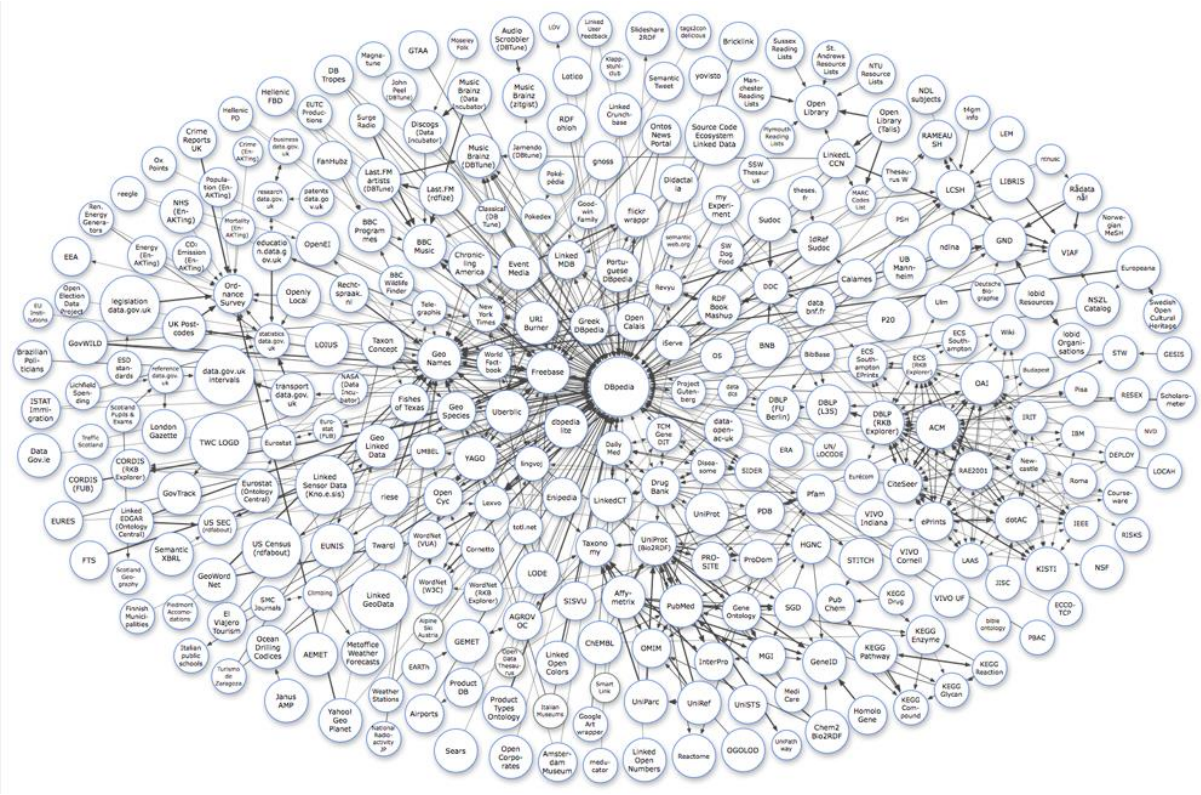
Ontology Base Requirement Management For Project Standardization

System Engineering For Infrastructure Using Semantic Web Technologies



MSc Construction Management & Engineering

Benjamin Herinckx, (b.herinckx@student.tue.nl)

Eindhoven University of Technology

Department of the Built Environment

Eindhoven, April 11th, 2017

COLOFON

Title	Ontology based requirement management for project standardization
Sub Title	SE for Infrastructure Using Semantic Web Technologies

Name	**Benjamin Herinckx**
Student #	0928261
Telephone	06-46710527
E-mail	b.herinckx@student.tue.nl

University	**Technische Universiteit Eindhoven**
Faculty	The Department of the Built Environment
Education	MSc Construction Management & Engineering

Project Company	**Rijkswaterstaat**
Work Group	MultiWaterWerk

Date	11-04-2017
Status	Master Thesis
Version	Final

GRADUATION COMMITTEE

Chairman:	**Professor Bauke de Vries**
1st Supervisor:	**Ass. Prof. Jakob Beetz**
2nd Supervisor:	**PhD Thomas Krijnen**
3rd Supervisor:	**Ing. Mick Baggen**

TU/e Technische Universiteit Eindhoven University of Technology

Where innovation starts

FOREWORD

A life-time of learning, innovations, challenges and rewards; that is the promise of the engineering life. This work is the sturdiest bridge I will ever build, the one joining years of training, exams and report writing to a career of construction and planning for the future. The connection between research and practice is never forgotten and the future of engineering lies within the collaboration between research and industry. The report you are about to read is the conclusion of a Master in Construction Management and Engineering at the Eindhoven University of Technology. It was carried out as an internship at Rijkswaterstaat within the MultiWaterWerk work group.

My passion for construction started when I was younger and I have since given myself the tools and ambition to succeed through higher education. In line with my personal development objectives, I seized opportunities to study abroad because I believe the future of engineering does not limit itself to borders. I have also carried multiple internships in order to contextualize my studies and relate my work to real professional practices. Altogether I am pleased with my academic path and I have plenty of hope for my future engineering career.

None of my achievements would have been possible on my own. I would like to take this opportunity to express my gratitude to Prof. Bauke de Vries, Ass. Prof. Jakob Beetz, Ass. Prof Qi Han and through them all the teachers that have helped me learn throughout university, high school and earlier. I am also grateful for the help and opportunity offered by Ir. Frans van Dam, Ir. Mick Baggen and all the people who have helped me seize the scope of expertise that the construction world offers through my internships. Last but not least, my family, friends and girlfriend who have offered me continuous support each to the full of their capacity. I hope that someday I will be able to repay this help and that my actions amount to a greater good.

ABSTRACT

In the context of building information modelling technologies more and more data is produced to represent the complex system of systems structures in architecture, engineering and construction domain projects. More and more, the issues of interoperability, accessibility and semantic meaning arise as complex analysis tools are used to complement the human expertise. The semantic web and its corresponding open standards and frameworks for data modelling has created opportunities to use an ontology driven approach for data modelling which promises greater expressivity and flexibility. This report shows that with careful considerations of the data structure the OWL logic vocabulary and SPIN rule notation can be used to elaborate a set of standardized project requirements.

SUMMARY

The adoption of digital design and business solutions has largely contributed to the development of safer and more efficient construction practices in the past. Innovations continue to affect the everyday environment of project managers. Commissioning bodies look to improve the quality of their requirement bills, architects to better communicate their designs, contractors adapt their construction methods, suppliers offer tailored solutions while asset managers have greater control over their project information.

Technology is the answer, the ICT domain has become a great deal of the construction industry and with its help the time, location and economic constraints have been shifted. Modern projects are ever, faster, safer, cheaper and more sustainable. The industry is charging ever forward and carries multitudes of initiatives. The multiplicity in systems is the reflection of the complexity and variety of the construction industry fields of expertise. The differentiation between the various construction titles, contract types and jobs (design, steel frame, concreted frame, cladding, piling, earthwork, drainage, testing, insurance, finance) resonate through the different approaches to define a standard model.

The different local initiatives have led to a conundrum where modelling has become a tool to standardize and align information between parties while the diversity of models renders the interoperability between models complex. The resulting trend is the advent of a need for an industry wide standard model. All seem to have found a common interest and placed their hopes in the Industry Foundation Class (IFC) standardization effort. This construction schema for geometric product is able to communicate complex 3D product geometry information between agents.

This leap forward has brought unparalleled interoperability between CAD type software and thus between design expertise. This interoperability is the first step of BIM adaptation. However, the future of BIM is to expand the interoperability to all project information (object properties, bill of requirements, bill of quantity, maintenance schedule, operation logs). The structuring and processing of all the project information falls into the scope of system engineering. Information that was once only available through stacks of printed documents will soon be centralized in a shared network type of database.

This work shows that, semantic web technologies can be implemented to develop project based ontologies to standardize soft information (requirements, activities). Furthermore, the reasoning capabilities of these technologies can be used to validate and verify a set of project requirements and constraints following best system engineering practices. The data is structured using RDF and semantically enriched using OWL description logic as well as SPARQL inferencing notation rules.

The rules are attached to defined classes that represent engineering products and systems in a standardized format. These rules can be used to run boolean tests and assign validation and verification statuses. Experimentation has produced encouraging results with the validation of all asserted information. Verification on the other hand is not always directly possible and requires further developments and intermediate processes. Beyond its functionality, the innovative use of RDF/OWL is the universality of its schema and the accessibility of the data structured as a network.

This brief exercise is a window on the opportunities that modelling efforts may bring to the construction industry and the performance improvements that quality design will bring to project products. This research domain is one that will yield many practical application in the industry and that will bear many fruits in the near future.

STATEMENT OF ORIGINALITY AND OWNERSHIP

This work is a submission for the completion of a MSc in Construction Management & Engineering. I hereby certify that I am the sole author of this thesis and that no part of this thesis has been published or submitted for publication.

I certify that, to the best of my knowledge, my thesis does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with standard referencing practices.

Table of Contents

1. INTRODUCTION

This chapter is intended to present the background of the research work as well as the everyday practice environment in which it is carried out. The idea is to introduce the concepts that this report elaborates on and to direct the reader towards the key questions/issues related to this research area.

1.1 Research Motivation

Rijkswaterstaat (RWS) is the engineering department of the Dutch Ministry of Infrastructure and Environment. As such, it assumes many critical missions in the construction industry. Its leading roles are that of Project Commissioner (PC) and Asset Manager (AM). Over the last centuries, RWS has been commissioning civil engineering works such as highways, railways, bridges, canals and dams for the development and maintenance of the Dutch territory. Because of the many environmental challenges that The Netherlands are facing, large scale infrastructure networks have proven to be a life-saving necessity over the centuries. Over the span of various regime, governments and nationalities, the RWS body has evolved to become the public supervising body for infrastructure constructions.

Throughout different periods, the governance role of RWS has varied but it has always maintained a strong academic interest for civil engineering and project management. Nowadays, RWS is a department of the Ministry and as such plays the role of 'administrator' for works of prime national interest. This position does not involve the physical construction of the works but rather the administration of complex multi-disciplinary projects as well as the day to day operations of critical infrastructure. As a result, RWS is bound to deal with information, communication and operation management.

Over the complete life-cycle of construction projects, a tremendous amount of information is produced. From the onset, the government is required to produce survey reports to identify the nation's infrastructural needs. Once identified, the RWS engineers need to assign

responsibilities and elaborate commissioning processes to develop a global project strategy. After a succession of topographical studies, commissioning the works to private contractors involves the draft of multiple legal documents in parallel with environmental and feasibility studies. The design and conception stages focus on the development of comprehensive requirements that ensure product quality and reliability as well as the lowest maintenance and operation costs. When RWS as AM takes on the responsibility of operating and maintaining these infrastructures directly or in collaboration with third parties, performance, incidents and operation information needs to be recorded. At last, scheduled maintenance works iterates through most of the above stages adding to the initial project information.

For administrative and maintenance reasons, project information should be fully accessible as well as fully operable in order to manage changes and new inputs. More and more, the role of asset manager requires state of the art information management technology. Key information management aspects are digital storage of data in smart databases, viewing and editing capabilities as well as access management. In parallel of smart information environments, the data itself is becoming smarter. Data can now be associated to definitions and semantic meanings as well as relate to other data in many various ways. All the technologies arising in the field of information management in the construction industry find their roots in a major shift to building information modelling (BIM). These developments help develop collaboration and preserve the trail of information in large scale projects.

RWS has always been at the centre of innovation for the construction industry and has immediately jumped on board the BIM development of the market. As of today, a couple of example projects have been developed using BIM and proto-BIM technologies (RWS 2017). On top of its technical know-how, RWS wants to become the expert leader of construction data management in The Netherlands. This new corporate strategy will be achieved through the development of construction technologies and the development of corresponding innovative work processes. From a corporate point of view the benefits are twofold, the innovation will bring a great leap forward in terms of work performance as well as a dominant market position.

1.2 MultiWaterWerk

In engineering, RWS is internationally recognized for the successful management of fresh- and seawater systems such as dams, canals, navigation locks, dredging and drainage systems (RWS 2016). The country engineers have become experts in hydraulic engineering by necessity. Throughout the national network, the water level and the ship navigation is regulated by locks. Individually these locks represent a node in a network between stretches of water. While the design of these locks is variable, they all serve the same fundamental functions (Rijkswaterstaat 2012):

- Protect from high-waters: the lock resists incoming water from high sea levels and extreme tide events.
- Manage water level: the lock helps maintain a constant upstream water height through the physical barrier of the lock door.
- Manage water discharge: the lock valves allow a constant water discharge volume to be released downstream of the system.
- Segregate fresh and sea waters: a physical barrier to help separate the inflow of sea waters and outflow of fresh waters.
- Accommodate ship navigation: passage and elevation between different water levels.

These fundamental functions help manage the natural flow of water. It is a critical task both in terms of the severity of failure and the complex and unpredictable nature of its dependency on environmental factors. Modern AM use mathematical models and expert meteorology to predict the behaviour of the tamed waters. Overall, the day to day running of the Dutch waterways is a great success and has helped tremendously to protect the citizens and develop the economy. The role of navigation locks is so fundamental indeed, that RWS has dedicated an entire department, the ‘Sluis Programma’, to monitor, study, develop and innovate on that front. The Dutch navigation lock expertise is certainly a national added value sparkling Dutch initiatives all throughout the world such as the largest navigation lock in the world, the Kieldrecht lock (Port of Antwerp 2016).

One of the most innovative initiative of the Sluis Programma is the MultiWaterWerk (MWW) work group. It was set up as a research task force to think the navigation lock of the future

(Rijkswaterstaat 2015). This corporate strategy comes about as 52 navigation locks throughout the Dutch water network are about to reach their theoretical life span between 2020-2040. Many of them are still fully operational but it is projected that the intensification of waterway traffic and the advent of larger ships will render these locks obsolete and/or inappropriate. It falls onto MWW to overcome the critical challenges that will arise in the near future from this ageing infrastructure.

Through modernisation of the navigation system arises the challenge of standardisation (Rijkswaterstaat 2016). Historically, all locks were designed individually which has led to a lot of variability in the system. This variability in the construction is a burden for the life cycle management. It not only requires individual running processes, operator training and maintenance, but also requires specific information storage and expertise captured as local knowledge. The opportunity arises to optimise the performance of the life-cycle costs (LCC) as the aging locks are replaced. This is achieved by devising a common framework for all the 52 projects (Rijkswaterstaat 2015). The result of a thorough and systematic overhaul of the intrinsic business processes and the use of modern building technologies. Projects are split in two conceptual group of information the static parts of the process that can be standardised and are shared between all projects. The dynamic part that is specific to single projects (local topography, environment). Whether static or dynamic, all project information is to be recorded along a single project-wide data structure.

Beyond the replacement of ageing lock, the main deliverable of MWW is a modern approach to all infrastructure project management. The optimization and predictability of LCC goes hand in hand with the improvement of information traceability, increased process reliability and maximum operation performance. In this quest RWS does not stand out, many AM organizations share the same common interest. RWS seeks to improve its learning capacity and build upon its corporate added value by collaborating with all national and international parties as well as reaching out to local stakeholders (Rijkswaterstaat 2015).

Internally, MWW is a collaboration between different RWS work groups and project product lines. The core of the team is composed of design, SE and BIM engineers as well as asset managers and sustainability experts. The work groups are joined under the supervision of both a contract and a project manager in order to cover the entirety of the project scope.

1.3 Problem Statement

Imagine the following scenarios: a medium-size on-measure concrete solution supplier is looking to adopt BIM technologies to promote their flexibility and adaptiveness to production uncertainties; a large-scale infrastructure contractor wants to express the competitive advantage of a particularly environmental friendly construction method; the architect of a novel city-centre opera needs to convey the complexity of the internal acoustic design and the innovative choice of materials; after 30 years of operations the asset manager of a high-speed railway track is planning to upgrade all the direction switch systems. These four realistic scenario have a vested interest in the development of a smarter construction model which includes complex construction properties.

The roadmap for a smarter construction industry needs to answer the following questions: how to insure accessibility, traceability, interoperability and universality of meaning of advanced construction data? What structure, or schema, will non-geometric project properties follow? What process to validate and verify project information? How to check for consistency and completeness of the information? How to implement the BIM process in different business cases? How to insure the continuity of the BIM process in the future?

- **Outlook on the future of construction data management:**

Building Information Model approaches aim to share all project data centrally in a universally operable format. The end goal is to improve the project collaboration process and streamline the workflow. These goals are achieved through better project visualisation (i.e. 3D modelling), decomposition & aggregation (i.e. System Engineering), traceability and continuity of the information. BIM processes virtually create a network type of governance and improves stakeholder management (Eastman, et al. 2011). As a result, projects benefit from lower rework, optimized cost/quality ratios and improved sustainability considerations.

BIM is the result of a long term modelling effort with critical deliverables such as STEP standards in the industry. The main innovation are the Industry Foundation Classes (IFC) schema which have now reached their fourth major update (i.e. IFC4). Its application and the use of BIM technologies are spreading thanks to national and international initiatives. In the infrastructure domain however, as opposed to the building domain, BIM technology is still at

its infancy. The IFC schemas for infrastructure objects and properties are still to be developed. Additionally, the interoperability of para-construction fields such as urban planning and GIS agents within the scope of infrastructure project is also a hot research domain.

- **Collaboration centric process introduce project complexity:**

The use of BIM in infrastructure is rapidly growing due to client specifications and stakeholder benefits. Project progress is iterative and participative. However, the use of BIM is still largely limited to one-to-one relationships between participants as opposed to many-to-many relations and exchanges that should take place in full networks. This barrier to adaptation is partly due to technology with the need for more flexible exchange protocols and more accessible server-like application. However by large, the evolution towards a full network activity is mainly limited by the semantics of the information. In order to be able to apply a standard schema (i.e. semantic structure) to complex infrastructure works (roads, rails, water systems...) there needs to be a commonly accepted ontology description of the concepts involved. Many initiatives (bSDD, COINS, CB-NL, PPBIM) develop common expert dictionaries, thesaurus, taxonomies or object type libraries to unify construction models.

The construction definitions are not sufficient on their own. Also being developed for good operability and market penetration are system engineering based information system that record the description of objects, systems, requirements and functions in alignment with these expert ontologies. Most project information is product and business specific and therefore the optimization and performance of the information system lies within the companies. It is their internal domain knowledge that has to be aligned in a standard way. Nowadays, many product information datasets are structured as hierarchical breakdown-trees which limits the expressivity of the information. Indeed, these trees can only represent ‘subsumption’ and ‘has part’ relationships. These trees fail to highlight the dependencies of concepts that are shared and the connections and dependencies between parts and/or levels.

- **Requirement as key project information:**

There are multiple types and versions of requirements (i.e. input, model, change, functional). Depending on professional preferences participants have various requirements managing systems (i.e. recording, validation, verification, testing). In order to be standardised the

complexity of requirement engineering need to be aligned with construction schema through enterprise ontologies in order to insure the universality of a requirement semantics.

SE appears early on in construction projects. So soon in fact that not all projects will be carried out. Perhaps a project will be deemed unfeasible or perhaps an entity decides to produce a 'standard case' project which it will use later on as a basis for similar projects. Depending on the contract type, the stakeholders of a project at the development stage and those at the production stage might differ. Alternatively, given the iterative nature of SE multiple versions of the project model might coexist.

In order to assure the traceability of the different updates and changes to the requirement work, the project information system needs to record all actions and modifications made to the model as well as their contextual reason. Ultimately though, BIM is only concerned with 'useful and practical' data. Perhaps the most complex issue is merely knowing what information is deemed useful enough to be shared through BIM. These specifications need to be drafted in adapted information delivery manuals.

- **Improvements through digitalization and flexibility:**

The development of requirements is the first stage of a construction, yet they are of paramount importance all throughout the project. Indeed, at all stages of a project, objects and functions need to be verified and the requirements must be validated. The performance of the end result is a function of the similitude between the requirement draft and the design product. Different technologies allow different levels of operability for the requirements.

The current most widespread performance checking is the human validation process. This intuitive process is easily implemented and does not require any out of scope skills, however becoming an expert at requirement checking is complex to learn and requires experience. Its greatest benefit is that a pen and paper might suffice to validate or refute an argument which is resource-efficient. However, over the span of a large project, this process is very tedious. The validation and testing are all but uniform and as a result, some decisions might be biased. All-in-all, a poor project performance analysis will yield a poor product.

The central innovation of BIM is to introduce digital processes to all aspects of project management. In particular, requirement checking process is one target area that could potentially be transcribed through a software algorithm. It defies the first approach because it requires more non-core expertise and it also demands critical thinking regarding the structure and standardization of the information. However, just like its CAD counterparts, the digital process benefits from greater flexibility and rework mechanisms. If all requirements are structured in a logic and systematic approach, then modifying one requirement should not alter the system's capacity to process the requirements while manually a change in requirement would require tremendous analysis rework.

- **System engineering approach to BIM:**

System engineering and BIM are intertwined. Indeed, BIM can be seen as the overhaul of standard practices through SE. BIM shall be the result, while SE is the working tool. The advent of parametric design, has had an impressive change on CAD practices. These changes now need to be transposed to other 'design' disciplines. The SE definition of smarter data models could allow a parametric definition of soft BIM information. Standard functions, requirements and system breakdown trees could be connected through semantics while queries relate instance objects and the meta-class definition. Engineering properties could be semantically defined and standardized while analysis results could be shared in an open formats.

There are a few obstacles to the implementation of a systematic "model requirement checker" and the development of a web of engineering. First, the conceptual vocabularies used by the various AEC actors could lead to confusions. There remains to be established a common international language of the construction industry that unambiguously refers to project resources while at the same time maintains the expressivity of various expertise fields. Secondly, roles and function definitions, for example for a bridge element, might be perceived and/or communicated differently between engineers, architects and contractors.

In addition to purely linguistic issues, there are operability and connectivity issues. How to relate model diagrams and SE breakdown structures and how to evaluate the match or disparities between the two? When multiple design options are evaluated, the different models need to be referenced in a similar fashion. Referencing and assignment mechanisms are non-trivial and require in-depth investigations.

- **Complex properties and performance checking:**

Within the context of this work, performance defines the level of correspondence between a design solution and design requirements. As such, the requirements set out by the PM on behalf of the client entity are the measuring stones of performance. There are several ways of measuring performance in design because requirements can have different attributes. Requirements are sometime compulsory (i.e. pass/fail) while others might be optimization ones (i.e. distribution around a target), some are balance requirements (i.e. if this compensate there and do that) others are optional and left to the discretion of the proposal designer. By nature, requirements can be qualitative, for example architectural ones, while others are purely quantitative, such as engineering capacities. Finally, requirements differ greatly in the testing methodology (calculation notes, on-site tests, 3D rendering). This complexity contrasts tremendously with the IFC schema description of project properties.

Additionally, the IFC schema is aligned in part with the level of details development phases of projects granted that most performance analysis is carried out by the PM during the development baseline of the project. However, contractors and sub-contractors often have options on material providers and/or on building technology choices. Late project developments may arise and change requests can modify parts of the project beyond the geometric scope of the IFCs. Once again the specificity of the schema shows a lack of continuity and flexibility.

- **Techno-centric perspective of information management**

Project management is dictated by process issues, when is performance checking the most critical, how are decision making responsibilities delegated? Project managers may have in-depth knowledge of the client needs, however, contractors might decide based on their practical expertise, for example the use of generic or tailored products, or the choice of using precast or in-situ concrete.

Beyond these process considerations there are technologic ones. Construction and design are governed by a multitude of software solutions, all use proprietary languages and data-structures. How then, insure the control over project information, how to insure the meaning and traceability of the data and, how to develop interoperability of the information and, last but not least, how to maintain the human expertise and critical thinking capabilities over the process?

1.4 Research Context

MWW establishes itself as an opportunity to innovate on the long term. Historically, the infrastructure industry has not changed drastically over time but rather has shown a constant and slow evolution (Rijkswaterstaat 2015). This overhaul initiative aims to drastically cut loose of the established processes and allow itself greater degrees of freedom to develop a durable and efficient project management process. The corporate strategy is to distil the technical expertise and empiric knowledge through a more streamlined and effective business process. As such, it does not aim to re-engineer the navigation lock but rather to redesign the communication channels and information management around the engineering product.

The MWW approach answers a black-box need for performance and efficiency from the RWS management team. This modelling approach focuses on business functionality and the quality of the delivered service. This worthy corporate goal however, cannot be achieved without modifying the organizational and structural model (i.e. the white-box model). The white-box model focuses not on functional goals, but rather on structural goals and aims to achieve a better symbiosis between the products and the services (Dietz 2005). Quite naturally, the white-box innovation is driven by system and requirement engineering because they have the specificity of dealing with decomposition structures, requirement transcription and business processes.

Improvement starts with a self-reflection and a deep analysis of internal processes. At RWS, system engineering is establishing itself as the reference discipline to improve on performance (Rijkswaterstaat 2016). A typical RWS project spans from the ‘plan development’ phase onto the ‘contract preparation’ phase. The two major information exchanges from the projects, see Figure 1.4, are when RWS puts out a ‘vraagspecificatie’ document to the market in which it has recorded all the project requirements and constraints and when it gets back an ‘oplever dossier’ from the contractor at the end of the production phase which contains all the product and construction information as well as miscellaneous reports.

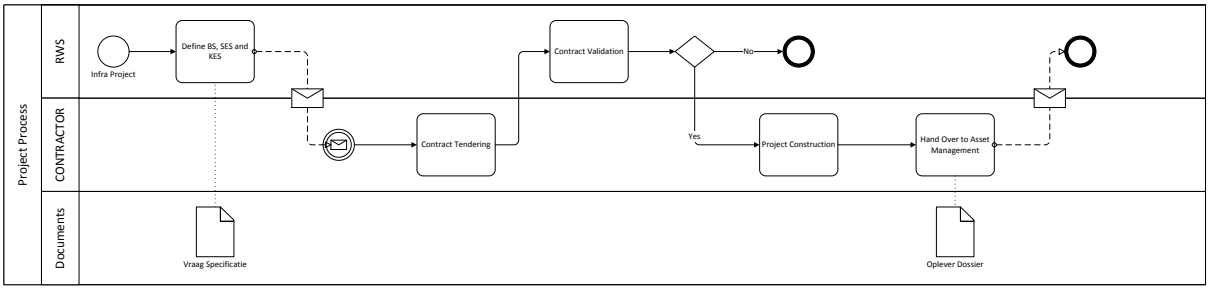


Figure 1.4 – Streamlined RWS Project Process

The greatest corporate challenge in terms of process innovation at RWS is dealing with project information in a smarter fashion. Because of their role duality (both client & asset manager) there is an intrinsic conundrum between the information that they deliver as a client entity, the information they receive from the contractors and the information they need as asset managers. The mismatch lies in the structure and the functionality of the information. Individually the three parties have different internal information structure that are best suited for their specific operations. This results from the organic needs of the independent organization to improve their own processes.

Beyond the measure of format and content, this challenge is a clear matter of systems engineering. The different entities are distinct parts in a more complex symposium system with communications, relations and dependencies between the parts. While RWS seeks to develop an internal SE process it also looks towards the integration of its own workflow into a greater collaboration meta-process. The enhanced process should augment the contribution of RWS and produce added value while in the meantime resolve the essential issues of continuity, consistency, coherency and granularity of the exchanged data.

2. RIJKSWATERSTAAT APPROACH TO DATA MANAGEMENT

This chapter outlines the current work process at RWS and covers a discussion of the pros and cons of the approach as well as a review of a few weak points. The key deliverable from this section is to mark a clear transition from the old ways to the new.

2.1 Requirement Based Design and Contracting

In design, all problems are described through requirements. A common saying is that requirements are the expression of what a customer has in mind (Hull, Jackson and Dick 2011). In essence, all project deliverables are a direct result of the requirement definition. Construction projects, more specifically, are centred on requirements. Indeed, the overall list of requirements is explicitly described in the initial project contract. It is therefore very accurate to describe construction as a requirement based activity.

A common paradox in the recording of requirements is that their intention, expression, interpretation and the delivering have different meaning based on the stakeholder. This is generally a result of natural language paper based documents which contain a large proportion of implicit or left to interpretation information. Requirements are so critical to good design that an entire science about how to manage the pre-construction stage of all projects has been developed over the year (Kossiakoff, Sweet, et al., Systems Engineering principles and practice 2011). SE is the methodology underlying the management of project knowledge and information according to project phases. The underlying thread of all information management is a mixture of ICT and requirement engineering.

The biggest challenges in requirement engineering are completeness and consistency (Siegemund, et al. 2011). In order to be universal, requirements need to be consistent, necessary, unambiguous, verifiable and complete. Completeness in itself is an unachievable

goal given they change over time. However, all projects should strive for complete information about the requirements structure (i.e. meta-model) rather than the requirement content.

At the core of SE lies 'how-to' guidelines regarding the drafting of requirements but as with every science more complex concepts have evolved. First the notion of concurrent design has introduced the complexity of involving different parties in a common effort to write down common project information. Concurrent design on its own has many risk factors: liability, traceability, compatibility, data integrity etc.

SE also covers the notion of project phasing and optimization. It has become common knowledge that as a project carries forward, the capacity for change reduces and therefore the cost of changes greatly increases, as shown in the McLeamy curve in Figure 2.1. Process analysis and workflow management techniques are used to minimize the intrinsic difficulties in network collaboration and to neutralize the effects of change requests. While it could be argued that on a higher level sudden changes in projects are a result of poor requirement analysis, these events can never completely be avoided an thus modern processes such as BIM need to incorporate their management.

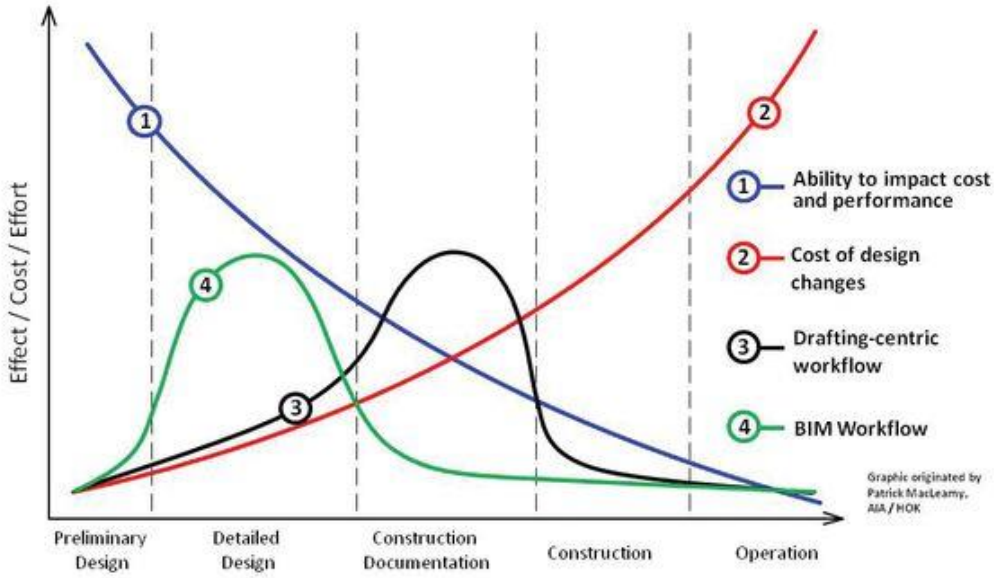


Figure 2.1 - McLeamy Curve

Another expert aspect of SE in the domain of the construction industry is the negotiation power of requirements. Most, if not all contracting is requirement based with the projects deliverables being described by functional requirements? Potential candidates for a given tendering process

typically base their estimates on a one-to-one mapping between the contract specification and units of production work (i.e. work packages).

On the one side, this natural contracting technique provides great visibility and a fair trade environment between competing actors. On the other side, this market process increases the economic risk of the client entity. Indeed, all requirements specified in the contract will be charged at a competitive rate while all 'extra-'requirements defined later will be negotiated at an arbitrary rate.

The cost of a poor requirement analysis can add up to a substantial level, so much so indeed, that many 'aborted' projects are a result of SE failure. Missing requirements can lead to time overruns and underestimated budgets (Siegemund, et al. 2011) while unnecessary ones will lead to additional and often futile workload. Initial errors in the capturing of requirement tend to propagate through entire systems and snowball to unmanageable proportions.

2.2 System & Requirement Engineering At RWS

RWS has a central role in all infrastructure works in The Netherlands and a key responsibility in the drafting of project requirements. More and more, over the years, RWS has shifted from a design-based (i.e. CAD) contractual process to a requirement based one (i.e. information). The modern drafting process at RWS heavily relies on SE and RE as their key development lines, see Figure 2.2. In order to optimize the quality - risk – functionality triangle of every project a lot of thinking effort is spent at the conception stage of the project. From a commissioning body's perspective, requirements are by far the greatest performance gainer.

In the case of sluice projects, RWS holds both title of project commissioner (PC) and asset manager (AM). As a result, RWS has a privileged perspective on these projects and has a greater technical knowledge about the design, operation and maintenance parts. However, the different phases of the project life-cycle are managed by different teams with specific objectives and performance indicators. A straightforward glance at the various project groups yields illustrative examples. The user/operator groups are concerned with the capacity of the sluice, the efficiency of the supporting systems (pumps, signals, motors) and the data management; the maintenance group is concerned with the durability of the materials,

accessibility to all parts of the design and repair schedules/ off-times; the environmental task force studies the impact of the product on the surrounding fauna, flora and the water quality etc. The specialization of the various groups is much appreciable since it provides greater expertise and depth of reasoning.

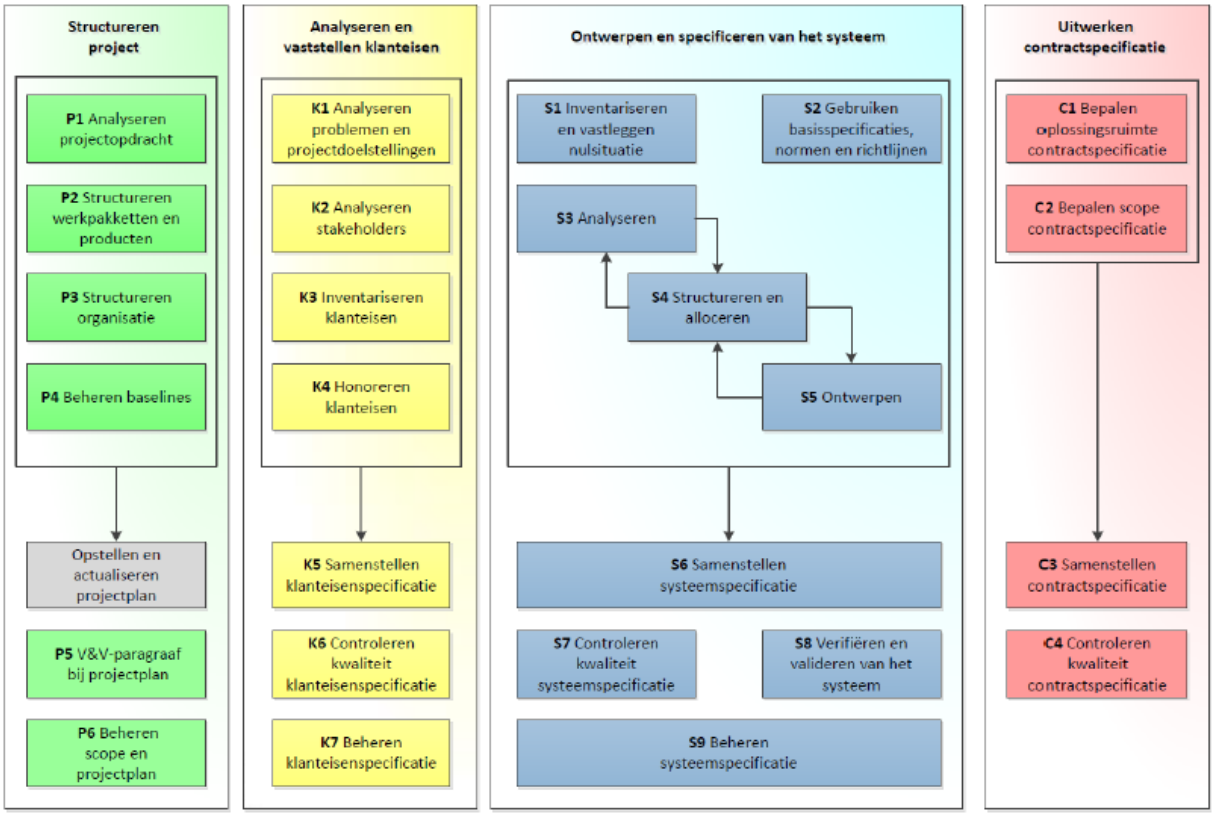


Figure 2.2 - RWS System Engineering Process (Rijkswaterstaat 2016)

However, the generation of all these oriented requirements may hinder the collaboration process through the complexity of the discussion and the production of redundant and/or contradictory information. This is where the structured approach of SE benefits the RWS groups. In particular, when it comes to information exchange the business processes are modelled on the V-shaped approach.

From Figure 2.2, it is clear that the different groups have different objectives. These individual objectives have for purpose to increase the performance of all the parts of the project. However, it is important to keep in mind the interaction of all the parts and specify the project as a whole. For this reason, a project is structured as a model. In the effort of standardizing, this model should contain the largest scope of project objects, functions and requirements.

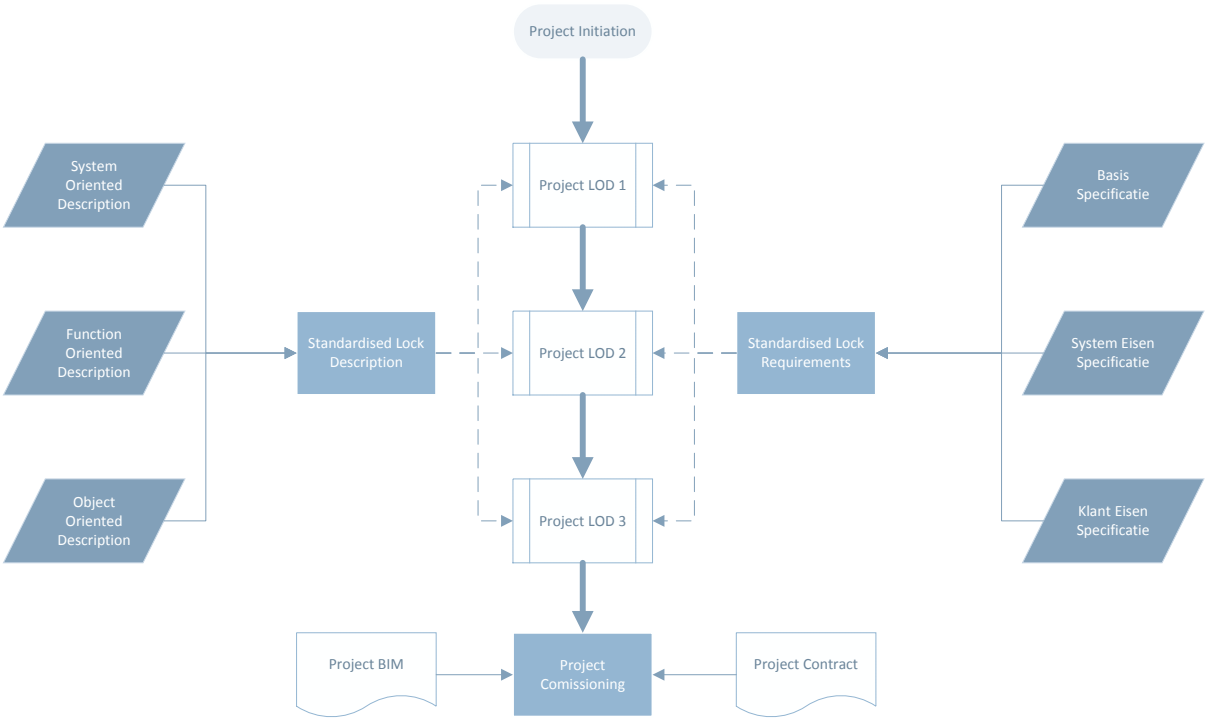


Figure 2.3 - Standard Project Development

These models are typically defined as tree hierarchies. Within the scope of standardizing a project process, see Figure 2.3, complexities arise when trying to link the object, function and requirement trees. Extensive system decompositions with many different levels of abstraction lead to granularity problems where information might not refer to the right abstraction level. It is not straightforward or sometime not even possible to assign a requirement to a single degree of abstraction in a hierarchy because the relationships and aggregation of the elements are not properly defined.

Additionally, requirements change over time, they may specialize or even contradict previous versions. This evolution is organic to the project and thus changes in terms of the object/system relationship to the requirements. One of the current problems is that the current tree hierarchy system does not account for the traceability of a particular requirement change, see Figure 2.4. The lack of requirement traceability is equivalent to the loss of design information. Indeed, when change requests are processed, typically, only the result of the requests is recorded. As time passes, the reason why the request had arisen is lost and the project is likely to undergo another cycle of troubleshooting in case a new change request is required. In the context of operation maintenance and performance, traceability is key.

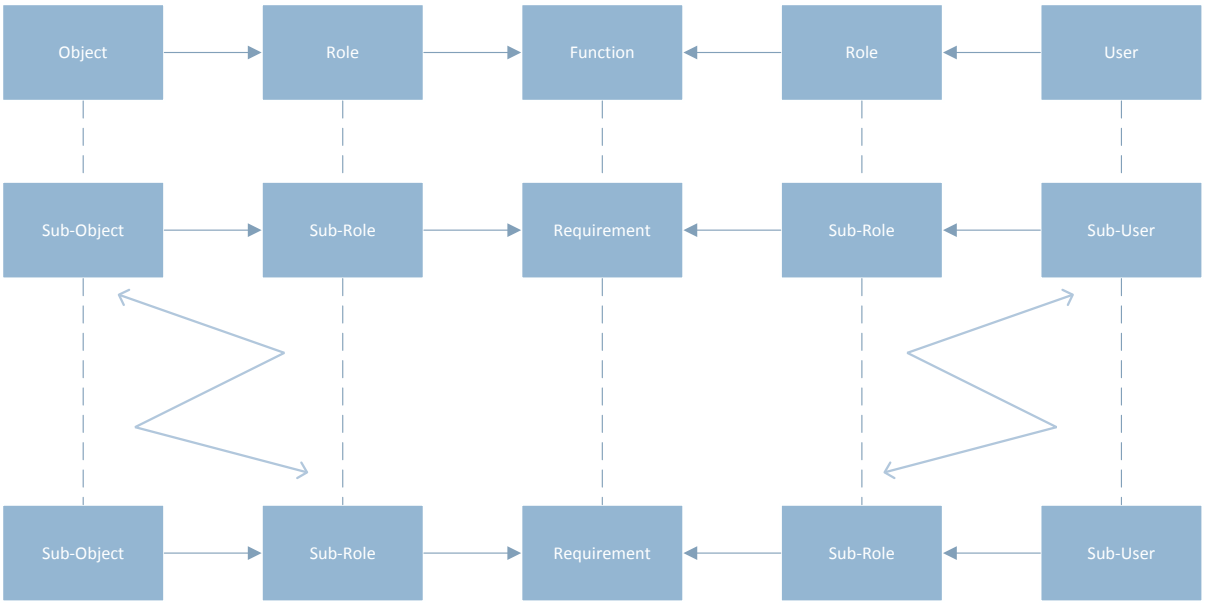


Figure 2.4 - SE Project Decomposition

2.3 System Decomposition of a Lock Project

As mentioned above, system engineering is the science behind requirements and has for main purpose to help translate client needs in terms of explicit requirements. It relies heavily on system decomposition (i.e. FBS, OBS, SBS) and modelling technology. The natural approach is to specify tree hierarchies of systems and parts in order to assign responsibilities between different working groups. In the current state, RWS project requirements are split in three different groups: 'basis specificatie' (BS), 'klanten eisen specificatie' (KES) and 'system eisen specificatie' (SES):

- SES are concerned with the functionalities and interoperability of systems (i.e. water discharge system, levelling system, emergency system).
- KES are concerned with the operability requirements of the asset manager perspective over the operation phase of the life-cycle (i.e. discharge speed, maintenance requirements, rate of opening).
- BS are concerned with the functionality of the different objects. The purpose of BS is to standardize as much of the functionalities as possible for a given project type (i.e. lock, dam, bridge) so as to minimize the amount of rework in between similar projects.

At the moment, there is a profound internal work being carried out to define a proper project decomposition. In the past, individual groups had their own project decomposition, see Appendix A. Beyond the issues on interoperability and consistency between the hierarchies, it was noted that the hierarchies had elements of different abstraction levels on a given modelling level (Dijkstra 2015). This issue had been solved by introducing a more transversal structure which considers all elements of a navigation lock project, see Figure 2.5.

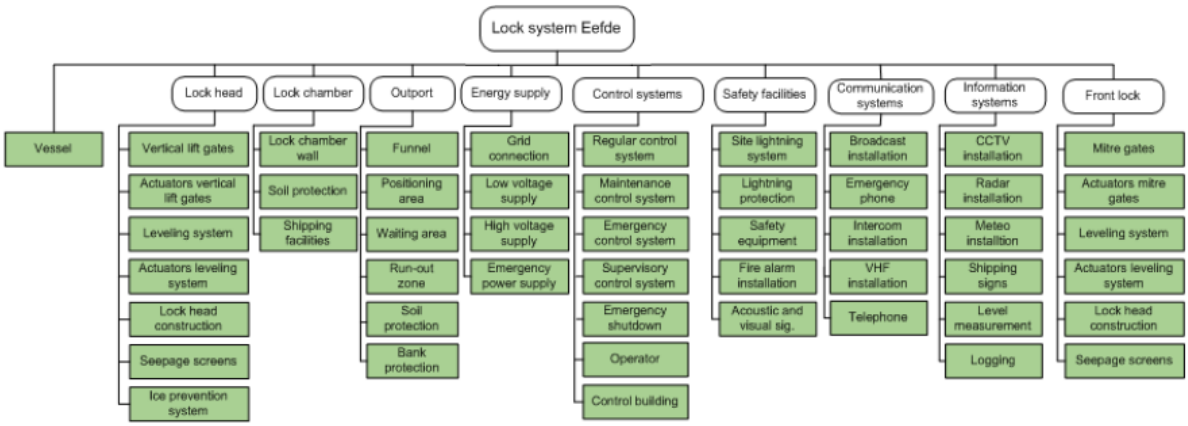


Figure 2.5 - Reviewed Lock Decomposition from (Dijkstra, 2015)

This new decomposition has the merit of being reviewed by the different services and valid according to the current standards (NEN 2767) which focus on civil components (Dijkstra 2015). However, this decomposition only represents a flagship model of a project, it does not develop the full complexity of the different components, objects and parts of any possible construction product. Moreover, its vertical structure does not show the horizontal relationship between the systems or the overlaps and intersections between them. For example this decomposition does not show that the operating system, the lift gates and the levelling system are related nor does it transpire that the lock chamber and the outport share construction elements.

While this static decomposition is suitable for a design approach, it is impractical for construction and or operation. Both construction and operation are event based processes. In a construction process, all systems and their parts are associated to a stage (i.e. a sequence or a date) in the production line. In order for one element to be validated, the elements of the previous stages have to be validated. In operation, all system events are activities, where within the scope of an activity a given system might be of significance while others aren’t. These

levels of complexity are not represented in the current proposed lock decomposition and it leads to a lack of continuity of the information.

2.4 The Structure of the Project Information

Internally, the information model at RWS is structured and available in a database of reports and files. More and more the paper based solutions are being complemented with digital models. The purpose of these digital models is to span the entire complexity of the project. They are meant to be used by all stakeholders at all stages of the product life-cycle. As a result, the data structure of the project should be abstracted of any particular 'internal' process and instead be able to accommodate them all.

Currently, the information sent out to potential contractors is a semi-structured report where none of the content can be used as input for further development. Instead, within the chain of development, the later stakeholders have to define their own information along the information produced by RWS. Naturally, the result are distinct and independent information models where the consistency, continuity and inter-operability are null. As developed before, the result of this independent process is a large mismatch between the vraagspecificatie and the opleverdossier.

On one side, the vraagspecificatie is somewhat structured around the MWW system decomposition. This information is valuable to both the designer and the asset manager who rely on these specification to produce their own process for testing, operating and maintaining the navigation lock. On the other side, the opleverdossier is only concerned with product information (i.e. geometry, materials) and little to no mention about functionality, hierarchy and properties, of the different systems.

In opposition with the requirement-based approach of RWS, the contractors and asset managers typically use an event-based approach decomposed into construction phases (i.e. Gantt Charts) and operating activity networks. These 'structures' are independent of the content information but are critical to the interoperability and continuity of project information. In the future, the MWW model for a standard navigation lock project should be flexible enough to implement these additional levels of complexity in the data models so as to improve the knowledge sharing between stakeholders.

3. MODERN MODELLING INITIATIVES AND INNOVATIONS

This chapter contextualizes the advent of data modelling in the construction industry, highlights the challenges that modellers have tackled and sets the scene for the modern research scene. This part also helps the beginner reader to develop a feel for modelling and appreciate its development.

3.1 The Advent of Product Information Modelling

3.1.1 The General AEC Reference Model

Product information models (PIMs) have been a primary research domain since the information, communication and technology development times. Early on ICT is perceived as a tool to improve engineering practices and develop innovative business processes. The core of the ICT domain tackles communication and interoperability issues between agents by specifying exchange formats and standardising exchange requirements. Exchanges are dynamic moments where digitally asserted information (i.e. data) is passed from one agent to another. In order for exchanges to be sensible, both the emitting end and the receiving end agents have to make sense of the data. Modelling is the scientific idea of defining, understanding, quantifying and visualizing knowledge. A model is a tool that agents can 'use' to verify their understanding of data as well as to introduce new data. In engineering domains, such as the AEC industry, models primarily focus on built products hence the denomination of product information models.

In 1988, Gielingh is the first to introduce the idea that in order to be efficient, the industry should develop a standard for PIM methods. His ideas have transpired to this day and he has had a major influence on the development of modern standards as the chairman for the AEC committee of ISO/STEP. Gielingh developed a general AEC reference model GARM (W. Gielingh, General AEC Reference Model (GARM) 1988). It introduces the concept of model definition space where entities are defined by their level of generalisation/specialization,

aggregation/decomposition, characteristics and life-cycle stages. More fundamentally it develops the idea of layered product definition units (PDU) which are domain and life-cycle specific definitions of product components.

The highest level of abstraction of the PDU is the Functional Unit (FU) which refers to the knowledge set of all product requirements and constraints. As such, it is fair to say that GARM is a requirement oriented modelling method. The FU is the basis against which lower abstraction levels are developed. The very next level is defined as the Technical Solution (TS). The technical solution is the part of the model that is concerned with the attributes and characteristics of the potential design components. The duality of the FU and TS represents the duality between a meta-class representation of an object and its instance representation.

This model is abstract enough (i.e. high level) that its framework is universally applicable to all engineering disciplines (i.e. construction, aeronautic, mechanical). The main contribution of this research is the development of a standard for the exchange of product-model data (STEP) and its associated sub-standards (i.e. file formats, Schemas, IDM).

3.1.2 The STEP family standards

The GARM model, as initially proposed by Gielingh is to abstract to be of any practical use but rather it is an integrator meta-model for industry oriented application models. The STEP initiative has for purpose to tackle the transitional gaps between conceptual and practical models. Its main innovation is that all its deliverables are computer interpretable and thus fit within the modern engineering design process. Over the years, the scope of STEP (ISO10303-1994) has evolved, from its holistic origin of exchanging information between CAD systems, to now including separate parts for implementation methods (part 21), conformance testing (Part 31), integration of resources, applications and protocols.

STEP is mainly addressed to CAD agents and is therefore mostly concerned with the representation of geometric representation of engineering physical products. It has had to adapt to the evolving technologies used to represent these objects, 2D wireframe, boundary representation, 3D solids, parametric objects etc. In parallel, it has had to develop a 2-step process to exchange the data in a neutral file format. This format, part-21 files (from the name

of the standard) is a very lightweight exchange language. All agents in a network can export their proprietary language into part-21 and the receiving end can then import it easily.

The innovation of STEP in the construction modelling domain is a tremendous breakthrough and is guiding the business process changes of today. However, these technological approaches are not sufficient and one important aspect discussed by meta-models is missing in the original STEP, that is, the representation of knowledge and meaning. In order to bridge that gap, the STEP framework has delegated spin-off organizations to design schemes of information classes for each industry.

3.1.3 BuildingSMART and the IFCs

The STEP sub-delegation that is responsible for the development of an AEC class scheme is BuildingSMART international and all its local representations. The model schema that aims to define all the construction concepts is named the Industry Foundation Classes (IFC) schema (buildingSMART 2017). This scheme is standardized as IAI/ISO 16739 and was first released in 1997. In the continuity of STEP, the IFC spin-off has recognized that the only way to implement a universal product model throughout the multitude of CAD agents is standardization (W. Gielingh 2008). Importantly this standard seeks to be free of vendor-specific applications and valid across organizational borders.

The idea of standardized data exchange is a breakthrough innovation but research has shown that its application in the industry was slow and that there were no signs of taking-off (W. Gielingh 2008). The observation led to critical informative statements about the drawbacks of IFC. Most of these drawbacks are implementation ones. The introduction of the standard was misdirected since none of the largest construction businesses saw a clear demand for the technology from their clients. Also, early adapters feared high uptake costs while other businesses would benefit from the lower implementation costs of a more mature technology. On the other hand, CAD vendors interpreted the open standard as a negation of their product added value and specificities.

In addition to the implementation problem, the product model technology laid the foundation of legal issues since construction court has historically relied on comprehensive and non-falsified paper-traces. The layers of complexity and the vendor dependent strategy of IFC

achieved the total alienation with regards to the smaller businesses and construction suppliers. In order to overcome these barriers to implementation the buildingSMART organization has had to revisit its distribution scheme and capture growing trends in the construction sector. The renewed standard effort strategy led to the development of BIM.

3.2 BIM: A New World Order

By far today, the most commonly shared trend in the AEC sector as well as many other industries is the digitalization of project information and domain knowledge (i.e. data). Many international, national as well as local initiatives are focusing their research efforts on developing industry wide universal information frameworks. The discussions are layered in many and sometime overlapping research areas. BIM is the domain that covers all these research innovations and seeks to implement them.

BIM is in essence a business process innovation rather than a technology innovation and would have benefited from a more representative denomination. BIM certainly relies on many technologies such as CAD, CAC, PIM innovations as discussed above but these innovations pre-date the notion of BIM as is common today. BIM specifications are in fact a set of answers to social and corporate needs that modern construction processes have developed. Over the last century, there is clear tendency for governmental bodies to retract from their positions of construction actors and rather focus on their role of construction regulator and supervisor. The developments of Public-Private-Partnerships (PPP) has meant that more and more construction business extend their core business capacities to new roles of the construction domain (i.e. designer, asset manager, certification).

In the meantime, due to economic instability, there has been a contradictory specialization effect where construction companies find refuge in their core competences to survive the fluctuation of construction domain. This duality in tendency has led to the development of much more collaborative and participative construction projects. These innovations have been embraced by businesses of all sizes that can rely on the specialization of the ones and or economic reach of the others. However, this added level of complexity can only be turned into added value if its challenges are dealt with seamlessly.

Intrinsic to a network collaboration is the idea of information exchange and product delivery in the shared context of a symposium. Naturally, the issues of who, what, when and how to exchange respective information come to mind. These problematics are precisely the type of problem statements that STEP initiatives had attempted to solve in the past. As a result the ideas and deliverables of STEP have gained the popularity among the industry that they struggled to capture in their beginnings.

However, in a sense, this process innovation has also highlighted the limitations of STEP. There seems to be a native contradiction between the ideas of modelling and standardizing. The first entails to represent the breadth of knowledge and complexity of the real world in the most detailed and efficient way while the second aims to tune digital information to universal but reduced concepts. As more and more organizations take part in the BIM developments it represents more and more views and domain knowledge that has to be implemented. This complexity and expressivity will never be achieved by a standard.

3.3 Modelling Technology Considerations

3.3.1 Modelling Process Abstraction Levels

Models are structures that represent contextual knowledge, as such, the context and/or environment in which the model is defined has a direct impact on interpretation of the semantics. The hamburger-model from GARM and the IFC schema lack the ability to represent the extent to which a given meaning is defined. In a corporate setting, individual companies have specific model definition and interpretation which suit their internal process. A supplier might have a model definition that matches its production chain knowledge (i.e. a given concrete type), while a contractor might interpret the same definition as a technical specification (i.e. a recognized concrete standard type). These business level models are critical because they represent the companies' best opportunity to share and transmit their added value and gain a strong market position. As a result, the data structure that aims to 'standardize' this knowledge needs to allow for the different perspectives.

The short example above is a representative case of different perspective on the same process abstraction level. Yet, in a standardization effort, it is required that on a higher abstraction level both these perspectives on the product merge to form a single representation that is readable

by all parties. This level approach allows the definition of 4 gross abstraction layers, the business layer, which is dedicated to the development of internal processes and models, the project layer, which aggregates the knowledge of the different parties of a common project, the type layer, which tunes the knowledge from different project of a similar type (i.e. a lock navigation type) and the industry level, which is a universal representation for all actors of a given domain (i.e. AEC). In this layering, see Figure 3.1, STEP and the IFC schema can be described as the least specific definitions that everyone can agree upon rather than an exhaustive description of the different perspectives.

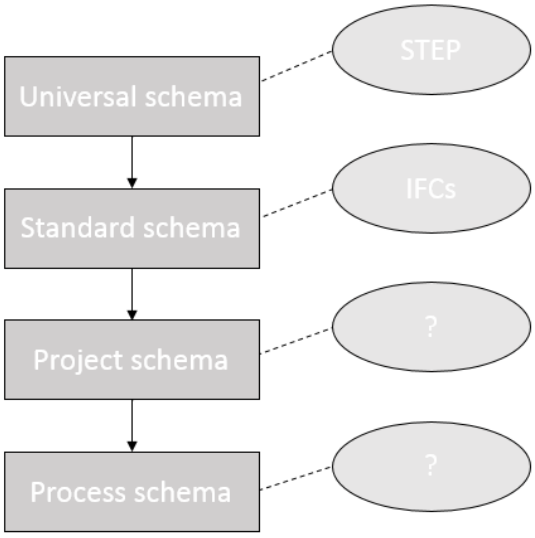


Figure 3.1 - Different Project Abstraction Layers

The holistic approach introduces new complexities. Horizontally for each level, should there be a unique data representation, or different functional ones, vertically, are lower models instances of the higher 'meta-models' or rather subsumption models and how to define the boundaries between the different models?

3.3.2 SysML Modelling For Engineering

SysML is a subset of the unified modelling language (UML) oriented for system engineering developed by the Object Management Group (OMG 2015). SysML is a language that supports the conceptual stages of a system rich project (Balmelli 2007). It allows the representation of a project decomposition and the description of project requirements. It is an object oriented language using an extension of the UML schema and a rich set of diagrammatic knowledge representation (Delligatti 2013). SysML is typically used to complement a V-shaped project

process, see Figure 3.2, or other concurrent and iterative processes (Kossiakoff, Sweet, et al., Systems Engineering principles and practice 2011).

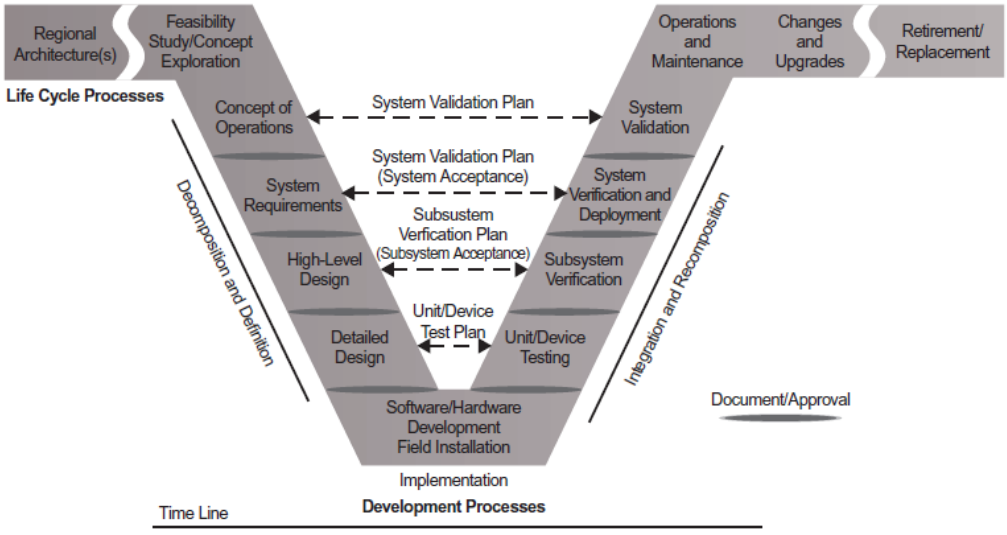


Figure 3.2 - V-shape Project Process (Kossiakoff, Sweet, et al., Systems Engineering principles and practice 2011)

SysML is a powerful modelling tool because it allows the definitions of classes and attributes (i.e. properties) as well as requirements and constraints. It also defines connection links between the different objects (n-ary relationships). As a result, historically, it has been extensively used to define the 'process schema' from the perspective of a central designer entity. However, SysML has some major technical drawbacks regarding its use for a project level schema across all stakeholders (Kiko and Atkinson 2008). The first is that SysML cannot link 'similar' concepts as being equivalent. As such, different classes will always be different concepts. This goes against the need to express the versatility in project knowledge.

Additionally, in sysML, project properties only exist within the context of a class attribute definition, they are not first-class primitives. This means that there can be no straightforward taxonomy or decomposition of properties and that all users have to conform to the use of stereotypical model defined properties. SysML offers no effective inferencing capacities because of its weak logic syntax. While sysML has proven to be a great tool for visualisation and effective communication of system complex product information it does not seem to be totally suitable for the concurrent design of multidisciplinary projects as it lacks the capabilities to structure and consistently equate different perspective on data.

3.3.3 Semantic Web Technologies in AEC

Already in the GARM model definition, Gielingh notes that ‘everything may be related with anything else’ or that ‘everyone should talk to everyone else to solve problems’ and even that ‘technical solutions are structured set of functional units’ while ‘the relations between functional units are modelled as a network’ (W. Gielingh 1988). Almost as a premonition of future developments, in its profound understanding of how project information is interrelated the father of STEP is formulating demands and requirements that are synonymous of a web of semantic data.

However, Gielingh very wisely notes that the standard way of doing business, ‘the divide and conquer approach’ to reduce the complexity of design problems requires a tree-like structure with strong hierarchical layers. This, ideology that the business processes dictate the product information modelling is strong throughout the GARM propositions and resonate in STEP and STEP by products. After all, product models are designed by industry experts for industry experts and it is only natural that all engineering parties want to express their knowledge in their own models. As was discussed earlier, there are strong limitations to the capacity of a standard model to develop a comprehensive range of expressivity and the IFC are struggling to incorporate all the industry classes (Beetz, Leeuwen and Vries 2009) (Pratt 1998).

Today’s limitations will eventually be resolved but by that time new limitations will have occurred. The speed at which standards deal with change and innovation will never capture the current trend and complexity of the industry (Beetz, Vries and Leeuwen 2007). Additionally, tools that are developed on these standards will only offer past business processes.

In the modern age of ICT, and within the scope of the BIM revolution, business models tend to evolve at a greater pace than knowledge domains. As a result, product modelling needs to be reinvented. The next generation of PIM is not a model that hopelessly tries to keep up with a standard business model, but rather a PIM that seeks to represent the entirety of a domain knowledge and offers its users the option of tailoring it to their own innovative business process. These state-of-the-art attributes are those of the semantic web.

4. CAPTURING, STRUCTURING AND MANAGING KNOWLEDGE

This chapter introduces the technologies of the semantic web and defines their usability within the scope of information modelling. Since this technology is the core of the product report of this work, it was deemed necessary to develop the ideas for a novice audience (i.e. from scratch). This approach is also the one followed by the author who had no previous experience using the following notions whatsoever.

4.1 The Emergence of the Semantic Web

The idea of ontologies has re-emerged recently in relation to the development of big-data and the concept of linked open data. The founder of the World Wide Web, Tim Berners-Lee started the grassroots initiative of the semantic web. His mission was to make sense of the tremendous amount of data that is available publicly on the web. In order to fulfil that mission the W3C has developed numerous standards on how to capture, share and use 'smart' data (W3C 2017).

4.1.1 Resource Description Framework

The Resources Description Framework (RDF) is a W3C initiative to develop a framework for the description of information through the web (Klyne, Carol and McBride 2014). The approach is simple in nature as it defines a standard data architecture to declare (i.e. assert) information. This architecture is based on tuples (or triples), see Figure 4.1.

<Subject> <Predicate> <Object>

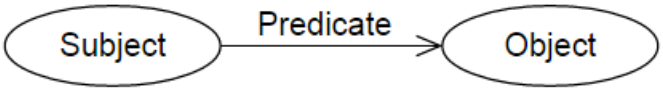


Figure 4.1 -RDF Triples syntax and graph representation (Klyne, Carol and McBride 2014)

In RDF all declarations (i.e. subjects, predicates, objects) are 'resources' and denote something in the universe of discourse (i.e. object, concept, idea, property). Resources are identified through Universal Resource Identifiers (URIs) which insure that all resources are unique and

cannot be misinterpreted. An RDF declaration represents a relationship of the type expressed by the predicate between the subject and the object resources. In graph description, the subject and object are nodes while predicates are vertices connecting the nodes. A node, might also represent an existing link between resources without being explicitly named or without relationship to a resource in the universe of discourse. In that case RDF makes use of blank nodes.

In RDF, datasets are a combination of graphs, graphs are the knowledge representation of a limited universe of discourse. It is preferable to work with multiple information graphs rather than combined datasets when dealing with large amounts of information. It follows that graphs are subsets of RDF datasets. If a graph is defined then it is given a name (i.e. URI) and referred to as a named graph while the triples that do not fit within a named graph are part of the default graph of a given RDF dataset.

In practice, RDF documents are written in a concrete RDF syntax. There exists different syntaxes with different purposes: RDF/XML is a XML compatible syntax that allows convenient exchange over http while; Turtle and N3 syntax were designed for human readability and introduce syntax shortcuts to reduce the overwhelming complexity of resource referencing; N-triples syntax, on the other hand, is devised to improve the machine readability and therefore improve RDF application performance.

The innovative aspect of RDF is the simplicity and open availability of its structure. As such, many agents are potentially free to choose to use such a standardized approach or at least implement compatibility measures. The non-restrictive approach in the modelling design and the limitless scope of expressivity makes it a very flexible framework applicable to large and complex domains such as engineering.

4.1.2 RDF Schema

The RDF schema (RDFs) is a semantic extension of RDF (Brickley, Guha and McBride 2014). It introduces the foundations of object oriented modelling such as the ideas of class and properties. RDFs is conceived as a meta-data description structure, that is, any project data can universally be classified as one of the RDFs primitives. As opposed to sysML, the aim of the

RDFs primitives is not to define a common modelling universe, but rather to provide the building blocks and the connection tools to develop independent modelling universes.

- **CLASSIFIERS**

As mentioned earlier, every information is defined as an instance of *rdfs:Resource*. 'Class' type of information are *rdfs:Class* elements. Note that *rdfs:Class* is reflexive, meaning that even itself is a *rdfs:Class* type. Properties are defined as instances of the meta-class *rdf:Property* while input information are classified as *rdfs:Literal* and described by *rdfs:Datatype* instances.

- **PROPERTIES**

Beyond the meta-classifiers RDFs defines a range of property specifications related to relationships between resources. A type relationship (i.e. instantiation) is asserted trough the *rdf:type* predicate. Through rdfs:subClassOf and *rdfs:subPropertyOf* one can establish subsumption relationships. It introduces more elaborate meta-data about how different resources are organized and related through their predication by defining *rdfs:domain* and *rdfs:range* properties.

The key difference between a typical object-oriented modelling language and the RDF schema is that the first defines its classes syntactically (i.e. human readable) while the second defines it semantically (i.e. machine readable). The difference might seem trivial but it yields considerable modelling opportunities. Syntactic models are descriptive information representation tools, while semantic models (or ontologies) can be used to support machine reasoning and operations.

4.1.3 Web Ontology Language (OWL)

The semantic web is structured in three layers, RDF defines a metadata layer, RDFs defines a schema layer and the web ontology language (OWL) defines a logical layer (Motik and Parsia 2012). In contrast to the lower layers OWL defines a logic vocabulary. It uses Manchester syntax (Horridge, et al. 2006) to describe axioms (i.e. logic rules) over the RDFs layer. For example, in OWL, one can express the equivalency between two classes, define the symmetry or restrict the cardinality of a property predication as well as restricting the range datatype.

The added value of OWL is the possibility to use inferencing mechanisms. Inferencing allows additional information to be 'deduced' from the original asserted information through an inferencing engine. For example, using range properties, one can infer the class type of a resource. One can also define transitive properties (such as subclassOf) where a continuous chain of property is inferred from the original subsumption assertion.

Finally, OWL differs from other object oriented modelling tools through two major differences. The first is that OWL doesn't make the unique name assumption, meaning that two different resources could potentially refer to the same concept. Explicit differences in concept have to be asserted through OWL distinct class or distinct individual statements. This property allows a versatility in the expressivity of the data structure. The second is that OWL follows an open world logic, which grossly translates to the idea that if a data structure is not explicitly constrained in scope the logic engine cannot define if a statement would potentially be valid out of the ontology scope.

Altogether, the differences between RDF/OWL and sysML syntaxes provide different functionalities. While sysML defines a static graphic description type of modelling the semantic web approach rather uses a dynamic vocabulary description ontology type of modelling. A mistake would be to think that one modelling approach is better than the other. In practice both have pros and cons and it is up to the judgment of the system engineer to select the modelling practice that best suits its needs. Further through this work, it will be demonstrated that the logic capabilities of the semantic web can be used to define and process project requirements and thus provide a large field of opportunities for ontology modelling in the AEC sector.

4.2 The Definition of Ontologies

4.2.1 Bottom-up ontology description

Ontologies are best understood as a combination of top-down and bottom-up approaches to specify a universe of discourse (UoD). In cognitive science, the UoD, is a domain of interest over which interpretations of concepts, variables and objects are universal. This universality of ideas and concepts is critical. It allows the domain users to develop extensive logical thinking and models. The rules, constraints, composition and all other processes rely on unambiguous statements (i.e. axioms). For example, in the UoD of mathematics, theorems, conjectures and

hypothesis are different concepts that are uniquely defined. All instances of these concepts represents a node in the mathematical network. From an axiomatic distribution it is generally possible to develop new statements (i.e. new information) using a deductive argument.

On one hand the UoD is a collection of objects discussed within a field, while on the other hand the ontology is a descriptive model of the relations between the concepts at various levels. Ontologies are directly related to SE because the essence of a good ontology is the decomposition of its entities in components (instance, class, attribute, relation, axiom)

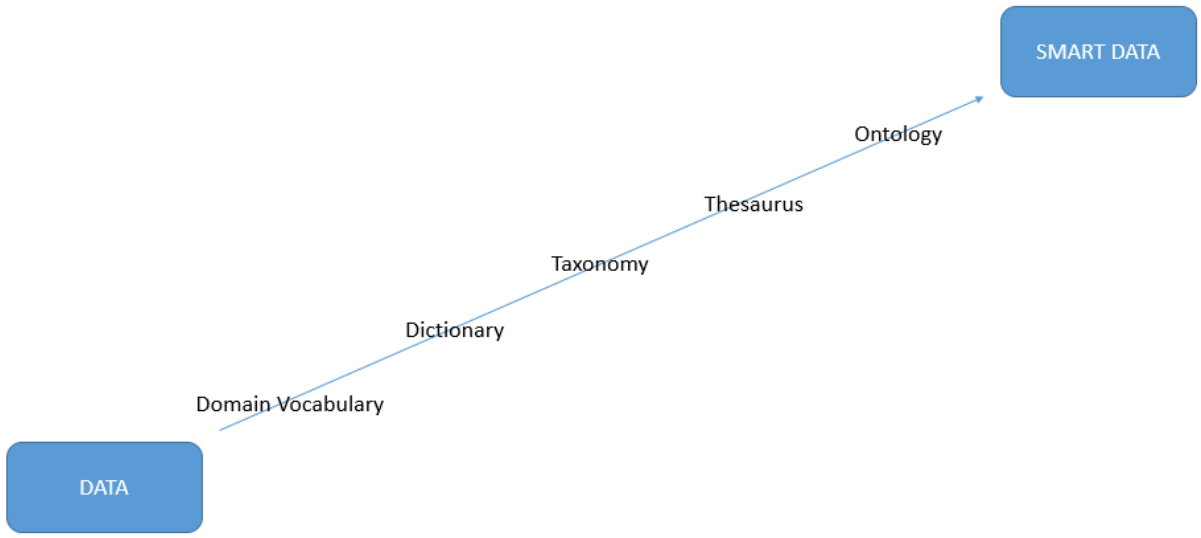


Figure 4.2 - Bottom-up approach to Ontologies

From Figure 4.2 and 4.3, one understands the sense of bi-directionality in the construction of ontologies. Indeed, over a given UoD, an ontology can be more or less smart. The 'smartness' of the ontology directly relates to how connected each of its elements are to one another. At the lower end, a **domain vocabulary** is merely a collection of domain specific words. These are words that are used to transcribe a given expertise from one expert to another. A **dictionary** is smarter than the vocabulary in the sense that it associates a definition to every elements of the UoD. However, within a UoD, definitions do not necessarily correspond to a single meaning. Indeed, a specific vocabulary can have a different meaning for different users. For example, in the AEC industry, the element 'wall' has different representations for engineers and architects.

In order to make the definitions relate to more powerful meanings it is necessary to introduce relationships between the elements. A **taxonomy**, introduces a hierarchy to the elements so as

to define on which abstraction level each vocabulary is situated. A **thesaurus**, goes a step further and introduces the idea of concept clusters. For example, in natural languages, a very common concept is etymology. Using etymology, it is possible to classify words, based on a shared historical root. This relationship goes beyond a simple hierarchy and as such allows more functionality.

In order to differentiate the meaning of 'wall' between engineers and architects it is necessary to introduce the 'concepts' of engineering and architectures, then it is simple to make the distinction between the different compositions. On one side, the composition of 'engineering' and 'wall' relates to a load-carrying element, while on the other side the composition of 'architecture' and 'wall' relates to a partitioning element.

The ultimate description level for a UoD is the **ontology**. It comprises all the properties of the lower levels and distinguishes itself by introducing the ideas of meta-data and logic modelling (i.e. rules). Meta-data is, put simply, 'data about data'. It supports the idea that all resources are part of a 'family of things'. There is no limit to what a class can represent, ideas, concepts, objects, documents, persons etc. That is why an ontology approach to modelling provides a vast scope of expressivity and inter-operability with other agents.

Beyond the linguistics of a model, an ontology also specifies rules, constraints and automated reasoning in the shape of axioms. These axioms may apply to all levels of abstraction of the data. For example, on an instance level, you could define parametric design such that a 'window' has to fit within a 'wall'. On a schema level, one could specify that a property can only apply from a specific class domain to a specific class range or that a relationship has a given directivity.

4.2.2 Top-down ontology description

In the grand scheme of things, it would be tempting to develop an ontology of everything. The framework of the W3C and the LOD initiative is such an attempt to design an architecture of everything, a place where 'anyone can say anything about anything' (Allemang and Hendler 2008). However, this abstraction level does not occur at the ontology level, see Figure 4.3. The main reason is that an ontology is above all a mean to an end. It is a tool that defines a set of

personal, corporate or institutional preferences to model a specific UoD and develop project specific functionalities.

The important notion is the idea of ‘tool’. One can easily imagine that different jobs require different yet specifically tailored tools. An ontology of everything is comparable to a Swiss army knife that can process a wide variety of applications but does not fully master any of them. The LOD approach to the issue is to develop a network of compatible ontologies in a comparable way to the World Wide Web.

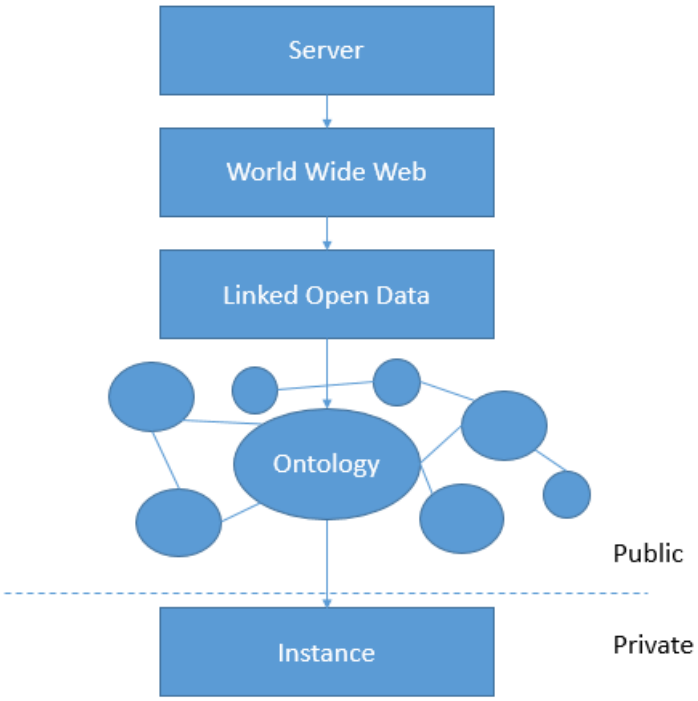


Figure 4.3 - Top-down approach to Ontologies

Allowing individuals to develop their own ontologies to suit real world applications is a technological enhancer and a social advantage. People and companies can model their individuality through their ontology. Businesses can exploit standard technologies to help promote their added value in a market economy. Instead of normalizing a standard ‘system engineering process’ it reflects upon its users and forces them to develop a SE process that is most effective for their domain of application. In essence the network approach promotes creativity and expressivity.

As of today, the LOD initiative has recognized that there is no standard way of measuring the openness or the linked nature of an ontology. Instead, it has decided to define a couple of

arbitrary factors to allow ontologies to be shared universally. The first rule concerns accessibility, the ontology must be resolvable through *http*. The second rule is conformity, the ontology must be a valid RDF format. The third rule governs the size on the ontology, it must be at least a 1000 triples long. The fourth covers the connectivity, it must refer to other ontologies a minimum of 50 times. At last, the ontology can be processed via a SPARQL endpoint.

This set of rules is an arbitrary threshold to confirm that an ontology is well defined, universally accepted, linked and 'open'. However, not all ontologies fit within the scope of the LOD. Indeed, it is very likely that a company ontology will not be publically published due to the proprietary content. Private ontologies share the same properties as 'open' ontologies but restrict the distribution to a distinct user group. In most cases, business ontologies can be divided in three categories, global, local and hybrid (Wache 2004), see Figure 4.4. A global ontology defines a universal data structure for multiple datasets while local ontologies only refer to one (or a limited number) of datasets. In practice a hybrid between the two is most often used, where a root level defines the ontology components, for example an object type library, then a lower level defines the semantics of these components within the scope of a given dataset. This hybrid approach particularly suits the system engineering approach to engineering projects because it helps structure the versatility between parties.

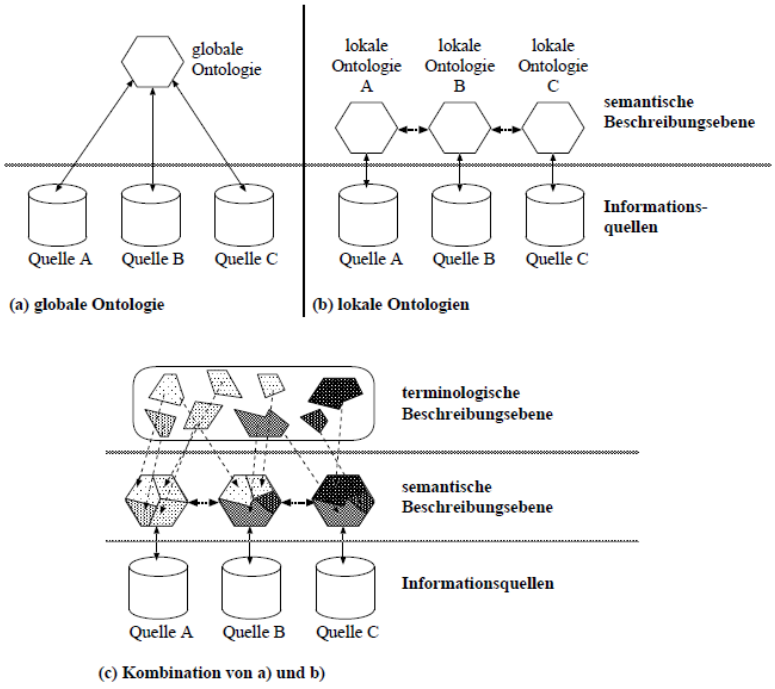


Figure 4.4 - Internal Ontology Architecture (Wache, 2004)

5. THE MULTIWATERWERK PROBLEM DOMAIN

This short chapter bridges the gap between the previous descriptive chapters and the following product chapters. It highlights the objectives and deliverables of this report by describing key information management aspects that the Navigation Lock Model seeks to improve upon.

5.1 Scoping the Smart MWW Model

5.1.1 Use-case Scenario of a Navigation Lock Model

Reference literature often argues that project participants are too quick to jump into the 'solution domain' rather than to ponder on the 'problem domain' (Hull, Jackson, & Dick, 2011). Indeed, there is a trend to glorify the realisation of the project product rather than to reinforce the project process. In the case of MWW this step is critical because the strategy adopted is not to approach the project as the development of 52 'products' but rather the development of a 'single process' applicable to 52 and more products in the future.

Before the discussion can shift towards the demonstration of the possible use of innovative technologies to produce solution results. It would be wise to extensively review what it is that the future information model is to do and possibly in what context. The side effects of not considering in full the problem domain would yield considerable rework, time loss and budget overdraft. In particular with a project such as MWW where project uncertainties are high (conditions of the products, road and maritime traffic, water level, environmental impact) the additional time spent in the initiation stage will be well worth the effort.

Defining the problem domain is essentially answering to the question: what do you want to be able to do? The answer should be straightforward and yield a single 'action' or 'functionality' per sentence (Hendler & Allemang, 2008). At this stage, the use of plain and simple vocabulary is preferred in order to avoid confusion and all meanings should be user-oriented. A standard process for that exercise is the 'use case scenario', see Table 5.1.

What do you want to be able to do? I want to be able to:
List project resources: elements, requirements, objects etc Structure the taxonomy (i.e. decomposition) of all the composite elements Categorize/partition the information into domains, events, views etc List all the relations that connect resources (i.e. links) Manage properties for individual classes to represent project complexity (i.e. attributes, object properties) Associate requirement-objects specification (i.e. n-ary relations) Assign responsibilities Test a requirement (i.e. rules) Manage changes and their impact on project performance Access, read & edit information Import/export capabilities Map internal information structure to the information system Share information distantly

Table 5.1 - Simple Use-case Scenario

The simple listing of rough capabilities is far from sufficient to design system requirements. A complete requirement specification is structured in a meaningful hierarchy. There are of course a multitude of way to define this requirement hierarchy, which is in itself a SE process. However, it would be premature to attempt to expertly structure these stakeholder specific hierarchies yet. In order to complete the use scenario model it is beneficial to scope the boundaries of the expected model. Additionally, it is necessary to realize that this is not a software development work, but rather, only currently missing functionalities will be studied.

5.1.2 Boundaries of the Navigation Lock Model

The smart model is a decision support tool that will help guide the design process by highlighting design and information conflicts. It is not meant to visualize and/or analyse geometric data, this achievement is done through IFC compatible agents. Similarly, the information model is not meant to analyse or operate over proprietary software language information. As far as the functionality, the model isn't meant to check or validate the IFC instance file, this is achieved through other initiatives such as the mvdXMLchecker (Zhang, Beetz, & Vries, 2014).

The model is primarily meant for essential CRUD (create, read, update & delete) as well as semantic reasoning and inferencing on asserted data. As such it is a tool that transforms requirements into rules that have to be checked against. On its own, the information system cannot 'come up' with the rules that it has to check against, rather these rules need to be asserted

by the user, as meaningful axioms and using simple arithmetic and/or string processing (i.e. literal calculations, argument checking).

5.1.3 Action Plan for the Development of a Standard Navigation Lock Model

As a summary of the previous discussions the necessary steps to developing a smart navigation lock model within the framework of the MWW project incorporating system engineering concepts and allowing the continuity of data between the stakeholders are:

1. **Develop a project decomposition system that expresses the full complexity of granularity and aggregation between the different project elements.**
2. **Restructure the arrangement of requirements so as to align them with the precise abstraction level of the model.**
3. **Find a 'mechanism' to assign a rule to each requirement so as to check for its validation.**
4. **Introduce a compatibility between an instance model (i.e. IFC instance file) with the information model.**
5. **Show that the semantic reasoning capacities are adequate to validate/discredit standardized project requirements.**

5.2 Information System Improvement Points

In the case of RWS, most of the requirements are defined at the level of functional activities and events. In the basis specification one can find, for example, EIS 051 and EIS 052. The first requirement specifies the number of vessels and the average maximum time that each passage should take as a function of the traffic intensity. The second requirement specifies a performance score as a statistical function with a percentage value of failure over a given time variable, see Figure 5.1.

Eis 051 **Gemiddelde passeertijd bieden aan scheepvaart**	**Bovenliggende eis(en)**	**Onderliggende eis(en)**
Het systeem schutsluis dient <ntb1> schip uit het scheepvaart-aanbod een gemiddelde passeertijd van maximaal <ntb> minuten te bieden, bij de maatgevende verkeersintensiteit <ntb>.	Eis 589	
Verificatiemethode	**Stakeholder(s)**	**Brondocument**
Ontwikkelingsfase: Realisatiefase: Gebruiksfase:		

Eis 052 **Maximale passeertijd scheepvaart**	**Bovenliggende eis(en)**	**Onderliggende eis(en)**
Het systeem schutsluis dient maximaal <ntb> promille van de schepen een passeertijd te bieden dat groter is dan <ntb> minuten, bij het maatgevende verval.	Eis 589	Eis 053
Verificatiemethode	**Stakeholder(s)**	**Brondocument**
Ontwikkelingsfase: Realisatiefase: Gebruiksfase:		

Figure 5.1 - EIS 051 & 052 for system performance (Rijkswaterstaat, 2012)

These requirements are only vaguely assigned to model 'objects'. In an effort to improve the comprehension and traceability of these requirements it would be beneficial to assign these requirements to a functional-role relationship where the subject of the requirement is a model product while the role of the object is the context of the relationship. By symmetry, in the behavioural domain of the model, this mechanism should also allow the definition of functional-process relationships where the subject remains a model product while the context of the relationship is defined by a activity/process, see Figure 5.2.

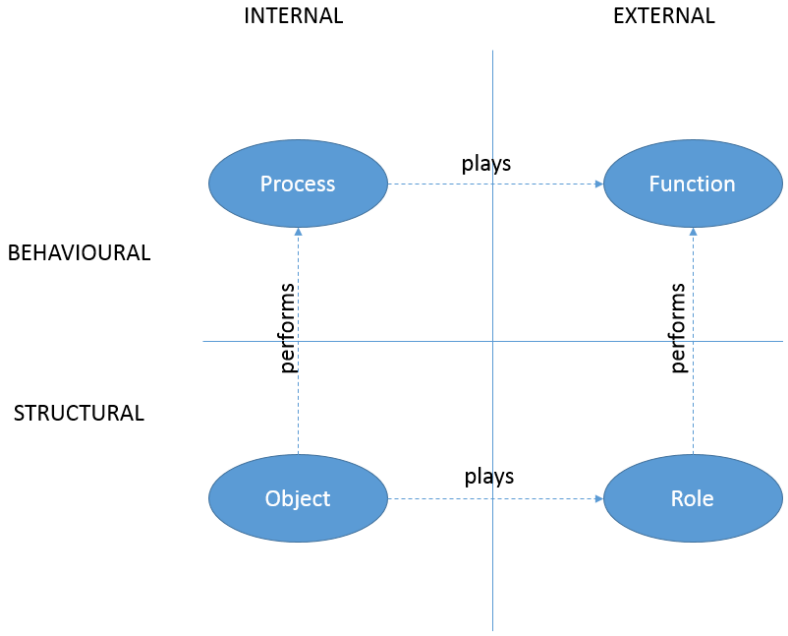


Figure 5.2 - SE Relational Model Structure

6. PRODUCT REPORT

This section portrays the different practical steps, deliverables and possible applications of the previously described technologies to develop an information system that links construction models and their requirements through engineering ontologies within the scope of Building Information Management applied for MultiWaterWerk.

6.1 Methodology

In semantic programming, information is split in three 'boxes'. Developing a coherent information system requires the definition of project specific terminological, assertion and rule boxes:

- The 'Terminological Component' (TBox) specifies the class and instance definitions of a taxonomy. The TBox only contains descriptive statements about the data and their property definition.
- The 'Assertion Component' (ABox) contains asserted facts associated with resources of the TBox. The ABox is the cornerstone of the semantic inference mechanism. It supports all the non-trivial data relationships. It makes a structural connection between the TBox elements and defines information about the data. As such it is comparable to a meta-model of the ontology.
- The 'Rule Component' (RBox) contains the rule and tests to be used in the framework of requirement checking and/or performance analysis.

The Tbox, ABox and RBox are not independent of one another and are not strictly coded separately. Instead, the information system is structured in engineering ontologies (Tudorache 2006). There are four distinct ontologies, the product, the connections, the requirements and the constraint ontologies. This arrangement allows greater flexibility and maintainability when working with the ontologies. In a corporate setting these different ontologies might be managed

by different working groups and specialists. The segregation of information is beneficial to the concurrent work processes. Additionally, in execution, the different ontologies represent different graphs of a dataset which makes running queries more efficient since the target graphs are reduced in size.

In addition to the core ontologies, the information system is also composed of a SPIN ruleset recorded as a spin file. This ruleset contains all the axiomatic rules that the core ontologies are designed against. The purpose of the SPIN ruleset is to define a group of standardized SPARQL queries (mostly ASK and CONSTRUCT) that need to be 'checked' and 'validated' by the inferencing engine. These rules are typically introduced as SPIN rules or SPIN constraints on classes of the engineering ontologies.

In order for the information system to be consistent in the checking process the ontology approach needs to implement both validation and verification mechanisms. On one hand, validation checks for the validity of the asserted model information against the set of requirements, that is, design information introduced manually by engineers, architects etc. On the other hand, verification checks for the consistency of the root information from the instance model against the requirements through an inference rule. A design solution is only viable if both the asserted and the inferred information meet the specification.

6.2 The Product Ontology

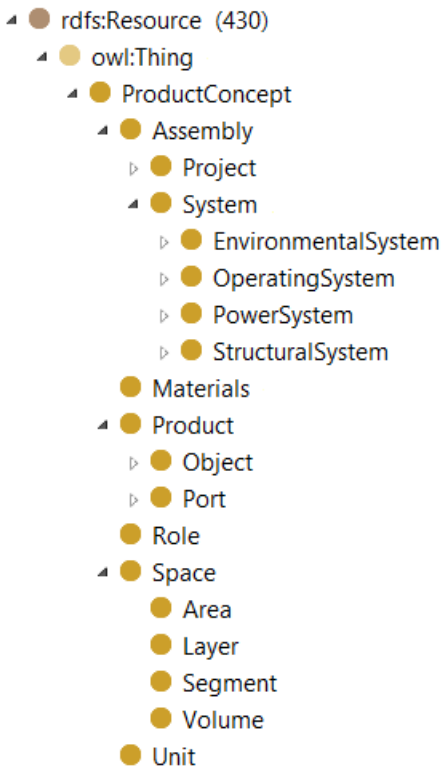


Figure 6.2 - System Taxonomy

The architecture of the TBox is the definition of a project taxonomy with the use of unary class definitions and binary relationship predicates such as *rdfs:subClassOf* and *rdfs:Property*. Since these binary relationships link RDF resources graphically and independently, subsumption and property relationships are convenient for the description of formal hierarchies. For example, a construction *Object* is the subsumption of a construction *Product* since it shares all the properties of the later while also having additional properties of its own.

The product ontology (namespace:mwwp) refers to all the elements used to describe the parts and systems of a navigation lock at different granular level, see Figure 6.2. All the classes in the ontology are references of a *mwwp:ProductConcept* class that identifies their universe of discourse. The *mwwp:NavigationLock* class is the *mmwp:Project* subclass of a *mwwp:Assembly* superclass. *mmwp:Systems* classes describe all the possible aggregations of *mwwp:Object* resources that can be organizationally relevant in a system engineering decomposition.

In parallel, some *mwwp:Object* instances can also be *mwwp:Port* instances, meaning that in the product system these link systems through a functional connector. These connectors are further classified by the type of port that they represent (i.e. mechanical, hydraulic, power, environmental).

The assembly of the different objects and sub-systems are defined using top object properties *mwwp:hasObject* and *mwwp:hasSystem* as well as further variations of these top-properties. These property(x, y) relationships are RDF restricted by range (respectively *mwwp:System* and *mwwp:Project*) and domain (respectively *mwwp:Object* and *mwwp:System*) statements. For example, the property *mwwp:hasOperatingSystem* has its range and domain set to

mwwp:OperatingSystem and *mwwp:NavigationLockSystem*. Indeed, there is no smaller decomposition of system that is composed of an operating system. This example shows how the property centric model definition helps to assert component types from characteristic properties. In addition, the property tree also includes inverse properties that are useful for navigation and inference.

NavigationLockSystem(x).hasOperatingSystem(x, y).OperatingSystem(y)
Is equivalent to:
OperatingSystem(x).isOperatingSystemOf(x, y).NavigationLockSystem(y)

On top of the physical elements of a typical lock project, the system ontology also includes dimensional elements that represent spaces and rooms that compose a navigation lock. These classes refer to functional spaces such as *mwwp:NavigationRoom* and *mwwp:WaitingArea*, rather than physical objects. The distinction between products and spaces is necessary since many system functions are explicitly defined at the spatial abstraction level rather than a product level (i.e. navigational functions).

The product ontology also contains information about the multiple roles that objects can fulfil throughout the project specification. A role is the unique title that a resource carries while assigned to a function. For example, in the context of managing the water level, the upstream door has the role of a weir, while in the context of a vessel passage, the upstream door has the role of a gate. These nuances are critical to the comprehension of project specifications, the functions and their assignments are further discussed in the connection ontology, refer to Section 6.4.

6.3 The Requirement Ontology

6.3.1 Requirement Structure

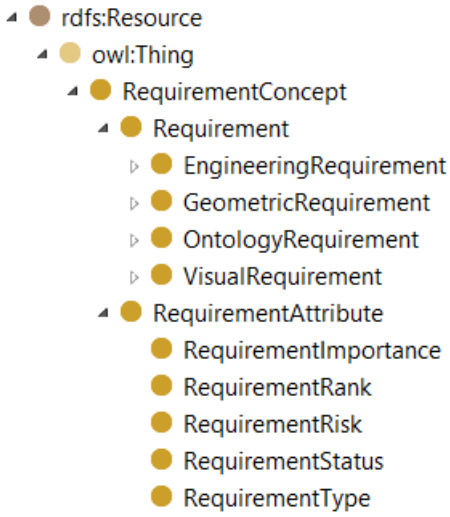


Figure 6.3 - Requirement Ontology

The requirement ontology (namespace: mwwr) is the structured listing of all possible design-property specifications. Requirements are explicitly expressed as *mwwr:Requirement* subclasses (i.e. *mwwr:HeightRequirement* , *mwwr:CapacityRequirement* etc). Requirements can also be composite objects predicated by *mwwr:hasPart* and/or *mwwr:isPartOf* to define a requirement decomposition into sub-requirements. Furthermore, requirements can be predicated by requirement engineering attributes (i.e. optional/compulsory, high/low risk, rank) through object property assertions of range *mwwr:RequirementAttribute*.

One important aspect of project traceability is to understand the evolution path of requirements. Simple requirements can be derived into more complex or more refined requirements. This development is denoted by the object property *mwwr:isDerivedInto* and its *mwwr:isDerivedFrom* inverse property.

6.3.2 Specification of Requirement

Throughout this project the authors have taken the position that project specifications are composed of two independent parts. The requirement, is the first part and captures the notion that a resource must have a property declaration (i.e. a hydraulic pump must have a capacity value). The second part is the constraint (Section 6.3) the constraint defines the range of valid solutions for that property declaration (i.e. 10 < capacity < 15 m^3/hour). The requirement ontology is only concerned with the first part of the specification. The aim is thus to precisely assert the declaration of a requirement to a product (i.e. the assertion of a triple statement between a subject resource from the system ontology and an object resource from the requirement ontology). This tuple is the expression of a design property, its assignment to a product resource and the need for an instance value to fulfil the property:

<Product Subject> <Design Property> <Some Value>

Requirement classes are the resources that capture and record whether a product resource matches the required graph pattern. Indeed, requirement classes are named (or defined) classes, for example, a design solution might be required to be an instance of a *mwwr:HeightRequirement* class and/or *mwwr:ColorRequirement* to check that the design has indeed a specified height and/or colour. This means that the requirements are specified using sufficient and necessary statements. The concept of sufficient and necessary conditions are expressed using *owl:equivalentClass* restrictions. In the case of requirements, one can express the fact that it is necessary that a resource would have some value (its constraint is not relevant at this stage) for a given design property. For example, the requirement that docking attachments need to be predicated with some strength value is shown through the axiom in Figure 6.3.

```
mwwp:DockingAttachment owl:EquivalentClass(
        mwwr:StrengthRequirement
        DataSomeValuesFrom (mwwp:hasStrength)
)
```

Figure 6.3 - Requirement Definition through Equivalent Class Restrictions

However, for reasons that are further elaborated in Chapter 7, this definition structure does not satisfy the functionality of the desired information system. As a result, the particular requirement classes are predicated by a *mwwr:isDefinedBy* which has a range of *rdf:Property* and is used to assign the engineering property from the product ontology that defines this requirement class.

6.3.3 Requirement Assignment

In the requirement ontology, the requirements are assigned through the *mwwr:hasRequirement* object-property predicate. The range of the predicate is an instance of a *mwwr:Requirement* class. This way of predicating requirements allows single objects to have multiple requirements of a same type (i.e. multiple height requirements) and also allows multiple objects to have requirements of the same type without referring to the same tuple.

On the other hand, the domain of the *mwwr:hasRequirement* property is a product *owl:Class*. In other words, the requirement instances are assigned directly to a product class rather than its instances. The justification is that project requirements stand true for all possible instance resources of a same class. In the case where multiple instances are proposed for a product class one can verify which instance best matches all its class requirements. The property relationship between the product class and the requirement instance can be thought of as a punning mechanism where on a meta-model level the product class is also an instance.

6.4 The Constraint Ontology

6.4.1 Property Value Constraint Concept

Requirements are predicated by constraints which define the range of solution that a requirement variable needs to be restricted to. In an OWL architecture, there are multiple approaches to limit the range and/or domain of property values. The native RDF approach is to use range and domain properties to limit the class type. However, in order to specify numeric ranges the reasoning must take place at the resource level rather than the meta-data level. A second native RDF approach would be to partition each requirement class into subclasses with a restriction condition between the valid and invalid state.

These RDF approaches have two inherent downfall. As described in other parts of this work, the problem of creating new RDF resources from model information leads to maintenance and consistency issues in the long run. The sub classing mechanism is one that indirectly creates new data information about the resource and complicates the referencing and reasoning. In addition, this classifying process is unidirectional and does not consistently insures that the resource is both ‘valid’ and ‘not invalid’. These semantics might seem obvious to the human reader but in an open world assumption logic the RDF/OWL reasoners cannot natively distinguish between the two states.

The OWL logic vocabulary allows property ranges to be expressed as *owl:equivalentClass* restriction properties on the classes, as shown in the requirements assignment section. For example, the possible range of values for a given property can be expressed as a restriction using the RDF/XML schemas *xsd:minInclusive* and *xsd:maxInclusive* and their exclusive counterparts.

However, this mechanism requires the assertion of each and every single specification because it does not explicitly make the distinction between the requirement assignment and the constraint range. The reason being that at the class level, the restriction condition does not allow the specification of a variable range. As for the requirements, the property path properties, BIND and FILTER functions of SPARQL can be used through a SPIN rule to transcribe the same logic expression using variable property assignments through a CONSTRUCT query to define constraint classes (note that, through SPARQL, the WHERE clause in the query defines the 'closed world' scope of an otherwise 'open world reasoning'). For example, one can infer that a given requirement fulfils an exclusive range restriction on the requirement property if the filter condition in the SPARQL query is true, see Figure 6.4.

```
# Exclusive Range Axiom
CONSTRUCT {
   ?instance a MWWConstraints:ExclusiveRange .
}
WHERE {
   ?instance MWWConnections:hasRequirement ?requirement .
   ?instance MWWConnections:hasObject ?object .
   ?requirement MWWRequirement:onProperty ?property .
   ?requirement MWWConstraints:hasConstraint ?constraint .
   ?constraint MWWConstraints:onConstraint MWWConstraints:ExclusiveRange .
   ?constraint MWWConstraints:hasParametric ?parametric .
   ?parametric MWWConstraints:minValue ?minparameter .
   ?parametric MWWConstraints:maxValue ?maxparameter .
   ?object ?property ?value .
   FILTER ((?minparameter < ?value) && (?value < ?maxparameter)) .
}
```

Figure 6.4 – Exclusive Range Constraint Inferencing Axiom

The parameters of the constraint are defined through a *mwwk:Paramatric* resource that has parametric attributes such as min, max and exact values. Using different filter composition, this mechanism can be expanded to infer all types of range requirements (upper & lower bound, exact value or inclusive range, see APPENDIX B). Furthermore, different arithmetic operations can be used to produce other value requirements. For example, one can specify a step constraint where the value of a property needs to be a multiple of a given parameter, see Figure 6.5.

```
# Step Range Axiom
CONSTRUCT {
   ?instance a MWWConstraints:StepConstraint .
}
WHERE {
   ?instance MWWConnections:hasRequirement ?requirement .
   ?instance MWWConnections:hasObject ?object .
   ?requirement MWWRequirement:onProperty ?property .
   ?requirement MWWConstraints:hasConstraint ?constraint .
   ?constraint MWWConstraints:onConstraint MWWConstraints:StepConstraint .
   ?constraint MWWConstraints:hasParametric ?parametric .
   ?parametric MWWConstraints:stepParameter ?parameter .
   ?object ?property ?value .
   BIND ((?value / ?parameter) AS ?result) .
   FILTER (abs((?result - round(?result))) = 0) .
}
```

Figure 6.5 - Step Constraint Inferencing Axiom

6.4.2 Parametric Constraint Concepts

Stakeholders might need to express composite or complex parametric constraint concept, for example, when restricting the chamber volume for the size of a given vessel. One needs to insure that the complete water volume is sufficiently low for the operation time while independently the width and height are also sufficiently high for the corresponding vessel size. Composite and complex constraints, differ from composite requirements in that they are not made of property-value relationships but rather they apply to multiple target resources at once.

In most cases, composite constraints can be expressed as a union or intersection of atomic constraint, for example, the height of a wall must have a valid range but also has to be a multiple of the standard brick height. These additional complexities can be expressed as compositions of statements using the Manchester OWL syntax operators AND (i.e. intersection) and OR (i.e. union).

In both the case of composite and complex requirements the modelling issues are not so much the assertion of the parametric testing rules (similar to property constraints) but rather the composition of the assignments to the proper resource and property. For example, the depth of the water in the lock is not only a function of the lock chamber but rather it is bound between the upstream water depth and the downstream water depth. These depth are properties of the navigation room rather than the lock product.

As a result, the specification applies both to the subject property and to a target object property. As always there are different ways one can model this complexity. One option is to assign the specification to the reified connection between the different resources, however, it is generally wiser to limit the use of reification to the reasoning engine. Indeed, without strict guidelines on reification, the information system is doomed to produce maintenance and logic issues. In this work the assignment is done through the assertion of a parametric property with *targetObject* and *targetProperty* predicates.

6.5 The Connection Ontology

6.5.1 The Limitation of Binary Relationships

In the description of a concept scheme (Miles, et al. 2005), there are multiple conceptual relationships between components. The fundamental RDF relationship *rdfs:subClassOf* defines class taxonomies as shown above. However this statement of subsumption where subclasses share all the properties of the superclass is not sufficient to describe a complex project. For example, while they are related, a *mwwp:LevelingSystem* is not a subsumption of the concept of a *mwwp:OperatingSystem*. These might share some common concepts and stand at the same abstraction level but the former does not have all of the later properties and they do not interchangeably stand for the same resource.

A major consequence of this approach is that the overlap between systems that share common objects either as boundary objects or as port objects through spatial or functional relationships cannot be captured with a simple subsumption relationship. In addition, while RDF statements can be graphed into a network of nodes and vertices, the individual tuples are coded independently. This results in the fact that two statements which are defined through binary predicates with the same subject might factually not refer to the same instance (i.e. same URI) of that subject. Again, this binary description limits the potential of ontologies to describe the complex relationship between systems. A remedy to this notion is critical since many functions or requirements are assigned to assemblies of systems rather than individual ones.

Another limitation of this semantic web feature in engineering practices is the complexity to assign multiple roles to resources as a function of their relationships. Projects are built from networks of component which carry specific roles within given functional or spatial settings.

For example, within the context of vessel passage a lock door has the role of a gate while outside the context of a boat passage a lock door has the role of a hydraulic weir. The functional assignments between the two contexts (or roles) on an individual instance are exclusive to one-another.

6.5.2 N-ary Relationships Between Engineering Systems

Complex relationships in real world models are n-ary relationships between resources (i.e. *objectProperty*$\{x_1,x_2,\ldots x_n\}$). They differ from binary relationship as they relate multiple resources to the same instance of an ontology resource (i.e. same URI). There is no native RDF functionality to specify these n-ary relationships, however, the W3C working group has defined a best practice standard to reify such n-ary relationships (Noy and Rector 2006). This best practice document defines three standard approaches to conceptualizing a n-ary relationship, see Figure 6.6 to 6.8.

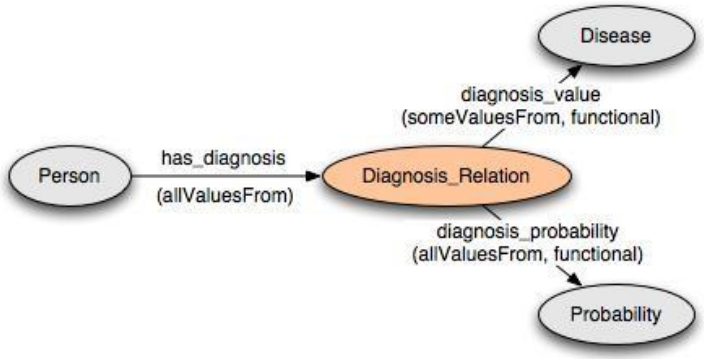


Figure 6.6 - Reification Mechanism 1 (Noy and Rector 2006)

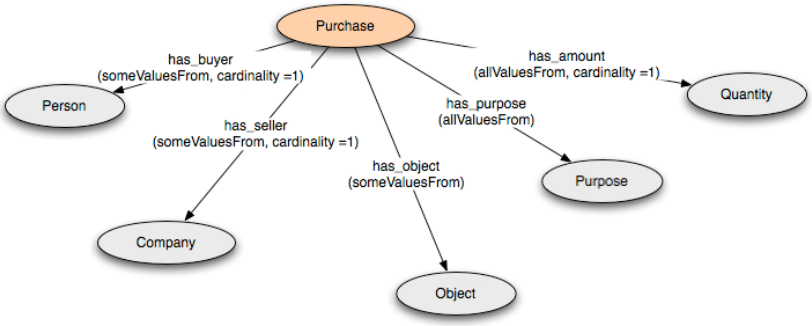


Figure 6.7 - Reification Mechanism 2 (Noy and Rector 2006)

The first pattern suggests the introduction of a relation resource between a subject resource and relationship attributes. This pattern is generally used in the case where the subject is the

conceptual ‘owner’ of the relationship. The second pattern is a variation of the first but does not introduce the subject owner relationship but rather considers all contextual information as relationship attributes. The third approach is to use a listing mechanism between multiple occurrences of similar resource.

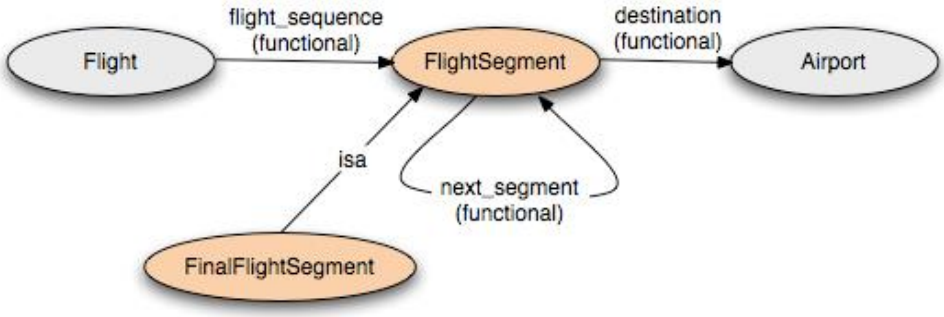


Figure 6.8 - Reification Mechanism 3 (Noy and Rector 2006)

The different mechanisms are suited to different purposes. In the case of a functional relationship between systems, the object, functional role and function classes are related through specific ‘fulfils’ relationship and are independently linked to object, functional role and function requirements. At the functional level, there may be several occurrences of relationships between different resources. For example, an object might play multiple functional roles and functional roles might perform different functions. This multiplicity of tuples may suggest a listing mechanism. In practice however, the critical information is not the multiplicity of triples that a resource might be subject to but rather the connections themselves and the conditions in which independent triples are stated.

This nuance in the focus of the reification mechanism tends to favour the graphic mechanisms 1 and 2. Between the two one can distinguish the different concepts by the importance of defining an ‘owner’ resource (i.e. mechanism 1) or not (i.e. mechanism 2). In the first method, there is a difference in the abstraction level between the owner resource and the attribute resources. This difference in granularity implies that the attribute properties only exist within the context of the existence of the owner resource. In the second method, all resources are specified at the same abstraction level. This perspective implies that the different resources exist independently of one another and are thus specified at the same granularity level.

Arguably, it is the author’s opinion that the first mechanism is best suited to the desired information system, because there is a clear abstraction gap between the specification resource and the product resource.

6.5.3 Application of the N-ary Reification to the Connection Ontology

The structure of the connection ontology (namespace: mwwc), is rooted by the superclass *mwwc:ConnectionConcept* which defines the universe of discourse. The N-ary relationships are defined as instances of *mwwc:Connection* subclasses. These subclasses are split into *mwwc:System-Boundary*, *mwwc:Resource-Specification* and *mwwc:Functional-Role*. The instances of these connection classes are inferred through reification object property predicates such as *mwwc:system-Boundary*, *mwwc:resource-Specification* and *mwwc:function-Role*. The classes are named through *owl:equivalentClass* restriction of the type *owl:someValueOf* on their respective reification properties.

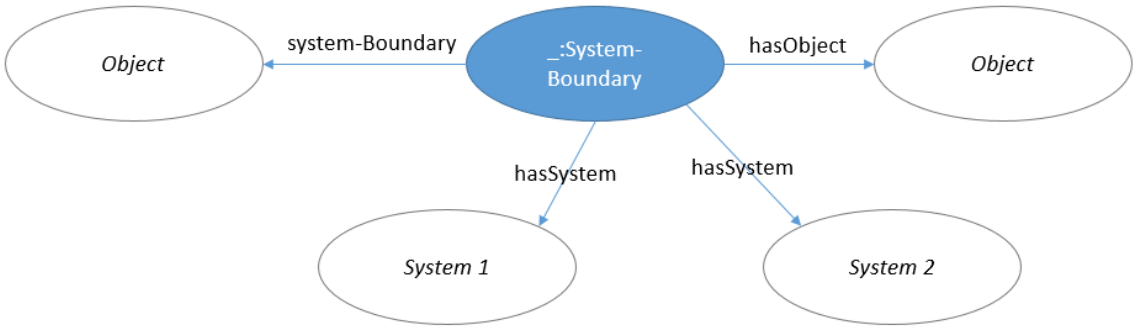


Figure 6.9 - System Boundary Graph Pattern

All the instances of the connection ontology are inferred as reified instances of the connection between concepts. For example, objects that are shared between different systems are considered boundary resources. The SPARQL engine can infer this information from the product ontology and instantiate it as a new resource, see Figure 6.9 & 6.10. The new resource specifically refers to the boundary role of the object rather than its product roles.

```
# System-Boundary Reification Axiom
CONSTRUCT {
   _:b0 MWWConnections:hasSystem ?system1 .
   _:b0 MWWConnections:hasSystem ?system2 .
   _:b0 MWWConnections:hasObject ?instance .
   _:b0 MWWConnections:system-Boundary ?instance .
   _:b0 rdfs:label ?label .
}
WHERE {
   ?system1 :hasObject ?instance .
   ?system2 :hasObject ?instance .
   BIND ((((str(?system1) + "_") + str(?system2)) + "_Boundary") AS ?label) .
   FILTER NOT EXISTS {
      ?node rdfs:label ?label .
   } .
   FILTER (?system1 != ?system2) .
}
```

Figure 6.10 - Boundary Instance Inferring Axiom

Similarly, the resource-specification class refers to all the instance of object-requirements-constraint specification, see Figure 6.11 & 6.12. The reified instance allows the information system to specify information about the specific relationship rather than the requirement in general. These reified resource are critical to the consistent processing of individual specification using standardised (i.e. share) requirement and constraint classes. Similarly, the functional-role class infers instances of object-role-function relationship.

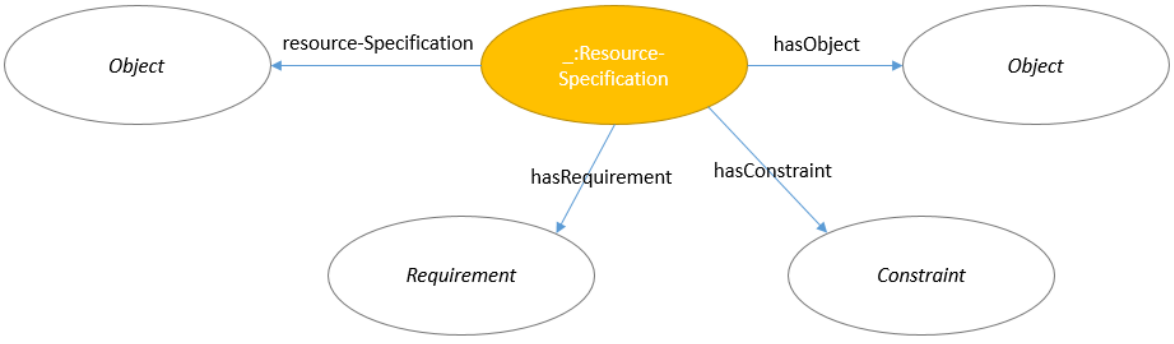


Figure 6.11 - Resource Specification Graph Pattern

```
# Resource-Specification Reification Axiom
CONSTRUCT {
   _:b0 MWWConnections:resource-Specification ?instance .
   _:b0 MWWConnections:hasObject ?instance .
   _:b0 MWWConnections:hasRequirement ?requirement .
   _:b0 MWWConnections:hasConstraint ?constraint .
   _:b0 MWWConnections:hasParametric ?parameter .
   _:b0 rdfs:label ?label .
}
WHERE {
   ?class MWWRequirement:hasRequirement ?requirement .
   ?instance a ?class .
   ?class (rdfs:subClassOf)* :ProductConcept .
   OPTIONAL {
      ?requirement MWWConstraints:hasConstraint ?constraint .
   } .
   OPTIONAL {
      ?constraint MWWConstraints:hasParametric ?parameter .
   } .
   BIND ((str(?instance) + str(?requirement)) AS ?label) .
   FILTER NOT EXISTS {
      _:0 rdfs:label ?label .
   } .
}
```

Figure 6.12 - Reification of Resource-Specification Instances

7. TESTING RULES & INSTANCE MODELS

This chapter defines the top-down validation and bottom-up verification processes that can be developed for checking the requirements & performance. In continuity with the use of semantic web technologies this chapter introduces the use of the W3C SPARQL Inference Notation (SPIN) to produce automated checking rules through a SPIN inferencing engine.

7.1 RBox: SPIN Ruleset

The RBox is the heart of the model functionality. It contains rules that are structured with a rule body, argument, assignment and conditions. These rules are expressed in the SPARQL query language. The RBox can relate to all the different levels of information. For example, one can implement rules about geometric data transpired by the IFC instance file or rules that check the consistency of the design assertions. The innovation consists in the use of SPARQL Inference Notation to test the requirements of a standardized navigation lock system.

7.1.1 Requirement Property Reasoning

In its native specification, OWL2 only supports object type property reasoning. As a result named classes with restriction conditions are only inferred where the property restriction is an object type property. In engineering however, most property specification are datatype properties with an infinite range of expression. For example, the strength of some product is a number comprised between 0 and infinity. It would be impractical to express all the possible strength value for a product as instance of a ‘number’ class.

A second downfall, is that owl restrictions cannot be specified with property chains whereas this functionality is offered by the path properties of SPARQL. As a result, the native OWL inferencing capacities need to be compensated with a SPIN axiom stating that named classes restricted by datatype properties can also be inferred and that, in order to maintain consistency, the inferencing is done at the reified specifications, see Figure 7.1. The axiom is 'equivalent' to an owl equivalent class restriction on a property.

```
# Requirement Property Inferencing Axiom
CONSTRUCT {
   ?this a ?class .
}
WHERE {
   ?class MWWRequirement:definedBy ?property .
   ?this MWWConnections:hasObject ?object .
   ?this MWWConnections:hasRequirement/MWWRequirement:onProperty ?property .
   ?object ?property ?unknown .
   FILTER NOT EXISTS {
      ?this ?property ?m .
      FILTER (?unknown = ?m) .
   } .
}
```

Figure 7.1 – Requirement Property Inferencing Axiom

7.1.2 Instance Resource Existence

The first step to validate a design proposal is to check that all the required elements do indeed exist (i.e. completeness). In an open world assumption RDF logic states that while a resource might not be defined within a dataset it does not mean that it has not been defined someplace else. As a result the notion of existence in a RDF graph is none trivial. In the scenario of an engineering ontology, the reasoning is that of a closed world assumption. In the case where something is not defined in the ontology it is a design fault rather than a statement of unknown as prescribed natively in RDF.

```
MWWRequirement:ExistenceRequirement
  owl:equivalentClass [
      rdf:type owl:Restriction ;
      owl:minCardinality "1"^^xsd:nonNegativeInteger ;
      owl:onProperty MWWProduct:hasCardinality ;
    ] .
```

Figure 7.2 - Existence Requirement Axiom

The generic open world assumption means that some additional existence conditions have to be defined in order to insure that RDF resources have well specified values rather than

unknowns. This existence query is defined in the requirement ontology using an *owl:equivalentClass* restrictions on the cardinality of the *mwwp:hasCardinality* property, see Figure 7.2. The *mwwp:hasCardinality* property is defined in the product ontology and assigned through a *spin:rule* as the count of resources that belong to a given product or assembly class, see Figure 7.3.

```
# Product Cardinality Axiom
CONSTRUCT {
   ?class :hasCardinality ?result .
}
WHERE {
   {
      SELECT ?class ((COUNT(?subject)) AS ?result)
      WHERE {
         {
            ?subject a ?class .
            ?class (rdfs:subClassOf)* :Product .
         }
         UNION
         {
            ?subject a ?class .
            ?class (rdfs:subClassOf)* :Assembly .
         } .
      }
      GROUP BY ?class
   } .
}
```

Figure 7.3 - Cardinality Assignment Axiom

Using these axioms the inferencing engine can evaluate the cardinality of a resource and classify it as an instance of the *mwwr:CardinalityRequirement* if its property and property value meet the named class necessary conditions. The checking mechanism is then simply done using an ASK *spin:constraint* on the resource, see section 7.1.5.

7.1.3 Model Validation (Atomic)

Validation of the project specification is the process of comparing the information asserted by design against the information laid out in the ontology requirements. In accordance with the differentiation between requirement-assignment and constraint-checking (refer to Chapter 6), the validation of requirement is the step where the reasoning engine validates that property requirements are indeed assigned to the instance resources in the product ontology. The axiomatic validation is expressed as a *spin:rule* on *mwwc:Resource-Specification* instances. A property relationship fulfils a requirement assignment when the instance resource is inferred as a *rdf:type* of the corresponding class in the requirement ontology defined by the proper

property. The validation, given that all conditions are met, is inferred as a CONSTRUCT query, see Figure 7.4. Overall, an instance resource can only be a design solution if all the listed object-property assignment requirements are validated.

```
# Specification Requirement Validation
CONSTRUCT {
   ?this MWWRequirement:isValid "True" .
}
WHERE {
   ?this MWWConnections:hasRequirement/MWWRequirement:onProperty ?property .
   ?this a ?requirement .
   ?requirement MWWRequirement:definedBy ?property .
}
```

Figure 7.4 - Property Assignment Validation Axiom

While, requirements define the assignment of properties to instance resources, constraints define the possible range of solutions of the property values. Similarly to the requirements, standard constraints have been defined as spin rules in the constraint ontology. The constraints on requirements are met by the design information when the reified specification is inferred as a *rdf:type* instance of the class that defines the proper constraint type, see Figure 7.5. At the specification level a valid instance resource has all its *mwwr:isValid* and *mwwk:isValid* boolean properties set to "True".

```
# Specification Constraint Validation
CONSTRUCT {
   ?this MWWConstraints:isValid "True" .
}
WHERE {
   ?this MWWConnections:hasConstraint/MWWConstraints:onConstraint ?class .
   ?this a ?class .
}
```

Figure 7.5 - Value Constraint Validation Axiom

7.1.4 Model Verification

The PMBOK defines verification as the evaluation of whether or not a product, service, or system complies with a regulation, requirement, specification or imposed condition. At the ontology level this can be transcribed as the need to evaluate whether an instance model verifies the specified project requirements. In opposition to validation, the verification process is a bottom-up approach where the source of the information is not the information asserted through design but rather information inferred directly from the source model. In the case of a

construction project, the typical source model is the IFC instance file. The design model is considered verified only if the inferred instance information match the ontological model.

Similarly to the model validation process, the SPARQL Inferencing Notation (i.e. SPIN) can be used to define verification functions over the instance model data. These functions specify templates of arithmetic operations and variable assignments to infer high-level data from low-level data. As a proof of concept, some basic geometric requirements specified in the ontology are translated into verification functions. These functions are assigned to product resources as a combination of *spin:rule* resources.

For instance, the type of vessel allowed to navigate through the navigation lock is partly a geometric requirement. The lock chamber and lock chamber doors need to be wide enough to allow for the size of the vessel and a safety margin. In parallel, the depth of the chamber needs to allow for a given water draft and an additional safety margin. This information can be inferred and extracted using the IFC schema path through its IFC-to-RDF mapping (Pauwels and Deursen 2012).

```
CONSTRUCT
{ ?proxy mwwp:hasXCoordinate ?x }

WHERE  {
{
?proxy  ifcowl:representation_IfcProduct/ifcowl:representations_IfcProductRepresentation
        /list:hasContents/ifcowl:items_IfcRepresentation/ifcowl:outer_IfcManifoldSolidBrep
        /ifcowl:cfsFaces_IfcConnectedFaceSet/ifcowl:bounds_IfcFace/ifcowl:bound_IfcFaceBound ?poly .
}
.
{
?proxy  rdf:type  ifcowl:IfcBuildingElementProxy ;
        ifcowl:objectPlacement_IfcProduct ?localplacement .
?localplacement ifcowl:relativePlacement_IfcLocalPlacement ?axisplacement .
?axisplacement ifcowl:location_IfcPlacement ?originpoint .
?placementlist  (^list:hasNext){0}/(^ifcowl:coordinates_IfcCartesianPoint) ?originpoint ;
                list:hasContents ?olengthmeasure .
?olengthmeasure express:hasDouble ?ovalue .
}
.
{
?lengthmeasure  rdf:type ifcowl:IfcLengthMeasure ;
                express:hasDouble ?value .
?measurelist    rdf:type ifcowl:IfcLengthMeasure_List ;
                (^list:hasNext){0}/(^ifcowl:coordinates_IfcCartesianPoint) ?point ;
                list:hasContents ?lengthmeasure .
?pointlist rdf:type ifcowl:IfcCartesianPoint_List ;
                (^list:hasNext)*/(^ifcowl:polygon_IfcPolyLoop) ?poly;
                list:hasContents ?point .
}BIND (round(?value + ?ovalue) as ?x)
}
```

Figure 7.6 - Extract x-coordinate from IFC instance file

The lock chamber is defined as a system of objects. The verification reasoning requires the extraction of x-, y- and z-coordinates of the objects, see Figure 7.6 (note that this query makes some simplifications that are discussed in Chapter 8). In order to verify that the instance width meets the requirement value it has to be inferred from the horizontal distance between opposing objects. RDF lacks the notion of matrix reasoning and complex computation. As such, the only possible inferencing mechanism is a minimax approach where the minimum distance between x-coordinates of opposing chamber walls is the threshold value for the maximum allowable width, see Figure 7.7.

```
# Object Width verification
CONSTRUCT {
   ?instance MWWRequirement:isVerified "True" .
}
WHERE {
   ?instance MWWConnections:hasObject ?object .
   ?instance MWWConnections:hasRequirement/MWWRequirement:onProperty :hasWidth .
   ?object :hasWidth ?value .
   FILTER (?result = ?value) .
   {
      SELECT ?object ((MIN(?distance)) AS ?result)
      WHERE {
         ?object :hasXCoordinate ?x1 .
         ?object :hasXCoordinate ?x2 .
         FILTER (?x2 != ?x1) .
         BIND (abs((?x1 - ?x2)) AS ?distance) .
      }
      GROUP BY ?object
   } .
}
```

Figure 7.7 - Chamber Width Verification Axiom

The CONSTRUCT query is implemented as a *spin:rule* on the resource-specification class. It targets the object-specification resource that contains the chamber system instance and the width requirement. In case the distance between the chamber walls in the instance file verifies the value width expressed in the design brief the spin inferencing engine will infer a “True” boolean value for the *mwwr:isVerified* property. The same query pattern can be expended to hasLength and hasHeight properties, see APPENDIX C.

7.1.5 Model Checking

The model information defines an acceptable design if all its components both validate (top-down) and verify (bottom-up) the ontology requirements and constraints. The checking approach presented here consists in asking if the requirements and constraints are both

validated and verified. These axiomatic statements are checked by a *spin:constraint* on resource-specification instances, see Figure 7.8 & 7.9.

```
# The requirements of this resource are not all valid
ASK WHERE {
   ?node a MWWConnections:Resource-Specification .
   ?node MWWRequirement:isValid ?boolean .
   FILTER (?boolean = "True") .
}
# The requirements of this resource are not all verified
ASK WHERE {
   ?node a MWWConnections:Resource-Specification .
   ?node MWWRequirement:isVerified ?boolean .
   FILTER (?boolean = "True") .
}
```

Figure 7.8 - Requirement Validation & Verification Checking Axiom

Using these universal checking mechanism the engineering ontology can validate the requirements and constraints. In the case where a requirement, or a constraint is not validated the spin inferencing engine will display a spin constraint violation. Alternatively, in a programming environment one could specify a CONSTRUCT statement to produce a constraint violation resource.

```
# The constraints of this resource are not all valid
ASK WHERE {
   ?node a MWWConnections:Resource-Specification .
   ?node MWWConstraints:isValid ?boolean .
   FILTER (?boolean = "True") .
}
```

Figure 7.9 - Constraint Checking Axiom

7.2 Tested Specifications and Instance Model

7.2.1 Tested Requirements and Constraints

As mentioned in other parts, this proof of concept is restricted to a prototype lock chamber design. As such, the requirements and constraints used in this analysis only relate to parts, and assemblies of the chamber system. Using some proxy instances it was also possible to test requirements that the chamber system or its parts have with external systems (i.e. the height of the chamber must be greater than the upstream water height).

The list of tested requirement is shown in Table 7.1, they represent a small but representative sample of typical standard engineering requirement in navigation lock projects. They range from property value range requirement to parametric requirement. As, explained earlier, the

Requirement #	Object	Label	On Property	Constraint	Target (object)	Target (parameter)
1	Lock Door (downstream)	The lock door has height	hasHeight	Inclusive Range		Literal
2		The lock door has height	hasHeight	Lower Para	Navigation Room (upstream)	hasWaterDepth
3		The lock door has strength	hasUltimateStrength	Lower bound		Literal
4		The lock door has operating time	hasOperatingTime	Exclusive Range		Literal
5	Lock Door (upstream)	The lock door has height	hasHeight	Inclusive Range		Literal
6		The lock door has height	hasHeight	Lower Para	Navigation Room (upstream)	hasWaterDepth
7		The lock door has strength	hasUltimateStrength	Lower bound		Literal
8		The lock door has operating time	hasOperatingTime	Exclusive Range		Literal
9	Chamber Slab	The chamber slab has thickness	hasHeight	Inclusive Range		Literal
10		The chamber slab has strength	hasUltimateStrength	Lower bound		Literal
11		The chamber slab has length	hasLength	Exact Para	Chamber System	Literal
12		The chamber slab has width	hasWidth	Inclusive Range		Literal
13	Chamber Wall LB	The chamber wall has height	hasHeight	Lower Para	Navigation Room (upstream)	hasWaterDepth
14		The chamber wall has strength	hasUltimateStrength	Lower bound		Literal
15	Chamber Wall RB	The chamber wall has height	hasHeight	Lower Para	Navigation Room (upstream)	hasWaterDepth
16		The chamber wall has strength	hasUltimateStrength	Lower bound		Literal
17	Chamber System	Distance between LB & RB	hasWidth	Range		Literal
18		Distance between LB & RB	hasWidth	Upper Para	Navigation Room (upstream)	hasWidth
19		Volume of water	hasWaterDepth	Upper Para	Navigation Room (upstream)	hasWaterDepth
20		Volume of water	hasWaterDepth	Lower Para	Navigation Room (downstream)	hasWaterDepth

Table 7.2 - List of Requirements and Constraints

requirements are attached to their respective object class and the validation is run through spin rules inferencing. In parallel, the verification are also run as spin rules.

7.2.2 Prototype Instance Lock Chamber System

The engineering ontologies define the data schema to be tested against but it does not produce the instance resources that are used to model the geometric representation of the product (i.e. the design). Instead, these instance resources are 3D geometric model semantically defined with the IFC schema and produced by commonly available CAD software. The IFC are not directly compatible with semantic web technologies. In an effort to combine the interoperability of IFC and the versatility of the semantic web, the IFC model can be converted into RDF by the use of the IFC-to-RDF tool (Pauwels, Meyer and Campenhout 2010).

For the purpose of testing the proof of concept ontology, two simplified lock chamber model have been designed in SketchUp 2015 and exported in IFC 2x3, see Figure 7.10 & 7.11. The first model is designed so that it meets all the geometric criteria set out in the design brief. The assignment between the instance file and the component ontology is done manually through the assertion of a *rdf:type* relationship between an ifcowl proxy element instance and a product class. These assertions are introduced through INSERT queries, see Figure 7.12.

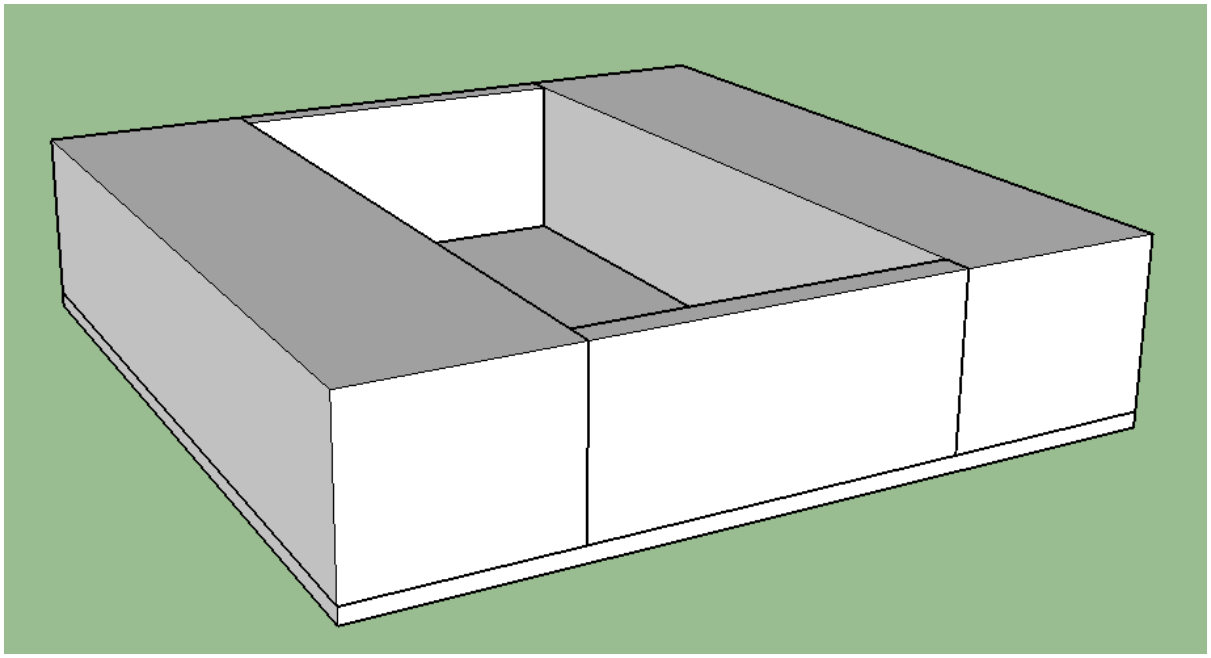


Figure 7.104 - First Prototype Design

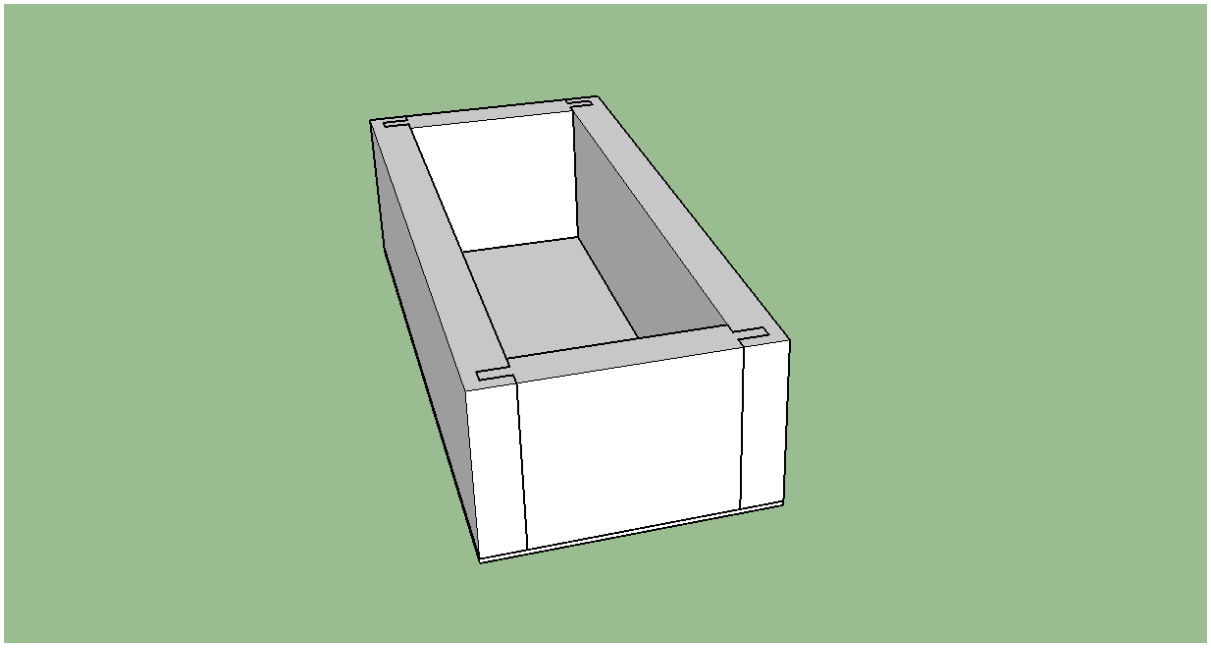


Figure 7.11 – Second Prototype Design

The purpose of the two prototype design is to test the consistency of the logic inferencing axiom stated in the ontology as well as the general behaviour of the information system. The second design brief still fits the requirements but the IFC model does not match the asserted information. This scenario is set up to test the verification capabilities of the ontology.

```
INSERT
{
inst:IfcBuildingElementProxy_47 a mwwp:DownstreamDoor .
inst:IfcBuildingElementProxy_128 a mwwp:UpstreamDoor .
inst:IfcBuildingElementProxy_209 a mwwp:ChamberWallLB .
inst:IfcBuildingElementProxy_290 a mwwp:ChamberWallRB .
inst:IfcBuildingElementProxy_371 a mwwp:ChamberSlab .
}
WHERE{}
```

Figure 7.12 - Semantic Assignment of Instance Resources

7.2.3 Asserting the Design Brief

As well as introducing the design model, the design stakeholders also report the design information as delivered by the architects and engineers. This information is typically made of a listing of all the object properties such as the dimensions and materials of the components as well as their strength and durability properties. These properties can directly be introduced as *xsd:Decimal* or *xsd:String* values onto the instance graph, see Figure 7.13.

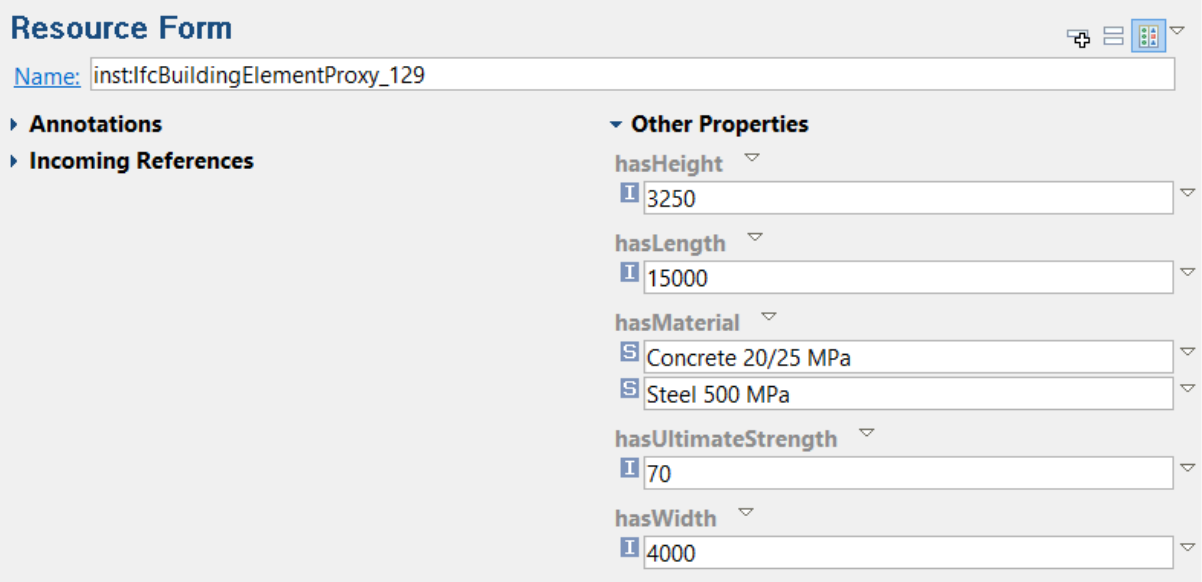


Figure 7.13 - Design Brief Assertion

The values of the design datatype property predicates are introduced as either string or decimal literals. Here the decimal standard is preferred over the float or double formats which are usually associated with engineering/science analysis for two independent reasons. The first is the way memory stores floats and doubles. The values never represent an exact figure but rather a very good approximation. This approximation adds a complexity layer to some of the comparison rules which is outside the scope of this work. The second, is the fact that in the XML schema, floats and doubles are disjoint from integers while decimals are not disjoint. This means that integers, may also be decimals and some decimals may also be integers which is a reality that best suits this ontology and will avoid some logic faults.

8. RESULTS & DISCUSSION

The purpose of this chapter is to present, in a first part, the results obtained through the ontology. In a second part, comments and developments are made about the obtained results in a structured discussion which relates to concepts discussed throughout the report. In parallel, some propositions are made regarding possible improvements and further works.

8.1 Presentation of Results

8.1.2 Test 1 Results

The first model inferencing validated the 20 property requirement assignment as well as the 20 constraint ranges, see Figure 8.1, both on property and parametric requirements. The verification axiom validated the minimax values of the *mwwp:hasWidth*, *mwwp:hasLength* and *mwwp:hasHeight* properties. On the other hands, properties such as the operating time, ultimate strength and water depth cannot directly be verified from the IFC instance file.

[specification]	reqID	requirement	constraint	verified
<@4ba817c3:15ab35a4dcf:-6c83>	Requirement_1	True	True	True
<@4ba817c3:15ab35a4dcf:-6c84>	Requirement_12	True	True	True
<@4ba817c3:15ab35a4dcf:-6c85>	Requirement_4	True	True	
<@4ba817c3:15ab35a4dcf:-6c86>	Requirement_15	True	True	True
<@4ba817c3:15ab35a4dcf:-6c87>	Requirement_2	True	True	True
<@4ba817c3:15ab35a4dcf:-6c88>	Requirement_6	True	True	True
<@4ba817c3:15ab35a4dcf:-6c89>	Requirement_17	True	True	
<@4ba817c3:15ab35a4dcf:-6c8a>	Requirement_19	True	True	
<@4ba817c3:15ab35a4dcf:-6c8b>	Requirement_10	True	True	
<@4ba817c3:15ab35a4dcf:-6c8c>	Requirement_8	True	True	
<@4ba817c3:15ab35a4dcf:-6c8d>	Requirement_18	True	True	
<@4ba817c3:15ab35a4dcf:-6c8e>	Requirement_11	True	True	True
<@4ba817c3:15ab35a4dcf:-6c8f>	Requirement_7	True	True	
<@4ba817c3:15ab35a4dcf:-6c90>	Requirement_16	True	True	
<@4ba817c3:15ab35a4dcf:-6c91>	Requirement_5	True	True	True
<@4ba817c3:15ab35a4dcf:-6c92>	Requirement_9	True	True	True
<@4ba817c3:15ab35a4dcf:-6c93>	Requirement_13	True	True	True
<@4ba817c3:15ab35a4dcf:-6c94>	Requirement_20	True	True	
<@4ba817c3:15ab35a4dcf:-6c95>	Requirement_14	True	True	
<@4ba817c3:15ab35a4dcf:-6c96>	Requirement_3	True	True	

Figure 8.5 - First Model Result Table

8.1.3 Test 2 Results

The second model inferencing also validated the 20 requirements and constraint given the fact that the asserted design brief is similar to the first model, see Figure 8.2. However, the verification axioms inferred that the IFC model proxy element's coordinates did not match with the asserted property values. As a result, none of the mismatched requirements were verified.

[specification]	reqID	requirement	constraint	verified
<@4ba817c3:15ab35a4dcf:-6fa0>	Requirement_1	True	True	
<@4ba817c3:15ab35a4dcf:-6fa1>	Requirement_12	True	True	True
<@4ba817c3:15ab35a4dcf:-6fa2>	Requirement_4	True	True	
<@4ba817c3:15ab35a4dcf:-6fa3>	Requirement_15	True	True	
<@4ba817c3:15ab35a4dcf:-6fa4>	Requirement_2	True	True	
<@4ba817c3:15ab35a4dcf:-6fa5>	Requirement_6	True	True	
<@4ba817c3:15ab35a4dcf:-6fa6>	Requirement_17	True	True	
<@4ba817c3:15ab35a4dcf:-6fa7>	Requirement_19	True	True	
<@4ba817c3:15ab35a4dcf:-6fa8>	Requirement_10	True	True	
<@4ba817c3:15ab35a4dcf:-6fa9>	Requirement_8	True	True	
<@4ba817c3:15ab35a4dcf:-6faa>	Requirement_18	True	True	
<@4ba817c3:15ab35a4dcf:-6fab>	Requirement_11	True	True	
<@4ba817c3:15ab35a4dcf:-6fac>	Requirement_7	True	True	
<@4ba817c3:15ab35a4dcf:-6fad>	Requirement_16	True	True	
<@4ba817c3:15ab35a4dcf:-6fae>	Requirement_5	True	True	
<@4ba817c3:15ab35a4dcf:-6faf>	Requirement_9	True	True	
<@4ba817c3:15ab35a4dcf:-6fb0>	Requirement_13	True	True	
<@4ba817c3:15ab35a4dcf:-6fb1>	Requirement_20	True	True	
<@4ba817c3:15ab35a4dcf:-6fb2>	Requirement_14	True	True	
<@4ba817c3:15ab35a4dcf:-6fb3>	Requirement_3	True	True	

Figure 8.6 - Second Model Result Table

8.1.3 Conclusion on Results

The results shown above have demonstrated that the description logic of the OWL vocabulary complemented with the SPARQL functionalities such as BIND and FILTER as well as its flexible property paths allow for the structuring of engineering requirements on properties and constraint. These requirements can be assigned at the meta-level on product class that correspond to a precise navigation lock system decomposition. The WHERE {} clause in the queries allows the reasoning to be restricted over a 'closed' set of data.

The reification mechanism successfully conceptualized the independent resource specifications and asserted them as ontology instances with their corresponding requirements, constraints and parametric. The unreadable URIs shown in the table results are compensated by natural language *rdfs:Label* assignments. The different rules and axioms elaborated in this ontology all behaved as expected and there was no significant room for consistency issues at this stage of the prototype.

The results show a certain confidence in the validation mechanism where most if not all requirements can be transcribed in some way through standardized parametric specifications. On the other hand, the verification is highly limited to the extent of the expressivity of the IFCs. In other words, while the ontology can infer information about the input data, it is not meant to extrapolate new data from the existing one. In parallel, where design information such as mechanical properties are required, the ontology is limited since the scope of IFCs is limited to geometric representation.

8.2 Discussion over the prototype information management system

8.2.1 Smart Data vs. Smart Logic

Arguably the most pondering difficulty when structuring an ontology is the inherent dilemma between more extensive and complex data or smarter processing rules. Is it that the corporation using the ontology expects users to specify very low level information on their work/products and will infer complex knowledge using advanced rules and mechanism. Or is it rather that the entity expects users to specify each and every detail of their work. If so, who is responsible for the validation of that information?

This intrinsic conundrum is highly related to the paradox between the need to standardize information while at the same time being able to consistently record all possible project environments and processes. The best case scenario is to let all the stakeholders work with their level of complexity on their own information while requiring that the exchanged functional part is conditioned (i.e. standardized) to be processed through minimal logic rules. Ultimately, this positioning is one that has to be decided by the commissioning body. As such, RWS has a central position in the development of smart technologies within the framework of the BIM development.

8.2.2 Availability, Complexity and Functionality

Semantic technologies are widely available given there open standard distribution (Hitzler, et al. 2012) which makes them a perfect candidate for the development of concurrent design model validation code. On the other hand, very few commercially available engineering and analysis software have yet developed a compatibility or the ability to export their results into RDF. This process gap has been the start of an industry wide opportunity with the

developments of semantic oriented businesses in the field of engineering information management.

By far, the greatest selling points of W3C technologies are their capacity to connect information resources into a network of data. RDF allows proprietary agents to share and communicate universally. This proof of concept shows that this innovation reduces the steps required to manually exchange information and therefore reduces the transcription errors that may naturally occur as well as improving the operability of the data. The web ontology language offers an added level of reasoning capabilities that extend the scope of RDF not only to an exchange framework but also allows some reasoning and operations over the data. Over the range of problems encountered throughout this report it was shown that the OWL Manchester syntax and the SPIN notation were sufficiently complementary to explicitly define necessary and sufficient conditions to assess whether an instance resource met its design requirements.

The reasoning behind most semantic applications programming is fairly similar to that of any other programming technology. There is no apparent complexity barrier to the adaptation of linked data initiatives to the domain of engineering. However, there is a fundamental difference in the storing mechanism and the relational referencing. These differences, namely, the use of native tuple stores and the graph based referencing, introduce different approaches to deal with, existence, completeness and consistency which are a critical part of any engineering design.

8.2.3 Data Consistency & Process Performance

As shown throughout this work, the description logic of the web ontology language can tackle some of these data issues. One can easily check for the cardinality and the completeness of a design. However, there is yet no way to check for the consistency between engineering units since the RDF framework does not allow to predicate literals (i.e. values). This drawback would have to be fixed through reification but as mentioned before it is the author's opinion that reification should be limited to engine inferencing to avoid maintenance and logic difficulties. As such, there is a contradiction between the need to reify and the manual input nature of the literals.

Additionally, lacking in this report is the ability to compare and map instance resources (i.e. IFC elements) directly to the ontology, or for that matter, to 'check' that instances are indeed

instances of the class they have been assigned to. This lack of functionality introduces consistency issues since it is impossible to check the similitude or difference between multiple resources introduced by different parties. One possible approach to tackle these consistency issues is to develop the use of OTLs. These project wide libraries are used to semantically enrich the definition of instance resources. As such, the precise meaning of an object definition is always accessible through a straightforward query. These OTLs, if aligned with the IFCs may prove particularly useful in the context of infrastructure. In parallel, the development of 3D OTLs will further help the classification of geometric object and may help improve the functionality and spread the use of W3C technologies in the field of construction within the context of BIM.

While the logic reasoning of the proof of concept can be transcribed to other applications, the process shown in the report is not yet universal. For example, the query shown in section 7.1.4 for model verification defines a very specific property path which assumes the use of a specific IFC type of object representation (i.e. IfcPolyLoop). As a result, this execution would fail in the case other representation types are used. Another downside, is the difficulty to map complex EXPRESS data aggregation types (list, array, sets) to RDF (Pauwels, Terkaj, et al. 2015). As proposed in this prototype the export mechanism offers weak consistency and would require further attention as to its strict semantics for universal applications. For example, there is no testing for the ‘well-definition’ or boundary conditions of the coordinate list.

At last, the proposed prototype ontology and the associated query were elaborated with little consideration of the execution performance. One might expect that as the dataset increases the lack of optimization will yield a poor operating performance. The possible optimizations are twofold, a stricter data structure regarding execution (segregation, classification) as well as smarter distribution patterns, particularly with the use of named graphs for the queries.

8.2.4 W3C and BIM

The prototype concept shows the limitation of verification methods with OWL logic vocabulary. Indeed, while a majority of requirements can be validated through the comparison of asserted information and asserted requirements, SPIN rules are inadequate to perform complex mathematical analysis. Instead, the most promising approach to extent the

applicability of RDF/OWL (Beetz, Vries and Leeuwen 2007) is the development of functional part specification from proprietary agents.

In the current state of affair various different work groups are trying to develop standardized lists of properties, in particular in the field of suppliers to promote the use of BIM. These lists are almost all exclusively transcribed into IFC properties. The conceptual paradigm is that IFCs do not and will never have the functionality to adequately transcribe these properties due to the lack of semantics of the schema. Ontologies on the other hand can help transcribe these properties into smart libraries with rich semantics. In the future, commonly available analysis software will develop semantic export capabilities through smarter functional parts. Complex analysis results could then be exchanged and verified directly through semantic ontologies based on these property libraries.

8.2.5 Scope & Scale Prospect

This work confirms that the semantic web technology can be used to improve the system engineering decomposition of infrastructure projects (Dijkstra 2015) and that the concepts laid out in (Tudorache 2006) are applicable to the information management system of RWS for the development of a standard navigation lock ontology. A major and recurrent complexity with linked data technologies however, is the expected size of a realistic ontology and the related maintenance of this database. In order to be more practical one needs to develop mechanism to segregate parts of the design model consistently (Beetz, Vries and Leeuwen, RDF-based distributed functional part specifications for the facilitation of service-based architecture 2007).

In parallel, this work shows to some extent that distributed graph reasoning over the RDF framework is applicable to corporate work. However, at the process level, one can imagine that a very rigorous work ethic and work method need to be achieved to avoid errors and misinterpretations. The development of SHACL specifications which is a RDF validation specification may introduce new tools to improve the maintainability of the models (Knublauch and Kontokostas 2017). In addition, the industry needs to develop best practices and protocols as to how to implement RDF/OWL technologies with current BIM technologies.

At the corporate level, the recording of requirement in the system, client and standard specification is currently inadequate. The natural language specification are too implicit and

not conceptualize enough to be stored as a library with an object-oriented modelling technology (Borst 2015). The experiment has shown that with the current reasoning level, the complexity to consistently define bottom-up axioms to check that bottom requirements meet the breadth and complexity of top requirements, or to check that derived requirements fulfil the scope of their parent requirements is beyond the capacity of the technology. As such, the human critical expertise is still fully complementary to the project information models

9. CONCLUSION

This work is a brief window on the complexity of data modelling in the field of the construction industry. It is evident that the topic will yet deliver many research opportunities in the future and bear many deliverables (standards, specifications, best practices) in the sole purpose of promoting quality design through a smarter project knowledge integration and consideration. At the level of a commissioning body such as RWS, the modelling process can benefit the drafting and checking of requirements.

Requirements can be recorded in an ontology where they are classified using the RDFs object-oriented modelling specifications. The meta-data and logic description assigned to the requirement classes using OWL vocabulary formulated through SPARQL inferencing notation rules are used as checking mechanism. The prototype experimentation yields encouraging results but also uncovers specificities and complexities of the initiatives that require further attention.

In highlights, the operability of the OWL logic vocabulary is reduced to validation of project information and is in no way designed as an analysis language which limits its use for verification. The process for data extractions (i.e. mapping between the OWL and IFC) need to be further elaborated to align the complex aggregation mechanisms (consistency and performance). The assertion of project requirements and of requirement decomposition is unrestricted and uncontrolled, while bottom-up validation of design information is still a structural challenge. At last, the property definition of project concepts and requirements needs a better semantic definition.

However, as the use of RDF spreads through CAD agents, its open distribution will allow greater inter-operability between proprietary tools and a more network-like concurrent work process. As such, the distribution and expressivity of semantic web technologies offer great opportunities in the field of BIM. Additionally, the semantic web could also play a role in connecting different AEC initiatives (internet of things, SMART cities) and the core production tasks.

Overall, the prototype delivered in this report could help develop a more consistent requirement drafting process, and as such, contribute to the development of better, safer and more performant project designs. Yet another application of ICT technologies that will benefit the engineering domain. However, the experiment has shown that the critical thinking power of the engineer is always required to oversee the mindless execution of the machine.

At last, to conclude this report, I would like to, again, express my sincere gratitude towards any and all the person who have brought me assistance in my endeavour. I strongly feel that this work is innovative and attempts to fulfil a growing industry need. I would love to see this excerpt continued and extended with a more comprehensive doctoral work. As I have carried this work I have met many industry actors that have been introduced to the ideas of the semantic web but who are still generally foreign to the core benefits of structured semantic data. I hope that this report has bridged the gap for some of them.

REFERENCES

Allemang, D, and J Hendler. 2008. *Semantic web for the working ontologist.* Morgan Kaufmann.

Balmelli, L. 2007. "An overview of the systems modelling language for products and systems development." *Journal of Object Technology.*

Beetz, J. 2014. "A scalable network of concept libraries using distributed graph databases." *International Conference on Computing in Civil and Building Engineering.*

Beetz, J, BD Vries, and JV Leeuwen. 2007. "RDF-based distributed functional part specifications for the facilitation of service-based architecture." *Proceedings of the 24th W78 Conference.* Maribor, Slovenia.

Beetz, J, JV Leeuwen, and BD Vries. 2009. "IfcOWL a case of transforming EXPRESS schemas into ontologies." *Developing and Using Engineering Ontologies.*

Borst, ED. 2015. *Koppeling bedrijfsdatabase met eisenbibliotheek.* BIM Locket.

Brickley, D, RV Guha, and B McBride. 2014. *RDF Schema 1.1.* W3C.

buildingSMART. 2017. *www.buildingsmart.org.* http://www.buildingsmart-tech.org/.

Castaneda, V, L Ballejos, ML Caliusci, and MR Galli. 2010. "The use of ontologies in requirements engineering." *Global Journal of Researches in Engineering.*

CB-NL. n.d. *www.public.cbnl.org.*

COINS Task Group. n.d. *www.coins.nl.*

Delligatti, L. 2013. *SysML Distilled.* Addison-Wesley.

Dietz, JLG. 2005. *Enterprise Ontology.* ICEIS 2005.

Dijkstra, M. 2015. “DSM based identification of reliability and availability risks within a lock's dependency structure: a case study on lock system Eefde .”

Eastman, C, P Teicholz, R Sacks, and K Liston. 2011. *BIM Handbook: A guide to building information modelling for owners, managers, designers, engineers and contractors.* Wiley.

EP Karan, J Irizarry. 2015. “Extending BIM interoperability to preconstruction operations using geospatial analyses and semantic web services.” *Automation in Construction.*

Gielingh, W. 2008. “An assessment of the current state of product data technologies.” *Computer-aided Design.*

Gielingh, W. 1988. “General AEC Reference Model (GARM).” *Construction Informatics Digital Library.*

Gielingh, Wim. 2008. “A theory for the modelling of complex and dynamic systems.”

Hendler, J, and D Allemang. 2008. *Semantic Web for the Working Ontologist* . Morgan Kaufmann.

Hitzler, P, M Krötzsch, B Parsia, and S Rudolph. 2012. “OWL2 Web Ontology Language Primer.” *W3C Recommendation.*

Horridge, M, N Drummond, J Goodwin, A Rector, R Stevens, and H Wang. 2006. “The Manchester OWL Syntax.”

Hull, Elizabeth, Ken Jackson, and Jeremy Dick. 2011. *Requirements Engineering.* Springer.

I Hijazi, M Ehlers, S Zlatanova, U Isikdag. n.d. “IFC to CityGML transformation framework for aeo-analysis: a water utility network case.”

J Beetz, W Coederbergh van den Braak, R Botter, S Zlatanova, R de Laat. n.d. “Interoperable data models for infrastructural artefacts - a novel IFC extension method using RDF vocabularies exemplified with quay wall structures for harbor.”

J Benner, A Geiger, K Leinemann. 2005. “Flexible generation of semantic 3D building models.” *1st international workshop on next generation 3D city models.* Bonn.

Karan, EP, and J Irizarry. 2015. “Extending BIM interoperability to preconstruction operations using geospatial analyses and semantic web services.” *Automation in Construction.*

Karlshoj, J. 2011. *Information Delivery Manual: Roadmap.* buildingSMART.

Kiko, K, and C Atkinson. 2008. *A detailed comparison of UML and OWL.* University of Mannheim .

Klyne, G, JJ Carol, and B McBride. 2014. *RDF 1.1 Concepts and Abstract Syntax.* W3C.

Knublauch, H, and D Kontokostas. 2017. “Shapes Constraint Language (SHACL).” *W3C.*

Kossiakoff, A, WN Sweet, SJ Seymour, and SM Biemer. 2011. *Systems engineering principles and practice.* Wiley.

Krogstie, J, C Veres, and G Sindre. n.d. “Interoperability through integrating Semantic Web Technology, Web Services, and Workflow Modelling.”

Lee, SW, and RA Gandhi. 2005. “Ontology-based active requirements engineering framework.” *APSEC 05.*

Lin, J, MS Fox, and T Bilgic. 2016. “A requirement Ontology for Engineering Design.”

Miles, A, B Matthews, M Wilson, and D Brickler. 2005. “SKOS core: simple knowledge organisation for the web.” *Proceedings International Conference on Dublin Core and Metadata application.*

Motik, B, and B Parsia. 2012. “OWL2 Web Ontology Language.” *W3C.*

Noy, N, and A Rector. 2006. *Defining N-ary Relations on the Semantic Web.* W3C.

OMG. 2015. “OMG Systems Modelling Language (OMG SysML).”

Pauwels, P, and D Van Deursen. 2012. “IFC-to-RDF: adaptation, aggregation and enrichment.”

Pauwels, P, T Mendes, C Zhang, A Roxin, J Beetz, and JD Roo. 2016. “Querying and reasoning over large scale building data sets: an outline of a performance benchmark.”

Pauwels, P, W Terkaj, TF Krijnen, and J Beetz. 2015. “Coping with lists in the ifcOWL ontology.” *Proceedings of the 22nd EG-ICE Workshop.* Eindhoven.

Pauwels, P., R. De Meyer, and J. Van Campenhout. 2010. “Interoperability for the design and construction industry through semantic web technology.” *Proceedings of the 5th International Conference on Semantic and Digital Media Technologies.*

Port of Antwerp. 2016. *Kieldrecht Lock | Port of Antwerp.* http://www.portofantwerp.com/en/news/inauguration-kieldrecht-lock.

Pratt, MJ. 1998. “Extension of the standard ISO 10303 (STEP) for the exchange of parametric and variational CAD.” *Proceedings of the tenth international IFIP WG 5.2/5.3 conference.* Prolamat.

Rijkswaterstaat. 2012. “Basisspecificatie Schutsluis.”

Rijkswaterstaat. 2015. “De Sluis van de Toekomst.”

Rijkswaterstaat. 2015. “MultiWaterWerk - Leren van Andere Beheerders.”

Rijkswaterstaat. 2016. “Procesbeschrijving systems engineering voor RWS projecten.”

Rijkswaterstaat. 2016. “Syntheserapport - Onderzoeksfase MultiWaterWerk.”

Rijkswaterstaat. 2017. *Bouwwerk Informatie Management.* http://www.rijkswaterstaat.nl/zakelijk/werken-aan-infrastructuur/efficienter-werken/bouwwerk-informatie-model/.

Rijkswaterstaat. 2016. *Publications | Rijkswaterstaat.* http://www.rijkswaterstaat.nl/english/about-us/publications/index.aspx.

Siegemund, K, EJ Thomas, Y Zhao, J Pan, and U Assmann. 2011. “Towards ontology-driven requirement engineering.”

Tudorache, T. 2006. “Employing Ontologies for an Improved development process in collaborative engineering.”

W3C. 2009. *SKOS Simple Knowledge Organization System Primer.* https://www.w3.org/TR/skos-primer/.

W3C. 2017. *World Wide Web Consortium.* https://www.w3.org/.

Wache, H. 2004. “Semantische Mediation fur heterogene informationsquellen.”

Zhang, C, J Beetz, and B de Vries. 2014. “An ontological approach for semantic validation of IFC models.”

APPENDIX A

RWS core SBS obtained from (Dijkstra 2015)

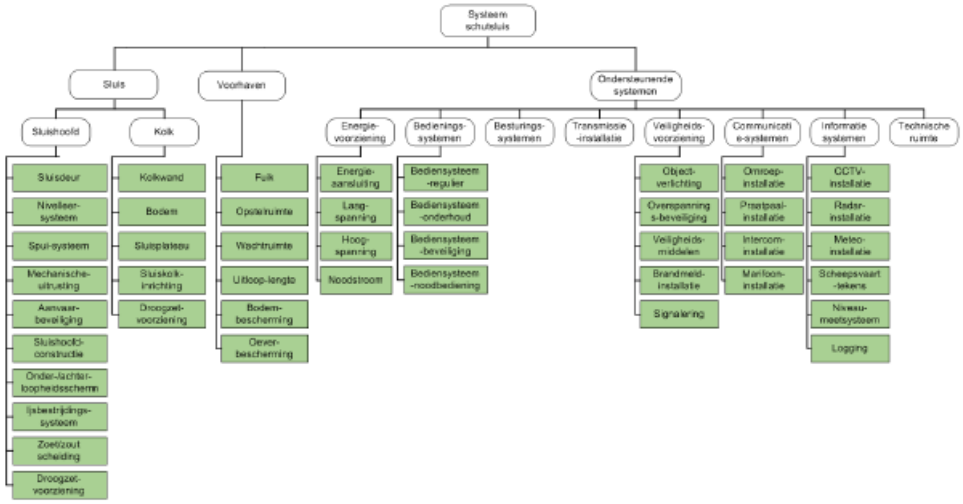


Figure 4.3: Basis specification decomposition of a generic lock (Rijkswaterstaat, 2012b).

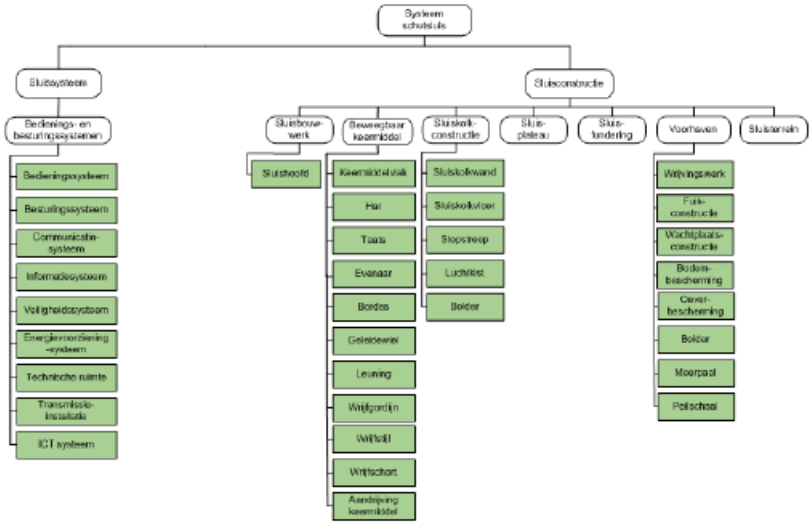


Figure 4.4: OTL decomposition of a generic lock (Rijkswaterstaat, 2014b).

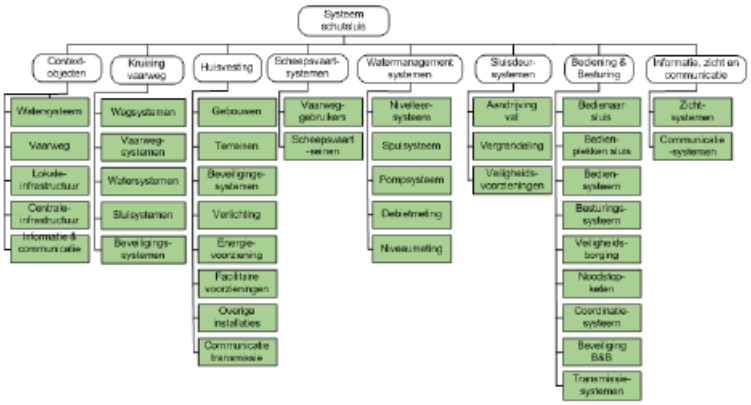


Figure 4.5: Grip decomposition of a generic lock (Rijkswaterstaat, 2014a).

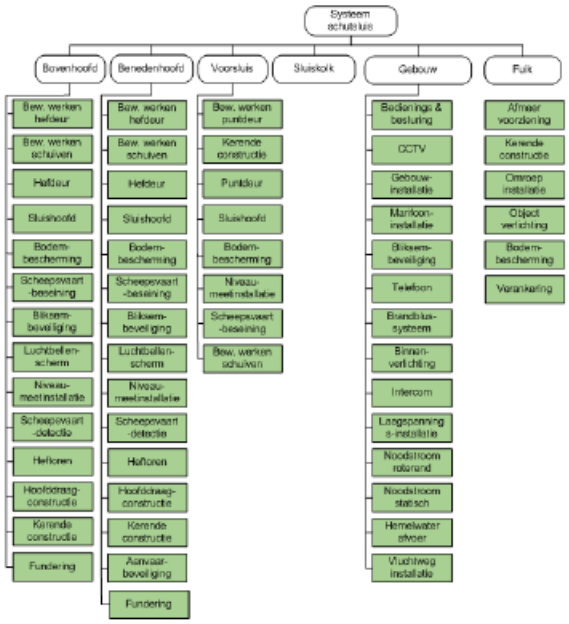


Figure 4.6: NEN 2767 decomposition of lock system Eefde (Rijkswaterstaat, 2013).

APPENDIX B

List of SPIN constraint query rules:

```
# EqualPara Axiom
CONSTRUCT {
   ?instance a MWWConstraints:EqualPara .
}
WHERE {
   ?instance MWWConnections:hasRequirement ?requirement .
   ?instance MWWConnections:hasObject ?object .
   ?requirement MWWRequirement:onProperty ?property .
   ?requirement MWWConstraints:hasConstraint ?constraint .
   ?constraint MWWConstraints:onConstraint MWWConstraints:EqualPara .
   ?constraint MWWConstraints:hasParametric ?parametric .
   ?parametric MWWConstraints:targetObject ?objectparameter .
   ?parametric MWWConstraints:targetProperty ?propertyparameter .
   ?objectparameter ?propertyparameter ?parameter .
   ?object ?property ?value .
   FILTER (?parameter = ?value) .
}
```

```
# Exact Value Range Axiom
CONSTRUCT {
   ?instance a MWWConstraints:ExactValue .
}
WHERE {
   ?instance MWWConnections:hasRequirement ?requirement .
   ?instance MWWConnections:hasObject ?object .
   ?requirement MWWRequirement:onProperty ?property .
   ?requirement MWWConstraints:hasConstraint ?constraint .
   ?constraint MWWConstraints:onConstraint MWWConstraints:ExactValue .
   ?constraint MWWConstraints:hasParametric ?parametric .
   ?parametric MWWConstraints:exactValue ?parameter .
   ?object ?property ?value .
   FILTER (?value != ?parameter) .
}
```

```
# Exclusive Range Axiom
CONSTRUCT {
   ?instance a MWWConstraints:ExclusiveRange .
}
WHERE {
   ?instance MWWConnections:hasRequirement ?requirement .
   ?instance MWWConnections:hasObject ?object .
   ?requirement MWWRequirement:onProperty ?property .
   ?requirement MWWConstraints:hasConstraint ?constraint .
   ?constraint MWWConstraints:onConstraint MWWConstraints:ExclusiveRange .
   ?constraint MWWConstraints:hasParametric ?parametric .
   ?parametric MWWConstraints:minValue ?minparameter .
   ?parametric MWWConstraints:maxValue ?maxparameter .
   ?object ?property ?value .
   FILTER ((?minparameter < ?value) && (?value < ?maxparameter)) .
}
```

```
# Inclusive Range Axiom
CONSTRUCT {
   ?instance a MWWConstraints:InclusiveRange .
}
WHERE {
   ?instance MWWConnections:hasRequirement ?requirement .
   ?instance MWWConnections:hasObject ?object .
   ?requirement MWWRequirement:onProperty ?property .
   ?requirement MWWConstraints:hasConstraint ?constraint .
   ?constraint MWWConstraints:onConstraint MWWConstraints:InclusiveRange .
   ?constraint MWWConstraints:hasParametric ?parametric .
   ?parametric MWWConstraints:minValue ?minparameter .
   ?parametric MWWConstraints:maxValue ?maxparameter .
   ?object ?property ?value .
   FILTER ((?minparameter <= ?value) && (?value <= ?maxparameter)) .
}
```

```
# Lower Bound Axiom
CONSTRUCT {
   ?instance a MWWConstraints:LowerBound .
}
WHERE {
   ?instance MWWConnections:hasRequirement ?requirement .
   ?instance MWWConnections:hasObject ?object .
   ?requirement MWWRequirement:onProperty ?property .
   ?requirement MWWConstraints:hasConstraint ?constraint .
   ?constraint MWWConstraints:onConstraint MWWConstraints:LowerBound .
   ?constraint MWWConstraints:hasParametric ?parametric .
   ?parametric MWWConstraints:minValue ?minparameter .
   ?object ?property ?value .
   FILTER (?minparameter <= ?value) .
}
```

```
# Lower Para Axiom
CONSTRUCT {
   ?instance a MWWConstraints:LowerPara .
}
WHERE {
   ?instance MWWConnections:hasRequirement ?requirement .
   ?instance MWWConnections:hasObject ?object .
   ?requirement MWWRequirement:onProperty ?property .
   ?requirement MWWConstraints:hasConstraint ?constraint .
   ?constraint MWWConstraints:onConstraint MWWConstraints:LowerPara .
   ?constraint MWWConstraints:hasParametric ?parametric .
   ?parametric MWWConstraints:targetObject ?objectparameter .
   ?parametric MWWConstraints:targetProperty ?propertyparameter .
   ?objectparameter ?propertyparameter ?parameter .
   ?object ?property ?value .
   FILTER (?parameter <= ?value) .
}
```

```
# Requirement Property Inferencing Axiom
CONSTRUCT {
   ?this a ?class .
}
WHERE {
   ?class MWWRequirement:definedBy ?property .
   ?this MWWConnections:hasObject ?object .
   ?this MWWConnections:hasRequirement/MWWRequirement:onProperty ?property .
   ?object ?property ?unknown .
   FILTER NOT EXISTS {
      ?this ?property ?m .
      FILTER (?unknown = ?m) .
   } .
}
```

```
# Step Range Axiom
CONSTRUCT {
   ?instance a MWWConstraints:StepConstraint .
}
WHERE {
   ?instance MWWConnections:hasRequirement ?requirement .
   ?instance MWWConnections:hasObject ?object .
   ?requirement MWWRequirement:onProperty ?property .
   ?requirement MWWConstraints:hasConstraint ?constraint .
   ?constraint MWWConstraints:onConstraint MWWConstraints:StepConstraint .
   ?constraint MWWConstraints:hasParametric ?parametric .
   ?parametric MWWConstraints:stepParameter ?parameter .
   ?object ?property ?value .
   BIND ((?value / ?parameter) AS ?result) .
   FILTER (abs((?result - round(?result))) = 0) .
}
```

```
# Upper Bound Axiom
CONSTRUCT {
   ?instance a MWWConstraints:UpperBound .
}
WHERE {
   ?instance MWWConnections:hasRequirement ?requirement .
   ?instance MWWConnections:hasObject ?object .
   ?requirement MWWRequirement:onProperty ?property .
   ?requirement MWWConstraints:hasConstraint ?constraint .
   ?constraint MWWConstraints:onConstraint MWWConstraints:UpperBound .
   ?constraint MWWConstraints:hasParametric ?parametric .
   ?parametric MWWConstraints:maxValue ?maxparameter .
   ?object ?property ?value .
   FILTER (?maxparameter >= ?value) .
}
```

```
# Upper Para Axiom
CONSTRUCT {
   ?instance a MWWConstraints:UpperPara .
}
WHERE {
   ?instance MWWConnections:hasRequirement ?requirement .
   ?instance MWWConnections:hasObject ?object .
   ?requirement MWWRequirement:onProperty ?property .
   ?requirement MWWConstraints:hasConstraint ?constraint .
   ?constraint MWWConstraints:onConstraint MWWConstraints:UpperPara .
   ?constraint MWWConstraints:hasParametric ?parametric .
   ?parametric MWWConstraints:targetObject ?objectparameter .
   ?parametric MWWConstraints:targetProperty ?propertyparameter .
   ?objectparameter ?propertyparameter ?parameter .
   ?object ?property ?value .
   FILTER (?parameter >= ?value) .
}
```

APPENDIX C

List of SPIN verification query rules:

```
# Object Width verification
CONSTRUCT {
   ?instance MWWRequirement:isVerified "True" .
}
WHERE {
   ?instance MWWConnections:hasObject ?object .
   ?instance MWWConnections:hasRequirement/MWWRequirement:onProperty :hasWidth .
   ?object :hasWidth ?value .
   FILTER (?result = ?value) .
   {
      SELECT ?object ((MIN(?distance)) AS ?result)
      WHERE {
         ?object :hasXCoordinate ?x1 .
         ?object :hasXCoordinate ?x2 .
         FILTER (?x2 != ?x1) .
         BIND (abs((?x1 - ?x2)) AS ?distance) .
      }
      GROUP BY ?object
   } .
}
```

```
# Object height verification
CONSTRUCT {
   ?instance MWWRequirement:isVerified "True" .
}
WHERE {
   ?instance MWWConnections:hasObject ?object .
   ?instance MWWConnections:hasRequirement/MWWRequirement:onProperty :hasHeight .
   ?object :hasHeight ?value .
   FILTER (?result = ?value) .
   {
      SELECT ?object ((MIN(?distance)) AS ?result)
      WHERE {
         ?object :hasZCoordinate ?x1 .
         ?object :hasZCoordinate ?x2 .
         FILTER (?x2 != ?x1) .
         BIND (abs((?x1 - ?x2)) AS ?distance) .
      }
      GROUP BY ?object
   } .
}
```

```
# Object length verification
CONSTRUCT {
   ?instance MWWRequirement:isVerified "True" .
}
WHERE {
   ?instance MWWConnections:hasObject ?object .
   ?instance MWWConnections:hasRequirement/MWWRequirement:onProperty :hasLength .
   ?object :hasLength ?value .
   FILTER (?result = ?value) .
   {
      SELECT ?object ((MIN(?distance)) AS ?result)
      WHERE {
         ?object :hasYCoordinate ?x1 .
         ?object :hasYCoordinate ?x2 .
         FILTER (?x2 != ?x1) .
         BIND (abs((?x1 - ?x2)) AS ?distance) .
      }
      GROUP BY ?object
   } .
}
```

APPENDIX D

List of SPIN reification axioms:

```
# Functional-Role Reification Axiom
CONSTRUCT {
   _:b0 MWWConnections:function-Role ?instance .
   _:b0 MWWConnections:hasObject ?instance .
   _:b0 MWWConnections:hasRole ?role .
   _:b0 MWWConnections:hasFunction ?function .
   _:b0 MWWConnections:hasParameter ?parameter .
   _:b0 rdfs:label ?label .
}
WHERE {
   ?class MWWProject:hasRole ?role .
   ?instance a ?class .
   OPTIONAL {
      ?role MWWProject:hasFunction ?function .
   } .
   OPTIONAL {
      ?function MWWProject:hasParameter ?parameter .
   } .
   BIND (((str(?instance) + "-") + str(?role)) AS ?label) .
   FILTER NOT EXISTS {
      _:0 rdfs:label ?label .
   } .
}
```

```
# Resource-Specification Reification Axiom
CONSTRUCT {
   _:b0 MWWConnections:resource-Specification ?instance .
   _:b0 MWWConnections:hasObject ?instance .
   _:b0 MWWConnections:hasRequirement ?requirement .
   _:b0 MWWConnections:hasConstraint ?constraint .
   _:b0 MWWConnections:hasParametric ?parameter .
   _:b0 rdfs:label ?label .
}
WHERE {
   ?class MWWRequirement:hasRequirement ?requirement .
   ?instance a ?class .
   ?class (rdfs:subClassOf)* :ProductConcept .
   OPTIONAL {
      ?requirement MWWConstraints:hasConstraint ?constraint .
   } .
   OPTIONAL {
      ?constraint MWWConstraints:hasParametric ?parameter .
   } .
   BIND ((str(?instance) + str(?requirement)) AS ?label) .
   FILTER NOT EXISTS {
      _:0 rdfs:label ?label .
   } .
}
```

```
# System-Boundary Reification Axiom
CONSTRUCT {
   _:b0 MWWConnections:hasSystem ?system1 .
   _:b0 MWWConnections:hasSystem ?system2 .
   _:b0 MWWConnections:hasObject ?instance .
   _:b0 MWWConnections:system-Boundary ?instance .
   _:b0 rdfs:label ?label .
}
WHERE {
   ?system1 :hasObject ?instance .
   ?system2 :hasObject ?instance .
   BIND ((((str(?system1) + "_") + str(?system2)) + "_Boundary") AS ?label) .
   FILTER NOT EXISTS {
      ?node rdfs:label ?label .
   } .
   FILTER (?system1 != ?system2) .
}
```