                       <http://www.w3.org/2003/11/swrl#argument1> <urn:swrl#x>
                                      ]
                            ]
] .
[ rdf:type <http://www.w3.org/2003/11/swrl#Imp> ;
  <http://www.w3.org/2003/11/swrl#head> [ rdf:type
<http://www.w3.org/2003/11/swrl#AtomList> ;
                              rdf:rest rdf:nil ;
```

```
                    rdf:first [ rdf:type <http://www.w3.org/2003/11/swrl#ClassAtom> ;
                            <http://www.w3.org/2003/11/swrl#classPredicate> :False ;
                            <http://www.w3.org/2003/11/swrl#argument1> <urn:swrl#x>
                          ]
                  ] ;
  <http://www.w3.org/2003/11/swrl#body> [ rdf:type
<http://www.w3.org/2003/11/swrl#AtomList> ;
                    rdf:rest [ rdf:type <http://www.w3.org/2003/11/swrl#AtomList> ;
                            rdf:first [ rdf:type
<http://www.w3.org/2003/11/swrl#DatavaluedPropertyAtom> ;
                                    <http://www.w3.org/2003/11/swrl#propertyPredicate>
:hasSpeedLimit ;
                                    <http://www.w3.org/2003/11/swrl#argument1>
<urn:swrl#x> ;
                                    <http://www.w3.org/2003/11/swrl#argument2>
<urn:swrl#y>
                                  ] ;
                            rdf:rest [ rdf:type
<http://www.w3.org/2003/11/swrl#AtomList> ;
                                    rdf:rest rdf:nil ;
                                    rdf:first [ rdf:type
<http://www.w3.org/2003/11/swrl#BuiltinAtom> ;
                                            <http://www.w3.org/2003/11/swrl#builtin>
<http://www.w3.org/2003/11/swrlb#notEqual> ;
                                            <http://www.w3.org/2003/11/swrl#arguments> [
rdf:type rdf:List ;
                                                    rdf:first <urn:swrl#y> ;
                                                    rdf:rest [ rdf:type rdf:List
;
                                                            rdf:first 120 ;
                                                            rdf:rest rdf:nil
                                                          ]
                                                  ]
                                          ]
                                  ]
                          ] ;
                    rdf:first [ rdf:type <http://www.w3.org/2003/11/swrl#ClassAtom> ;
                            <http://www.w3.org/2003/11/swrl#classPredicate>
:National_Motorway ;
                            <http://www.w3.org/2003/11/swrl#argument1> <urn:swrl#x>
                          ]
                  ]
] .

###  Generated by the OWL API (version 3.4.2) http://owlapi.sourceforge.net
```