MASTER

Automatic validation of technical requirements for a BIM model using semantic web technologies
Description of a linked building data workflow using SHACL as checking mechanism to improve information exchanges between actors in the AEC industry

van den Bersselaar, E.

Award date:
2022

Link to publication

Disclaimer
This document contains a student thesis (bachelor's or master's), as authored by a student at Eindhoven University of Technology. Student theses are made available in the TU/e repository upon obtaining the required degree. The grade received is not published on the document as presented in the repository. The required complexity or quality of research of student theses may vary by program, and the required minimum study period may vary in duration.

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
AUTOMATIC VALIDATION OF TECHNICAL REQUIREMENTS FOR A BIM MODEL USING SEMANTIC WEB TECHNOLOGIES

Description of a linked building data workflow using SHACL as checking mechanism to improve information exchanges between actors in the AEC industry

Student
Bersselaar, E. van den
0894574
e.v.d.bersselaar@student.tue.nl

Graduation committee
Prof. Dr. Ir. B. de Vries (TU/e)
Ir. Dr. P. Pauwels (TU/e)
Prof. Dr. M.R.V. Chaudron (TU/e)
Ir. B.J.A. van Thiel (BIM-Connected)

Supervisor company
Ing. L.B. Verhelst (BIM-Connected)

University
Eindhoven University of Technology
Master Construction Management and Engineering

Company
BIM-Connected

ECTS
40

Academic year
2021/2022

Date final presentation
29-06-2022
PREFACE

Dear reader,

After eight years of walking around at Eindhoven University of Technology, this master thesis marks the closing of a fantastic time as a student in the Built Environment faculty. I am proud to present this work as the final project for the master Construction Management and Engineering following nine months of reading, writing, experimenting, and learning. During this time, I have developed myself on both a professional level as well as personally and discovered that the scope of this thesis is the direction I want to pursue while taking the first baby steps towards a professional career.

In my opinion, it is the perfect ending of my student life, which started in 2014 when I still thought I was going to be an architect. It only took a few courses and a single project to realize that architecture and Ellen were not a great match, so the rest of my bachelor my mind was seduced by mechanics, material properties and other structural design related themes. Then, I got distracted for two and a half years by what in my opinion turned out to be the best extra-curricular activity one can experience as a built environment student. Together with 15 others I founded VIRTUe, a student team that ended up participating in the Solar Decathlon competition in 2018 in Dubai with their own prototype of a sustainable home. During that period, I developed myself as project manager of the team, which made me realize that that was the course I should take for my master program. Construction Management and Engineering as master’s program thus was the logical decision, where the especially the lectures of Pieter Pauwels piqued my interest. The challenges for the construction industry with regard to Building Information Management, the difficulties in data and information modelling, and the issues around exchanging that information between involved parties in a project; those became the topics that gave me drive and energy.

And they still do. I have often heard peers, friends and other graduates cursing their graduation subject towards the end of their process, but for me that is definitely not the case. I have really enjoyed working on this thesis and that is, among other things as well, also surely attributable to the pleasant sessions I have had with my supervisors. For that, I want to specifically thank Pieter Pauwels for inspiring me with your lectures, articles and in depth knowledge, and additionally for all the feedback sessions, fast replies on MS Teams, and general support during these two years at CME. Then, I also want to express my appreciation to my second TU/e supervisor, Michel Chaudron. Thank you for providing monthly feedback that helped me in managing the scope of the project and kept me from diving too deep into detailed matters. Next up are Bob van Thiel and Lucas Verhelst, my supervisors from BIM-Connected, the company where I have executed my graduation project and which will be the company I will join after finishing it. Bob, thank you for the opportunity to work on this thesis at BIM-Connected, for being able to partly work at the office and for always radiating positive energy. Lucas, thank you for stimulating me to get the most out of this and for cracking my brain by asking critical questions during our sessions. I think you kept a fantastic balance between guiding me into the right direction while focusing on all the technical details.

Lastly, I would like to express my gratitude to every other person that somehow contributed. My colleagues at BIM-Connected for the table football sessions, my friends for supporting me and distracting me with fun talks and activities and my family for understanding and listening to the difficulties I had during this time.

I hope you enjoy reading this work as much as I had when making it.

Ellen van den Bersselaar

Eindhoven, June 2022
TABLE OF CONTENTS

SUMMARY ...................................................................................................................... 7
SAMENVATTING ........................................................................................................... 10
ABSTRACT ...................................................................................................................... 11
OVERVIEW - ABREVIATIONS ...................................................................................... 14
OVERVIEW - FIGURES .................................................................................................. 15
OVERVIEW - TABLES .................................................................................................... 17
OVERVIEW - LISTINGS ................................................................................................. 18

CHAPTER 1 - INTRODUCTION ...................................................................................... 19

1.1. Background context
1.2. Problem statement
1.2.1. Research gap
1.2.2. Research objective
1.3. Research design and outline

CHAPTER 2 - LITERATURE RESEARCH - INFORMATION MODELING AND EXCHANGE IN THE AECO INDUSTRY .................................................................................. 23

2.1. Building Information Modeling
2.2. Industry Foundation Classes
2.3. Semantic Web and Linked Data
2.3.1. Resource Description Framework
2.3.2. Resource Description Framework Schema and Web Ontology Language
2.3.3. SPARQL
2.3.4. SHACL
2.4. Conclusion

CHAPTER 3 – LITERATURE RESEARCH - THE BIM BASED AUTOMATED QUALITY CHECKING PROCESS .............................................................................................. 38

3.1. Rule interpretation
3.2. Building model preparation
3.2.1. From IFC to Linked Building Data
3.2.2. Geometry
3.2.3. BIM model integrity
3.3. Rule execution
3.4. Rule reporting
3.5. Solutions for BIM compliance checking
3.6. Conclusion
CHAPTER 4 - METHODOLOGY .................................................................50

4.1. Methodological justification
4.2. Data collection
4.3. System architecture design
4.4. System architecture functionalities
   4.4.1. Rule interpretation
   4.4.2. Model preparation
   4.4.3. Rule execution
   4.4.4. Rule reporting
4.5. Case descriptions

CHAPTER 5 – PROTOTYPE PREREQUISITES .............................................61

5.1. Rule analysis
   5.1.1. Document S5084
   5.1.2. Document ID437
5.2. Transformer Ontology
   5.2.1. Class definitions
   5.2.2. Object-, Data and Datatype property definitions
5.3. Conclusion

CHAPTER 6 – PROTOTYPE DEVELOPMENT .............................................71

6.1. Case 1: Semantic check
6.2. Case 2: Geometric check
6.3. Case 3: Final prototype

CHAPTER 7 – REVIEW .................................................................94

7.1. Conclusion
7.2. Discussion and recommendations

REFERENCES .................................................................98

APPENDICES .................................................................102
Projects executed in the architecture, engineering, and construction (AEC) industry require a high level of collaboration between all involved parties. Due to the AEC industry’s nature, projects go through multiple phases and many experts with varying domain-specific knowledge are involved. During handover moments, information whether it is in the form of a drawing, calculation, analysis, report, or visualization, is passed from point A to point B. Imaginable, the amount of data and information that is generated during the course of a project, and thus also is exchanged, is enormous and it is desirable that information created in the beginning of the life cycle is still available at the end of the trajectory when the maintenance phase of the building starts.

By applying Building Information Management (BIM) strategies to an AEC project, the consequent re-use of digital information provides improvement in design activity coordination, calculation and simulation integration, construction management and in the handover to the building owner. To encourage this, the Industry Foundation Classes (IFC) were developed by buildingSMART. IFC is a data schema, defined and standardized in ISO 16739-1:2018 that can be used to describe construction-related information using an object-based hierarchy that encompasses a large number of entities. By following the IFC data structure, its entities can be used to describe tangible objects such as a ‘wall’ and ‘door’, but it is also capable of describing the more abstract relationship between these objects. In addition to its ability for expressive object modelling, the strength of the IFC schema lies in the way it can define a detailed representation of those objects’ geometrical properties. Eventually, a rich BIM model is created that can be saved in a neutral file format ‘IFC-Step Physical File’ (IFC-SPF) with the extension ‘.ifc’. The entire underlying vision for this was to enable the exchange of building information between different software applications, which would up until then all save their contents in a proprietary file format that is not readable by non-native programs.

Although this implementation of BIM improved the quality of information exchange drastically, the academic world suggests that an opportunity lies in the Semantic Web (SW) Technology Stack, which includes but is not limited to the Resource Description Framework (RDF), the Resource Description Framework Schema (RDFS), the Web Ontology Language (OWL) and the Shapes Constraint Language (SHACL). RDF in combination with its subsequential building blocks from the SW Technology Stack are a strong data modelling technique. The combination of declaring logic-based statements with RDF and possibility to do this while combining information from multiple application areas can play a key role in the future of information exchange in the AEC industry. Several of the current difficulties that the IFC standard imposes on Building Information Modeling could be diminished by integrating SW technologies into the process of creating BIM models. The option to validate these statements with a rule language built on the same logical principles allows to set up of a strong rule checking environment which can significantly improve the efficiency and quality of data exchange activities.

Especially in regard of the latter, a great urgency exists for the improvement of the Quality Control (QC) process in the AEC industry since the number of requirements in projects is majorly exceeding the capabilities of a manual checking procedure, causing unwanted risks and errors in projects. Already in 2009, a framework was presented that is still supported by today’s literature, which describes four essential aspects of a rule checking system and their interdependencies. These aspects are (1) rule interpretation, (2) model preparation, (3) rule execution and (4) reporting results. The QC solutions presented in literature or those that are available on the market, often focus on one or only a few aspects of the complete QC process. Furthermore, a lack of options is recognized that allow for the programming of project specific rules, creating a dependency on the proprietary rulesets that are included in the software. Additionally rules are often documented only in human readable, textual documents for which an interpretation step is necessary in order for a computer to work with them. No general and standardized
classification exists regarding the typology of all these requirements, which would be useful for automating this rule checking process in the future.

A chance lies in more public, open and vendor-neutral approach, using SW technologies. By combining the strengths of these technologies with the established IFC data model which excels in geometric data modeling, a new rule checking process could be designed that proves the advantages of both approaches and additionally takes into account all necessary and relevant aspects. The main research question of this thesis arises from the gap in the academic field that neglects to address the challenges as stated above as a whole and is therefore: ‘What value can be created for the rule-compliance checking process during the design phase/handover phase of a construction project by using a workflow that involves semantic web technologies?’ The answer of this question will contribute to the ongoing improvement that is necessary in the AEC industry regarding information management in projects. This is done by developing a prototype which demonstrates the potential of using semantic web technologies in this industry. The prototype will have a narrowed scope that focusses on quality control of BIM models regarding technical requirements that are imposed on said model. This rule checking system will use semantic web technologies and shows the possibilities for quality control using SW technologies.

To answer this question, first a literature review is conducted that aims to establish a clear overview on the existing research that is done regarding BIM and information exchange within the building industry. Following that, the current state of the art is researched regarding the process of quality control of said information. Based on the conclusions drawn from this literature review, a system architecture is developed for a rule checking system that uses SW technologies in combination with IFC that goes through all four earlier mentioned aspects of such a system. Additionally, two prerequisite steps are executed before being able to develop the rule checking system according to that architecture. These two prerequisites are a thorough rule analysis and the creation of a vocabulary that bridges between the BIM model and the ruleset.

A method is created to make the resulting prototype of this thesis. The data that will be used is on the one hand a BIM model consisting of four separate IFC-SPF files and on the other hand a set of requirements that this BIM model should be validated against. This set of requirements are human written text documents saved in .pdf and .doc formats. The goal of the rule checking system prototype is to validate the BIM model against different types of rules, using the system architecture that followed from the literature review. To achieve this, three cases have been set up, which all cycle through the system architecture, but focus on different challenges that lie within the process. The first case performs a semantic check and concentrates on enriching the BIM model with properties that map targeted objects to the vocabulary as specified in the created ontology. In the second case, a technical requirement is validated that uses implicit geometric information from the BIM model. A process is demonstrated that captures this information from the IFC data, translates this into RDF data and thereby enriches the information model with otherwise hidden knowledge. Lastly, a set of rules is composed that encompasses all types as defined in the rule analysis. These are interpreted, the BIM model is prepared correctly for the check, the rule check is executed and a human readable report is generated that communicates eventual violations of the rule set. This completes the third case of this research.

By going through this process as a whole, this research contributes to the implementation of SW technologies into the current AEC industry process of data exchange and quality control. It is found that SHACL is a suitable rule checking language that can validate a substantial part of a document containing technical requirements. Furthermore, a clear observation can be made regarding the dependencies that lie in the BIM modelling phase that precedes the QC phase. The rule interpretation and model preparation steps are heavily reliant on the people who are involved and the industry’s nature makes it so that the agreements about these two steps are different for each different project. Due to a lack of understanding about the added value of data driven processed instead of working in a document based manner, there are little initiatives in standardizing and regulating the QC steps, which keeps the vicious circle alive and
makes it nearly impossible to automate this process. However, this is something that is essential to move forward as an industry as a whole, because the increasing complexity of projects and the ever growing set of demands and requirements which must be complied to cannot be handled manually.

This thesis ends with a final review containing a conclusion that answers the main research question and its corresponding sub research questions. This is followed by a discussion regarding this conclusion, the limitations involved in performing the research and recommendations that suggest further research on the topic of using SW technologies for QC in the building industry.
SAMENVATTING

Projecten die uitgevoerd worden in de architectuur-, ingenieurs-, en constructie- (AEC) industrie verlangen een hoog niveau betreft de samenwerking tussen alle belanghebbende partijen. Door het karakter van de AEC industrie hebben projecten meerdere opeenvolgende fases en is de betrokkenheid nodig van veel specialisten die variërende kennis hebben op hun domein van expertise. Op verschillende punten in het project zal informatie, of dit nu in de vorm van een tekening, berekening, analyse, rapport of visualisatie is, overgedragen moeten worden van punt A naar punt B. De hoeveelheid data en informatie die wordt gegenereerd over de tijdspan van een project en dus ook wordt uitgewisseld is enorm en het heeft de voorkeur dat informatie uit het begin van de levenscyclus van een project nog steeds beschikbaar is aan het einde van het traject wanneer de onderhoudsfase begint.

Door Building Information Management (BIM) strategieën toe te passen binnen een AEC project, kan het consequent hergebruiken van digitale informatie bijdragen aan de verbetering van onder andere coördinatie van het ontwerpproces, nauwkeurigheid van berekeningen, integratie van disciplines en de overdracht van informatie naar de gebouweigenaars. Om dit extra te stimuleren zijn de Industry Foundation Classes (IFC) ontwikkeld door buildingSMART. IFC is een data schema, gedefinieerd en gestandaardiseerd in ISO 16739-1:2018 en kan gebruikt worden om bouw gerelateerde informatie te beschrijven door middel van een object-gebaseerde hiërarchie die een grote hoeveelheid entiteiten omvat. Door de structuur van het IFC datamodel te volgen, kunnen deze entiteiten gebruikt worden om tastbare objecten te beschrijven zoals ‘muur’ en ‘deur’. Het is daarnaast ook mogelijk om meer abstracte concepten uit te drukken, zoals de relatie die bestaat tussen deze objecten. Naast de mogelijkheden voor het expressief modelleren van objecten, ligt de kracht van het IFC schema ook in de manier waarop het een gedetailleerde representatie kan definiëren van de geometrische eigenschappen van genoemde objecten. Uiteindelijk kan zo een draagkrachtig BIM model gemaakt worden dat in het neutrale bestandsformaat ‘IFC-Step Physical File’ (IFC-SPF) met de extensie ‘.ifc’ opgeslagen kan worden. De onderliggende visie is om het uitwisselen van gebouw gerelateerde informatie tussen verschillende software applicaties mogelijk te maken, wat voorheen alleen kon in het proprietaire formaat van de betreffende applicatie en dus niet leesbaar was voor andere programma’s.

Ondanks de verbetering in de kwaliteit van informatie uitwisseling die teweeg is gebracht door de implementatie van BIM, suggereert de academische wereld dat een extra kans ligt in de groep technologieën die bij het Semantische Web (SW) horen. Hieronder vallen onder andere het Resource Description Framework (RDF), het Resource Description Framework Schema (RDFS), de Web Ontology Language (OWL) en de Shapes Constraint Language (SHACL). RDF kan samen met de daaropvolgende bouwblokken van de groep SW technologieën een sterke manier voor data modelleren voorschrijven. De combinatie van het maken van op logica gebaseerde statements met RDF en de mogelijkheid om dit te doen terwijl data vanuit diverse applicatie gebieden gekoppeld wordt, speelt potentiële een sleutelrol in de toekomst voor informatie-uitwisseling in de AEC industrie. Meerdere van de moeilijkheden die het IFC data model opleggen op het BIM proces zouden teniet gedaan kunnen worden door correcte implementatie en integratie van SW technologieën in het huidige proces van BIM. De daar bijkomende mogelijkheid om deze statements te valideren met een op regels gebaseerde taal die gebouwd is op dezelfde logische basisprincipes biedt de mogelijkheid om een sterke eisen-check omgeving op te zetten die de efficiëntie en kwaliteit van data uitwisseling en controle significant kan verbeteren.

Vooral aangaande dit laatste bet staat er een hoge urgentie binnen de AEC industrie om het Quality Control (QC) proces te verbeteren, omdat de hoeveelheid eisen in projecten de grens van de menselijke capaciteit om deze manueel te checken overschrijdt. Dit zorgt voor onnodige en ongewilde risico’s en fouten in projecten. Al in 2009 is een kader gepresenteerd, dat hedendaags nog steeds door literatuur onderschreven wordt, dat vier essentiële aspecten en de onderlinge relatie daartussen beschrijft welke
terug moeten komen in een eisen-check systeem. Deze aspecten zijn (1) regel interpretatie, (2) model voorbereiding, (3) uitoever check, en (4) rapporteren van de resultaten. De QC oplossingen die in de literatuur voorgesteld worden of die al beschikbaar zijn in de huidige markt focussen vaak maar op één of een aantal van die vier aspecten. Verder wordt er een tekort aan mogelijkheden opgemerkt waarbij project specifieke eisen gecontroleerd kunnen worden, waardoor men afhankelijk is van de regelssets die vooraf geprogrammeerd zijn in de software applicatie die gebruikt wordt voor de validatie. Ook worden eisen vaak alleen op een door mensen te begrijpen manier gedocumenteerd in tekst-gebaseerde document waardoor een interpretatiestap vereist is voordat een computer ermee kan werken. Er bestaat geen generieke en gestandaardiseerde classificatie die eisen onderverdeeld in verschillende typen. Iets wat nuttig zou zijn voor het automatiseren van dit QC proces in de toekomst.

Er ligt een kans voor de verbetering hiervan in een generiekere aanpak, waarbij de focus ligt op open standaarden en onafhankelijkheid van proprietaire software leveranciers. Deze kans ligt het combineren van de kracht van SW technologieën met het bewezen IFC data model dat uitblikkt in het modelleren van geometrische eigenschappen. Zo kan een nieuw proces voor het valideren van eisen ontworpen worden dat de voordelen van beide aanpakken meeneemt en daarbij rekening houdt met de vier eerder genoemde aspecten. De hoofdvraag van dit onderzoek komt voort uit het missende stuk in de literatuur dat zich onthoudt van het aanstippen van bovenstaande uitdagen als integraal probleem en is daarom: *Welke waarde kan gecreëerd worden voor het eisenvalidatie proces tijdens de ontwerpfase/het overdrachtsmoment van een constructieproject door technologieën vanuit het Semantische Web toe te passen in de workflow?* Het antwoord op deze vraag draagt bij aan de continue verbetering die nodig is in de AEC industri betreft informatiemanagement in projecten. Dit wordt gedaan door een prototype te ontwikkelen dat de potentie demonstreert van de toepassing van SW technologieën in deze industrie. Het prototype wordt gemaakt binnen een begrensde scope die de focus legt op de kwaliteitscontrole van een BIM model aangaande technische eisen die aan dit model gesteld zijn. Dit systeem gaat SW technologieën toepassen en laat zien wat de toepassingsmogelijkheden hiervoor zijn betreft QC.

Om de hoofdvraag te beantwoorden wordt eerst een literatuur onderzoek uitgevoerd waarbij het doel is om een duidelijk overzicht te genereren van het bestaande onderzoeken die gedaan zijn op het onderwerp van BIM en informatie-uitwisseling binnen de AEC industrie. Daarop doorgaande wordt de ‘state of the art’ status van het proces rondom kwaliteitscontrole van die informatie bestudeerd. De conclusies die volgen vanuit dit literatuur onderzoek zijn leidend in de ontwikkeling van een systeem architectuur dat een regelvalidatie systeem beschrijft wat SW technologieën gebruikt in combinatie met IFC en door alle vier eerder genoemde aspecten heen beveegt. Bovenop dit regelvalidatie systeem worden twee vooraf vereiste stappen uitgevoerd die nodig zijn voordat aan de systeem architectuur gestart kan worden. Deze twee randvoorwaarden zijn het uitoeren van een regel-type analyse en het maken van een vocabulaire in de vorm van een ontologie welke een brug vormt tussen het BIM model en de set regels waaraan deze gecheckt wordt.

Een methodologie is vastgesteld om dit prototype te kunnen ontwikkelen als eindresultaat van dit onderzoek. De data die hiervoor gebruikt gaat worden aan de ene kant een BIM model dat opgebouwd is uit vier aparte IFC-SPF bestanden en aan de andere kant een set eisen waartegen dit model wordt gevalideerd. De set regels bestaat uit door mensen geschreven tekstuele documenten die opgeslagen zijn in .pdf en .doc formaten. Het doel van het regelvalidatie systeem is om het BIM model te kunnen valideren tegen verschillende typen regels waarbij de ontwikkelde systeem architectuur gebruikt wordt. Om dit te bereiken zijn drie casussen opgesteld welke allen de systeem architectuur doorlopen, maar alle drie op verschillende uitdagingen binnen het proces focussen. De eerste casus voert een semantische validatie uit en concentreert op het verreiken van het BIM model met eigenschappen die een mapping creëren tussen bepaalde benodigde objecten en de vocabulaire die gespecificeerd staat in de ontwikkelde ontologie. In de tweede casus wordt een technische eis gevalideerd waarbij impliciete geometrische informatie uit het BIM model gebruikt wordt. Een proces wordt gedemonstreerd dat deze informatie onttrekt aan de IFC data en dit converteert naar RDF data om zo het informatiemodel te verreiken met kennis die anders verborgen
was gebleven. Ten slotte wordt een set eisen samengesteld die alle typen regels omvat welke gedefinieerd waren in de regel type analyse. Deze worden geïnterpreteerd, het BIM model wordt correct voorbereid voor de validatie, de check wordt uitgevoerd en een door mensen te begrijpen rapport wordt gegenereerd dat eventuele overtredingen van de eisen communiceert. Dit completeert de derde casus van dit onderzoeksverslag.

Door dit hele proces in zijn geheel door te gaan, draagt dit onderzoek bij aan de integratie van SW technologieën met het huidige proces van de AEC industrie voor het uitwisselen van data en de controle daarvan. Het kan geconcludeerd worden dat SHACL een geschikte op regels-gebaseerde taal is dat een substantieel deel van een document met technische eisen kan valideren. Ook wordt geobserveerd dat er bepaalde afhankelijkheden bestaan voorafgaande aan het QC proces die al beginnen in de fasen waarin het BIM model gemaakt wordt. De regel interpretatie- en model voorbereidingsstappen zijn enorm afhankelijk van de mensen die betrokken zijn. Daarbij komt nog dat het karakter van de industrie ervoor zorgt dat er geen consistentie afspraken gemaakt worden over deze twee stappen en dus voor ieder project weer anders zijn. Door een gebrek aan begrip en kennis betreft de voordelen die een manier van data gedreven werken met zich mee kan brengen ten opzichte van een manier gebaseerd op werken met documenten, zijn er weinig initiatieven om de QC stappen te reguleren en te standaardiseren. Dit houdt deze vicieuze cirkel in stand en vermoeilt de automatisering van dit proces die nodig is om de industrie als geheel vooruit te bewegen. Dit zal uiteindelijk nodig gaan zijn, omdat de complexiteit van projecten blijft toenemen en de eeuwig groeiende set aan eisen en regels niet manueel behandeld kan worden.

Dit verslag eindigt met een review waarin antwoord gegeven wordt op de hoofdvraag en deelvragen in de conclusie en de discussie die daarop volgt wordt beschreven. Verder staan in deze review nog de beperkingen die van toepassing waren tijdens het uitvoeren van dit onderzoek en worden suggesties gegeven voor toekomstig onderzoek binnen het onderwerp van het gebruik van SW technologieën voor QC in de constructie-industrie.
ABSTRACT

Due to the multidisciplinary nature of the architecture, engineering, and construction (AEC) sector, each phase of a project requires the involvement of different specialists. Inherently, an intensely well-managed, and flawless stream of information between all stakeholders during the entire project is needed to achieve a successful project. While progressing through the different stages, a project must continuously comply with several sorts of requirements. Often, the involvement of the cognitive skills of people is still essential when it comes to validating these requirements, which is error-prone and contributes to the downfalls in the information stream of a project. It is evident that the implementation of Building Information Modeling (BIM) and the Industry Foundation Classes (IFC) standard already drastically increased the efficiency of data exchange in the industry. However, the descriptive underlying language of IFC, EXPRESS, has a limited expression range. An alternative should be able to provide a flexible manner to describe concepts from several domains using a logical basis that enables users to work with this information in a modular way. Possibilities lie in the technology stack offered by the Semantic Web (SW). Therefore, this thesis aims to demonstrate the added value in using SW technologies for quality control in the AEC sector. A rule checking system prototype is developed which validates technical requirements that are imposed on an IFC-based BIM model using the Shapes Constraint Language (SHACL). The data used to build a small test case are four IFC-STEP files which compose a BIM model of a transformer in a transformer space. This BIM model is validated against a selection of requirements gathered from two human-written documents. The prototype is designed by going through four necessary steps that are needed to develop a functioning rule checking system, (1) rule interpretation, (2) model preparation, (3) rule execution and (4) rule reporting. Before starting these steps, a thorough rule analysis is performed as well as the development of an ontology which are both prerequisites needed for the rule interpretation and model preparation steps. Following this, three cases have been set up, which all cycle through the system architecture, but focus on different challenges that lie within the process. The first case performs a semantic check and concentrates on enriching the BIM model with properties that map targeted objects to the vocabulary as specified in the created ontology. In the second case, a technical requirement is validated that uses implicit geometric information from the BIM model. Lastly, a set of rules is composed that encompasses all types as defined in the rule analysis. These are interpreted, the BIM model is prepared correctly for the check, the rule check is executed and a human readable report is generated that communicates eventual violations of the rule set. This completes the third case of this research. In addition to the rule checking prototype, this thesis results in a workflow schematic of BIM model requirement validation using IFC data, linked building data, Python scripting and SHACL. Finally, this thesis ends with a final review containing a conclusion that answers the main research question and its corresponding sub research questions. This is followed by a discussion regarding this conclusion, the limitations involved in performing the research and recommendations that suggest further research on the topic of using SW technologies for QC in the building industry.
**OVERVIEW - ABBREVIATIONS**

This list presents the abbreviations that were used in this thesis report and their meanings.

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIM</td>
<td>Building Information Modelling</td>
</tr>
<tr>
<td>IFC</td>
<td>Industry Foundation Classes</td>
</tr>
<tr>
<td>STEP</td>
<td>Standard for the Exchange of Product data</td>
</tr>
<tr>
<td>IFC-SPF</td>
<td>IFC-Step Physical File</td>
</tr>
<tr>
<td>RDFS</td>
<td>Resource Description Framework Schema</td>
</tr>
<tr>
<td>RDF</td>
<td>Resource Description Framework</td>
</tr>
<tr>
<td>Turtle</td>
<td>Terse RDF Triple Language</td>
</tr>
<tr>
<td>OWL</td>
<td>Web Ontology Language</td>
</tr>
<tr>
<td>QC</td>
<td>Quality Control</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible Markup Language</td>
</tr>
<tr>
<td>XSD</td>
<td>XML Schema Definition Language</td>
</tr>
<tr>
<td>SHACL</td>
<td>Shapes Constraint Language</td>
</tr>
<tr>
<td>SPARQL</td>
<td>SPARQL Protocol and RDF Query Language</td>
</tr>
<tr>
<td>LD</td>
<td>Linked Data</td>
</tr>
<tr>
<td>LBD</td>
<td>Linked Building Data</td>
</tr>
<tr>
<td>OTL</td>
<td>Object Type Library</td>
</tr>
<tr>
<td>ILS</td>
<td>Informatie Leverings Specificatie</td>
</tr>
<tr>
<td>WWW</td>
<td>World Wide Web</td>
</tr>
<tr>
<td>W3C</td>
<td>World Wide Web Consortium</td>
</tr>
<tr>
<td>SW</td>
<td>Semantic Web</td>
</tr>
<tr>
<td>OWA</td>
<td>Open World Assumption</td>
</tr>
<tr>
<td>CWA</td>
<td>Closed World Assumption</td>
</tr>
<tr>
<td>LOD</td>
<td>Level of Development</td>
</tr>
<tr>
<td>CDE</td>
<td>Common Data Environment</td>
</tr>
</tbody>
</table>
OVERVIEW - FIGURES

This list presents an overview of the figures that were used in this thesis report.

Figure 1: Schematic representation of the research design
Figure 2: Schematic depiction of BIM use cases and their dependencies during the life cycle of an AEC project (Bormann et al., 2018)
Figure 3: Graphical depiction of the comparison in information loss during handover moments in an AEC project between BIM-based and traditional workflows
Figure 4: Little BIM, big BIM & closed BIM, open BIM diagram of Jernigan (2008)
Figure 5: BIM maturity level diagram developed by Bew & Richards (2008)
Figure 6: Schematic showing a part of the IFC data model structure with several important entities
Figure 7: Schematic showing the general principle of creating an objectified relationship between two IFC entities (IfcWall and IfcWindow)
Figure 8: 5 stars model of achieving linked
Figure 9: The semantic web stack visualized
Figure 10: Structure of an RDF triple consisting of a subject, predicate and object
Figure 11: RDF graph example showing the possibility of adding metadata to resources in the form of different literals
Figure 12: Visual schematic showing the use of the rdf:type and rdfs:SubClassOf properties
Figure 13: Schematic of an RDF graph showing the principle of using OWL statements in the Tbox and its effect on the instances in the Abox
Figure 14: Visualization of the BOT ontology including the implementation of the bot:Interface class
Figure 15: Visualization of the BOT ontology including the implementation of the bot:Zone class
Figure 16: Schematic overview of the four sub-steps which are necessary for a functioning rule-based checking system. Taken from Eastman et al. (2009)
Figure 17: Schematic representation of a semantic rule checking system. taken from Pauwels et al (2017b)
Figure 18: The IfcOpenHouse model, used for an exercise to create new geometry using IfcOpenShell and pythonOCC.
Figure 19: The new geometry created for the representation of the interior volume of the IfcOpenHouse model
Figure 20: Methodology according to the engineering cycle
Figure 21: Visual representation of structural part of the IFC model
Figure 22: Visual representation of the building related installation of the IFC model
Figure 23: Visual representation of the cable infrastructure of the IFC model
Figure 24: Visual representation of the transformer part of the IFC model
Figure 25: Resulting model of combining the four separate IFC files as described in section 4.2.
Figure 26: Addition of preliminary steps to the system architecture
Figure 27: Four main components of the system architecture
Figure 28: Case 1 of the prototype development
Figure 29: Case 2 of the prototype development
Figure 30: Case 3, the prototype development going through the complete system architecture
Figure 31: Graphical representation of the system architecture for developing a rule checking prototype
Figure 32: Example graph rule type 1
Figure 33: Example graph rule type 2
Figure 34: Example graph rule type 3
Figure 35: Example graphs rule type 4
Figure 36: Example graph rule type 5
Figure 37: Protégé interface showing the SimpelGGI ontology
Figure 38: List of the modelled classes in the ModularTransformerDemo ontology
Figure 39: Object properties created in Protégé
Figure 40: Data properties created in Protégé
Figure 41: Datatypes listed in Protégé
Figure 42: The building related installation of the transformer vault
Figure 43: GGI BIM model opened in DDS-CAD Viewer
Figure 44: 109 instances of IfcFlowSegment inherited from IfcRoot
Figure 45: DDS schema navigator revealing the IFC schema behind the model
Figure 46: IfcFlowsegment with the name 'Conduit with Fittings...'
Figure 47: The 5 'Klemmendozen' in the BIM model, of which one is highlighted in blue and zoomed in onto
Figure 48: SPARQL query in GraphDB selecting the five klemmendozen
Figure 49: VSC environment showing the rule checking system
Figure 50: The three placeholders in the TRAFO150_20kV_PRIMAIR.ifc model that are required to have a center-to-center distance of 2400 mm
Figure 51: Part of the properties of the placeholder objects as shown in the DDS-CAD viewer, where -x, -y, and -z coordinates are listed as properties relative to the PEIL-point of the floor.
Figure 52: Placeholders are of the type IfcBuildingElementProxy
Figure 53: Graphical representation of the system architecture for developing a rule checking prototype
Figure 54: Visual of the position of the local coordinate system of an IfcBuildingElementProxy
Figure 55: Capture of the terminal output of executing the script as shown in Listing 22
Figure 56: Schematic of the RDF structure that will be used to transform the Python output values into the RDF format
Figure 57: 3D view of the BIM model in Solibri showing the properties of a Lightning striker object.
Figure 58: Schematic showing the steps to automate the prototype
OVERVIEW - TABLES

This list presents an overview on the tables that were used for this thesis report.

Table 1: Requirement types and their definition, taken from Kovacs & Micsik (2021)
Table 2: Classified rule types based on the PCE EM 58 document. Taken from Lee et al. (2016)
Table 3: Comparative analysis on available quality checking solutions by category. Taken from Kovacs & Micsik (2021)
Table 4: Demonstration of implicitly hidden rules in the written requirements from document S5084
Table 5: Demonstration of implicitly hidden rules in the written requirements from document ID437
Table 6: Indication of names given to objects according to IFC, its RDF conversion and the Simple GGI ontology
OVERVIEW - LISTINGS

This list presents an overview on the listings that were used for this thesis report.

Listing 1: Example of turtle serialization of the RDF graph in Figure 11

Listing 2:

Listing 3: SPARQL query example

Listing 4: SPARQL example result of the query from Listing 3 on the data of Listing 2

Listing 5: SPARQL data example

Listing 6: Example set of RDF data which can be validated by the SHACL shapes in Listings 7 and 8

Listing 7: Example of a SHACL node shape

Listing 8: Example SHACL node shape with two property shapes

Listing 9: Example of a validation report showing a violation in the data graph. Taken from the W3C SHACL Recommendation

Listing 10: Shape for rule type 1 validation

Listing 11: Shape for rule type 2 validation

Listing 12: Shape for rule type 3 validation

Listing 13: Shape for rule type 4 validation

Listing 14: Shape for rule type 5 validation

Listing 15: Part of the result of the IFC to LBD convertor, showing an instance of a mep:FlowSegment

Listing 16: SPARQL query inserting the GGl:Kabelbaan property into all triples where the object is of type mep:FlowSegment

Listing 17: Turtle snippet describing one of the surge protectors in the converted RDF graph from the original IFC file

Listing 18: SHACL shape that can be used to verify the requirement ‘The transformer space must include a building-related installation consisting of (emergency)lighting, cable carriers and service power outlets’

Listing 19: Snippet of the IFC step file describing surge protector OSA 150kV fasen:691755 and its IfcLocalPlacement

Listing 20: Snippet of the IFC step file describing surge protector OSA 150kV fasen:691756 and its IfcLocalPlacement

Listing 21: Snippet of the IFC step file describing surge protector OSA 150kV fasen:691757 and its IfcLocalPlacement

Listing 22: Python script that calculates the center-to-center distance of the bounding boxes of the three surge protector objects in the BIM model

Listing 23: Python script using the RDFLib library to transform values and variables into triples for in an RDF graph

Listing 24: Resulting Turtle file showing the created RDF graph following the structure from Figure 56
CHAPTER 1 - INTRODUCTION

This first Chapter provides the background information for the research problem that is assessed in this study. From this context description, a problem statement follows in which a research gap is present. Considering the latter, a main research question and accompanying sub-research questions are formulated that form the basis of this thesis. Lastly, to give the reader an overview of the structure and purpose of the complete report, a research design is drafted accompanied by an outline.

1.1. Background context

Projects executed in the architecture, engineering, and construction (AEC) industry require a high level of collaboration between all involved parties. A typical project will go through multiple phases, starting with the design development phase, then the actual construction of the design and lastly the operation and maintenance phases. Due to the multidisciplinary nature of the sector, each phase requires the involvement of different specialists and experts on certain topics to ensure an overall level of quality regarding the end-product. Inherently, a continuous, intensely well-managed and flawless stream of information and data between all those stakeholders during the complete lifecycle of the product is needed to achieve an as successful project as possible in the end (Borrmann et al., 2018).

A Building Information Model, or BIM model, is envisioned to represent the physical and functional characteristics of a facility. It should function as a shared knowledge resource in which information can be inserted, extracted, or modified by different stakeholders during the complete lifecycle of that facility (NIBS, 2012). Borrmann et al., (2018) state that the BIM-way of working aims to minimize manual re-entering of information and tries to enable the re-use of digital information, thereby avoiding laborious and thus error-prone work. This drastically improves the collaboration between stakeholders in a construction project, because it allows for better coordination and a reduction of data loss during information exchange moments between parties (Pauwels & Petrova, 2018). Next to the fact that a properly modelled BIM model correctly features the 3D geometry of a project; it also considers its semantics. An instance of a certain object can have its own specific properties assigned to it and can have relationships to other elements.

Non-profit organization buildingSMART developed a vendor-neutral data format that enables the exchange of digital building models and the information they contain. This data model is object-oriented and describes building data in a neutral format. Information can be shared by saving the model as a STEP Physical File (SPF) and can be opened again in any software application that supports this extension. In 2013, the Industry Foundation Classes (IFC) data model was accepted as an ISO standard and became the de facto manner of data exchange.

Even when all requirements for a project have been defined in detail, ranging from the desired datatype of an object’s property in the BIM model to more technical requirements such as a material’s minimal value for structural strength, it is necessary that these requirements are checked at certain points in the process. These checks assure the quality of the data during the handover to another party and are essential in preventing errors during construction which cause a loss in time and money. The BIM-based workflow that is currently standard practice in AECO projects results in the creation of large, comprehensive BIM models that contain detailed information of the building. Managing all this available data manually and making sure a sufficient level of quality is achieved, takes a considerable amount of effort. It would therefore be beneficial to use the information incorporated in a BIM model for automated checks that ensure the quality of the project instead of doing error-prone manual checks (Kovacs & Micsik, 2021; Borrmann et al., 2018).
1.2. Problem statement

For most projects, the efficient exchange of data without information losses during the complete lifecycle of the building is still a utopian goal to strive for. Although the introduction of BIM and IFC drastically improved the quality of collaboration between stakeholders of the project, it is still relevant and necessary for the industry to lift this to a higher level.

The Industry Foundation Classes (IFC) standard was developed as a response to the interoperability issues that occurred due to the lack of a universal standard for describing building information. IFC matches the BIM way of working and the multidisciplinary nature of the AEC industry due to its ability to facilitate information sharing ranging from topics such as the actors involved in a certain phase of the process to the geometry of structural elements and their properties. There are however several issues recognized in recent literature regarding the IFC data model emphasizing on several aspects impacting its usability in practice. These issues include but are not limited to: (1) the lack of modularity of the schema and its file-based exchange nature making it impossible to link small data portions of a BIM model to another dataset, (2) the fact that EXPRESS is the modelling language used in the IFC standard which is not logic-based, and (3) the complexity and massiveness of the schema causing problems with querying specifically needed information from a BIM model as well as problems with software implementations.

It is for that reason that the focus for the future of data exchange in the building industry is put on semantic web technologies. The semantic web is the vision of Berners Lee et al. (2001) who have introduced this term as a next step regarding the development of the World Wide Web (WWW), aiming to transform the web of documents that it currently is into a meaningful web of data. The international World Wide Web Consortium (W3C) contributes to this vision by developing several standards and best practices that describe technologies that are part of the so-called semantic web stack that enable people to create linked data and eventually a complete web of linked data (W3C, 2019). The Resource Description Framework (RDF) is among those technologies and provides a framework for expressing information about resources, which can be people, objects, relationships or really anything else. By describing information about the world according to RDF, a labelled graph is created, which is readable by computers as well as humans (W3C, 2014). To enhance this graph with semantic structure, Resource Description Framework Schema (RDFS) can be used to specify classes, subclasses, and datatypes further (W3C, 2014a). OWL is the next instrument in the semantic toolkit when even more expressive elements are needed to describe ontologies for RDF (W3C, 2012).

By using RDF, RDFS and OWL to describe building data instead of just the IFC schema, progress can be made in the fields of interoperability, linking across domains and logical inference and proofs (Pauwels et al., 2017). In theory, any type of data can be described in RDF and linked to each other, which would be ideal for the AEC industry with all their different domains and disciplines. Adding to that, the semantic web offers a modular solution for all the data that is involved in construction projects, which is contradicting the gigantic IFC schema that lacks modularity and extensibility. Lastly, the semantic web technologies have a formal logical basis, which EXPRESS does not have. This can aid in an improved rule checking process for the industry.

1.2.1. Research gap

Within the semantic web stack, theShapes Constraint Language (SHACL) was introduced by the W3C in 2017 as a standard for validating RDF data. Several research initiatives recommends using SHACL for data validation in the construction industry because it has advantages regarding data interoperability (Stolk & McGlinn, 2020; Werbrouck et al., 2019; Kovacs & Micsik, 2021).

In practice, this can already be used to validate data regarding its quality on data-structure level. However, a research gap still exists in the field of checking the compliance of the data with certain codes and standards that are often used in the design and construction phases of a project. This means that SHACL...
is not used to check the quality of the data itself but will be used to check the contents of the data. Some vendor-specific applications, for example Solibri, already exist to check whether a BIM model complies with certain rules from for example the Dutch Building Decree. The Model View Definition (MVD) and BIM Collaboration Format (BCF) standards add to this rule checking process by providing a specified workflow. However, this is based on an IFC framework and suffers from the issues that this schema brings along. Hence, no vendor-neutral, semantic web-based approach exists that enables a project manager or information manager to quickly generate a report that states whether the design matches all the necessary requirements which are set for the project.

1.2.2. Research objective

The research will focus on the following main research question, which is supported by four sub-questions.

Main research question:

‘What value can be created for the rule-compliance checking process during the design phase/handover phase of a construction project by using a workflow that involves semantic web technologies?’

Sub-questions:

1. What is the semantic web and why is it promising for compliance checking in the AEC industry?
2. What is the current process of rule checking and data validation in a construction project?
3. What challenges exist in implementing semantic web technologies for rule checking?
4. What is a suitable workflow for using semantic web technologies for rule checking?

In addition to finding the answer to these main- and sub-research questions, the second research objective is to develop a rule checking system that is based on semantic web technologies with its purpose being to aid AEC projects’ stakeholders who are involved in the quality control process.

1.3. Research design and outline

In conclusion, the research documented in this thesis aims to contribute to the ongoing improvement that is necessary in the AEC industry regarding information management in projects. This is done by developing a prototype which demonstrates the potential of using semantic web technologies in this industry. The prototype will have a narrowed scope that focusses on quality control of BIM models regarding technical requirements that are imposed on said model. This rule checking system will use semantic web technologies and shows the possibilities for quality control using linked data.

The successive chapters that follow this introduction are structured in such manner so that gradually steps are made towards previously stated goals. Figure 1 presents this structure and the end products of each phase graphically. Chapters 2 and 3 first create a better understanding on the known research and theories that exist regarding information exchange and the quality control of said information. Through literature research, these two topics are studied and by documenting this, a conclusion is formed which answers sub-research questions 1 and 2. Additionally, this conclusion forms the basis for the system architecture that is the resulting product of Chapter 4. This chapter also elaborates on the data that is used to build three case studies that are used to develop the rule checking system prototype. Chapters 5 and 6 then execute the system architecture step by step. The first use case runs through the system architecture while focusing on enriching the BIM model, translating it to RDF and aiming for a semantic check using SHACL. The second use case aims to produce a recommended method for validating a technical requirement that uses geometrical information. Lastly, the third use case combines the knowledge that is gained from the first two cases and uses this to define the final prototype for a workflow that uses Semantic Web
technologies for the automatic validation of technical requirements for a BIM model. This concludes the results-section of this project. Chapter 7 then completes this research with a discussion on the results and the process, a conclusion that answers the main- and sub research questions and lastly, recommendations for further research on this topic.

**SECTION 1: INTRODUCTION**

Problem statement → Research gap → Research objectives

**SECTION 2: LITERATURE RESEARCH**

Article selection → Literature review

1. Information modelling and exchange in the AECO industry
2. The BIM based automated quality checking process → Conclusion

**SECTION 3: METHODOLOGY**

Case study → Methods description → Process description → System architecture

**SECTION 4: PROTOTYPE DEVELOPMENT**

First iteration development cycle → Expanding prototype → Functioning prototype

**SECTION 5: REVIEW**

Discussion → Conclusion

*Figure 1: Schematic representation of the research design*

CHAPTER 2 - LITERATURE RESEARCH
Information modeling and exchange in the AECO industry

The literature research starts with this Chapter which aims to construct a knowledge framework regarding the topics of Building Information Modeling as well as the Semantic Web and Linked Data. These state-of-the-art technologies both aid the AEC industry in structuring, documenting, validating, and exchanging the data that they work with daily. BIM is the current de facto procedure for that, and research as well as the industry point out that the Semantic Web Technology Stack offers multiple additional advantages to the current way of working, which will be highlighted in this chapter. Eventually, the information gathered through this literature review will both help in answering sub research question 1 as well as with the design of the methodology for the development part of this project.

2.1. Building Information Modeling

Building Information Modeling (BIM) is one of the most noteworthy recent initiatives that has improved information management in the AEC industry and has resulted in the development and wide usage of BIM authoring and application tools (Pauwels et al. 2010; Pauwels et al. 2017; Werbrouck et al. 2019). Research in BIM and its application in practice are relevant for all disciplines involved throughout the building life cycle, as becomes clear from the research of Yalcinkaya & Singh (2015). They state that topics which vary from facility management to the architectural design process and from safety management to design codes and code compliance can benefit from working according to the vision of BIM.

According to the definition of the United States National Building Information Modeling Standard (NIBS, 2012), a BIM model is defined as being a ‘digital representation of physical and functional characteristics of a facility’. The actions that are being undertaken to achieve this fall under the process of Building Information Modeling. A BIM model should facilitate a shared knowledge resource about a facility from the moment of its initiation up until its demolition and depending on their role in the project, stakeholders can utilize this BIM model in different manners to optimize collaboration among each other. Due to the AEC industry’s nature, projects go through multiple phases and many experts with different domain-specific knowledge are involved. Stakeholders and involved companies typically cooperate only for the duration of a single project and move over to new collaborations afterwards, forming new contracts with new agreements about data modelling and data exchange causing fragmentation throughout the industry (Werbrouck et al. 2019). Figure 2 from Bormann et al (2018) shows the dependencies present in the timespan going from start to end of an average AEC project. Each arrow in the schema of Figure 2 indicates a handover moment within a certain phase of the project between two or more stakeholders. During such a handover, information whether it is in the form of a drawing, calculation, analysis, report, or visualization, is passed from point A to point B. Imaginable, the amount of data and information that is generated during the course of a project is enormous and it is desirable that information created in the beginning of the life cycle is still available at the end of the trajectory when the maintenance phase of the building starts. Figure 3, taken from Eastman et al. (2008) shows the difference between a digital information flow and a conventional information flow where the latter demonstrates significant data drops each time a new phase of the project begins. By applying BIM to an AEC project, the consequent re-use of digital information provides improvement in design activity coordination, calculation and simulation integration, construction management and in the handover to the building owner (Bormann et al., 2018).

In their book, Jernigan (2008) introduce a differentiation in levels of BIM integration that companies can achieve in their organization. A distinction is made between ‘open’ and ‘closed’ BIM and ‘little’ and ‘big’
Figure 2: Schematic depiction of BIM use cases and their dependencies during the life cycle of an AEC project (Bormann et al., 2018)
BIM, as is visible in Figure 4. Ideally, the AEC industry works towards achieving big, open BIM, which is in line with Bew & Richards’ (2008) wedge of BIM maturity levels. In this ramp, four levels of BIM maturity are defined where the last level indicates a complete integration of BIM using open ISO standards, cloud-based project data storage and fully enriched digital BIM models that are re-usable throughout the complete life cycle. The BIM maturity model is visible in Figure 5. Reaching maturity level 3 as well as a big, open BIM workflow requires the use of open standards as well as a shift from file-based collaboration to object-based collaboration. This is the point of focus of BuildingSMART, a nonprofit organization aiming to solve the AEC’s interoperability challenges by initiating, developing, creating, and adopting open digital standards for BIM processes (BuildingSMART International, 2022).
2.2. Industry Foundation Classes

The Industry Foundation Classes (IFC), are the current standard for data exchange in the building industry. It is a data schema, defined and standardized in ISO 16739-1:2018 that can be used to describe construction-related information using an object-based hierarchy that encompasses a large number of entities. The IFC standard is largely based on the ISO 10303 standard that is the 'Standard for the Exchange of Product data, or STEP. In the STEP standard, EXPRESS is introduced as general modelling language for the description and exchange of product data. BuildingSMART used the EXPRESS language to form the first version of the IFC schema. By following the IFC data structure, its entities can be used to describe tangible objects such as a ‘wall’ and ‘door’, but it is also capable of describing the more abstract relationship between these objects. Figure 6 (Bormann et al., 2018) depicts a selection of the most important entities and their inheritance structure in the data model. Figure 7 (Bormann et al., 2018) shows the general principle that is used to form relations between those entities.

![Figure 6: Schematic showing a part of the IFC data model structure with several important entities](image)

![Figure 7: Schematic showing the general principle of creating an objectified relationship between two IFC entities (IfcWall and IfcWindow)](image)

Eventually, a rich BIM model is created that can be saved in a neutral file format ‘IFC-Step Physical File’ (IFC-SPF) with the extension ‘.ifc’. The entire underlying vision for this was to enable the exchange of building information between different software applications, which would up until then all save their contents in a proprietary file format that is not readable by other programs. Thus, by means of IFC it was now possible for BIM modelers to make their models accessible outside their BIM modelling environment for those software applications that have implemented the IFC schema. This results in a bi-directional,
efficient communication flow of AEC-related data from stakeholder A of Discipline B to stakeholder C of discipline D and so forth, and this aids in minimizing the loss of data during the project’s life cycle.

It seems however, that the architecture of the IFC schema itself is the limitation in achieving this vision. Starting with the EXPRESS language that is used in the IFC standard, which directly impacts on some issues (Pauwels et al., 2010). Beetz et al (2009) conclude in their paper that the specification of IFC in the EXPRESS language imposes limitations on the usability of IFC, mainly because the language is not logic-based. The reaction to that was the development of the XML Schema Definition (XSD) and Web Ontology Language (OWL) schema versions of the original EXPRESS version (Nisbet & Liebich, 2007; Pauwels & Terkaj, 2016). While the recent development of the OWL version of the IFC schema is promising, it is still based on the original structure of the EXPRESS schema. Curry et al. (2012) mention interoperability as a large issue regarding data management for finished projects that move to the maintenance phase in their life cycle. Different stakeholders from the AEC industry as well as from other domains are often interested in small portions of information captured in a BIM model. Due to the lack of modularity in the IFC data model, and the file-based nature of data exchange in the AEC industry, it is impossible to link a small portion of required building data to another, let alone if this regards heterogeneous data formats (Pauwels & Roxin, 2016a; Krijnen et al, 2020). It prevents users to access a complete digital representation of the product, whether it is still in development or finished. This limits the possibilities for data querying and analysis, since many relationships and properties within building models are difficult to retrieve (Zhang et al., 2015; Schneider et al., 2018; Zhang et al, 2018). Van Berlo et al. (2019) add that it is possible for modelers to implement certain parts of the IFC schema in different manners, imposing even more challenges when using the building data for quality checks, validations, or simulations in a computational environment. This also puts several implications on application developers, who must deal with this monolithic data model and often have no domain knowledge to understand the approximately 1200 classes that need a correct implementation in the software they are building (Bonduel et al. 2018). Lastly, although IFC is especially powerful in the representation of geometric data of BIM models, it is quite weak in representing other types of data such as temporal-, real-time- and geographic information system (GIS)-data (Pauwels et al. 2022).

2.3. Semantic web and Linked Data

It is evident that the implementation of Building Information Modelling and the Industry Foundation Classes drastically increased the efficiency of data exchange in the AECO industry. It seems however, that the descriptive language EXPRESS, which is the underlying basis of IFC, has a limited expression range. An alternative for this might be found when looking at the World Wide Web (WWW) and specifically at the Semantic Web (SW). With the Semantic Web, it is possible to describe an infinite number of concepts and relate them to each other in one big system. This linking of domains is especially interesting for the AECO industry, since this sector is characterized by the involvement of many different disciplines throughout the building life cycle. In each stage of a project, different sets of data are relevant and should be transferred as complete as possible to the next stage. And although each project differs from the previous one, some parts of information could be reusable. An alternative for EXPRESS should thus be able to provide a flexible manner to describe concepts from several domains using a logical basis that enables users to work with this information in a modular way.

The Semantic Web is an extension of the WWW. Berners Lee et al. (2001) introduced the term SW, and they envision it as a method to make the WWW ‘smart’ by enabling machines to comprehend the data that is given to them as input instead of only displaying it on websites. Eventually this should transform the web of documents that is the WWW into a meaningful web of data, forming a semantic network that is the SW. Essential for this principle is the use of linked, open data which just as BIM, has certain levels of maturity, or stars in this case (Berners-Lee, 2006). Figure 8 shows a classification of 1 to 5 stars indicating the level of linked data. Each star adds a new requirement to which the data must comply. These rules are:
1 star: data is available on the WWW independent of the format, but under an open licence
2 stars: data is structured (e.g. Excel)
3 stars: data is available in a non-proprietary and open format
4 stars: data is formed using the W3C open standards and uses URI’s to identify resources
5 stars: data is linked to other datasets

Specifically, the Semantic Web can be seen as a set of standards which describe best practices for sharing data and the semantics of that data over the web using applications. These standards and best practices, which are necessary for achieving at least 4 star linked data, are actively developed and maintained by the...
World Wide Web Consortium (W3C) and belong to the set of technologies called the Semantic Web Stack. Using the Semantic Web Stack, people can build vocabularies and write rules that enable computers to handle data in a more useful manner and to execute trusted interactions over a network of information (W3C, 2015). Figure 9 shows this complete stack of technologies and standards as well as the hierarchy that exists between them. Each layer builds further on the capabilities of the previous ones, eventually enabling the full potential of the Semantic Web. In the following sub-chapters, the most important building blocks of the Semantic Web are elaborated upon in more detail such as the Resource Description Framework (RDF), Resource Description Framework Schema (RDFS), Web Ontology Language (OWL) and Shapes Constraint Language (SHACL).

2.3.1. Resource Description Framework

The main data model for describing information on the SW is the Resource Description Framework, or RDF (W3C, 2014a). This model states a method of writing down information about resources, ultimately allowing the information modeller to establish directed and labelled graphs. An RDF resource can be anything ranging from tangible daily life objects to more abstract concepts such as relationships or mathematical objects. To compose an RDF graph, resources are formulated in the form of RDF triples, which are made of the combination of a subject, a predicate, and an object. An example of a triple in the form of a graph is given in Figure 10. The two nodes are formed by the subject- and object-entities of the triple (‘el:Ellen’ and ‘el:Houseplant_001’) and the predicate (‘pl:owns’) gives meaning to the edge of the nodes. All individual entities in an RDF graph are assigned a specific Uniform Resource Identifier, or URI. Using this basic structure, so called labelled directed graphs are created which can be understood by both humans as well as machines.

![Figure 10: Structure of an RDF triple consisting of a subject, predicate and object](image)

The URI assigned to each node and edge of a graph ensures that every described concept is unique, comparable to the function of a GUID within IFC, but then on the World Wide Web instead of in a local file. An URI almost always consists of an HTTP address, linking the resource to a findable webpage which contains the definition of said resource. The HTTP address should be made dereferenceable, meaning that one can find out its definition by using a HTTP protocol to look up the address (Pauwels et al, 2022). This mechanism is key to the vision of the semantic web of providing people worldwide access to a web of linked, openly available data. Apart from URI’s, a graph can contain literals, which are basic values that can be represented in the forms of a string, boolean, integer and many other datatypes. Literals can only appear in triples as the object, and never in the subject position. Figure 11 shows that the graph in Figure 10 is expanded by adding additional information about the instances ‘el:Ellen’ and ‘el:Houseplant_001’.
To write down RDF triples, multiple formats or serializations can be used which each have their own advantages over the other options. Out of these syntaxes, the Terse RDF Triple Language (Turtle) is chosen as the serialization that is used in the remainder of this thesis, since it is considered the most suitable regarding human readability as well as being relatively easy to work with (W3C, 2014b). To give an example on the usage of Turtle, the RDF graph in Figure 11 is written down in Turtle syntax as shown in Listing 1.

Figure 11: RDF graph example showing the possibility of adding metadata to resources in the form of different literals

Listing 1: Example of turtle serialization of the RDF graph in Figure 11
As is visible in Listing 1, Turtle serializes the triples of an RDF graph into a textual file. This example also gives a good impression of the namespaces that are used to make the text better readable for humans. By using prefixes, which can be seen in lines 1-6 of Listing 1, one can abbreviate a complete URI into the prefix + the name of the resource. For example, to abbreviate the URIs used to indicate the predicate ‘FirstName’ that connects the subject ‘Ellen’ to the literal “Ellen”, respectively the prefixes ‘foaf:’ and ‘el:’ are used.

### 2.3.2. Resource Description Framework Schema and Web Ontology Language

The FOAF vocabulary is an example of the use of an ontology. Within the Semantic Web Technology Stack, the Resource Description Framework Schema and the Web Ontology Language are technologies that can be used to build these ontologies which can be used to improve the semantics of an RDF graph (W3C, 2014c; W3C, 2012). According to Gruber (1993), the definition of the word ontology is ‘an explicit specification of a conceptualization’. Using conceptualization, one can represent a view of the world as it is in an abstract or even simplified manner for a certain purpose. One example illustrating the purpose of an ontology could be an organization that wants to improve the design process for one of their products. By developing an ontology representing their product, they can provide the designers with a data structure which describes the formal naming of all entities within the product, the relationships between those entities and their properties. In doing so, concrete agreements can be made about the representation of the product, making it easier to eventually produce its real-life version by minimizing miscommunications and misinterpretations. Since RDFS and OWL both are building blocks of the Semantic Web and use the RDF triple structure to build ontologies, they can be reused and extended by anyone who wishes to do so. Berners-Lee et al., (2006) envision ontologies to therefore play a significant role in creating mappings and interactions between data with inherently different formats. They are important for applications with the function to search or merge data and information from varying communities (W3C, 2004b).

RDFS provides the basic elements for describing an ontology. With RDFS, classes, subclasses, comments, and datatypes can be specified in RDF of which an example is given in Figure 12. Figure 12 shows that the resources el:Ellen and el:HousePlant_001 are typed as instances of the respective classes foaf:Person and pl:Strelitzia. This is done by using the property rdf:type, which is interchangeable with the property a, as predicate between the nodes of the instance and the class it belongs to. A class itself can belong to a class too via the rdfs:subClassOf property. All instances of the subclass, in the case of Figure 12 el:HousePlant_001 being an instance of pl:Strelitzia, are automatically instances of the super-classes as well. This means that the instance el:Houseplant_001 is an instance of pl:Zingiberales and pl:Plantae as well. This naturally introduces the concepts of the Terminology Box (Tbox) and Assertion Box (Abox), visualized with a dashed separation line in Figures 12 and 13. Statements made in the Tbox are associated with a domain of interest and they describe a certain truth about that domain. In the Abox, instantiations are made which comply to the statements in the ontologies from the Tbox. Another functionality of the RDFS vocabulary are the rdfs:range and rdfs:domain properties. These can be either used to denote that a resource which has a given property is an instance of one or more classes or on the other hand that the values of a given property are instances of certain classes. Figure 13 includes an example of the use of rdfs:range and rdfs:domain.

OWL further enhances these concepts by adding the possibility to restrict data constructs regarding cardinality, property characteristics, or (in)equality. Forming rich ontologies using RDFS and OWL enables the representation of semantics which are understandable and usable by computers and applications.
Because OWL adds logical expressiveness to the world of semantic modelling, it opens up the potential for computers to infer implicit data from asserted information (Beetz & Borrmann 2018). Semantic Web technologies usually work with the assumption of representing knowledge according to an Open World Assumption (OWA). This is logical, since the whole concept of the SW is to publish all information on the web, which is never finished. OWA assumes that if something is not known to be true, it is not per se false, but unknown. It might thus be either true or false in the future if more information is provided. On the contrary, the Closed World Assumption (CWA) assumes that if something is not known to be true, it is per definition false. This latter knowledge representation approach is often the one which traditional software applications, including BIM authoring tools, adopt. This implies that for a BIM model in which some objects are not present, or simply modelled incorrectly, this is interpreted by the application as truth. By adopting SW technologies in a correct manner, and thus the Open World Assumption as well, the SW can be a fruitful addition to existing (BIM, IFC and EXPRESS)-technologies in the AEC industry (Pauwels & Terkaj 2016b). Figure 13 shows an example of how extra information can be inferred by using OWL statements in the datamodel as part of the RDF graph. `owl:inverseOf` is used to indicate that the `pl:hasChild` and `pl:hasParent` properties have an inverse relationship with one another. This means that if in the Abox of the RDF graph the triple statement `el:HousePlant_001 → pl:hasChild → el:HousePlant_001Cutting_001` is made, it is possible to infer that `el:HousePlant_001Cutting_001` has a parent which is `el:Houseplant_001` due to the OWL inverse property characteristic that is modelled in the Tbox. In Figure 13, this example is visualized by the dashed line between the houseplant and houseplant cutting nodes.
Beetz et al. (2005) propose and discuss a describe a mapping structure between several EXPRESS and OWL constructs aiming to create an OWL notation of the IFC schema in addition to the already existing XML IFC schema representation. Pauwels & Terkaj (2016) used this to further work on developing an official ifcOWL ontology which aims to find the connecting point between the IFC standard and the possibilities that semantic web technologies have to offer. Eventually this ifcOWL ontology is used by Pauwels & Roxin (2016) to initiate an IFC-to-RDF converter to translate IFC4 files into RDF based files in a bidirectional manner. One of the main starting points for this research was that the ifcOWL ontology should follow the original EXPRESS schema as closely as possible. However, this leads to two main consequences according to Pauwels & Roxin (2016). Namely (1) the maintaining of EXPRESS-specific constructs lead to unintuitive OWL and RDF constructs and (2) the resulting Abox instance graphs are still as large and monolithic as when the original IFC standard is followed. In addition to this, the leading software vendors in the AECO industry do not support the ifcOWL version of the IFC schema. By cutting away geometric data and several EXPRESS related instances, Pauwels & Roxin (2016) aim to resolve these drawbacks. Rasmussen et al. (2021) still think that the efforts which are being done to nudge the AEC industry towards using linked data for information exchanges are too extensive and lack modularity and thus the possibility to link with other often relevant domains. Therefore, the first initiative on the Building Topology Ontology (BOT) was made by Rasmussen et al. (2017a; 2017b) which introduces a simple ontology describing the most basic building structure using the classes a bot:Building which can
have a bot:Storey that has a bot:Space which in its turn has a bot:Element. Figures 14 and 15 show the most recent status of the BOT ontology, adding several classes and object properties to the data model, making this a promising development in achieving BIM maturity level 3 with proper Linked Building Data (Rasmussen et al. 2021).

This BOT ontology is the core of the W3C LBD community group (2022) which was initiated to create ontologies for the built environment which conform with the following three key-aspects:

1. Simplicity
2. Extensibility
3. Modularity

Using these in combination with the BOT ontology, a clear and understandable Linked Data basis exists on which new ontologies can be developed for different domains and functionalities.

**2.3.3. SPARQL query language**

RDF, RDFS and OWL are present among the building blocks to create linked data. However, this is without use if this data cannot be accessed, evaluated, or analyzed to fulfill a certain purpose. The SPARQL Protocol And RDF Query Language (SPARQL) is published and maintained by the W3C as the available standard for placing inquiries on RDF datasets (W3C, 2013). A SPARQL query makes use of the triple structure that is introduced by the RDF standard. For each element of the triple in a query (object, predicate, subject) an empty variable can be inserted, for which the matching patterns will be sought in the dataset by the SPARQL processor and thus giving the result for the query. An example of the result of a simple query on a simple set of data taken from the W3C SPARQL standard (2013) is given in Listings 3 to 5.

For the AEC industry, query languages could prove to be very useful to improve the usability of information embedded in a BIM model. Since data from various domains is often stored in these models in an IFC-STEP file format, it is difficult to retrieve specific domain related information in a modular manner without needing the data model in its entirety. It is for that reason that Zhang et al. (2018) researched the possibilities to extend the SPARQL 1.1 standard with domain specific functions in order to query BIM data. In their study, it is proven that by using SPARQL in combination with these extra developed functionalities, information from BIM models can be queried with spatial and logical reasoning, and additionally it is possible to query data from different sources, which is not doable with conventional techniques in the AEC industry.
2.3.4. Shapes Constraint Language

In addition to describing and querying information using Semantic Web technologies, it is also important to be able to validate the data describing this information. However, it is a challenge in the context of the Semantic Web to perform dedicated checks on data because its whole principle relies on the fact that an Open World Assumption is common. The RDF, RDFS and OWL technologies enable one to retrieve, add and possibly change data stored on different servers on the WWW. This poses the question on when a graph is ‘complete’ and can be checked for validation purposes or how one can ‘close the world’ to do so.

The Shapes Constraint Language (SHACL) is recommended by W3C to use as a language for validating RDF graphs (W3C, 2017). Using SHACL, a Closed World Assumption is made instead of OWA that is made when using other SW technologies (Knublauch, 2017). The language can validate a set of data written in RDF against certain constraints. This principle works by checking an RDF graph, called the data graph, against another RDF graph containing the constraints, also called the shapes graph. After the check, a validation report is generated that displays the violations of the dataset against the ruleset. Using a SHACL shape, a restriction, or set of restrictions can be placed on data elements, such as classes, properties, or specific nodes in the data graph. There are two main types of shapes, namely a node shape and a property shape. The first places a constraint on a node of a data graph where the latter puts a constraint on the values associated with a node using a so called SHACL property path. Listing 7 provides an example of a node shape that targets the objects in a triple with pl:Name as predicate. The targeted nodes are then validated on whether they are of the type sh:Literal and not for example of type sh:IRI. Listing 6 provides a small set of RDF data written in Turtle which the SHACL shape from Listing 7 can validate.

Listing 3: SPARQL data example.

<table>
<thead>
<tr>
<th>name</th>
<th>mbox</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Johnny Lee Outlaw&quot;</td>
<td><a href="mailto:jlow@example.com">mailto:jlow@example.com</a></td>
</tr>
<tr>
<td>&quot;Peter Goodguy&quot;</td>
<td><a href="mailto:peter@example.org">mailto:peter@example.org</a></td>
</tr>
</tbody>
</table>

Listing 4: SPARQL query example.

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>

SELECT ?name ?mbox
WHERE
```

Listing 5: SPARQL example result of the query from Listing 3 on the data of Listing 2.
Most shapes, however, are a combination of a node shape and a property shape. An example of this is given in Listing 8. In here, a SHACL node shape is created which includes two property shapes which can also validate the dataset from Listing 6. This shape first targets all instances of the type foaf:Person which in the case of the example is the node el:Ellen. Then two times a construct with sh:path is used to indicate which properties of instances of foaf:Person will be validated. In this case, these are the foaf:FirstName and foaf:Age properties. The check that eventually is performed, validates whether a person has exactly one first name and whether their age lies within the range of 0 and 150 years. Using SHACL, various types of constraints can be constructed to serve various purposes. An overview of the constraint types and the possibilities for data validation is made based on the W3C standard (2017) and is included in Appendix A. A more elaborated view upon the use cases of SHACL and the state of the art within scientific literature is given in the next part of this literature review which will focus on the process of BIM based automated quality checks.

2.4. Conclusion

An overview is given on BIM, IFC and SW technologies. This shows the possibilities for the AEC industry when moving towards a Linked Data based data model in comparison with the currently used IFC standard based on the EXPRESS language. It can be concluded that RDF in combination with its subsequent building blocks from the Semantic Technology Stack are a strong data modelling technique. By attaching URIs to the modelled information it becomes attainable to link data originating from different domains and
to easily reuse a part of information in another project. Also agreed upon by Pauwels et al. (2017a), the combination of declaring logic-based statements with RDF and possibility to do this while combining information from multiple application areas can play a key role in the future of information exchange in the AEC industry. Several of the current difficulties that the IFC standard imposes on Building Information Modeling could be diminished by integrating SW technologies into the process of creating BIM models. The option to validate these statements with a rule language built on the same logical principles allows to set up of a strong rule checking environment which can significantly improve the efficiency and quality of data exchange activities.
CHAPTER 3 – LITERATURE RESEARCH

The BIM based automated quality checking process

The previous Chapter explained the relevant technologies regarding the act of modeling and exchange of information in the AEC industry. While this is an important factor in the digitization process this industry is currently experiencing, it is just as important to know whether that information was modelled in a correct manner. This is where the process of Quality Control (QC) comes into play that helps ensuring the quality of the data in a BIM model and aids the industry in the prevention of mistakes made regarding incorrect or missing information at some point in the lifecycle of a project. In the literature that is studied for this part of the thesis, several terms are used for the BIM based automated quality checking process: among others BIM model validation, validation checking, model checking, quality control, rule checking, code compliance checking. This thesis uses BIM compliance checking and BIM quality checking to reference to the QC process of a BIM model. This chapter provides an overview of the literature that is written regarding this topic and aims to add to the answering of sub research questions number 2 and 3.

While progressing through the different stages of design, construction, and operation, a project in the AECO industry must continuously comply with several sorts of requirements. These can among others be project- or client specific, they can arise from building codes and standards or are set by the architect or structural engineer working on the design. The BIM model should satisfy these requirements when the project is handed over to another party or is moving forward to a next phase. Often, the involvement of the cognitive skills of people is still essential when it comes to validating a BIM model against the relevant rules. This method is prone to errors and contributes to the downfalls in the information stream of a project as depicted and explained in Figure 3 in the previous section. These information losses regarding the data in a BIM model can eventually cause unexpected costs for the project for example due to a delay in the planning, a necessity to buy extra materials or even lawsuits. It is therefore relevant to consider the state

Figure 16: Schematic overview of the four sub-steps which are necessary for a functioning rule-based checking system. Taken from Eastman et al. (2009).
of the art regarding the way that the industry deals with complying to all these requirements that are involved in projects.

In their article, Eastman et al. (2009) recognize this matter and propose a necessary structure for the implementation of a complete and functioning rule checking and reporting system. This structure was acknowledged and adopted by others as well (Pauwels et al., 2010; Sohilin & Eastman, 2015; Pauwels et al., 2017b; Kovacs & Micsik, 2021; Hagedorn & König, 2021) and forms a solid basis for development around BIM model validation and quality control. It describes four sub-processes that are required for a rule-based checking system which are (1) rule interpretation; (2) BIM model preparation; (3) rule execution and (4) reporting the results. Figure 16 gives a schematic overview of these four steps. Pauwels et al. (2017b) provide a clear schematic, as shown in Figure 17, that shows the components necessary for a semantic rule checking system.

![Figure 17: Schematic representation of a semantic rule checking system, taken from Pauwels et al. (2017b).](image)

They state that by using a language-based rule checking process, the three bottom components (ABox, TBox and RBox) in Figure 17 can all consist of the same, identical language. These three components can be created in various manners. De Meester et al. (2020) give a clear and quite chronological overview on the existing approaches specifically for the validation of RDF graphs, which are (1) hardcoded, (2) integrity constraints, (3) query-based, (4) high-level language. SHACL belongs to the last of those four categories. Pauwels et al. (2017b) conclude that by using the same language and data model for the schema, its instances, and the rules they are validated against, the following advantages can be achieved as mentioned in Eastman et al. (2009):

1. It is possible to retarget implementations to different source formats
2. The rules written are portable between different platforms and applications
3. If well written, it is possible to apply the language in all different context and domains

In the parts 3.1. till 3.4., the four steps required by a functioning rule checking system from Figure 16 are explained in more detail and lessons learned from previous studies are captured to form a basis for the framework of a rule checking system based on semantic web technologies. Additionally, an insight will be given in the current ideas and tools that exist in the field of BIM compliance checking in both the academic world as well as on the commercial market. The remainder of this chapter will summarize the presented challenges and form a conclusion by also considering the retrieved information in chapter 2.

### 3.1. Rule interpretation

Rules and requirements are constructed by people and can be captured in documents that are human-readable. When these documents are bundled in a specific code, norm or standard, they gain a legal status and are of major influence on the AECO industry that should comply with them. Since Building Information
Models are becoming increasingly more complex and detailed, the requirements they must adhere to are growing in complexity as well. It is impossible to manually verify whether the BIM model conforms to all these requirements which do not only deal with properties, but also with semantically rich information that is not explicitly present in the digital drawings (Solihin & Eastman, 2015). Still, it is at present day not yet possible for a computer to interpret these rules and perform an assessment on the data stored in a BIM model, which would require the transformation of these rules from natural language into a computer-readable format.

Solihin & Eastman (2015) report two main challenges that come into play regarding this subject. The first is to be able to implement the complexities inherent to the rules themselves and the second has to do with the wide scope of the conditions that need to be met. Because a BIM model generally goes through multiple design phases, the objects in the model may vary in level of development, or LOD, throughout time. Additionally, a rule or requirement never touches the complete range of content of the BIM model but is only relevant for a certain sub-part of the dataset. A rule checking system has to deal with these building model characteristics and should only result in feedback that is reliable for the relevant context of the model. On top of this, the interpretation of rules and requirements to make a computer understand them, often needs domain expertise. Next to syntax and grammar of the rules, the semantics of the rules play a role too, which would need a method for capturing this domain expertise into computer readable definitions. Multiple options exist in theory to perform the translation from human readable rules to computer understandable rules, such as through certain software programs, by an object-based approach, a logical approach, or an ontological approach (Kovacs & Micsik, 2021). However, the choice of the language that will be used during the rule interpretation step highly impacts the outcome of the translation, and decides the level of flexibility, modularity, and functionality of the output from this phase (Pauwels et al., 2010, Hagedorn & König, 2021). Additionally, it is important to be aware of the completeness of the set of information that is to be checked and how the rule checking system handles in that regard. Using the power of reasoning within an Open World Assumption, additional information can be generated for the existing knowledgebase, which depending on the specific use-case can be desired or not.

### 3.1.1. Requirement types and rule classification

An important first step in structuring the rule interpretation step is the systemization and classification of the infinite number of rules and requirements that can be derived from building codes which moreover will differ for each project. In their article, Kovacs & Micsik (2021) lay out an overview of the different types of requirements that can be checked in a Quality Control (QC) process for the AECO industry. Table 1 shows these seven requirement types and an elaboration on what they encompass.

From Table 1, it can be concluded that this typology of different requirements is still very broad. Within the seven categories, still rules can be distilled that fall under more specific groups for example subjective versus objective requirements, BIM model related requirements versus non-BIM model requirements, and technical requirements versus non-technical requirements. As already was mentioned, an essential step in moving towards automated rule checking is the classification of the rules that must be interpreted by computers. No official standard or other regulatory document exists that provides a uniform way to create a division for these classes. However, in literature as well as the industry, several attempts have been made to capture a system for classification. Below, an overview is given on some of the most notable ones.
<table>
<thead>
<tr>
<th>#</th>
<th>Requirement</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Requirements regarding the building or space program</td>
<td>States which spaces are required and what properties they should have. These space requirements can entail requests about dimensions, number of occupants, proximity, etc.</td>
</tr>
<tr>
<td>2</td>
<td>Functional requirements</td>
<td>A client can provide requests about the usage of certain spaces. This has to do with the occupants of the space and their desires about the functionalities of the space.</td>
</tr>
<tr>
<td>3</td>
<td>Financial requirements</td>
<td>These are the requirements that state something about the financial goals of the project and can be made regarding the overall project as well as smaller portions such as on material level or building element level.</td>
</tr>
<tr>
<td>4</td>
<td>Aesthetical requirements</td>
<td>Requirements regarding color, style, or shape for example which often cannot be interpreted in a single manner and depend heavily on the architects’ interpretation of the clients’ wishes.</td>
</tr>
<tr>
<td>5</td>
<td>Building performance requirements</td>
<td>These are a set of requirements that can originate from multiple sources such as rules of thumb, client-specific wishes, and regulation documents. Also, the area of the building and the elements that are involved in a certain requirement can vary as well as the discipline that they touch.</td>
</tr>
<tr>
<td>6</td>
<td>Building code and regulations</td>
<td>Requirements stated in a building code document are often legally binding, standardized, and relevant for most projects. They could apply to very different scales of the project, varying from urban to domain specific requirements. Some are relevant for all projects in a nation, some only for projects in a certain environment.</td>
</tr>
<tr>
<td>7</td>
<td>Complete discipline assessments</td>
<td>A complete methodology is provided by a certain body or organization to evaluate a project or part of a project on certain aspects. An example is the BREEM certification which rates a complete project against several types of requirements.</td>
</tr>
</tbody>
</table>

Solihin & Eastman (2015) developed a classification for BIM model rules according to their complexity regarding computing in combination with certain requirements that are demanded by the rule execution environment needed. The first two classes, and the two least complex ones, encompass rules that require explicitly stated entities and attributes available from the model or attribute values that can be derived in a simple manner. Example generic forms of these class 1 and class 2 rules are respectively: ‘Object x should have an y amount of objects of type z to fulfill function p’, and ‘The distance between two objects of type x should by y and they should both have attribute z or q’. The classes 3 and 4 are more complex and lack the most progress regarding automated checking according to Solihin & Eastman (2015). The tools that must handle the class 3 rules need a connection with a solid geometry modeler, often in combination with a system that can deal with spatial information of objects. Class 4 rules do not require a ‘yes/no’ answer.
but a proof of solution, which translates specific domain knowledge into design solutions for all classes 1 through 3.

Lee et al. (2016) developed a Model View Definition (MVD) validation process and used the MVD of the Precast Concrete Institute (PCI) PCI EM 58 rule sets to classify the different rule types that are relevant for this validation. Table 2 shows the result of this typology, in which a general distinction can be made between value validation and type validation. Using this classification, the authors hope to take a step towards a more modularized rule-based validation method for the validation of IFC models in regard to their MVD specification.

Table 2: Classified rule types based on the PCE EM 58 document. Taken from Lee et al. (2016)

<table>
<thead>
<tr>
<th>Rule type</th>
<th>Type</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Checking method</td>
<td>V</td>
<td>Checking a correct value</td>
</tr>
<tr>
<td>Check value</td>
<td>D</td>
<td>Checking a value of an attribute to be of a defined type</td>
</tr>
<tr>
<td>Check existence of a value</td>
<td>N</td>
<td>Checking null, existence of an instance attribute</td>
</tr>
<tr>
<td>Check uniqueness of a value</td>
<td>M</td>
<td>Evaluating a lower and upper bound: setting a limit on the number of attributes (cardinality)</td>
</tr>
<tr>
<td>Check uniqueness in aggregation</td>
<td>U</td>
<td>Checking uniqueness within a file: comparing a value within a model</td>
</tr>
<tr>
<td>Check type</td>
<td>T</td>
<td>Checking a correct type of an entity</td>
</tr>
<tr>
<td>Check referential entity</td>
<td>S</td>
<td>Checking a subtype entity</td>
</tr>
<tr>
<td>Check referential entity</td>
<td>R</td>
<td>Checking a referencing entity</td>
</tr>
<tr>
<td>Check referential entity</td>
<td>I</td>
<td>Checking an inverse relationship</td>
</tr>
</tbody>
</table>

Pauwels et al. (2017b) aim to compare three existing rule checking approaches for semantic rule checking with regard to the AEC industry. In order to do so, a rule classification was made beforehand that is used to set up a reliable test environment for the comparison experiment. This resulted in a differentiation between seven rule sets which all vary in content but have the same purpose of rewriting complex ifcOWL graphs into more simplified versions. An example is a rule that transforms a certain complex IFC property set construct into a more simple variant which is able to directly link objects with their respective property sets and individual properties instead of via a n-many relation. The seven rule sets that focus on this are all set up with this purpose and are heavily subjected to the structure of the IFC data model and the consecutive ifcOWL ontology that was developed to resemble this structure as closely as possible.

Lastly, an example of rule classification is given by making a step into the industry rather than the academic world. Already in 2008, construction company BAM made a typology for rules as well, indicating the societal relevance on this topic as well. The goal of their typology is for the designing and contracting parties to gain insight in the exact needs and requests of the client, since these are often formulated in quite an abstract manner. By defining the existing relationships between requirements, the aim is to structure them into a hierarchical rule-tree. Within their method, the BAM-group recognizes five requirement types which can have multiple levels of detail and can be allocated to one or multiple objects within the project. By creating a table which takes these three aspects into account, a clear overview can be created that aids in structuring the rules that need to be validated. An example of such a table from the BAM SE-guide is depicted in appendix B.

3.2. Building model preparation

BIM models are composed of objects, which belong to a certain type and have numerous properties. The rules and requirements that are constructed for a project are directly connected to the semantics that lie in the objects that together make up the BIM model. It is therefore necessary that the information in such a model is adequate and matches the agreed upon structures for writing down data. It could for example be the case that a check must be performed on the existence of a guardrail next to all instances of a staircase in the model. The guardrail could be modelled as a wire-shaped geometry that visually matches the characteristics of one, but if it is not classified as an instance of the type ‘guardrail’, the checking
system will not recognize to perform the check on that object. Misclassifications like these can cause severe interoperability issues between the individually modelled BIM models in a collaboration environment and become increasingly problematic in later stages of the construction- and operational phases of the process (Koo & Shin, 2018; Hesselink et al., 2021). Before performing a rule check, it is thus necessary to extract the correctly modelled information from a BIM model. Eastman et al. (2009) mention three strategies for adhering to this availability of information. First, a responsibility lies with the designer or modeler that provides the information in a model. However, some attributes of the data are too detailed and can be derived implicitly by a computer, which is the second strategy. The third option comes into play when complex simulations or computations are needed to extract the necessary values for checking. These cases require an additional application that is capable in doing so.

As was elaborated upon in Chapter 2, the de facto standard for exporting and importing BIM models among different software applications is IFC. The IFC schema was not modelled deliberately for import into rule-based QC environments, because its specification is not based on logic-theory (Pauwels et al., 2010). Next to this lack of a formal logical basis, the IFC schema was proven to be limited in its semantic expressiveness, causing (1) the inability to describe all necessary elements that are relevant for the BIM model as well as (2) difficulties in generic reasoning and querying methods on IFC based BIM models. Adding to this, the schema in its entirety is too rigid and monolithic, preventing modularity in the sense that users of certain blocks of data cannot extract this from the full IFC-file. Lastly, the IFC standard provides modelers with several options to create the same piece of information to be used in a BIM model. Software applications that are used to analyze, view, or edit the BIM model often do not have all those options implemented, which causes inconsistencies during the complete lifecycle of the project, among which is the QC process that is necessary during handover moments.

3.2.1. From IFC to Linked Building Data

To tackle these issues, it could be a solution for the industry to transition to semantic web technologies for describing building data. This domain provides a strong, logical basis with RDF, RDFS, and OWL that can be used in the modelling phase to describe information inherent to the project. As stated in Chapter 3.1., regarding the rule interpretation phase, the semantic web domain also provides options for describing rules based on the same logic-based theory. The combination of describing both the data in the Building Information Model as well as the rules needed to check the quality of the BIM model with the same logic theory is ideal for the development of a complete rule checking system that is more cohesive than current solutions. To convert BIM models from the IFC format into a Linked Building Data (LBD) format, first consensus had to be reached on the general data structure or ontology (Tbox) that would be used for this conversion (Beetz et al., 2009). This resulted in the development of the IfcOWL ontology by Pauwels & Terkaj (2016).

- **IFC to RDF converter**

  Using the IfcOWL ontology, a tool was developed by Pauwels and Terkaj (2016), that converts an IFC BIM model into an RDF graph. This enabled the possibilities to use querying with SPARQL and reasoning techniques on the building information data. Furthermore, inherent to the RDF standard, other RDF graphs could now be linked to the graph of the converted BIM model. However, one of the limitations remains, which is the size and complexity of the converted models due to the usage of the IfcOWL schema. Because it was developed to be an as close as possible duplicate of the original EXPRESS schema, the modularity issues still prevail in this IFC to RDF convertor.
- **IFC to LBD convertor**

Building on this modularity challenge, several ontologies were developed specifically to help overcome this limitation by aiming to modularize the IfcOWL schema and subsequently rethinking the necessary concepts for an AEC-specific ontology. Examples of this are the developed BOT, BEO, MEP, and BACS ontologies, as well as the development of an algorithm to automatically generate a modular version of an OWL ontology (Rasmussen et al., 2021; Pauwels, 2018; Terkaj et al., 2017; Terkaj & Pauwels, 2017). In their research, Bonduel et al. (2018) focus on converting BIM models in IFC format to modular Linked Building Data (LBD) Abox RDF graphs by making use of the BOT, PROPS and PRODUCT ontologies. This thus differs from the IFC to RDF converter which directly maps an IFC BIM model to an IfcOWL BIM model. Within this IFC to LBD converter it is possible to modify several settings such as the level of complexity in regard to the PROPS ontology. Just as was done in the simpleBIM methodology of Pauwels & Roxin (2016), the IFC to LBD convertor leaves out the geometrical- and representation parts of the BIM data, since these types of numerical information are not very suitable to use in a Linked Data and Semantic Web context. This means that that part of the BIM model information must be handled in a different manner compared to its non-geometrical counterpart.

### 3.2.2. Geometry

Geometric data can be 2-dimensional as well as 3-dimensional. Because of the abundance of geometric modelling tools and applications that are available in the AEC industry, there exists a challenge utilizing the information in this type of data due to the different representations of the data caused by the variety in modelling kernels implemented in these software tools. Four types of geometries can be distinguished, which are explained in more detail, but in an understandable manner in Pauwels et al. (2022). These four types encompass complex constructive solid geometry, a less detailed 3D geometry in the form of boundary representations (BREPs), 2D geometry consisting mainly of lines and points, and lastly point cloud models form the fourth geometry type.

As already specified extensively, the Semantic Web offers great possibilities in the field of modelling and exchanging the semantic, non-geometrical portion of BIM model data. However, geometrical data also needs to be handed over multiple times between stakeholders of a project. This data often consists of long, non-human readable lists of numbers representing the points, polylines and surfaces that together make up the geometry of a BIM model. It is not efficient to directly transform these lists into knowledge graphs in such a way as one would with semantic data. Efforts do exist in previous studies on how to tackle this problem when aiming for using Linked Data for the AEC industry. A concrete example can be seen in the process of developing the ifcOWL ontology. Within the IFC data model, the EXPRESS schema heavily relies on the LIST datatype to represent numeric values. Pauwels et al. (2015) sought a solution for the fact that this datatype is difficult to map directly to an OWL equivalent and came up with four conversion procedures to represent numerical IFC data in OWL aggregation types. The main conclusion from that research was that an inevitable trade-off exists between staying true to the EXPRESS schema and computational efficiency. Pauwels et al. (2017c) continue on this after the approval of the ifcOWL ontology as an official recommendation by further investigating the optimization of the representation of geometric data in a semantic context. The exact implications on the size of the instance model by using the four different conversion procedures are quantified, which leads to the recommendation of modularizing the ifcOWL ontology into five main parts. Using that structure would enable an end user to decide on the complexity of the geometry representation, depending on their particular use case.

This recommendation of Pauwels et al. (2017c) is in line with the development of several ontologies that specialize in capturing geometrical information in a semantic manner. Bonduel et al. (2019) describe an approach which embeds data from existing geometry formats inside RDF literals. Based on five listed requirements that followed from existing geometry-related ontologies, they conclude that RDF literals are
suitable to exchange geometry of BIM models. In their study, a new ontology (FOG) which stands for File Ontology for Geometry formats is presented that enables a modeler to embed geometry in RDF literals. The FOG ontology is an addition on the Ontology for Managing Geometry (OMG) which was published by Wagner et al. (2019). By using the datatype property `omg:hasSimpleGeometryDescription` 2D and 3D geometry description of any format can be stored in an RDF literal. The FOG ontology adds flexibility to this by including dedicated geometry ontologies such as GEOM and OntoBREP which are introduced by Perzylo et al. (2015), making the combination of the OMG and FOG ontologies usable for the exchange of a wide variety of cases that involve geometric building data.

- **IfcOpenShell**

In addition to the methods as described above, where the goal is to find a method to represent geometrical data in an RDF graph by using ontologies, there exists a second approach to work with the geometry related data of a BIM model. This is by using the open-source software library of IfcOpenShell, which aims to help users as well as software developers to work with the IFC file format (IfcOpenShell, n.d.). This toolkit includes the 3D geometry library provided by Open CASCADE Technology (n.d.) and can therefore analyze, transform, validate and generate geometry in IFC files. A Python wrapper is available for this Open CASCADE (OCC) library as well as for IfcOpenShell itself, making it possible to create a Python development environment to work with the geometry in IFC files. A few examples of the potential of the combination of IfcOpenShell with the Python wrapper of Open CASCADE (PythonOCC) are listed below.

The IfcOpenShell academy provides several examples and tutorials on how to get the most out of the toolkit. Krijnen (2015) posted a short exercise on how to enrich an existing IFC model with new geometry. Figure 18 below shows the geometric features of a house on a hill. The IFC file of this model does not define an instance of IfcSpace which describes the interior of this house bounded by the walls and roof. Therefore, a lot of information is not explicitly embedded in the IFC file and prohibits usage of this information. Using a python script that makes use of the IfcOpenShell and OCC technologies, first the relevant objects are filtered and the necessary geometry to create the eventual interior volume is extracted from those objects. Based on that, a new solid is made, as shown in Figure 19, which represents the interior volume of the original house. This enables the user to perform additional analysis on the IFC model, using this newly gained information that was hidden in the original models' geometry.

Hesselink et al. (2021) developed a platform using OpenCascade and IfcOpenShell to be able to perform an analysis regarding the placement of balcony elements within an IFC BIM model. To do so, an automatic
classification method was used that can extract attributes like coordinates and volumes from balcony elements which are not perse modelled in a similar manner. However, they still encountered situations where misclassifications and misattributions caused unreliability in this automated process. The authors therefore recommend a process with prerequisites that impose certain exchange requirements that help in the provision of the correct BIM model information.

3.2.3. BIM model integrity

As written by Eastman (2009) and earlier elaborated upon in Section 3.2., it is important that the information in a BIM Model is correct and suitable for the check that will be performed on said information. Several standards as well as less officially binding best practices exist in helping organizations to explicitly state the information and data they require from subcontractors, modelers, or any other party working with the digital representation of a certain project. Using these by implementing them in an organizations’ BIM process can positively impact the validation part of that process by standardizing a substantial amount of the data. The buildingSMART International standards are a clear example of this and provide the AEC industry with guidance on an industry specific data model schema (IFC), a method the definition and documentation of processes and data requirements (Information Delivery Manual; IDM), exchange specifications for data models (Model View Definitions; MVD), software-independent communication protocols (BIM Collaboration Format; BCF) and a library that defines standardized BIM objects (buildingSMART Data Dictionary; bSDD). buildingSMART often works closely together with the International Organization for Standardization, or ISO. However, several ISO standards exist as well as European (EN) and national (NEN) ones that benefit the AEC industry regarding data quality and data consistency.

3.3. Rule execution

The third act in the rule checking process combines the results of the previous two steps; rule interpretation and building model preparation, which are carried out in parallel. Several options exist for executing this third step. Zazuko (n.d.) built a ‘SHACL Playground’, which is a web application in which you can copy-paste your shapes graph and data graph to generate a validation report. When opening the website, an example is visible that shows that in a shapes graph an entity ‘person’ can have a minimum and maximum amount of the property ‘name’ that is equal to 1. The data graph on the right, however, has two values for the property ‘name’, which makes the result of the validation report ‘unsuccessful’. Next to this example of a SHACL-processor, it is possible to use a python- or java-implementation of SHACL (Sommer & Car, 2021; Knublauch et al., n.d.).

A challenge exists regarding false positives that are outputted by the rule-execution step. All false negatives will be checked, but false positives are not which can cause rule violations to go unnoticed. This is something that is relevant for the verification of the eventual rule checking system.

3.4. Rule reporting

Finally, the results that are the outcome of the rule execution phase must be reported back to the person performing the check. This report should give a clear overview of the rules that are applied in the check as well as the instances that were checked. Additionally, a distinction should be made between a rule that is passed, a rule that is not passed and rules that could not be checked because of a certain error or inconsistency in the data. It is thus required that the information is mapped in reverse back from a computer-readable format into a human-readable format. This helps users to understand the problems that
are present in the BIM model and provides a steppingstone towards effectively solving them. Each case in a rule checking process might require different feedback, depending on the type of checks that are executed. SHACL uses the SHACL Validation Report Vocabulary to describe the validation report that is the result of the validation process. This vocabulary defines the available RDF properties that make up this validation report. An example of this is given in Listing 9, which shows a validation report for a certain set of data that does not conform with the shapes graph against which it was checked.

```json
[  a sh:ValidationReport ;
   sh:conforms false ;
   sh:result [  
      a sh:ValidationResult ;  
      sh:resultSeverity sh:Violation ;  
      sh:focusNode ex:Bob ;  
      sh:resultPath ex:age ;  
      sh:value "twenty two" ;  
      sh:resultMessage "ex:age expects a literal of datatype xsd:integer." ;  
      sh:sourceConstraintComponent sh:DatatypeConstraintComponent ;  
      sh:sourceShape ex:PersonShape-age ;  
   ]
].
```

Listing 9: Example of a validation report showing a violation in the data graph. Taken from the W3C SHACL Recommendation

The validation report indicates a violation by stating a Boolean true or false value for the sh:conforms property, which in the case of Listing 8 is false. The property sh:ValidationResult then reports per violation which node, path, and/or value causes the violation. This is accompanied by a message stating extra textual details about the violation which are understandable by humans. The values for sh:resultMessage can be produced by adding an sh:message in the shapes graph that is used for the validation and can thus be customized per SHACL shape.

This SHACL validation report is quite understandable for humans. However, this is heavily dependent on the naming of the nodes and edges in the data graph, so this might be improved by transforming this SHACL validation report from RDF to another format, which is more user friendly.

3.5. Solutions for BIM compliance checking

Various options are available that can go through all four steps explained in sections 3.1 to 3.4 and thus facilitate BIM compliance checking in one of the many fields of BIM QC. A distinction can be made between commercial solutions and solutions that make use of open standards or are only worked out conceptually in academic papers. Kovacs and Micsik (2021) present an analysis in their paper where they compare the options as listed above by grading them against eight different principles that they think should be satisfied by a state-of-the-art BIM compliance checking system. In summary, these eight principles state that the QC process should be software independent and interoperable in the sense that it is available for all included parties and the digital tools they use. To achieve this, a LD approach is recommended to keep the QC process transparent and to minimize data translations between involved tools. The requirement
input method should be automated to prevent spending huge amounts of time on the administration of this. The BIM quality checker must be able to detect geometry clashes and discrepancies in attribute information. The results of this conflict detection should be reported back to the user through multiple visualized output manners. The checker should expand the context that the IFC data model provides, as this is limited in analyzing and querying. In addition, data loss must not occur anywhere between the steps of Eastman et al. (2009) of ‘rule interpretation’ and ‘rule reporting’. All of this should happen in an environment where stakeholders are responsible of keeping the requirements up to date and are notified of changes made by others. The results of this analysis can be seen in Table 3.

From Table 3, it can be concluded that none of the presented solutions for a BIM compliance checker system satisfy the eight principles they were tested against. Especially principles six, seven and eight seem to lack a positive marking. Those are exactly the features that can be supported using a Linked Data approach, underpinning the need to shift to the use of RDF and OWL for rule-based QC checking of BIM models.

### 3.6. Conclusion

A great urgency exists for the improvement of the QC process in the AECO industry since the number of requirements in projects is majorly exceeding the capabilities of a manual checking procedure, causing unwanted risks and errors in projects. Already in 2009, Eastman et al. (2009) conclude that proprietary platforms and applications that focus on rule checking, are limited in their functionality. No program exists that encompasses all four functionalities that should be incorporated in a rule checking system. In 2021, Kovacs and Micsik (2021) conclude again that the solutions presented in literature or those that are available on the market, often focus on one or only a few aspects of the complete QC process. Furthermore, a lack of options is recognized that allow for the programming of project specific rules, creating a dependency on the proprietary rulesets that are included in the software. A chance lies in more public, open and vendor-neutral approach, using Linked Data approaches.

It is important to consider the geometrical part of a BIM model in a different manner than that one would with the semantic part of said model. Geometry can be represented in several ways due varying implementations of 3D modeling software applications. When considering using geometric data in a linked building data context, a decision should be made on the necessary complexity of that data in order to decide on the manner of transforming this into an RDF graph. Several ontologies are available to aid in this, but it is also possible to execute more complex operations using the geometric power of IFC itself in combination with IfcOpenShell to directly extract the desired values of certain BIM model element properties. By combining the strengths of the currently available Semantic Web technologies with the established IFC data model which excels in geometric data modeling, a new rule checking process could be designed that proves the advantages of both approaches.

<table>
<thead>
<tr>
<th></th>
<th>I.</th>
<th>II.</th>
<th>III.</th>
<th>IV.</th>
<th>V.</th>
<th>VI.</th>
<th>VII.</th>
<th>VIII.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Briefing software</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>CAAD and BIM software solutions</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Partly</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Model Checkers</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Partly</td>
<td>No</td>
</tr>
<tr>
<td>IFC doctypeChecker Workflow</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Experimental research projects</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Building SMART file format</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>
Furthermore, it is important to focus on several other challenges that exist in the field of rule checking. Rules are often documented only in human readable, textual documents for which an interpretation step is necessary in order for a computer to work with them. No general and standardized classification exists regarding the typology of all these requirements, which would be useful for automating this rule checking process in the future.
CHAPTER 4 - METHODOLOGY

The fourth chapter describes how this study was conducted, and which methods were used to answer the sub- and main research questions as well as to achieve the final goal of this research: developing a functioning prototype of a rule checking system using semantic web technologies.

A graphical visualization of the methods used for the development part of this study is presented. This is an extension of the earlier presented research design in the introductory section. It shows the different steps that are necessary to successfully conclude this project. Based on the conclusions of the literature research, a system architecture is proposed, and the methods used for each phase of the development part are elaborated upon. By executing these consecutive phases step by step, a functioning prototype can be made that aids in automated quality control of projects in the AECO industry.

Subsequently, the case study and its complementary data will be described in this section and additionally a consideration is made of the potential challenges and risks that this methodology imposes on the total research.

4.1. Methodological justification

The literature research that is documented in Sections 2 and 3 anticipated on laying a knowledge basis that proved to be helpful in answering sub research questions 1, 2 and 3 of this thesis. Additionally, the conclusions from these preceding sections aid in the design of the system architecture that is necessary for the prototype development to achieve the desired result. This methodology that seamlessly fits this prototype development process is a classical engineering cycle. This means that it complies with the following four steps: (1) data collection, (2) system architecture design, (3) prototype development and (4) prototype validation, also visible in Figure 20.

As can be seen in Figure 20, the cycle of system architecture design followed by prototype design and prototype validation can be iterated through until the desired result is achieved in the validation step and the prototype is functioning properly. The methodology for this research describes three use cases which will be used to iterate through the system architecture. Each use case and thus every iteration will focus on a different challenging aspect of the rule checking system. The schematic overview of the system architecture is shown in Section 4.3. and its functionalities are described in Section 4.4. Section 4.5. elaborates on the exact details of the three use cases. Before that, the next Section of this Chapter elaborates on the data that is used for the cases.
4.2. Data Collection

To be able to develop the prototype rule checking system that is to be the resulting product of this thesis, data is needed to build a clear and workable case to prove the concept. The envisioned system requires two main types of input data which are (1) a BIM model and (2) rules and requirements for that particular BIM model.

The data is provided by Qirion, a company that is involved in designing, building and operating energy grids. They have provided the data files containing the design of a transformer vault including the technical design of the transformer itself. Complementary to this BIM model are two text files in .pdf format, which describe several requirements that the design must comply with. These requirements are set up by TenneT, which is the eventual asset manager of the constructed project. The transformer vault BIM model is delivered in the form of four separate IFC STEP files which when opened similarly merge into the entire transformer design. These files are:

1. TRAFO150_20kV_BOUWKUNDE.ifc
2. TRAFO150_20kV_GGI.ifc
3. TRAFO150_20kV_KABEL.ifc
4. TRAFO150_20kV_PRIMAIR.ifc

The first file encompasses the structural outer shell that protects the more vulnerable technical parts of the transformer. The ‘_GGI’-file contains the design for the so called ‘gebouw gebonden installatie’ (GGI) which translates to the installations that are relevant for the usability of the structural part of the design from the ‘_BOUWKUNDE’-file. Installations that are part of the GGI can for example be the lighting, electrical components such as power outlets, ventilation, or air-conditioning, depending on the requirements for the building. The third file of which the name ends on ‘_KABEL’ includes the cables and their necessary infrastructure which accommodate transport of the transformed power out of the transformer towards its next destination. The fourth and last IFC file makes up the (preliminary) technical design of the transformer itself. It consists of multiple placeholders indicating the dimensions of the components that should be

![Figure 21: Visual representation of structural part of the IFC model](image1)

![Figure 22: Visual representation of the building related installation of the IFC model](image2)
installed there. Additionally, multiple elements are modelled in detail. Together they make up the primary design of the transformer. Figures 21 to 24 give a visual impression of each IFC file and the model that they describe. The combined result of all four files is visible in Figure 25 below.
As mentioned, the design of this BIM model should comply with several requirements that have been set up by TenneT. They are aiming for more standardization regarding the designs of the transformers and their respective vaults. The goal eventually is that the different types of transformer vaults and transformers consist of modular and standardized so called ‘building blocks’ that make the design which Qirion is responsible for, is less prone to errors. One of the documents contributing to this is the ‘S5084 Modulebeschrijving – Module 02 Transformatorruimte’, which states several requirements to which a design of a transformer vault must comply. This so-called S-document is part of a series of documents which together describe the full set of requirements to achieve the goal of modular transformer design and building. Document ‘S5084 Modulebeschrijving – Module 02 Transformatorruimte’ is added and can be seen in Appendix B. In addition, an amendment proposal called ID437 is present in which three changes to the complete set of S-documents are submitted to which the designers of the BIM model should abide. Document ID437 is added as well and can be seen in appendix C. These two documents will form the basis of describing the cases that will be used during the development phase. The cases are explained in more detail in Section 4.5.

4.3. System architecture design

After the input data (BIM model & requirements) is obtained, the architecture is made to help in developing the prototype system that aims to use semantic web technologies for BIM model quality control checking. Figure 26 shows that the system consists of four main components, indicated by the different colored blocks. Together, they conform to the framework of Eastman et al. (2009) and therefore comply with what literature considers to be the building blocks for a proper rule checking system.

Before these steps are executed, two prerequisites should be met which will contribute to a more efficient development process. These prerequisites follow from the literature research, in which is concluded that the first two steps of Eastman’s (2009) model should be aligned as much as possible. Therefore, an analysis will be performed on the requirement documents with the aim of structuring the ruleset as much as possible before translating the rules to a computer understandable language. The second prerequisite is to analyze the contents of the requirements as well as the BIM model in order to provide the BIM model (A-box) with a T-box including an R-box that forms a general basis for the rule checking system. This leads
to Figure 27, which includes these two preliminary steps that precede the main process. Then, in Figure 28 the next step is visualized, which is the first iteration through three of the four steps. The focus will lie on the model translation step, model enrichment step, the definition of computer readable constraints and the generation of a first validation report.

**Iteration 1:**

![Diagram](image1)

**Figure 28: Case 1 of the prototype development**

The second iteration adds a component that tackles the geometric part of a BIM model and cycles through the same steps as the first iteration, but focuses on validating a geometry-related requirement. This is visible in the schematic of Figure 29.

**Iteration 2:**

![Diagram](image2)

**Figure 29: Case 2 of the prototype development**
Figure 30: Case 3, the prototype development going through the complete system architecture
Finally, the third iteration follows the complete system architecture which is shown in Figure 30. The symbols used are arrows to indicate the order of activities throughout the system, light-grey colored 3D objects to indicate data containers and dark-grey colored fields to indicate processes. The different shades of yellow, orange, and red refer to the four rule checking system functionalities and stay consistent throughout the three iterations.

4.4. System architecture functionalities

Figure 31 shows the complete and general rule checking system leaving out the iterations that are purely necessary for the development part of this project. Each of the four functionalities of the system is elaborated upon below.

4.4.1. Rule interpretation

The first step, after completing the two prerequisites, which starts in parallel with the model preparation step, is focusing on one of two required types of input data; the requirements against which the BIM model needs checking. In this case, this input is two textual documents which are hand-written in a human-readable format saved in .doc and .pdf extensions. Rules and requirements regarding a design are often defined by humans and are thus not automatically interpretable by computers. Thus, in order to develop a system that checks whether the model complies with these requirements, the rules need to be translated into a format that can be interpreted by a computer.

The language that will be used for the interpretation of the requirements is SHACL. This means that the goal for this step is to begin with a portion of data as shown in the left side of the rule interpretation block in Figure 30 that are the client’s requirements and to transform it into a so-called shapes graph, the portion of data depicted on the right side of that block in Figure 30. This is done by firstly filtering the rules that are in the requirement documents, based on the rule analysis. As became clear during the literature research, there are different types of rules which need other methods of interpretation. So, during this rule type filtering step, the requirements are filtered by type and it is decided whether SHACL can be used to check them. Then, the actual translation process begins to convert the rules into an RDF compliant format using SHACL. The result is a single or multiple shape graphs, depending on the amount and types of rules that are relevant for the case, which contain the requirements to which the data graph following from the model preparation step will be checked against.

The ontology formulation step which is part of the prerequisites is relevant for the rule interpretation step as well as for the model preparation step. Since no ontology exists for the input data that is used in this thesis, one must be created separately. The function will be to provide a common data structure to which the information in the BIM model should adhere to. It is the common ground that ensures that the computer will understand what the semantics of an object in the model are, so that certain objects that are relevant for the requirements can be verified. It could for example be the case that an object that represents a high voltage electrical wire in real life is modelled as an ifcBeam in the BIM model. The ontology that is developed indicates what objects should be part of the model and specifies how they are related to each other, what data type they should be, ensuring that during the rule execution phase, the ruleset data matches with the instances data.

4.4.2. Model preparation

In parallel with the rule interpretation step, the model preparation step takes place. Since the functionality of the rule checking system is dependent on the consistency of the data output of both the rule interpretation step as well as the model checking step, it is important that these are executed regarding
Figure 31: Graphical representation of the system architecture for developing a rule checking prototype
one another. As visible in Figure 30, this step starts with a data container which represents the original IFC BIM models that are received. These are then transformed to fit the RDF data schema with a convertor tool as discussed in the previous literature section. The output of this convertor is the BIM model in a turtle extension. Following that, two additional steps are necessary to ensure a proper model for the next phase in the process: rule execution. These data enrichment and geometry engine steps are worked out respectively in iterations 1 and 2, which are depicted in detail in Figures 28 and 29. For the first iteration, which will be elaborated in more detail in Section 4.5 of this Chapter, only the data enrichment step will be done since no checks will be performed yet which involve geometric aspects. The second iteration however, will focus on the geometry specifically.

The data enrichment step must be done because no ontology is present for the input data. If the BIM model is not modelled correctly, it could be the case that the modelled object that represents that specific conductor cannot be recognized by a computer system as the object that must be checked. Therefore, it is important to add a semantic basis that overlaps with the modelled information as well as with the computer readable ruleset. Regarding the second iteration of the development process, it will be necessary to work with the geometrical data in the BIM model. To make this set of information useful for the rule checking system, an interpretation should be done which filters out the necessary portions of data and makes them understandable for the rule checker.

When the data enrichment step and eventually the geometrical data processing step as well have been executed, the data graph conforming the original IFC BIM model is complete, and it can be used in the next phase of the process.

4.4.3. Rule execution

As soon as the two RDF graphs are constructed; the data graph from the model preparation step and the shapes graph from the rule interpretation step, the rule execution phase can begin. The data model will be validated against the rules that are captured in the shapes graph. This is done in a SHACL processor, which will ultimately produce a validation report in the form of an RDF graph as well. This validation report communicates the results of the check. In the case where none of the SHACL restrictions are violated, the report tells the user that their data conforms with the set requirements. If these requirements are violated, the validation report provides an overview of the details of the exact violation or violations. For the iterations in this development process, this is done using a Command Line Interface (CLI) in Microsoft Visual Studio Code, built on the pySHACL implementation of Sommer & Car (2021). Eventually, this should be expanded and/or implemented in a larger framework describing a rule checking process that has the appropriate user interfaces and works largely on automated procedures.

4.4.4. Reporting results

The last and fourth step of the rule checking system is reporting the results back to the user of the system. During this step, it is preferred that the results of the SHACL validation report are communicated in an understandable manner, which means that the validation report needs to be converted from its RDF format to a format that is more intuitive for the end user. The implementation options for this range from relatively simple variants such as the creation of an Excel sheet with the results to developing a more visual manner of presentation including a user interface and/or visual feedback that appears in the original BIM model.

As is visible in Figure 30, these options are both incorporated in the process, and will both ensure that an end user can use the results from the rule checker. Eventually, this last step aims to result in a quick process which helps the BIM model designer or validator to quickly see the level of quality of the model.
4.5. Case descriptions

Since the methodology plans on doing three iterations through the architecture that is designed for the development phase, also three cases are needed to provide a clear goal for the prototype. This helps in framing the process and will minimize the risk of staying in an ongoing cycle without end goal. This subsection describes these cases, of which the first two are relatively small and simple ones compared to the third, which will result in the final prototype of this research.

4.5.1. Case 1: Semantic validation

The first iteration will focus on the validation of the following requirement to which the BIM model must comply:

‘The transformer space must include a building related installation consisting of (emergency) lighting, cable carriers and service power outlets’.

Focusing on this requirement, the first three steps of the system architecture will be taken which results in the first SHACL validation report. The attention for this case lies on enriching the converted IFCtoLBD data with additional properties which ensure a correct mapping between the A-box data and the R-box data, using the T-box transformer ontology.

4.5.2. Case 2: Geometry validation

The second iteration will validate a requirement which involves the BIM models’ geometry features. This requirement is:

‘The three surge protectors and potheads that must be attached to the TenneT grounding transformer should have a center-to-center distance of 2400 mm’.

This center-to-center distance is not explicitly present as a property with a value in the data of the BIM model, so it must be extracted from the data. The latter is just one of the many examples that can be thought of which need a similar approach within the full process of quality compliance checking. The result of this second iteration, is the description of a method which is able to retrieve desired implicit geometric information, insert it as RDF data in the graph representing the data of the BIM model, and finally uses SHACL to validate this newly created graph against a shapes graph.

4.5.3. Case 3: Combined validation and reporting

The third case combines several requirement types into one check while trying to limit the amount of manual work needed to perform the validation, thus aiming to automate the process as much as possible. Additionally, in this last iteration the fourth step of the framework will be executed resulting in a report that improves the usability of the validation results for the end user, which was not done for cases 1 and 2 yet. The requirements that will be checked simultaneously for case 3 cover the complete typology of rules that is made in Section 5, demonstrating the options for BIM model validation with SW-technologies. The set of requirements for case 3 is:

- Rule type 1: The transformer space must include a building related installation consisting of (emergency) lighting, cable carriers and service power outlets
- Rule type 2: Four lighting strikers are placed on a transformer space
- Rule type 3: The value of the property diameter is of datatype float

59
• Rule type 4: The wiring harness is provided with cable coils and copper connection vanes- (or forks)
• Rule type 5: The earthing ball's diameter is 25mm
• Rule type 6: The three surge protectors and potheads that must be attached to the Tennet grounding transformer should have a center-to-center distance of 2400 mm.
CHAPTER 5 - PROTOTYPE PREREQUISITES

This Section covers the effort that is done to meet certain prerequisites that are necessary for the development of the rule checking system prototype. Since the focus lies on two documents containing certain requirements for the transformer, the first preliminary step is an analysis of these documents regarding the types of rules that are embedded in it. Different types of rules might require a different translation step or call for changes to the BIM model. Therefore, this analysis must be done before the start of the development of the actual rule checking model. The second prerequisite is the development of an ontology which serves a connecting purpose between the requirement documents on the one hand and the BIM model on the other. The results of the requirement analysis as well as the ontology that is used for the development of the rule checking system are elaborated in this Chapter and aim to bring structure in the three cases as described in Chapter 4. This will be of value for the results, discussion and conclusion which respectively are written down in Chapters 6 and 7.

4.1. Rule analysis

For the development of this prototype, two sets of requirements are used which are documented in text in a word and a pdf file. An analysis was performed on these two documents aiming to determine the types of requirements that are used and how often. The purpose of this is to aid in the rule interpretation and model preparation phases of the development methodology. Knowing the requirement types will provide information about the rule checking system and the functions it needs.

Initially, a distinction is made in the analysis between three types of requirements namely model content requirements (1), technical requirements (2), and data quality requirements (3). As can be seen in Appendix B: Requirement analysis, each category is given a specific color and is used to mark the requirements that fit into the corresponding group. The model content requirement (1) should conform the structure ‘x is connected to y’ or ‘p contains q, r, and s’. This type of requirement is used to validate the completeness of a BIM model to ensure the desired information is present. The technical requirement (2) has a structure such as ‘x should have a minimum distance to y of z’ or ‘p should be modelled according to section q of standard r’. It is focused more on the values of the data and their implications on the finalized product of the BIM model. Lastly, the data quality requirement (3) checks whether data is entered or modeled in a certain required manner. A rule could have a form such as ‘the value of x should be a string containing no more than y characters’. During this process, it appeared that for the requirements in both documents it was often the case that more distinct rules could be derived implicitly, providing the possibility for a clearer rule classification structure. Table 4 shows an example from the S5084 document, where a piece of written text the size of half an A4 paper is dissected into multiple rules. To do this, first a translation step took place from Dutch to English. Then, the sentences were written down separately and dissected into the different rules embedded in them.

Following this, a classification is made that describes the different types of rules that are present in the two requirement documents. Since SHACL is chosen as language to translate the human understandable rules into computer understandable ones, the classification system falls back on the structure of the SHACL primer as much as possible. Rule types that are impossible to capture with SHACL, and do not fall in the geometric rule category, are categorized as ‘other’ and fall outside the scope of this thesis. Appendix C: SHACL Primer rule classification summary shows a summary of the SHACL primer which depicts the different constraints that SHACL can put on RDF graphs in a compact manner. Each type of constraint is translated to a minimalistic mathematical expression aiming to make the different types more recognizable when translating human written text into SHACL shapes. Sections 5.1.1. and 5.1.2. contain the results of the analyses of respectively the S5084 and the ID437 requirement documents. The first Section uses the derived rules from Table 4 and demonstrates per rule type what the general form of the RDF graph would be, accompanied with the SHACL shape that matches this requirement.
Table 4: Demonstration of implicitly hidden rules in the written requirements from document S5084

<table>
<thead>
<tr>
<th>ID</th>
<th>Requirements from document S5084 (dark blue) &amp; derived rules</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.000</td>
<td>On all the copper connection vanes 20 kV and 10 kV as well as on the cable carriers, Ø25mm earthing balls will be attached.</td>
</tr>
<tr>
<td>1.001</td>
<td>A 20 kV copper connection vane is provided with an earthing-ball</td>
</tr>
<tr>
<td>1.002</td>
<td>A 10 kV copper connection vane is provided with an earthing-ball</td>
</tr>
<tr>
<td>1.003</td>
<td>A cable carrier is provided with an earthing-ball</td>
</tr>
<tr>
<td>1.004</td>
<td>The earthing ball’s diameter is 25 mm</td>
</tr>
<tr>
<td>1.005</td>
<td>The earthing ball’s diameter will be modelled in mm</td>
</tr>
<tr>
<td>1.006</td>
<td>The value of the property diameter is of datatype float</td>
</tr>
<tr>
<td>2.000</td>
<td>Earthing of components should be done as much as possible though earthing plates which are embedded in concrete, aiming to limit the amount of visible copper</td>
</tr>
<tr>
<td>2.001</td>
<td>There should be as little as possible visible copper</td>
</tr>
<tr>
<td>2.002</td>
<td>Earthing plates should be embedded in concrete as much as possible</td>
</tr>
<tr>
<td>3.000</td>
<td>On the corner points of the transformer spaces without a roof, lightning strikers will be placed</td>
</tr>
<tr>
<td>3.001</td>
<td>Lighting strikers are placed if a transformer space has no roof</td>
</tr>
<tr>
<td>3.002</td>
<td>Four lighting strikers are placed on a transformer space</td>
</tr>
<tr>
<td>3.003</td>
<td>One lighting striker is placed on each corner of the transformer space</td>
</tr>
<tr>
<td>4.000</td>
<td>The lighting strikers should be connected according to S2001 to an earthing rod which is placed near the lighting striker.</td>
</tr>
<tr>
<td>4.001</td>
<td>The connection between the lighting striker and the earthing rod is according to S2001</td>
</tr>
<tr>
<td>4.002</td>
<td>An earthing rod is placed near a lighting striker</td>
</tr>
<tr>
<td>4.003</td>
<td>A lighting striker is connected to an earthing rod</td>
</tr>
<tr>
<td>5.000</td>
<td>In all cases, the earthing rod is connected to the earthing raster inside and/or around the building</td>
</tr>
<tr>
<td>5.001</td>
<td>An earthing rod is connected to an earthing raster</td>
</tr>
<tr>
<td>5.002</td>
<td>This earthing raster can be inside the building, around the building or both</td>
</tr>
<tr>
<td>6.000</td>
<td>In case of a transformer space with a roof, a roof-earthing net needs to be placed according to S2001.</td>
</tr>
<tr>
<td>6.001</td>
<td>If the transformer space has a roof, a roof-earthing net should be placed</td>
</tr>
<tr>
<td>6.002</td>
<td>This earthing net is placed according to S2001</td>
</tr>
</tbody>
</table>

4.1.1. Document S5084

The first analyzed document is the S5084 document that describes what the module ‘transformer space’ should encompass. These S-documents together should form the basis for a standardized model for a transformer station. Figures 32 to 36 below show the different rule types that are present in a part of the S5084 document. For each rule type, an example is given in Listings 10 to 14 of how such a rule would typically be modelled in RDF and what the complementary SHACL shape could look like.

- Rule type 1: Object relation requirement

Document S5084 contains many variations of the following type of sentence: ‘object x has relationship y with object z’. An example from Table 4 is the rule 1.001: ‘A 20 kV copper connection vane is provided with an earthing-ball’. The relationship ‘is provided with’ can be replaced by other predicates like ‘is connected to’, ‘contains’, ‘consists of’, etc. Figure 32 shows
one of the possibilities to model this in an RDF format and Listing 10 displays the SHACL shape needed to check this type of requirement.

- **Rule type 2: Cardinality requirement**

In addition to the object relation requirement, cardinality constraints can be placed on the relationship between the objects. For example, a requirement can be that a certain object x should be connected to at least four instances of object y, but never more than eight instances. Rule 3.002 from Table 4 is an example of this. These types of requirements can be solved with the sh:minCount and sh:maxCount constraint components. Figure 33 shows one of the possibilities to model this in a linked data format and Listing 11 displays the SHACL shape needed to check this type of requirement.

![Figure 33: Example graph rule type 2](image1)

**Listing 11: Shape for rule 2 validation**

```shacl
prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
prefix sh: <http://www.w3.org/ns/shacl#> .

@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix foaf: <http://xmlns.com/foaf/spec/> .
@prefix sh: <http://www.w3.org/ns/shacl#> .

<a:CardinalityRule a sh:NodeShape ;
  sh:targetClass a:TransformWorkspace ;
  sh:property [ 
    sh:path sh:InversePath pl:PlaceOn ] ;
  sh:class a:LightningStirker ;
  sh:minCount 4 ;
  sh:maxCount 4 ;
].
```

- **Rule type 3: Data type requirement**

The third type of rule contains all types of requirements that check the datatype of a certain value in the model. This can vary from a check on the number of characters in a string, to whether the datatype of a value should be a float, integer, or Boolean. Figure 34 shows one of the possibilities to model this in a linked data format and Listing 12 displays the SHACL shape needed to check this type of requirement. Rule 1.006 from Table 4 is used as an example.

![Figure 34: Example graph rule type 3](image2)

**Listing 12: Shape for rule 3 validation**

```shacl
prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
prefix sh: <http://www.w3.org/ns/shacl#> .

@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix foaf: <http://xmlns.com/foaf/spec/> .
@prefix sh: <http://www.w3.org/ns/shacl#> .

<a:DataTypeRule a sh:NodeShape ;
  sh:targetClass a:EB ;
  sh:property [ 
    sh:path pl:Diameter ] ;
  sh:datatype xsd:float ;
].
```

- **Rule type 4: Logical requirement**

Often, a requirement is more complicated than the examples from Figures 32 to 34 and Listing 10 to 12. For those situations, SHACL provides constraint components which are comparable to the common logical operators ‘not’, ‘and’, ‘or’ and ‘exclusive or’. These can be used to construct more complex shapes that require certain exceptions from the general rule. Next to these, a qualifiedValueShape component can be used that is an equivalent of an ‘if-statement’. The example
used is the following requirement from Document S5084: ‘The wiring harness is provided with cable coils and copper connection vanes- (or forks)’. This implies that for both the modelled information in the top graph of Figure 35 as well as for the bottom one, the SHACL shape in Listing 13 should result in a positive validation, as well as in the case when the CCF and CCV instances are both used.

- **Rule type 5: Value and unit requirement**

The document also contains rules that require a certain property to have value x with unit y. Rules 1.004 and 1.005 from Table 4 combined form such a requirement. It is often the case that a value and unit are combined in technical requirement documents, which is why it is convenient to have a SHACL shape that covers both in the same check. In the example provided by Figure 36 and Listing 14 a solution for this is demonstrated which is used in the NEN2660 (2021) as well.
In this solution, a blank node is used to objectify the desired quantity, such as length. This node has in turn two properties, namely the value of the quantity and the unit of the quantity.

- **Rule type 6: Geometrical requirement**

A requirement can also refer to certain geometrical properties embedded in a BIM model. These examples give an impression of the requirements that fall under the type ‘geometrical requirement’:

- The minimum required distance between conductors and walls is 400 mm, or 750 mm center-to-center, for 20 kV.
- The height of the walls is based on the dimensions of the transformer and have a minimum height of the maximum specified height of the transformer.
- At least 850x2300 mm (width x height) free space is required around the transformer for escape routing, calculated from the side of the transformer and the opened doors of the cabinet attached to the transformer.

Depending on the way the information is modelled, geometry can be explicitly stated in the data for example in a triple in the form of ‘object X \( \rightarrow \) hasDistancefromObjectY \( \rightarrow \) 100mm’. However, it often is the case that the desired information must be derived or computed because it is hidden in an implicit manner. Additionally, the IFC data model is better in describing geometrical data than the RDF data model, which is why during the conversion process of IFC to RDF the geometry data usually is not converted and thus missing in the resulting RDF graph. It is therefore not possible to directly check geometrical requirements such as the ones put in the example above with SHACL. However, use case 2 which is documented in Section 6.2 aims to develop a structure which eventually enables to perform SHACL-checks on this kind of requirement and thereby thus extends the possibilities for SHACL.

- **Rule type 7: Referral to another document**

A second type of requirement that cannot be checked with SHACL directly is the referral to another technical document or standard. Rule 6.002 from Table 4 shows an example of this. The requirement states that something must be placed according to a certain document ‘S2001’. To be able to conform to the contents of this S2001 document, extra information is needed to decide what type of check eventually needs to be performed.

- **Rule type 8: Subjective requirement**

The last rule type that is part of this classification is a so-called subjective requirement. This is a requirement that is written down with too little detail, making it impossible for a computer to interpret it let alone to check it. An example is the requirement from Table 4 with ID 2.001 ‘There must be as little as possible visible copper’. The usage of the construction ‘as little as possible’ causes this requirement to classify as subjective, because it is unclear whether a rule checking system validates this requirement positively or negatively.

4.1.2. Document ID437

Document ID437 is the second of two documents which are used as input for creating a rule checking system for the transformer and transformer space BIM model. It is a written document that captures the minutes and agreements that are made on a few design changes that are supposed to make certain
elements more interoperable with each other during the building and maintenance phases of the projects’ life cycle. Document ID437 is visible in Appendix C and is highlighted in the same manner as document S5084 to provide a first impression on the types of requirements present in the text. This document is significantly different from the first analyzed document, because it specifies in detail what the agreements are that the attending parties talked about content wise as well as the reasoning behind the decisions. Each topic is accompanied by complementary drawings and Figures which point out the issues elaborately. Ultimately, three requirements can be distilled from ID437 which are captured in Table 5.

Table 5: Demonstration of implicitly hidden rules in the written requirements from document ID437

<table>
<thead>
<tr>
<th>ID</th>
<th>Requirements from document ID437 (dark blue) &amp; derived rules</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.000</td>
<td>The three surge protectors and potheads that must be attached to the TenneT grounding transformer should have a center-to-center distance of 2400 mm and not of 2250 mm which was the case before.</td>
</tr>
<tr>
<td>7.001</td>
<td>Three surge protectors must be present.</td>
</tr>
<tr>
<td>7.002</td>
<td>Three potheads must be present.</td>
</tr>
<tr>
<td>7.003</td>
<td>A surge protector is attached to a TenneT grounding transformer.</td>
</tr>
<tr>
<td>7.004</td>
<td>A pothead is attached to a TenneT grounding transformer.</td>
</tr>
<tr>
<td>7.005</td>
<td>A surge protector should have a center-to-center distance of 2400 mm to a neighboring surge protector.</td>
</tr>
<tr>
<td>8.000</td>
<td>The minimum distance between phase- and neutral points of the transformer connection cable and the transformer conserver should be 1500 mm.</td>
</tr>
<tr>
<td>8.001</td>
<td>A radius of 1500 mm around the corner point of the transformer conserver should be free of transformer connection cables.</td>
</tr>
<tr>
<td>8.002</td>
<td>A transformer conserver should be present.</td>
</tr>
<tr>
<td>8.003</td>
<td>A transformer connection cable should be present.</td>
</tr>
<tr>
<td>9.000</td>
<td>A 150(110)kV neutral point grounding placeholder should be present in the model in accordance with relevant existing TenneT standards.</td>
</tr>
<tr>
<td>9.001</td>
<td>A 150(110)kV neutral point grounding placeholder should be present.</td>
</tr>
<tr>
<td>9.002</td>
<td>The placeholder is modelled according to relevant TenneT standards</td>
</tr>
</tbody>
</table>

All requirements from document ID437 and its derived requirements can be classified according to the system that is elaborated upon in the previous Section for document S5084. Two of the three requirements (ID 7.000 and ID 8.000) have a large geometrical component embedded in them, which makes the validation of them largely dependent on the ability to check these geometric values. The other requirements such as ID 7.001 can either be checked with one of the five defined types of SHACL shapes or need extra information such as with rule ID 9.002.
4.2. Transformer Ontology

To be able to perform a check against the rules as stated on the previous page, it is necessary to define the concepts and relationships that are embedded in those requirements so that the rule checker will recognize the input data. In other words, a mapping should exist between the data that will be validated and the requirements that are used for that validation. One way to do this is via an ontology, which can serve as a general information model regarding a certain topic. With OWL, an ontology can be developed using Linked Data principles. Besides providing a mapping between the semantics, terms and definitions used in the BIM model versus those in the rule documents, this ontology can serve other purposes too using the inferencing power of OWL. This will however not be within the scope of this project and therefore the focus of this subsection will be to develop an ontology that fits the goal of Qirion to standardize and modularize the design of transformers and transformer stations.

No fitting ontology exists that defines the semantics of a transformer space, transformer, building related installation and their underlying components and relationships. Additionally, the conversion of the BIM model from IFC to LBD is done using the BOT, PRODUCT and MEP ontologies, which are focused on buildings specifically and not assets from the energy sector. With that in mind, a small information model was created in Protégé that can be used to check a selection of requirements from the two rule documents. This selection will cover rule types 1 through 6 that are defined in the previous subsection 5.1. The name of this ontology is ModularTransformerCheckOTL and entails all the elements that are necessary for checking the requirements for cases 1, 2 and 3. Figure 37 shows the main page in Protégé, including the IRI of the ontology.

![Figure 37: Protégé interface showing the SimpeIGGI ontology](image)

4.2.1. Class definitions

Through the SHACL shapes, often nodes will be targeted through the class to which they belong. It is a likely scenario that the type of the objects in the instance data will not match with the description of the objects that are specified in the requirement documents. The instance data will therefore be enriched by giving the objects an additional class and thus mapping them to the vocabulary used in the rules. Figure 38 shows the classes that have been modelled in Protégé and are part of the ModularTransformerDemo ontology.
4.2.2. Object-, Data and Datatype property definitions

Object properties describe the different relationships that can be modelled between the classes. Two individuals in the Abox are linked to each other via an object property. Data properties on the other hand can link an individual to a data literal. Datatype properties add metadata to literals by stating information about its datatype. Figures 39, 40 and 41 depict an overview of those created properties in Protégé which add to the ontology used for the prototype development.
Now that the basic structure of classes and their relationships among each other is defined, this structure can be used to enrich the input data from the BIM model so that it matches this ontology. This ensures that during the rule execution phase, the rule interpreter understands which objects need to be checked, provided that the defined SHACL rules also take this ontology into account.

4.3. Conclusion

In this chapter, two steps are taken that are necessary to begin with the development of the rule checking system. First, the rule analysis resulted in the recognition of eight different types of requirements that exist within the two requirement documents of TenneT. Out of these eight types, five are possible to validate directly by using SHACL. These are the types (1) object relation requirement, (2) cardinality requirement, (3) data type requirement, (4) logical requirement and (5) value and unit requirement. The other three types are (6) geometric requirement, (7) referral to another document and (8) subjective requirement. For (6), a workflow will be developed so that the rule checking system can validate this type as well. Types (7) and (8) can indirectly lead to new rules that fall under categories (1) to (6), or will end up to be part of the few requirements for which human intervention is necessary. By making this typology, insight is created in the amount of requirements that are automatable. By extrapolating the information that is captured in Table 4 to the rest of document S5084, an estimation is made that approximately 10 out of
18, or 55.5% of the requirements can be validated with SHACL directly. If a workflow is used that enables the system to validate geometrical requirements as well in an indirect manner but eventually also by using SHACL, this percentage increases to 12 out of 18, or 66.6%. This means that two third of the requirements can be validated in an automated manner which thus reduces the amount of human involvement drastically.

The second prerequisite entails the creation of an ontology that specifies several standardized classes and object properties which form a T-box for the A-box information inside the BIM model. The definition of these 17 classes, 4 object properties and 2 data properties form the basis of the Modular Transformer Ontology that can be used in the rule interpretation and model preparation steps of the rule checking system.
CHAPTER 6 - PROTOTYPE DEVELOPMENT

This sixth Section documents the process of a first iteration through the methodology, specifically regarding two rule types that the final prototype should be able to check. For this proof of concept, the steps of three of the phases that are captured by the framework will be executed. These steps are model preparation, rule preparation and rule execution.

5.1. Case 1: Semantic validation

For this first part of the development process, the following rule should be checked:

The transformer space must include a building-related installation consisting of (emergency) lighting, cable carriers and service power outlets.

This sentence alone contains multiple elements that need to be verified:

- The BIM model must contain a transformer space.
- A building related installation must be present in the transformer space.
- The building-related installation should contain at least:
  o Emergency light fixtures.
  o Regular light fixtures
  o Cable carriers
  o Service power outlets.
- The elements that make up the building-related installation all need to be more than one instance, since the sentence specifies that multiple instances of the same entity should be present.

For this part, the BIM model of the building related installation is relevant. A visual representation of this is depicted in Figure 42.

Figure 42: the building related installation of the transformer vault
5.1.1. Model preparation

The BIM model as presented in Figure 42 that needs to be validated is delivered as an IFC-SPF file and is called TRAFO150_20kV_GGI.ifc. Figure 43 shows the model being opened in DDS-CAD Viewer, which is a BIM visualization tool and gives the user also the option to browse through the IFC file as can be seen on the left-side window in Figure 43. From there, it appears that the following objects with certain quantities are present in the model:

- IfcLightFixture, 12 times
- IfcBuildingElementProxy, 7 times
- IfcCableCarrierSegment, 63 times
- IfcCableSegment, 46 times
- IfcOutlet, 2 times
- IfcSwitchingDevice, 1 time

However, it seems that the IFC browser function is simplified, because some of the normal IFC objects that would need to be present, such as geometry, are not. This becomes clear when opening the BIM model in the DDS Schema Navigator, which gives the result as depicted in Figure 44. This program shows the complete IFC schema structure and when navigating this, eventually the instantiations of the IFC objects are reached. In Figure 44, it can be seen that the model contains 51,982 instances of the object IfcRepresentationItem, which covers most of the geometric aspects of the model. Furthermore, 4,520 instances of IfcRoot are present, under which we should be able to find the items that appeared in Figure 43. Figure 45 shows the expanded tree structure that lies under IfcRoot, eventually showing 109 instances of IfcFlowSegment. When clicking on these Ifc Flowsegments, it seems that they are the same objects as the 63 instances of IfcCableCarrierSegment and 46 instances of IfcCableSegments.
from the BIM Model when opened in DDS-CAD Viewer. This is finally ensured when looking at the properties of the objects. Figures 46 and 47 show that both the IfcFlowSegments as well as the IfcCableCarrierSegments are the same entities. In any way, this proves how important it is to model the information in a correct manner so that miscommunications due to data inconsistencies are prevented. This can be solved with data enrichment following the ontology that was designed in Section 5.2.
5.1.2. Model translation

For now, this analysis is put aside, and the focus turns to the conversion of the IFC-STEP file into a data model that follows the RDF data structure. From the literature review, it became apparent that this could be done best with the branched version of the IFCtoLBD convertor developed by Pauwels (2021). The conversion process transforms the building data from being written down in STEP Physical File Format to being written down in Turtle, which is a serialization of the RDF data model. The convertor uses several existing ontologies that are used to create Linked Building Data such as the BOT, BEO, and MEP ontology. During the conversion, the geometrical data is excluded from the rest of the data, resulting in a Turtle file describing the objects, their classes, their properties and the values of those properties.

As can be seen in Listing 15, when staying on the IfcFlowSegment and IfcCableCarrierSegment subject from before, this conversion stays true to the DDS Schema navigator program and uses the IfcFlowSegment naming.

5.1.3. Model enrichment

After converting the BIM model to an RDF format, the information needs to be enriched in order to comply with the required data structure that is set up. It appeared during the analysis of the received IFC models, that the classes used by the rule ontology are different from the classes used in IFC. Therefore, a mapping should be made between the converted BIM model and the ontology that is written in Protégé. Table 4001 on the next page gives an indication of the mapping that should take place for the rule that is used for this concept-proving first iteration of the system architecture.

To enrich the RDF data so that it adheres to the Simpel_GGI ontology, GraphDB is used. By using several SPARQL commands, the existing RDF data can be filtered, and additional data can be added. The goal is to add the different classes and relationships that are present in the ontology to the BIM model, so that it matches the ontology structure. A test was done with the objects in the BIM model that are named as 'NLRS_62_EF_klemmendoos…'. These objects are the service power outlets and should be a part of the building-related installation according to the to be checked requirement. As can be seen in Figure 48.
Table 6: Indication of names given to objects according to IFC, its RDF conversion and the Simple GGI ontology

<table>
<thead>
<tr>
<th>Name IFC</th>
<th>IFC type</th>
<th>#</th>
<th>Name RDF converted</th>
<th>IFC type</th>
<th>RDF Type converted</th>
<th>RDF Simple Instance</th>
<th>RDF Simple Ontology</th>
</tr>
</thead>
</table>

there are 5 service power outlets present in the BIM model. They look like small, rectangular boxes in the visual model, as depicted in blue in Figure 48.
As can be seen in Figure 48, there are two types of 'klemmendozen', namely type a and type b. They are modelled in such a manner that they don't have a common property, which is problematic in the rule definition phase. Figure 48 also shows that the class IfcBuildingElementProxy under which these 'Klemmendozen' fall, also is the upper class for two other entities. This makes it hard to define a SHACL shape that specifically targets the klemmendozen only. Therefore, these five klemmendozen need to be enriched with a common property, which follows the designed ontology, making them belong to the class 'Wandcontactdoos' or 'service power outlets'.

To achieve this data enrichment, first SPARQL is used in GraphDB to filter the five klemmendozen out of the rest of the dataset. Since there are two types of klemmendozen, a and b, a Regular Expression or RegEx SPARQL function was used to filter on a certain part of the string of one of the properties of the klemmendozen. This results in retrieving all desired five objects from the dataset, as can be seen in Figure 49.
Following this, a SPARQL INSERT function can be used to add the desired property or predicate and subject to the selected objects. An example of this can be seen in Listing 16 below, where the subject GGI:Kabelbaan is added as a type to all the mep:FlowSegments.

Listing 16: SPARQL query inserting the GGI:Kabelbaan property into all triples where the object is of type mep:FlowSegment

These described steps of data enrichment can be used to make the relevant instances of the data set follow the ontology, making the data suitable for automatic rule checking later in the complete process.

5.1.4. Rule preparation

Constraints for the data will be modelled in SHACL. This means that the information from the ‘rule to check’ part needs to be transformed into a SHACL shape. After writing the rules in SHACL, the rule check can be performed. Listing 17 shows the SHACL shape that was written to verify the requirement stated at the beginning of this section. It uses test data which is instantiated data of the SimpelGGI ontology made in Protégé.
The transformer space must include a building-related installation consisting of (emergency)lighting, cable carriers and service power outlets.

Listing 17: SHACL shape that can be used to verify the requirement 'The transformer space must include a building-related installation consisting of (emergency)lighting, cable carriers and service power outlets'
5.1.5. Rule execution

The check will be executed in the Visual Studio Code SHACL environment that was set up. This environment enables the user to paste a data graph and a shapes graph in separate screens, which will be checked against each other. This produces a validation report in the third screen, stating the results of the validation. PySHACL is used as SHACL processor during this phase.

The full set up can be seen in Figure 50 below. By typing ‘validate’ in the command line, a validation report pops in the screen on the most right side, showing the results of the validation.

![VSC environment showing the rule checking system](image-url)
5.2. Case 2: technical check

This part of the report aims to perform a check of a rule that belongs to the category of technical requirement checks. More specifically, this focuses on a requirement that involves the geometrical part of the BIM model’s data.

The requirement that is of concern for this is taken from amendment proposal document ID437 which proposes the following:

‘The three surge protectors and potheads that must be attached to the TenneT grounding transformer should have a center-to-center distance of 2400 mm’

This requirement was freely deducted from the document since it focused more on explaining the situation and necessary design changes instead of posing hard textual requirements. In figure 51, the surge protector and potheads are visible in blue. The BIM model was designed in such manner that these components are grouped in a so-called placeholder. The TRAFO150_20kV_PRIMAIR.ifc model therefore contains three placeholder objects representing these surge protectors and potheads. They are named respectively ‘Placeholder OSA 150kV fasen:Placeholder OSA 150kV fasen:691755’, ‘…-56’ and ‘…-57’, which is visible in Figure 51.

Figure 51: The three placeholders in the TRAFO150_20kV_PRIMAIR.ifc model that are required to have a center-to-center distance of 2400 mm

When looking at the property list as shown in Figure 52 of these placeholder objects, in the DDS-CAD viewer in which the model is opened, it shows that the placeholders have certain coordinates for their x-, y-, and z- positions relative to the ‘PEIL’ point. The properties X-, Y-, and Z-coordinate respectively have the values 2.375 m, 3.700 m, and 3.750 m, indicating the object’s location relative to ‘PEIL’. However, Listing 18 shows that these properties and their values are not visible in the IFC-to-RDF converted file by the IFC to LBD convertor. This means that the RDF graph in its current state does not contain the necessary data for checking the center-to-center distance between the surge protectors. In addition to that, Listing 18 lacks information about the exact location of the coordinate system in which the X-, Y-, and Z-coordinates lie. It is not clear whether these points specify a vertex of the object, the centroid, or another point.
Listing 18: Turtle snippet describing one of the surge protectors in the converted RDF graph from the original IFC file.

```turtle
instance:element_327823
  a:obo:Element;
  dbo:label "Placeholder OSA 150kV fasen:Placeholder OSA 150kV fasen:691755"^^xsd:string;
  dcterms:subject ""^^xsd:string;
  dcterms:reference ""^^xsd:string;
  dcterms:hasGif "f7a3b71-753c-4a7f-9261-c86d5f8de3615"^^xsd:string;
  dcterms:hasCompressedGif "3teuJ7YqV9X6G0X4r"^^xsd:string;
  dcterms:phased ""^^xsd:string;
  dcterms:reference "Placeholder OSA 150kV fasen"^^xsd:string;
  dcterms:reference "Placeholder OSA 150kV fasen"^^xsd:string;
  dcterms:reference "Placeholder OSA 150kV fasen"^^xsd:string;
  dcterms:area "3.565453718800115"^^xsd:double;
  dcterms:hook ""^^xsd:double;
  dcterms:volume "0.10370113322916"^^xsd:double;
  dcterms:level "PEIL"^^xsd:string;
  dcterms:moveWithinElements false;
  dcterms:elevationFromLevel "3750"^^xsd:double;
  dcterms:offsetFromHost "3750"^^xsd:double;
  dcterms:phaseCreated "Fase 1"^^xsd:string;
  dcterms:category "Generic Models"^^xsd:string;
  dcterms:family "Placeholder OSA 150kV fasen:Placeholder OSA 150kV fasen"^^xsd:string;
  dcterms:type "Placeholder OSA 150kV fasen:Placeholder OSA 150kV fasen"^^xsd:string;
  dcterms:typeName "Placeholder OSA 150kV fasen"^^xsd:string;
  dcterms:placeholder "00_ClashedObject"^^xsd:string;
  dcterms:category "Generic Models"^^xsd:string;
  dcterms:family "Placeholder OSA 150kV fasen"^^xsd:string;
  dcterms:reference "Placeholder OSA 150kV fasen"^^xsd:string;
  dcterms:category "Generic Models"^^xsd:string;
  dcterms:geojson "{}"^^xsd:string;
```

Figure 52: Part of the properties of the placeholder objects as shown in the DDS-CAD viewer, where -x, -y, and -z coordinates are listed as properties relative to the PEIL-point of the floor.
The next step is looking for these properties inside the original IFC file. When following the IFC schema structure, one needs to go through the following super-types to reach the placeholder entities: IfcRoot \(\rightarrow\) IfcObjectDefinition \(\rightarrow\) IfcObject \(\rightarrow\) IfcProduct \(\rightarrow\) IfcElement \(\rightarrow\) IfcBuildingElement \(\rightarrow\) IfcBuildingElementProxy. The Placeholders are of the type IfcBuildingElementProxy as can be seen in Figure 52. This is a type within IFC that can be used in a similar fashion as the IfcBuildingElement type, but does not have a pre-specified building type such as ‘floor’ or ‘beam’ (buildingSMART, 2013).

The IfcBuildingElementProxy type can have several properties. The ones that are relevant for the geometric representation of an IfcBuildingElementProxy are the IfcProductDefinitionShape and IfcLocalPlacement. The first describes the geometric shapes of which the element is composed, whereas the latter defines the local coordinate system describing the location of the element in the BIM model.

Listings 19, 20 and 21 show snippets from the IFC-step file describing the three surge protectors in the model. When looking at Listing 21, SPF line #837823 describes one of the surge protectors and refers to #837822 in order to find its IfcLocalPlacement. Following that to lines #837821 and #837819, we find the IfcCartesianPoints 2375, 3699, and 3750, which match the coordinate values that were found earlier in Figure 52.

The difference in distance between the Y-coordinates of these Cartesian points and thus the distance between the three surge protectors is 8199 – 5949, and 5949 – 3699 which both results in 2250. This is coherent with the 2250 mm that needs to be adjusted to 2400 mm according to document ID437. When looking into the geometric placement of an IfcBuildingElementProxy entity, it appears that its placement is indeed defined from the centroid of the XY-plane of the object, as can be seen in Figure 54 below (buildingSMART, 2017).
To prepare the BIM model for the rule checking part of the process, it needs to be enriched with the value that represents this distance between the centroids of the surge protectors. This value is implicitly present in the information of the BIM model and to make the data more valuable it should be made explicit so that the desired rule check can be executed. There are multiple options available which can succeed in extracting geometrical information from IFC data, converting it to RDF and adding it to an RDF database where the data can be used for queries, checks or other purposes. For this proof of concept, it is however chosen to write a python script that calculates the difference between the centroids of the surge protectors.

To access the data of the IFC file using python, first a new project is initiated in PyCharm, which is configured so that it uses a Conda environment. This environment is created through Anaconda Prompt in which the packages ‘ifcopenshell’ and ‘pythonocc-core’ are installed. Of these, the first adds the IfcOpenShell library to the environment which enables one to work with the IFC file format. The latter of the two adds the possibility to use Open Cascade 3D geometry modeling features and can be used together with IfcOpenShell to get the most out of geometric IFC data. By installing these two packages simultaneously in an environment through Anaconda, all dependencies on which they rely are automatically found and installed as well. Additionally, the correct version of Python that will work with both packages is installed in this environment.

When setting up the new PyCharm project, the created Conda environment is chosen for the configuration and its accessory Python version as interpreter for the project. When this configuration is completed, a python script can be written in PyCharm that can access the IfcOpenShell and OpenCascade external libraries to work with the IFC data and specifically the IFC geometrical data.

Listing 22 shows a script that calculates the center-to-center distance of the bounding boxes of the three surge protector objects in the BIM model. Lines 1-3 import several modules from the ifcopenshell and pythonocc packages and thereby give access to the functions included in these modules. Following that, some settings are configured in line 5-6 that ensure that the geometry that is returned from ifcopenshell.geom.create_shape() are PythonOCC shapes instead of standard IfcOpenShell geometry. Line 8 then specifies the IFC file of which the data is to be used in the consecutive lines that call this ifc_file variable. In lines 10-12, variables SP1, SP2 and SP3 are created which represent the three surge protectors. They are found in the data of the ifc_file by selecting the objects that have the GUID’s written as strings between the brackets in the ifc_file.by_guid function. Then in lines 14-16, three variables ‘shape1’, ‘shape2’ and ‘shape3’ are made, which are OCC geometries of SP1, SP2 and SP3. Consecutively, the ‘bbox1’, ‘bbox2’ and ‘bbox3’-variables represent three void box-geometries, which are added to the

Figure 54: Visual of the position of the local coordinate system of an IfcBuildingElementProxy
three OCC shapes of SP1, SP2 and SP3 by using the brepbdlib_Add method from the BRepBndLib module and thus creating a bounding box around the three surge protectors. Lines 26, 31 and 36 define the functions that determine the center points of those bounding boxes, of which the X-, Y-, and Z-coordinates are printed in the console screen as depicted in Figure 55. Lastly, center-to-center distance is calculated since that is the value of interest for the rule that is to be checked. This is defined in lines 41 and 42, of which the result is printed in the PyCharm console as indicated in Figure 55 as well.

```
import ifcopenshell.geom
import OCC.Core.Bnd
import OCC.Core.BRepBndLib

settings = ifcopenshell.geom.settings()
settings.set(settings.USE_PYTHON_OPENCASCADE, True)

ifc_file = ifcopenshell.open("IFCModels/TRAFO150_20kV_PRIMAIR.ifc")

SP1 = ifc_file.by_guid("3teujT3nArv9X6r0tX0L")
SP2 = ifc_file.by_guid("3teujT3nArv9X6r0tX0I")
SP3 = ifc_file.by_guid("3teujT3nArv9X6r0tX0J")

shape1 = ifcopenshell.geom.create_shape(settings, SP1.geometry)
shape2 = ifcopenshell.geom.create_shape(settings, SP2.geometry)
shape3 = ifcopenshell.geom.create_shape(settings, SP3.geometry)

bbox1 = OCC.Core.Bnd.Bnd_Box()
bbox2 = OCC.Core.Bnd.Bnd_Box()
bbox3 = OCC.Core.Bnd.Bnd_Box()

OCC.Core.BRepBndLib.brepbdlib_Add(shape1, bbox1)
OCC.Core.BRepBndLib.brepbdlib_Add(shape2, bbox2)
OCC.Core.BRepBndLib.brepbdlib_Add(shape3, bbox3)

bounding_box_center1 = ifcopenshell.geom.geom_utils.get_bounding_box_center(bbox1)
print("'X-coordinate SP1':", bounding_box_center1.X(),
      "'Y-coordinate SP1':", bounding_box_center1.Y(),
      "'Z-coordinate SP1':", bounding_box_center1.Z())

bounding_box_center2 = ifcopenshell.geom.geom_utils.get_bounding_box_center(bbox2)
print("'X-coordinate SP2':", bounding_box_center2.X(),
      "'Y-coordinate SP2':", bounding_box_center2.Y(),
      "'Z-coordinate SP2':", bounding_box_center2.Z())

bounding_box_center3 = ifcopenshell.geom.geom_utils.get_bounding_box_center(bbox3)
print("'X-coordinate SP3':", bounding_box_center3.X(),
      "'Y-coordinate SP3':", bounding_box_center3.Y(),
      "'Z-coordinate SP3':", bounding_box_center3.Z())

DiffSP1SP2 = ((bounding_box_center2.Y()-bounding_box_center1.Y())*1000)
DiffSP2SP3 = ((bounding_box_center3.Y()-bounding_box_center2.Y())*1000)

print("The center-to-center distance of SP1 and SP2 is:", DiffSP1SP2, "mm",
      "The center-to-center distance of SP2 and SP3 is:", DiffSP2SP3, "mm")
```

Listing 22: Python script that calculates the center-to-center distance of the bounding boxes of the three surge protector objects in the BIM model

Figure 55: Capture of the terminal output of executing the script as shown in Listing 22
Figure 55 shows that the distance between the center points of the surge protectors is indeed 2250 mm, which is coherent with the earlier deducted 2250 mm from the IFC files.

The next step in the model preparation phase is to ensure that this information is present in the data graph which will be checked in the rule execution phase. The value ‘2250’ with datatype ‘mm’ is now just a result printed by a python script, but it needs to become part of the transformed (IFC-RDF) BIM model.

To achieve this, RDFLib (RDFLib, 2021) will be used, which is a python library for working with RDF data. Listing 23 shows the follow up of Listing 22 whereby the necessary variables are saved in a graph which is eventually serialized in Turtle and saved on a specified location. Using RDFLib, one can create temporary and local containers in which triples can be added to form an RDFLib graph. The library also provides options for querying the data, adding existing RDF data from web locations, and changing triples using various operations. In Listing 17, a variable $g$ is created, which is a container that can be filled with RDF data. Lines 52 to 58 then first assign variables to the namespaces which are used for the ModularTransformer OTL as well as the RDF instance data that is to be checked. Using the bind function, the Turtle output file uses the assigned strings as namespace URIs. Following this, the desired structure of the RDF graph is made using two nodes ‘Distance_X’ and ‘DistanceValue’. Via a ‘RelatesObject’ predicate, two Objects can be linked to this Distance node which points to a value and unit eventually. This structure is visible in the schematic as shown in Figure 56.

Then, the actual triples that form the structure as depicted in Figure 57 are written down and added to the graph($g$) using the add() function. For the actual value that represents the distance between the Surge Protector objects in the IFC model, the variables $DiffSP1SP2$ and $DiffSP2SP3$ are called from the script in Listing 23 and transformed to the datatype ‘float’. Lastly, the function serialize() is used to save...
the graph in the desired format and to the correct location. The result of this is visible in Listing 24, which shows the Turtle file that is the result of executing the script in Listing 23.

```
from rdflib import URIRef, BNode, Literal, Namespace, Graph

g = Graph()

Data = Namespace("http://example.org/InstanceDataModularTransformer#")
Q = Namespace("http://www.qirion.nl/ModularTransformerQTL#")
q = Namespace("http://qudt.org/vocab/unit/"

q.bind('data', q)
q.bind('qtl1', Q)
q.bind('data', Data)

distanceOfDiffSP1SP2 = BNode()
distanceOfDiffSP2SP3 = BNode()

g.add((Data['SurgeProtec001'], Q['HasDistanceToSP_002'], distanceOfDiffSP1SP2))
g.add((Data['SurgeProtec002'], Q['HasDistanceToSP_001'], distanceOfDiffSP1SP2))
g.add((Data['SurgeProtec002'], Q['HasDistanceToSP_003'], distanceOfDiffSP2SP3))
g.add((Data['SurgeProtec003'], Q['HasDistanceToSP_002'], distanceOfDiffSP2SP3))

q.add((distanceOfDiffSP1SP2, q['unit'], q['MillIM']
q.add((distanceOfDiffSP2SP3, q['value'], Literal(float(DiffSP1SP2)))))
q.add((distanceOfDiffSP2SP3, q['value'], Literal(float(DiffSP2SP3)))))

s.serialize(destination = r'C:\Users\SJ44931\OneDrive - TU Eindhoven\Construction Management and Engineering\Graduation\r\"6. Development RC System\Final prototype files\Rule execution\Instance data\r\"geometrygraph.ttl', format='ttl')
```

Listing 23: Python script using the RDFLib library to transform values and variables into triples for in an RDF graph

```
prefix Data: <http://example.org/InstanceDataModularTransformer#> .
prefix Qtl1: <http://www.qirion.nl/ModularTransformerQTL#> .
prefix qudt: <http://qudt.org/vocab/unit/> .
prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

Data:SurgeProtec001 Qtl1:HasDistanceToSP_002 _:N374362ae7b25402aa7a7af7abb2864f5 .
Data:SurgeProtec002 Qtl1:HasDistanceToSP_001 _:N374362ae7b25402aa7a7af7abb2864f5 ;
Qtl1:HasDistanceToSP_003 _:Nae3c341967c045c91aa98d3e9797af .

Data:SurgeProtec003 Qtl1:HasDistanceToSP_002 _:Nae3c341967c045c91aa98d3e9797af .

_:N374362ae7b25402aa7a7af7abb2864f5 qudt:unit qudt:MillIM ;
qudt:value 2.25e+03 .

_:Nae3c341967c045c91aa98d3e9797af qudt:unit qudt:MillIM ;
qudt:value 2.25e+03 .
```

Listing 24: Resulting Turtle file showing the created RDF graph following the structure from Figure 56
5.3. Final prototype

In subsections 6.1. and 6.2. two major aspects of the rule checking system have been investigated and a proof of concept was made for a semantic check as well as a check using geometry data from the BIM model. The focus was primarily on checking two specific requirements by iterating through the first three steps of the system architecture. To complete the methodology and to produce the final prototype of a rule checking system for this thesis, a check will be performed on all 6 defined rule types. Additionally, the fourth step in the system architecture, rule reporting, is considered which will produce a human readable result that communicates the outcome of the BIM model check. Furthermore, a schematic is presented which shows the automation process that is required for this complete system. Lastly, this rule checking system is put into a systems engineering context, which places the narrowed scope of this project into the bigger picture of information exchange and data management within the AEC industry.

5.3.1. Rule interpretation

The following rules are selected for this final check:

1. The transformer space must include a building related installation consisting of (emergency) lighting, cable carriers and service power outlets.
2. Four lightning strikers are placed on a transformer space.
3. The wiring harness is provided with cable coils and copper connection vanes - (or forks).
4. The earthing ball's diameter is 25mm.
5. The value of the property diameter is of datatype ‘float’.
6. The three surge protectors and potheads that must be attached to the TenneT grounding transformer should have a center-to-center distance of 2400 mm.

Listing 25: SHACL shape representing rule 1.
These rules have been translated into SHACL shapes and follow the vocabulary as indicated in the ModularTransformerDemo OTL, which was specifically made for this check to map the objects in the BIM model to the SHACL rules for checking them. Listings 25 to 29 show the SHACL shapes for each of the six rules.

Listing 26: SHACL shape representing rule 2.

Listing 27: SHACL shape representing rule 3.

Listing 28: SHACL shape representing rules 4 and 5.
Listing 29: SHACL shape representing rule 6.
5.3.2. Model preparation

In the first case that is described in Section 6.1., the first step regarding the preparation of the model, all four original IFC files are converted into RDF files with a `.TTL` extension. Subsequently an enrichment step is necessary to map the object naming in the RDF dataset to the ModularTransformerDemo OTL. This ensures that in the rule execution step the correct and complete set of objects, properties and relationships are checked against the ruleset listed in Section 6.3.1. Using SPARQL queries, the necessary objects and their properties for this prototype are selected using the query form SELECT and an additional property is inserted using the INSERT operation, which maps the objects to the correct typology following the ontology. After finishing case 1 and 2 it seemed that this enrichment process, earlier done with the help of the SPARQL functionality of GraphDB, could also be done in a more fitting manner to the rest of the system architecture by using Python scripts, making the step automatable. Therefore, in this section, the model preparation step is done using the RDFLib Python library, which is also capable of querying RDF graphs using SPARQL. An example is given where, as can be seen in Figure 57, the lightning striker objects in the IFC model are not of the desired type as specified in the ModularTransformerOTL. The from IFC to RDF converted data model should thus be enriched with a triple stating that all lighting striker objects (Type Name NLRS_28_SC_TLB_kolom_buis_generiek_ISR:kolom_gen_ø88.9x2.5) are also of type QotL:LightningStriker. Listing 30 shows the Python script that will add a graph from a defined location to the temporary container. Additionally, it queries said graph on the four lighting striker objects and inserts the desired triple statement to those objects.

![Figure 57: 3D view of the BIM model in Solibri showing the properties of a Lightning striker object.](image-url)
Listing 30: Python script that uses SPARQL queries to find the correct objects in a loaded graph and maps them to the correct types according to the ModularTransformerOTL.
5.3.3. Rule execution

Section 6.3.2. describes and visualizes the mapping process which is used to enrich the objects in the dataset with additional properties which link them to the ModularTransformer OTL. Within the scope of this project, it would not be fitting to fully complete this mapping for every object that is relevant for the rule checking process regarding the six rules as defined in Section 6.3.1. Therefore, the rest of the data is manually generated, serialized in Turtle, which results in the dataset as displayed in Appendix D: Prototype data. It uses the mapped objects and their OTL-corresponding properties, as well as the generated geometrical RDF data of Section 6.2, and is thus extended with the missing necessary triples for the validation of the

![Image of a Turtle file](image.png)

Listing 26: SHACL Validation report that is the result of the final check

The dataset, as well as the SHACL shapes are placed in the PySHACL environment, and the underlying script is executed via the Command Prompt Terminal. The resulting SHACL validation report is shown in Listing 26. Listing 26 shows a violation of the ruleset, more specifically of the geometrical rule. This is as expected, since the rule states that the distance between the powersurge poles should be 2400 mm, but as already became clear in section 6.2., this model still contains the old distance of 2250 mm.

5.3.4. Reporting results

Finally, in this last section describing the final prototype, the focus is put on reporting the results back to the potential user of the prototype. An assumption is made that this user often has no deep knowledge of Linked Data, RDF and SHACL, so it is not suitable to directly feed them the SHACL validation report that is the result of the rule execution step. Therefore, an interface should be developed, or the information should be converted into a more human readable manner. It should at least contain the following information regarding a violation that has been made:
- Information about the object or object property for which the error occurs
  - Unique ID
  - Object type
- Information about the error that occurs
  - Unique ID of rule that is violated
  - Textual explanation of the rule that is violated

Multiple manners can be used to create this user-friendly report. Using IfcOpenShell together with PythonOCC, it is also optional to implement this reporting in such a manner that the violations visually pop up in a graphical viewer of the IFC model, so that the user can see where the violation occurs.

In addition to this change in format regarding the reporting of the results of the rule validation, it is also necessary to automate these four steps of the system, so that the usage of the system is as simple as possible for the user and includes as little as possible necessary manual actions. Figure 58 shows a schematic diagram which includes the conceptual architecture regarding the necessary steps for this automation process.

Figure 58: Schematic showing the steps to automate the prototype
CHAPTER 7 - REVIEW

This seventh and last chapter closes of this thesis by providing the reader with a review that places the methodology and subsequent results in a perspective regarding the scientific and societal relevance of this research. Looking back on the main research question and associated sub research questions, the added value to the current state of the art is discussed as well as the limitations that were present during the research process. Then, a final summary containing the most important findings is drawn up and is followed by the recommended steps that one can take when continuing future research on the topic of data driven validation on BIM models.

6.1. Conclusion

This study aimed to analyze the potential benefits for the AEC industry regarding the implementation of a decentralized rule-checking system for handover moments in a construction project that uses SW technologies. To gain insight in this matter, the following research question resulted as the main point of focus for this thesis:

‘What value can be created for the rule-compliance checking process during the design phase/handover phase of a construction project by using a workflow that involves semantic web technologies?’

To investigate this, a framework is developed that comprises of several essential elements for such a system using conclusions drawn from studying preceding relevant scientific research. This basis is expanded by two main parts. The first being an extensive analysis on rule typology, with the purpose to classify the various rules according to the SCHAL recommendation primer. The second expansion of the system concerns the creation of a method that enables the validation of technical requirements which rely on geometrical information in the BIM model. The final prototype of this system is made by going through three cycles, whereby each is focusing on a distinctive difficulty within the system. All of this comes together in the last of those three cycles, which results in the completion of the execution of the methodology.

The first sub question that accompanies the main research question is ‘what is the semantic web and why is it promising for usage in compliance checking in the AEC industry?’ Through literature review several reasons are found that substantiate arguments for using SW technologies to encourage a data driven workflow in the AEC industry. The RDF data model provides a strong, logic-based modelling technique which enables the linking of data that originates from various domains. A key concept that is essential for the future of information exchange in the building industry (Pauwels et al, 2017a). Especially in the handover moments between actors during the different project phases the accuracy of this exchange is a necessity to prevent data losses. This, in combination with the increasing complexity of projects for which the amount and difficulty of its requirements massively exceeds the capacity that lies within manual validation by people. The technology stack that is developed for the semantic web can offer relief for this issue by steering the process to a more data driven way of working instead of the currently used document based one. BIM models that use the RDF data model as underlying technology, and additionally the technologies embedded in RDFS, OWL, SPARQL and SHACL can be understood by computers. This makes it possible to infer new information that can enrich said model, to query the information stored in the data of the model, and to automatically validate requirements while using the same language.

To be able to enforce change or optimization in a process, it is necessary to have insight in the current situation. The second sub research question, ‘what is the current process of rule checking and data validation in a construction project’ intends to provide that insight. Via the study of Kovacs and Micsik (2021) it can be concluded that proprietary solutions as available on the market all lack at one or more functionalities which are described by Eastman et al. (2009) in their still leading conceptual structure for rule checking systems. These four functionalities are (1) rule interpretation; (2) BIM model preparation; (3) rule execution and (4) reporting the results the four functionalities and should all be incorporated in
an integral manner in these QC systems to ensure full functionality of these systems. Furthermore, despite the features that these systems offer, the current process of rule-based quality control is heavily dependent on human interpretation of rulesets and requirements and the modelling decisions that they make based on that. From the literature review it became clear that no standardized manner exists to transform rule-based requirements written down in human readable textual documents into a language that can be interpreted by a computer. The consequence is that each project is handled in a different manner which makes it difficult to automate this process. On top of that, this is an investment in time that both commercial and governmental organizations are not yet willing to make, because of a lack in understanding concerning the added value that a shift towards data driven processes can bring them.

The third sub research question is  ‘what challenges exist in implementing semantic web technologies for rule checking?’: Both processes of solving the necessary prototype prerequisites (rule analysis and OTL development) as well as going through the three use cases (semantic check, technical check, and full prototype design) aided in finding the answer to this question. It can be concluded that a challenge lies in the BIM modelling activity itself, because of a lack in standardization regarding the Quality Control process. Because rules, rulesets and rule types are not standardized, the BIM modeler designs the BIM model according to project specific guidelines, which often do not match the vocabulary that is used in the requirements. An ontology or Object Type Library are solutions for this. However, to match the requirements to the vocabulary of this ontology as well, an interpretation step is necessary to transform human readable requirements into computer readable ones. The rule type analysis resulted in a rough estimation which states that approximately 66.6% of the requirements in document S5084 can be automated in a process that uses SHACL. This includes geometrical requirements, which can also be validated by using the workflow that is described in Section 6.2. Eventually, this issue circles back to the problem that revolves around the lack of standardization, making it difficult to automate this process. Furthermore, a challenge lies in the representation of geometric data within RDF. For use case 2, the decision was made to write a hardcoded script using Python and the RDFLib library that can transform Python values and variables into RDF triples, which proved to be a successful method for this prototype.

Lastly, to be aid in answering the main research question, this thesis helped in gaining insight for the fourth sub research question: ‘what is a suitable workflow for using semantic web technologies for rule checking?’: The schematic that is shown in Figure 63 shows the technical workflow that is developed for this research. This is the implementation of the system architecture that is presented in Chapter 3 and the result of the three use cases. The components and arrows form a framework that can be integrated within an existing information management process that involves rule-based quality control of BIM models. It contains all the necessary four aspects for a rule checking system as well as the two prerequisite steps as defined in the methodology.

In conclusion and to answer the main research question ‘What value can be created for the rule-compliance checking process during the design phase/handover phase of a construction project by using a workflow that involves semantic web technologies?’: it can be said that involving semantic web technologies in a rule checking system improves the workflow for quality control. The prototype that is described in this thesis shows that it is possible for such a system to be based solely on open standards, which makes the process transparent for everyone involved. By using the SW technology stack, it is possible to create more valuable data because the strong logical basis of this technology stack enables querying and rule checking. It has also been proven that it is possible to separate the geometric and semantic data from an IFC file and only use the desired parts of that geometric data within the RDF data representing the original IFC data. This ensures that only the necessary information is present in the data, making this more structured and prevents the dataset from becoming too large. By using a rule checking system according to the architecture that is proposed, it is furthermore possible to automate the QC process which limits the amount of manual, and thus error prone, labor that is necessary for people and thereby this workflow is adapted to the ever-increasing amount of data that lies within AEC projects. Lastly, it is possible to implement this process in a step-by-step manner within the current workflow that companies use, which
will directly improve the quality of the produced BIM models and helps with a graduate transitioning from a document driven workflow towards a data driven one.

6.2. Discussion and recommendations for further research

This research’s added value lies in the complete dissection of the necessities for the development of a rule checking system. Each aspect is considered, and two additional steps are added to the recommended process that should be followed when designing such a system. This contributes to the scientific knowledge base as well as to professionals working in this field that deal with the previously indicated issues every day. It is however important to notice certain limitations that are inherent to the method used in this thesis and therefore the resulting prototype and its functioning should be discussed.

First, the method that is used to capture geometric information from the IFC file and to eventually transform this into RDF data is a method that uses hardcoded techniques. As soon as the input data, for example when a new project is started, the Python script must be adapted to these new circumstances as well. Therefore, this prototype only works within the boundaries of the cases that were set up for this research. This is also the case for the process of mapping the object types to the correct vocabulary as defined in the information model. Adding to this topic, the geometric data is now linked to the rest of the RDF data by writing new triples that use the vocabulary as specified in the created ontology. No comparison has been made with other methods of linking geometric data to the RDF data model, such as is done in Bonduel et al. (2019) and Wagner et al. (2019). Research specifically focusing on the combination of geometric information and its integration into the RDF data model should be executed to find the optimal way in doing this.

From the results and conclusions that follow from the research done for this thesis, several propositions can be made for additional topics that are suitable for future studies. These topics fit in different fields of expertise, such as (automation of) quality control, rule analysis and translation, BIM model creation, standardization and legislation, and process and information management. For each of these fields an example for future research is given.

Regarding the automation of quality control processes, it is recommended to research the technical possibilities for automation and the technologies that would be most suitable in doing so. Especially within the context of working within a Common Data Environment (CDE), it is necessary to investigate the possibilities for the integration of automated rule checking systems in this environment. To add on this, additional research could be done on the automation of the enrichment of BIM models. This research did not focus on the inferencing power that lies within the Web Ontology Language, which could possibly be valuable in the automatic inferencing of information that is not explicitly modelled, thereby making the BIM model more complete for the validation that follows this step. All these steps can eventually be implemented in one and the same CDE in which it is possible to view the BIM model, execute validations, see the resulting violations, and display the BIM models quality control related statistics on a user-friendly dashboard.

Then, a big part of this thesis focused on the translation of the rulesets against which the BIM model had to be checked. It was concluded that there is not yet a proper classification that describes different types of rules. A suggestion is made in this research to classify the rules with the SHACL primer in mind, but this could be done differently as well. Therefore, more research is needed in the field of rule classification. This could contribute to standardizing the automation process of rule translation, minimizing the amount of human bias that is involved with this. These last three suggestions for further research are especially relevant for the scalability of the resulted workflow using semantic technologies for quality control.

Lastly, it is recommended to continue researching the possibilities for standardization and legislation around the topic of information management in the built environment and other related sectors. This should
be the starting point when thinking about integrating the technologies as used in this thesis for industry cases, since it first and foremost depends on the agreements between the involved stakeholders in a project. They should become convinced of the change that is necessary to improve data and information management. Of course, this research focused on a very specific case using an example of TenneT and Qirion, or a client and a contractor. A solution for rule-based quality control should however be scalable throughout the whole of the AEC industry, where quality control is one of the main processes that is of importance for every project. It is therefore useful to zoom out and to try to place the result of this thesis in a bigger picture. If such a system should be implemented on the scale of the complete AEC industry in the Netherlands, a good starting point would be to separate this into several phases and sub steps with partial goals working towards one main purpose. A first division could for example be made when looking at the different levels that exist regarding involved rules and regulations in a project. The highest level are rules that originate from organizations and the standards they maintain such as the NEN, BIM-Loket and ISO. Two other levels can be defined for rules that are organization specific or even more detailed when they are project specific. In a parallel fashion, the involved people on these three levels can start working on the rule interpretation step by analyzing the documents and translating them into SHACL shapes with the help of information management experts. This is an investment, just like the implementation of BIM as a way of working, which eventually pays off and results in less errors and information losses. Eventually, by showing companies practical implementations such as would be the case when the results of this research are developed further, the added value can be recognized and made visible which encourages the industry to make this necessary investment to push the AEC industry towards new heights regarding the efficiency and accuracy of information exchange.

I hope that this research contributes to this as well, might it be indirect or direct, by demonstrating the possibilities that lie within the usage of Semantic Web technologies in the Quality Control process of BIM models.

Thank you for reading.
REFERENCES


W3C LBD community group (2022). Retrieved on 20-4-2022 from https://www.w3.org/community/about/


APPENDICES

A: BAM SE-guide rule analysis table
B: Requirement analysis
C: SHACL constraint types overview
D: Prototype data
Appendix A: BAM SE-guide rule analysis table

<table>
<thead>
<tr>
<th>Els</th>
<th>Type</th>
<th>Allocatie (OBS)</th>
<th>Werkpakket (WBS)</th>
<th>Niveau</th>
<th>SBS</th>
</tr>
</thead>
<tbody>
<tr>
<td>De veiligheid van motorrijders dient gewaarborgd te zijn</td>
<td>Aspect</td>
<td>BAM Wegen</td>
<td>Prestatie</td>
<td>L</td>
<td>X</td>
</tr>
<tr>
<td>De geleiderail is in bochten voorzien van een onderplank</td>
<td>Objectieis</td>
<td>BAM Geleiderail</td>
<td>Aanbrengen berm-beveiliging</td>
<td>Constructie</td>
<td>L</td>
</tr>
<tr>
<td>De onderplank wordt uitgevoerd in hetzelfde materiaal als de geleiderail</td>
<td>Objectieis</td>
<td>BAM Geleiderail</td>
<td>Materlaal</td>
<td>L</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Wegen</th>
<th>Bormbeveiliging</th>
<th>Markering</th>
<th>Kunstwerken</th>
<th>VTTI</th>
<th>Kabels en leidingen</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Appendix B: Requirement analysis

Document S5084

S5084

Modulebeschrijving

Module 02 - Transformatorruimte
Onder- en Regelstations

<table>
<thead>
<tr>
<th>Versie</th>
<th>Datum</th>
<th>Auteur(s)</th>
<th>Opmerking</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.0</td>
<td>31-12-2017</td>
<td>J. van Heumen</td>
<td></td>
</tr>
<tr>
<td>3.01</td>
<td>14-05-2018</td>
<td>J. van Heumen</td>
<td>Verwerking model 150(110)/10 kV</td>
</tr>
<tr>
<td>3.02</td>
<td>23-08-2018</td>
<td>J. van Heumen</td>
<td>Verwerking reviewpunten MDT</td>
</tr>
<tr>
<td>4.0</td>
<td>20-09-2018</td>
<td>J. van Heumen</td>
<td>Definitief</td>
</tr>
<tr>
<td>4.1</td>
<td>04-10-2018</td>
<td>J. van Heumen</td>
<td>Verwerking gesloten transformatorruimte</td>
</tr>
<tr>
<td>4.2</td>
<td>19-10-2018</td>
<td>J. van Heumen</td>
<td>Verwerking reviewpunten MDT</td>
</tr>
<tr>
<td>5.0</td>
<td>01-11-2018</td>
<td>R. Salomé</td>
<td>Definitieve versie</td>
</tr>
<tr>
<td>5.1</td>
<td>20-03-2019</td>
<td>H. Schröder</td>
<td>Wijzigingen Modular Bouwen versie 1.1</td>
</tr>
<tr>
<td>6.0</td>
<td>16-05-2019</td>
<td>H. Schröder</td>
<td>Accorderen van het MDT MB</td>
</tr>
<tr>
<td>6.2</td>
<td>21-07-2021</td>
<td>W. Mielkert</td>
<td>MB Release 1.3 ID32_ID03_ID116</td>
</tr>
<tr>
<td>7.0</td>
<td>29-07-2021</td>
<td>R. Salomé</td>
<td>Def. goedkeuring MDT en manager</td>
</tr>
</tbody>
</table>
## Meta data & Abstract

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>SS5091</td>
<td>Modulebeschrijving: Module 03:</td>
</tr>
<tr>
<td>Overzicht ontwerptekeningen Modulair Bouwen</td>
<td>Transformatieruimte Onder- en regelstations</td>
</tr>
</tbody>
</table>

### Categorie
Transportnet Elektriciteit

### Abstract

In de modulebeschrijvingen wordt de technische uitvoering van deze modules vastgelegd.
Documentautorisatie

<table>
<thead>
<tr>
<th>Document</th>
<th>Modulebeschrijving Module 02- Transformatie- en regelstaties</th>
<th>7.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>APM</td>
<td>Modulair Bouwen en Productmanagement</td>
<td></td>
</tr>
<tr>
<td>Geldig vanaf</td>
<td>29-07-2021</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Naam</th>
<th>Akkoord</th>
<th>Datum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Auteur</td>
<td>W. Melkert</td>
<td>21-07-2021</td>
</tr>
<tr>
<td>Goedkeuring</td>
<td></td>
<td></td>
</tr>
<tr>
<td>APM Modulair Bouwen</td>
<td>X</td>
<td>29-07-2021</td>
</tr>
<tr>
<td>en Productmanagement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(secundair)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Meico Boegems</td>
<td>X</td>
<td>29-07-2021</td>
</tr>
<tr>
<td>APM Modulair Bouwen</td>
<td>X</td>
<td>29-07-2021</td>
</tr>
<tr>
<td>en Productmanagement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(bovenduurde)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alexander Wolf</td>
<td>X</td>
<td>29-07-2021</td>
</tr>
<tr>
<td>APM Modulair Bouwen</td>
<td>X</td>
<td>29-07-2021</td>
</tr>
<tr>
<td>en Productmanagement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(Netstructuur)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enika Piga</td>
<td>X</td>
<td>29-07-2021</td>
</tr>
<tr>
<td>APM Klant en Ontwerp</td>
<td>X</td>
<td>29-07-2021</td>
</tr>
<tr>
<td>Ongevaarmanagement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Daphne Goethart</td>
<td>X</td>
<td>29-07-2021</td>
</tr>
<tr>
<td>APM Opdracht &amp; Monitoring</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Heleen Schiltier</td>
<td>X</td>
<td>29-07-2021</td>
</tr>
<tr>
<td>APM Instandhouding</td>
<td>Frank de Groot</td>
<td>29-07-2021</td>
</tr>
<tr>
<td>OIV</td>
<td>Paul Jansen</td>
<td>29-07-2021</td>
</tr>
<tr>
<td>Energiecontrole en balancering</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>mib</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Girion Technisch Beheer</td>
<td>Reinder Peters</td>
<td>29-07-2021</td>
</tr>
<tr>
<td>Girion Engineering</td>
<td>Tim Jansen</td>
<td>29-07-2021</td>
</tr>
<tr>
<td>Alliander Inkoop</td>
<td>Arloan de Jong</td>
<td>29-07-2021</td>
</tr>
<tr>
<td>Goedgekeurd namens MDT B&amp;P</td>
<td>Alex Geschiere</td>
<td>29-07-2021</td>
</tr>
<tr>
<td>Transportnetten E</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Goedgekeurd namens APM Modulair</td>
<td>Wouter de Wit</td>
<td>02-08-2021</td>
</tr>
<tr>
<td>Bouwen en productmanagement</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Disclaimer
Aan de inhoud van dit document kunnen alleen rechten ten opzichte van Liander nv worden ontleend, indien zij door rechtsge Burgess onderdeel worden ondersteund. De informatie kan van vertrouwelijke aard zijn en alleen bedoeld voor gebruik door de geadresseerde(n). Indien u dit document onterecht in uw bezit heeft, wordt u verzocht deze te vernietigen. Het is niet toegestaan dit document, of delen ervan, te wijzigen, te kopiëren of buiten zijn context te gebruiken.
Inhoudsopgave

META DATA & ABSTRACT ...........................................................................................................2

1 INLEIDING ...............................................................................................................................5
   1.1 BRIEFLY .........................................................................................................................5
   1.2 DOCUMENTEN ................................................................................................................6

2 MODULE TRANSFORMATORRUIMTE ...............................................................................7
   2.1 INTRODUCTIE ................................................................................................................7
   2.2 BEZOEKERSRUIMTE .......................................................................................................7
      2.2.1 Afspraken ................................................................................................................7
      2.2.2 Vennootschapstransformator ....................................................................................7
      2.2.3 Transformatorruimte ...............................................................................................7
      2.2.4 Sfeerimpressie .........................................................................................................8
      2.2.5 Kabelbouw ..............................................................................................................9
      2.2.6 Aardings- en blikseminstallatie ..........................................................................9

3 UITGANGSPUNTEN ..............................................................................................................10
   3.1 EXPLORING ..................................................................................................................10
   3.2 PREVIEW .......................................................................................................................12
   3.3 SEQUENCE ....................................................................................................................14
   3.4 GOL ................................................................................................................................14
   3.5 OVER ................................................................................................................................14

---

model content check

"x is converted to y with calling p, y, z, b.

data quality check

"x is an integer

technical check

"x is replaced according to this block."
1 Inleiding


In de modulebeschrijvingen wordt de technische uitvoering van deze modules vastgelegd.

1.1 Opbouw

In dit document wordt de opbouw van de module “Transformatorruimte” (gecodeerd 02) beschreven. De module “Transformatorruimte” wordt in verschillende varianten gebruikt in de volgende stationsmodellen:

- 150(110) kV KOP & RAP onderstation 80 MVA
- 150(110) kV KOP & RAP onderstation 160 MVA
- 150(110)/10 kV KOP & RAP onderstation 53 MVA
- 150(110)/10 kV KOP & RAP onderstation 106 MVA
- 50/10 kV KOP onderstation 40 MVA
- 50/10 kV KOP onderstation 80 MVA
- 20/10 kV KOP & RAP regelstation 20 MVA
- 20/10 kV KOP & RAP regelstation 40 MVA
- 10/10 kV Regelsation 20 MVA
- 20/20 kV Regelsation 40 MVA

Daarnaast zijn de volgende losstaande modules “Transformatorruimte” opgezet:

- 10/20 kV Transformatorruimte 20 MVA
- 20/10 kV Transformatorruimte 40 MVA
- 20/10 kV Gesloten transformatorruimte 20 MVA
- 50/10 kV Transformatorruimte 20 MVA
- 50/10 kV Gesloten transformatorruimte 40 MVA
- 50/10 kV Vrijstaande transformatorruimte 40 MVA
- 150(110)/10 kV Gesloten transformatorruimte 53 MVA
- 150(110)/10 kV Transformatorruimte 53 MVA
- 150(110)/10 kV Transformatorruimte met 150(110) kV rail aansluiting
- 150(110)/20 kV Transformatorruimte 80 MVA
- 150(110)/20 kV Transformatorruimte met 150(110) kV rail aansluiting

Deze losstaande modules zijn niet in een model toegepast. De module 50/10 kV transformatorruimte 20 MVA is alleen voor vervanging. De losstaande modules 10/20 kV transformatorruimte 20 MVA, 20/10 kV transformatorruimte 40 MVA, 150(110)/10 kV en 150(110)/20 kV transformatorruimten met en zonder rail aansluiting, worden zowel in nieuwbouw als ombouw toegepast, echter zijn dit geen reguliere standaard stations modellen.
Voor de stations waar omgevingsrechtelijk een gesloten transformatorruimte een vereiste is, moeten bij nieuwbouw en ombouw de modellen van de gesloten transformatorruimte worden toegepast.

1.2 Documenten

Bij deze modules horen de ontwerpen volgens S5091 “Overzicht ontwerptekeningen Modulair Bouwen” met de begincode ‘02’.

Verder worden in onderstaande algemene PV’s de functionele eisen opgenomen die van toepassing zijn voor de in deze module beschreven bouwstenen en componenten. In deze standaarden wordt ook doorgezet naar normen en voorschriften.

- S2001 - Voorschrift voor aarding bliksembeveiliging en stelpuntsbehandeling
- S5001 Algemeen PV E-Kabels
- S5002 Algemeen PV E-Primaire componenten
- S5003 Algemeen PV E-Secundaire componenten
- S8014 Algemeen PV E-Gebouw gebonden installaties
- S8010 Algemeen PV E-Terreinen en gebouwen.
2 Module Transformatorruimte

2.1 Inhoud

De module transformatorruimte beschrijft de uitgangspunten voor indeling en afmetingen van de transformatorruimten in verschillende modellen. De module transformatorruimte is opgebouwd uit een aantal bouwstenen die te onderscheiden zijn op basis van verschillende typen stations (stationsmodel), transformatoren en stapuitsbehandelings

2.2 Beschrijving bouwstenen

2.2.1 Algemeen

Elke vermogenstransformator wordt in een transformatorruimte geplaatst. Deze
transformatorruimte bestaat uit 3 wanden en een open voorzijde, al dan niet afgeschermd
met een hekwerk. In uitzonderlijke gevallen, waarbij de omgeving technisch noodzakelijk
is, kan een transformatorruimte voorzien zijn van 4 wanden waarvan één uiteenbaar om de
transformator te kunnen sleiden. In de bouwstenen is de standaardvariant met 3 wanden
uitgewerkt.

Voor de situaties waar omgeving technisch het niet mogelijk is om een transformatorruimte
zonder dak te plaatsen kan de gesloten transformatorruimte worden toegepast.

In de bouwstenen is rekening gehouden met de opstelling van de benodigde componenten,
slagafstanden, vluchtwegen, kabelverbindingen en werkruimte. Ook is, in geval van de
150(110)/20 kV en 150(110)/10 kV modules, rekening gehouden met de opstelling van 150
(110) kV componenten.

Uitgangspunt is dat de primaire transformatorm aansluiting (U1, V1, W1) aan de linker zijde
gezien vanaf de voorzijde van de transformatorruimte zijn geplacéerde. Hierdoor zijn de
regelkast en secundaire aansluitkast aan de voorzijde van de transformatorruimte
gepositioneerd. De 150(110)/20 kV en 150(110)/10 kV transformer is een uitzondering hier
op. Voor deze transformers is de positie van de regelkast en secundaire kast ten opzichte
van de positie van aansluiting (U1, V1, W1) vanuit het verleden anders gespecificeerd.

2.2.2 Vermogenstransformator

De verschillende vermogenstransformatoren zijn gespecificeerd in de
transformatorspecificaties SS076. De maximale afmetingen zoals genoemd in deze bestekken
zijn overgenomen in de bouwstenen.

De transformers zijn voorzien van een conservatortafel, primaire en secundaire bushings met
aansluiting: MK (Rheinhausen) regelschakelaar met regelkast en secundaire kast. Daarnaast
zij de transformers voorzien van beveiligingen en appendages zoals beschreven in de
transformatorspecificaties.

2.2.3 Transformatornruimte

Iedere transformatorruimte heeft een eigen fundatie, los van de fundatie van de overige
transformatorruimten en gebouwen.

De transformatorruimte bestaat uit een bouwbijlage van beton waarin een aardanker is
opgenomen bestaande uit herkenbare aardstaven. Deze staven zijn volgens het
aardingsvoorschrift S2001 "Voorschrift voor aardingen en bliksembeveiliging"
sterpunthandeling" verbonden met het betonijzer. De bakconstructie zijn aardplaten gestort die direct zijn verbonden met deze aardstaven.

De losse prefabricaties zijn voorzien van aardplaten. Deze platen worden onderling verbonden en aan de onderzijde verbonden met het aardcasser van de bakconstructie.

In de bouwsteen transformatorruimte zonder dak of voorgevel, daar waar regenwater in de olieopvangbak kan komen, worden voorzieningen voor de olie-waterscheider opgenomen. Voor de olie-waterscheider is standaard een uitvoering met vijf verval aansluiting op de verdeling. Indien een verval niet mogelijk is dan wordt een olie-waterscheiderpomp met type bundguard toegepast.

De olieopvangbak dient vloeistofdicht uitgevoerd te worden. Er mogen geen MS-labels of HS-kabels door de olieopvangbak lopen. Secundaire en GGI kabeldoorkomsten in de olieopvangbak dienen boven de olieniveau Grens geplaatst worden.

De transformatorruimte moet voorzien worden van en gebouw gebonden installatie bestaande uit (nood)verlichting, kabelbaren en service wandcontactdozen.

2.2.4 Sterpuntsaarding

Per stationsmodel is de sterpuntsaarding van de transformator uitgevoerd volgens het BS5000 Aanleg Transport. De sterpuntsaarding wordt via aardplaten gezaard aan het aardcasser van de transformatorbak. De sterpunstartom voorzien van 2-wikkelingen.

150/110/20 kV transformator:
- De 20 kV zijde heeft een driehoekskwikkeling en wordt gezaard met een separate aardingstransformator. De aardingstransformator is gespecificeerd in de S5013 "Specifications for 20 kV and 10 kV Earting transformers".
- De fundatie van de aardingstransformator wordt op afschat gestort naar de olieopvangbak van de vermogenstransformator. (gecombineerde olieopvang)

110/110/10 kV transformator:
- De 10 kV zijde heeft een driehoekskwikkeling en wordt gezaard met een separate aardingstransformator. De aardingstransformator is gespecificeerd in de S5013 "Specifications for 20 kV and 10 kV Earting transformers".
- De fundatie van de aardingstransformator wordt op afschat gestort naar de olieopvangbak van de vermogenstransformator. (gecombineerde olieopvang)

50/10 kV transformator:
- De 50/10 kV transformator is voorzien van een ingebouwde spoel (impedantie) in het sterpupt aan de 10 kV zijde.
- Het 10 kV sterpupt wordt rechtstreeks met aarde verbonden of voorzien van een op afstand bedienbare lastscheider (indien parallelbedrijf mogelijk is).

20/10 kV transformator:
- De 20/10 kV transformator is voorzien van een ingebouwde spoel (impedantie) in het sterpupt aan de 10 kV zijde.
- 10 kV sterpupt wordt rechtstreeks met aarde verbonden of voorzien van een op afstand bedienbare lastscheider (indien parallelbedrijf mogelijk is).
10/20 kV transformer:

- Het 10 kV stelpunt wordt niet aangesloten.
- Het 20 kV stelpunt wordt aangesloten op een externe stelpuntspoel welke tegen de transformatorwand wordt bevestigd. De grootte/impedantie van deze externe spoel, zal met een kortsluitberekening bepaald moeten worden.

10/10 kV en 20/20 kV regeltransformator:

- De 10/10 kV en 20/20 kV regeltransformator is niet voorzien van stelpuntaarding.

2.2.5 Kabeljuk

De bouwteken kabeljuk is uitgevoerd in thermisch verzinkt staal. Dit kabeljuk is geplaatst op funderingsbalken aan de laagspanningszijde van de 150(110)/20 kV, 150(110)/10 kV transformator en aan beide zijden van de 20/10 kV, 10/10 kV, 20/20 kV en 50/10 kV transformator. Het kabeljuk wordt voorzien van kabelaad, steunisolatoren, kopen aansluitvlaggen (of vork) en aardbuit (kogels). Het kabeljuk wordt geaard via aardplaten in de bakomnuster. De transformator wordt met litten en aansluitbomen aangedragen op het koperwerk van het kabeljuk.

Op het kabeljuk aan de HS zijde van de 5010 kV transformator wordt ook een aardingschakelaar en faseoverspanningsafleiders geplaatst.

De HS kabel worden afgemonteerd op de koper aansluitvlaggen (of vork) en vastgezet in de kabelaad.

2.2.6 Aardings- en blikseminstallatie

De aarding van componenten dient zoveel mogelijk via aardplaten in het beton uitgevoerd te worden om de hoeveelheid zichtbaar koper te beperken (t.v.m. diefstal).

Op de hoekpunten van de transformatorruimte zonder dak worden bliksemkappen geplaatst. De bliksemkappen dienen volgens het aardingsvoorschrift S2001 te worden aangesloten op een aardpunt welke in de directe nabijheid van de bliksemkappen gelegen is. Indien deze aardpunten niet buiten de transformatorruimte gelegen kan worden, wordt deze in de transformatorruimte gelegen. Hiervoor is een bouwkundige voorzieningen opgenomen te worden. In alle gevallen wordt de aardpunt ook verbonden met het aardraster in en/of om het gebouw.

Bij de gesloten transformatorruimte dient conform de S2001 een daknet, t.b.v. de bliksemafvang, te worden aangebracht.

Een zichtbaar aardingsnetwerk is alleen toegestaan waar zich een grote concentratie van te aarder componenten / gebouwstelen bevindt.

In de 150(110)/20 kV en 150(110)/10 kV transformatorruimte worden door TenneT componenten geplaatst. Deze worden, conform TenneT, geademd. Het zichtbare deel van deze aardingsinstelling dient, t.v.m. diefstal, bij voorkeur gemaakt te zijn, en wordt verbonden via aarden platen aan het liander aardraster.

Op zowel de koper aansluitvlaggen 20 kV en 10 kV als op de kabelaad, worden aardkogels Ø25mm aangebracht om bij werkzaamheden een zichtbare aarding aan te brengen. In de transformatorruimte (en) wordt geen aarders toegepast op 10 kV en/of 20 kV niveau.
3 Uitgangspunten

De module transformatormuimte is op basis van de uitgangspunten ontworpen.

3.1 Bouwkundig

Maatvoering:
- De maatvoering is gebaseerd op de gekozen maximale afmeting per transformatorenuitvoering zoals opgenomen in de transformatorbestekken S5076 (inclusief koelers).
- Ten minste 850x2300 (bxh) vrije ruimte (vluchtwegen) rondom de transformatoren aanhouden, gerekend vanaf de zijkant van de transformatoren en openstaande deuren van de kast aan de transformatoren;
- De minimale slagafstand (buiten) f-f 20 kV 400 mm (center to center = 750 mm);
- De minimale slagafstand (buiten) f-f 10 kV 300 mm (center to center = 500 mm);
- Bij de gesloten transformatormuimte is voor de hoogte ten onder aan de maatvoering uitgezet naar bouwkundige vaste maten vermeerderd met 3 m welke nodig is voor uitgutheden van de regelschakelaar; daarboven is nog 500 mm aangehouden voor opvoorzien.

Center to center is de minimale afstand tussen de pitten van de transformatoren of de afstand tussen middelpunt primaire aansluiting/geleider.

De minimale slagafstand moet worden toegepast tussen geleiders en wanden.

Olieopvangbak
- Bij het ontwerp van de olieopvangbak is rekening gehouden met sproeiën van olie vanuit het hoogste punt van de transformatoren. De afstand tussen de olekeerwand en transformatoren is, aan alle zijden, tenminste 20% van de hoogte van de transformatoren (incl. conservator). E.e.a. conform de NEN-EN-IEC 61936-1. Indien de buitenwanden van de olieopvangbak aan de bovenzijde schuin afgestopt naar de oliebak zijn uitgevoerd zal eventueel sproeiende olie in de opvangbak belanden. Dan wordt deze wand bij het olieopvang gerekt nodig worden. Rondom fundament looproosters tot aan wanden;
- De looproosters moeten, waar nodig, voorzien zijn van sporingen en schopranden voor aarding-en kabeldoornvloeiingen.
- Ter plaatse van de roostervloer en de betonrand is een hoogte verschil van 20 mm aangehouden; dit in verband met het niet volledig vlak kunnen storten van het beton en de vellingkant (20x20 mm) die nodig zijn om het beton te kunnen ontkisten. Zie onderstaand detail. Er ontstaat door de vellingkant een opstap/afschuining waardoor er geen struikelgevaar is van een opstaande rand.
Figuur 1: Detail van de roostervloer en de betonrand.

- Alle betonconstructies voorzien van aardraster in beton en met aardplaten naar buiten gevoerd;
- De opvangbak uitvoeren met in het werk gestort beton.

**Bovenbouw 3 zijden omsloten**

- De wanden voor de 150(110)/20 kV, de 150(110)/10 kV en de 50/10 kV onderstations en de 20/10 kV regelstations uitvoeren in prefabricat beton;
- De wanden voor de 150(110)/20 kV, 150(110)/10 kV en 50/10 kV onderstations uitvoeren in 3 dichte wanden en één open zijde;
- Bij de 20/10 kV, 10/20 kV, 20/20 kV en 10/10 kV regelstations 3 dichte wanden en de open zijde voorzien van een spijlen hekwerk;
- De wanden voor de 20/20 kV en 10/10 kV regelstations uitvoeren in gemetselde wanden;
- De hoogte van de wanden is gebaseerd op de afmeting van de transformator en heeft minimaal de hoogte van maximaal gespecificeerde hoogte van de transformator;

**Bovenbouw volledige gesloten transformatorruimte**

- De wanden voor de 150(110)/20 kV, de 150(110)/10 kV en de 50/10 kV onderstations en de 20/10 kV regelstations uitvoeren in prefabricat beton;
- De wanden voor de 150(110)/20 kV, 150(110)/10 kV en 20/10 kV stations uitvoeren in 4 dichte wanden, de te openen wand voorzien van een uitneembare paneel waarin twee (vleugeldeuren en AKR roosters onderin en bovenin zijn gesitueerd;
- De hoogte van de wanden is gebaseerd op de afmeting van de transformator en aanvullende hoogte voor uitblijven van de regelschakelaar;
- In het uitneembare paneel zitten AKR roosters, in de bovenzijde van de wanden zitten Coulissen roosters zodat de luchtstroom geventileerd wordt, maar wel geluidsdeemiging plaats vindt. Bij de toepassing van deze roosters is de koeling van de transformator en transformatorruimte afdoende.
Dak volledige gesloten transformatorruimte

- Het dak uitvoeren in beton, overheliend aan de wanden (zodat er geen goten noodzakelijk zijn). Indien dit niet mogelijk is, in verband met de positionering van deze module, dan het dak uitvoeren als een inzandig dak met afvoeren aan de buitenzijde;
- **Geen dakbedekking toepassen k v m onderhoud en inspeclie**

50kV KOP station Liander (componenten niet gestandaardiseerd in modulair ontwerp)

- Opstelling componenten volledig buiten de transformator olie opvangbak;
- Opstelling componenten en ondersteuningsconstructie voor kabeleindsilting, motoredeelende aarder, sterpunts-overspanningsafleider (voldoende ruimte voor sterpuntsbaarder), fase-overspanningsafleider;
- Kabel (geen lijn), invoer kabel op projectbasis te bepalen;
- **Fase-afstand componenten: 1.500 mm**

150/110 kV TerneT (niet in scope van standaardisatie)

- De opstelling van de componenten is volledig buiten de transformator olie bak;
- Er is ruimte voor de opstelling van componenten en ondersteuningsconstructie voor: kabeleindsilting, aarder, sterpunts-overspanningsafleider (voldoende ruimte voor sterpuntsbaarder), fase-overspanningsafleider;
- De invoer van kabel (geen lijnen), dient op projectbasis te worden bepaald;
- De fase-afstand van de componenten is 2.250 mm.

3.2 Primair

Uitvoering transformator

- **Conform transformerspecificaties S5076**

Stroomtransformator voor sterpuunt

- **Conform de S6032 Stroomtransformatoren**;
- Gehard ringkern uitvoering geschikt voor buitenopstelling;
- Twee kernen in één behuizing;
- **Geschikt voor doorvoeren aardstrip 30x5mm**;
- Belastbaarheid: 300 A / 120 sec 2500 A / 6 sec;
- Kabelinvoer: 2 EMC wartels onderzijde;
- P1 bovenzijde / P2 onderzijde.

20 kV en 10 kV kabels

- Voor de 20 kV en 10 kV transformatkabels zie de S5090 Modulebeschrijving kabels
- **Flexibele aansluiting (Lizze) van kabelrek naar transformator**
- 20 kV en 10 kV kabels in buitenopstelling tegen UV beschermen met UV bestendige krimpkous;
- Kabeldoorvoeringen vloeistofdicht afwerken;
Sterpuntsaarding 150(110)/20 kV en 150(110)/10 kV transformator:

**10 kV en 20 kV aardingstransformator:**
- Conform de S6013 Eerhoud transformers gespecificeerd.
- Aansluiten met UV bestendige kabel aan koper aansluitvlaggen kabeljuk.
- UV bestendige kabel in geschikte kabelblokken monteren aan wand transformatornuimte.
- Aansluiting per fase met stekker eindsluiting.
- Sterpunkt aardingstransformator voorzien van viagaanluiting.
- Sterpunkt aardingstransformator met koper strip 30x5 mm, geisooleerd, geïnstalleerd, doorringkernstroomtransformator voeren en aansluiten op aardraster transformatorbak.
- Ringkernstroomtransformator op staalconstructie monteren aan wand transformatornuimte.

**Sterpuntsaarding 50/10 kV transformator:**
- De 50/10 kV transformator is voorzien van sterpuntsaansluiting, in het sterpunkt van aan de 10 kV zijde is een sterpuntspool in de transformatorengebouwd.
- Sterpunkt 50/10 kV transformator is voorzien van viagaanluiting.
- Sterpunkt 10 kV zijde transformator rechtstreeks verbonden met aardraster transformatorbak.
- Sterpunkt gesloopt, geïnstalleerd, doorringkernstroomtransformator voeren en aansluiten op aardraster transformatorbak.

**Sterpuntsaarding 20/10 kV transformator:**
- De 20/10 kV transformator is voorzien van sterpuntsaansluiting, in het sterpunkt van aan de 10 kV zijde is een sterpuntspool in de transformatorengebouwd.
- Sterpunkt 20/10 kV transformator is voorzien van viagaanluiting.
- Sterpunkt 20 kV wordt niet aangesloten.
- Sterpunkt 10 kV zijde transformator rechtstreeks verbonden met aardraster transformatorbak of 10 kV zijde transformator, indien parallelbedrijf mogelijk is, verbonden met aardraster transformatorbak via een lastscheider geplaatst aan de wand van de transformatornuimte.
- Sterpunkt gesloopt, geïnstalleerd, doorringkernstroomtransformator voeren en aansluiten op aardraster transformatorbak.

**Sterpuntsaarding 10/20 kV transformator:**
- De 20/10 kV transformator is voorzien van sterpuntsaansluitingen. In de transformatoren is aan het 10 kV sterpunt een sterpuntspool gebouwd.
- Sterpunten van de 20/10 kV transformator zijn voorzien van viagaanluiting.
3.3 **Secundair**

- Secundaire bekisting gecombineerd met GGI bekisting in thermisch verzonken ladderbaan onder roostervloer.

3.4 **GGI**

Verlichting/brachtstroom:

- 2 Stuks CEE-form service WOD, 6302 A • 2 stuks 230 V 16 A voorzien van benodigde aardstekbeveiligingen en minimale kortsluitstroom van 10 kA;
- Verlichting vanuit beloopbaar oppervlak te onderhouden;
- Voorzien van noodverlichting met een minimale autonomie van 3 uur;
- Bekisting GGI installatie gecombineerd met secundaire bekisting in thermisch verzonken ladderbaan onder beloopbaar oppervlak.

3.5 **Overig**

Aarding

- Aarding van: kabelvloeren, kabeladders, staalconstructies, aardingstromator, vermogenstransformator via aardplaten in het beton volgens S2001 'Voorschrift voor aarding, bliksembeveiliging en sterpuntsbehandeling'.

---

APM Modulair Bouwen & Productmanagement

Page 34 van 34
### Wijzigingsvoorstel B&P Transportnetten E

**ID437 Poolafstand 150kV component MB Transformatorruimten**

<table>
<thead>
<tr>
<th>Status log</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Toegewezen ID in B&amp;P Transportnetten E bevindingenlijst</td>
<td>ID437</td>
</tr>
<tr>
<td>Melding van bevinding</td>
<td>5-2-2021 door W. Melkert</td>
</tr>
<tr>
<td><strong>Classificatie van bevinding door het kernteam</strong></td>
<td></td>
</tr>
<tr>
<td>□ Urgent:</td>
<td></td>
</tr>
<tr>
<td>□ Energielevering of veiligheid wordt negatief beïnvloed (niet IBN waardig); of</td>
<td></td>
</tr>
<tr>
<td>□ MB standaard is technisch niet realiseerbaar in het realisatieproject én IBN-datum in direct gevaar.</td>
<td></td>
</tr>
<tr>
<td>□ Anders namelijk:</td>
<td></td>
</tr>
<tr>
<td>☒ Optimalisatie: Verbetering of additionele functionaliteit, maar niet urgent.</td>
<td></td>
</tr>
<tr>
<td>Documentversie en laatste wijzigingsdatum</td>
<td>0.2 op 01-06-2021</td>
</tr>
<tr>
<td>Auteur(s)</td>
<td>W. Melkert</td>
</tr>
<tr>
<td>Behandeld door het kernteam</td>
<td>Versie 0.2 op 9-6-2021</td>
</tr>
<tr>
<td>Product Owner akkoord met indiening bij het MDT</td>
<td>09-06-2021 door Marco Bongers namens Alex Geschiere</td>
</tr>
<tr>
<td>Ter goedkeuring aangeboden aan voorzitter MDT</td>
<td>09-06-2021 door Tom Janssen</td>
</tr>
<tr>
<td>Besluit MDT en vervolgstappen</td>
<td>Besluit op 17-06-2021 gedeeld door Renata SalaMé: 09-07-2021</td>
</tr>
<tr>
<td></td>
<td>De goedgekeurde versie van dit wijzigingsvoorstel</td>
</tr>
<tr>
<td></td>
<td>Vervolgstappen:</td>
</tr>
<tr>
<td></td>
<td>1. Communicatie met het kernteam B&amp;P Transportnetten E en overige belanghebbenden (w.o. ‘teken’-team, realisatieprojecten) m.b.t. MDT besluit, impact en vervolgstappen. [actiehouder Tom Janssen]</td>
</tr>
</tbody>
</table>
Verantwoording inhoud en kwaliteit door kernteam

De inhoud en kwaliteit van dit wijzigingsvoorstel wordt volledig gedragen door het kernteam B&P Transportnetten E: een regio dekkende afvaardiging van specialisten van Orion Engineering, Assetbeheer en het calculatiebureau. Zie onderstaande tabel voor de samenstelling.

<table>
<thead>
<tr>
<th>Naam</th>
<th>Rol</th>
<th>Akkoord</th>
<th>Datum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tom Janssen</td>
<td>Voorzitter</td>
<td>Ja</td>
<td>08-06-2021</td>
</tr>
<tr>
<td>Wilfred Melkert</td>
<td>Module engineer primair</td>
<td>Ja</td>
<td>08-06-2021</td>
</tr>
<tr>
<td>Jeroen Butter</td>
<td>Module engineer primair</td>
<td>Ja</td>
<td>08-06-2021</td>
</tr>
<tr>
<td>Arjan Klein Holte</td>
<td>Module engineer BTI</td>
<td>Ja</td>
<td>08-06-2021</td>
</tr>
<tr>
<td>Erik de Kaper</td>
<td>Beheerder</td>
<td>Ja</td>
<td>08-06-2021</td>
</tr>
<tr>
<td>René van Etten</td>
<td>Consultant primair</td>
<td>Ja</td>
<td>08-06-2021</td>
</tr>
<tr>
<td>Luki Budiarto</td>
<td>BIM Consultant</td>
<td>Ja</td>
<td>08-06-2021</td>
</tr>
<tr>
<td>Richard Wismans</td>
<td>Engineer GGI</td>
<td>N.v.t.</td>
<td></td>
</tr>
<tr>
<td>Aant Klaas de Jong</td>
<td>Module engineer secundair</td>
<td>Ja</td>
<td>08-06-2021</td>
</tr>
<tr>
<td>Bram Staarink</td>
<td>Module engineer kabel</td>
<td>Ja</td>
<td>08-06-2021</td>
</tr>
<tr>
<td>Remco van Loon</td>
<td>Module engineer BTI</td>
<td>Ja</td>
<td>01-06-2021</td>
</tr>
<tr>
<td>Robert-Paul van Campen</td>
<td>Cost engineer</td>
<td>N.v.t.</td>
<td></td>
</tr>
</tbody>
</table>
Inhoudsopgave

1. BEVINDING ........................................................................................................................................... 4
   1.1 OMSCHRIJVING BEVINDING ........................................................................................................... 4
   1.2 WAAROM IS DE BEVINDING EEN PROBLEM ............................................................................. 5

2. MOGELIJKE OPLOSSINGSRICHTINGEN ......................................................................................... 5
   2.1 OPLOSSINGSRICHTING 1: AANPASSEN POOLAFSTAND, SLAGAFSTAND EN STELPUINTAARDING .............................................................................................................................. 5
   2.1.1 Omschrijving ............................................................................................................................ 5
   2.1.2 Voor- en nadelen t.o.v. huidige situatie ..................................................................................... 6
   2.1.3 Impact op standaarden ............................................................................................................... 7

3. UITGEWERKT WIJZIGINGSVOORSTEL VOOR VOORGESTELDE OPLOSSINGSRICHTING ............ 8

4. IMPACT OP DE STANDAARD ............................................................................................................. 11
   4.1 DOCUMENTEN IN DE AM SHOWROOM ...................................................................................... 11
   4.2 OVERIGE BELIJD- OF STANDAARDISATIEDOCUMENTEN IN DE AM SHOWROOM ......................... 11
   4.3 BEHEER (O.a. BEI, DOCUMENTATIE) .......................................................................................... 12
   4.4 PROJECTEN .................................................................................................................................. 12
   4.5 TRAININGEN ................................................................................................................................. 12
   4.6 EVENTUELE OPMENNINGEN ......................................................................................................... 12

5. FINANCIËLE IMPACT ......................................................................................................................... 13
   5.1 EENMALIGE KOSTEN ...................................................................................................................... 13
   5.2 WEDERHERSTELDOEPEN PER PROJECT ...................................................................................... 13

6. IMPACT OP DE PROJECTPLANNING ............................................................................................... 14
1. Bevinding

1.1 Omschrijving bevinding

Ten aanzien van de 150 kV aansluiting in de 150(110)/10 kV en 150(110)/20 kV Modulaire Transformerruimten zijn drie samenhangende onderwerpen aangedragen in één bevinding. In dit WV zijn deze onderwerpen apart van elkaar genoemd en behandeld.

- Onderwerp 1: Poolafstand 150(110) kV componenten TenneT vergroot van 2250 mm naar 2400 mm
- Onderwerp 2: Verbeteringen en correctie in transformerator place-holder
- Onderwerp 3: Toevoegen place-holder 150(110) kV sterpunt aarder/OSA combinatie.

Onderwerp 1 Poolafstand 150(110) kV componenten

De 150(110)/10 kV en 150(110)/20 kV transformeratorruimte ontwerpen in Modulair Bouwen bevatten enkele 150(110) kV componenten die door TenneT geleverd en geplaatst worden zoals de 150(110) kV overspanningsafleiders, kabeleindsluitingen en transformerator-aarder. Deze componenten staan in de Modulair Bouwen transformeratorruimte opgesteld als place-holder met een onderlinge poolafstand van 2250 mm. De TenneT standaard transformerator-aarder voor deze toepassing, heeft echter een minimale poolafstand van 2400 mm. Deze afstand is nodig om de aarder te kunnen voorzien van een standaard TenneT aansluitvlag in plaats van een stift aansluiting. Bij een kleinere poolafstand (<2400 mm) is de f-f slagafstand tussen de fasen, door de breedte van deze uitvoering, niet goed te borgen.

Onderwerp 2 Slagafstanden in ontwerp transformerator place-holder.

In de 150(110)/10 kV en 150(110)/20 kV transformeratorruimten, is onvoldoende rekening gehouden met de slagafstand tussen de transformerator aansluitkabel 150(110) kV en de transformerator conservator van het model. In deze modellen blijkt tussen de transformerator place-holder en de aansluitkabel een slagafstand van ca. 1375 mm aangehouden te zijn. De minimale slagafstand fase-aarde moet 1500 mm zijn. Daarnaast bevatten de transformerator place-holders enkele fouten waardoor de slagafstand tussen conservator en 150(110) kV bushing te klein is aangehouden in het ontwerp. In de 150(110)/10 kV 53 MVA place-holder, is de maatvoering uit de betreffende transformerator specificaties ten aanzien van de middelste fase positie verkeerd overgenomen. (Zie hoofdstuk 3 figuur 5)

Onderwerp 3 Sterpunt aarding 150(110) kV

In de ontwerpfase van Modulair Bouwen is onvoldoende aandacht besteedt aan de 150(110) kV sterpunt-aarding. Er is een poer opgenomen in het ontwerp, maar er lijkt geen toetsing plaatsgevonden te hebben op de inrichting van de sterpunkt-overspanningsafleider in combinatie met een sterpunkt-aarder. Er is ook geen place-holder aanwezig in het ontwerp. Dit leidt tot ontwerpproblemen en veel project specifiek overleg tussen Reddyn en TenneT.
1.2 Waarom is de bevinding een probleem

Onderwerp 1 Poolafstand 150(110)/20 kV componenten.
De TenneT standaard aarder heeft een poolafstand van 2400 mm. Hoewel de huidige MB-poolafstand van 2250 mm maakbaar is door toepassing van niet TenneT standaard aansluitklemmen met stift uitvoering, geeft TenneT aan dat een poolafstand van 2400 mm beter past binnen de TenneT standaarden. Per project wordt nu intensief gediscussieerd met TenneT over deze poolafstand. Het Modulair Bouwen transformatorruimte ontwerp sluit niet meer aan bij de TenneT standaarden op dit vlak.

Onderwerp 2 Slagafstanden in ontwerp transformator place-holder.
De te kleine fase-aarde slagafstand uit het model vertaalt zich in de praktijk tot noodgrepen om de minimale slagafstand van 1500 mm alsnog te realiseren. Het standaard Modulair Bouwen ontwerp schiet op dit punt tekort. De uit de transformerspecificatie verkeerd overgenomen maatvoering van de 150(110)/10 kV doorvoer isolatoren leidt tot te kleine 150(110) kV slagafstanden. Ook is het 10 kV kabelkruk hierdoor niet in lijn met de middelste fase van de transformator. (Geen direct technisch probleem)

Onderwerp 3 Sterpunt aarding 150(110) kV.
De standaard Modulair Bouwen transformatorruimte ontwerpen ten aanzien van de 150 kV sterpunt aansluiting van transformatoren schiet te kort. In gevallen waar TenneT een schakelbaar sterpunt verlangt ontbreekt een goed getoetste sterpunt-aarder in combinatie met sterpunt-overspanningsafleider concept opstellingstekening.

2. Mogelijke oplossingsrichtingen

Overzicht oplossingsrichtingen:
Oplossingsrichting 1: Aanpassen poolafstand, slagafstand en sterpuntaarding

2.1 Oplossingsrichting 1: Aanpassen poolafstand, slagafstand en sterpuntaarding

2.1.1 Omschrijving
De oplossingsrichting is per onderwerp beschreven. In hoofdstuk 3 zijn afbeeldingen weergegeven van de voorgestelde oplossingen.
Onderwerp 1.
Binnen de Modulair Bouwen standaard transformatorruimte is voldoende ruimte aanwezig om de poolafstand te vergroten van 2250 mm naar 2400 mm. Hierdoor kan de TenneT standaard aarder toegepast worden. Om deze oplossing door te voeren is het nodig om de Modulair Bouwen ontwerpen aan te passen ten aanzien van de fundatie waarop de componenten van TenneT geplaatst worden inclusief de 150(110) kV place-holder. De poeren in het beton zullen harl op har een afstand krijgen van 2400 mm net als de place-holder van de 150(110) kV opstelling waarin de transformator-aarder, overspanningsafleider en kabel-eindsluiting zijn opgenomen.
Het ontwerp wordt afgestemd met TenneT zodat dit binnen TenneT breed gedragen wordt.

Onderwerp 2.
Binnen de Modulair Bouwen standaard transformatorruimte is voldoende ruimte aanwezig om de poeren van de 150(110) kV componenten te verplaatsen richting de open zijde van de transformatorruimte in combinatie met het vergroten van de poolafstand uit onderwerp 1. Daarnaast zal de place-holder verbeterd worden zodat de fase-aarder slagafstand tussen doorvoerisolator en conservator, in het ontwerp goed wordt weergegeven. Ook wordt de maatvoering van de 150(110)/10 kV transformator aangepast zodat deze overeenkomt met de transformator specificaties.

Onderwerp 3.
Aan de hand van standaard TenneT componenten is een 150(110) kV place-holder ontworpen voor de 150(110) kV stelpuntsaarding van de transformator. Deze place-holder wordt in de transformator overzichtstekeningen weergegeven. Het ontwerp wordt afgestemd met TenneT zodat dit binnen TenneT breed gedragen wordt.

De 150(110) kV componenten vallen, zoals eerder omschreven, buiten de scope van Modulair Bouwen. Daarom wordt alleen een place-holder ontworpen en met TenneT afgestemd of deze maakbaar is. Staalconstructies en detail opstellingstekeningen worden hiervan niet gemaakt binnen Modulair Bouwen.

2.1.2 Voor- en nadelen t.o.v. huidige situatie

<table>
<thead>
<tr>
<th>Voordelen</th>
<th>Nadelen</th>
</tr>
</thead>
<tbody>
<tr>
<td>Financieel</td>
<td>N.v.t.</td>
</tr>
<tr>
<td>Engineering vergt minder afstemminguren met TenneT</td>
<td></td>
</tr>
<tr>
<td>Doorlooptijd project</td>
<td>N.v.t.</td>
</tr>
<tr>
<td>Risico op vertraging in project neemt sterk af.</td>
<td></td>
</tr>
<tr>
<td>Er is geen goedkeuring van TenneT nodig om af te wijken van hun standaard.</td>
<td></td>
</tr>
<tr>
<td>Betrouwbaarheid</td>
<td>N.v.t.</td>
</tr>
<tr>
<td>(O.a. EMC-invloed, reductie van SVBM en/of onderbrekingsfrequentie)</td>
<td></td>
</tr>
<tr>
<td>Juiste slagafstand 1500 mm faseaarder tussen conservator en blanke aanslukkabel transformator leidt tot minder kans op overslag.</td>
<td></td>
</tr>
<tr>
<td>Veiligheid</td>
<td>N.v.t.</td>
</tr>
<tr>
<td>Vluchtweg achter transformator beter geborgd in standaard ontwerp indien transformer niet dieper in de transformatorruimte geplaatst hoeft te worden om slagafstand te halen.</td>
<td></td>
</tr>
<tr>
<td>Beheer/onderhoudbaarheid</td>
<td>N.v.t.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### 2.1.3 Impact op standaarden

<table>
<thead>
<tr>
<th></th>
<th>Positief</th>
<th>Negatief</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>S-documenten</strong></td>
<td>• Verbetering standaard ontwerp transformatorruimten</td>
<td>• Aanpassing van SS091 nodig.</td>
</tr>
<tr>
<td><strong>Raamcontracten</strong></td>
<td>• N.v.t.</td>
<td>• N.v.t.</td>
</tr>
</tbody>
</table>
3. Uitgewerkt wijzigingsvoorstel voor voorgestelde oplossingsrichting

Oplossingsrichting 1

De standaard module transformatorruimte ontwerpen beschikken over voldoende ruimte om in dit geval de ppoolafstand van de 150(110) kV TenneT componenten te vergroten. De totale diepte van de transformatorruimte wordt bepaald door de afmetingen van de transformator place-holder, op basis van de transformator specificaties, en de benodigde minimale vluchtwegen rondom de transformator.

In onderstaande figuur 2 en 3 zijn de oplossingen van alle onderwerpen in één afbeelding weergegeven. De slagaafstanden in onderstaande afbeeldingen zijn gemeten vanaf de buitenzijde van spanning voerende delen naar aarde.

Onderwerp 1, de ppoolafstand vergroten naar 2400 mm
Onderwerp 2, de aansluiting van het juk naar de transformator in rechte lijn aan conservator zijde.
Onderwerp 3, een place-holder voor de sterpunt aarding opgenomen.

![Diagram](image)

Figuur 2 oplossingsrichting 1 150/20 kV 80 MVA transformator opstelling
Noot: door het corrigeren van de 150/110/10 kV place-holder, is de middelste fase ca. 500 mm verschoven t.o.v. de huidige standaard. In het nieuwe model het 10 kV juk 500 mm verschoven zodat het kabeljuk in lijn is met de middelste fase.

In onderstaande figuren 4 en 5 zijn de wijzigingen ten aanzien van de transformator place-holders weergegeven. De place-holders sluiten met deze wijzigingen aan bij de transformator specificaties en de typicals zoals opgenomen in de MB-standaarden.

Figuur 4 laat de 150/20 kV 80 MVA place-holder zien. Hierin zijn de conservator afmetingen aangepast naar afmetingen die dichter bij de transformatoren liggen die binnen het huidige contract worden geleverd. Het oude model is gebaseerd op oudere ontwerpen. De afmetingen van de conservator zijn niet gespecificeerd. De globale maten zijn uit het oude model zijn afgerond, tekst aanduidingen onderdelen toegevoegd en OLTC verplaatst zodat deze overeenkomt met de huidige standaarden.
In figuur 5 laat de place-holder van de 150(110)/10 kV transformator zien. De maatvoering van de transformator is verbeterd en komt in het nieuwe model overeen met de specificaties ten opzichte van waar de middelste fase zit. Als gevolg hiervan schuift het 10 kV juw op in het standaard ontwerp. Maatvoeringen zijn verbeterd aan de hand van de transformator typicals. De aansluitvlaggen zijn gewijzigd van verticale vlaggen naar een weergave van horizontale aansluiting. Tekst aanduiding onderdelen toegevoegd waar deze nog ontbraken.
4. Impact op de standaard

Dit wijzigingsvoorstel heeft impact op:

4.1 S-documenten in de AM Showroom

☐ SS080 Algemeen PVE EMC en aarding OS RS en SS
☐ SS081 Algemeen PVE Kabels OS RS en SS
☐ SS082 Algemeen PVE Primaire componenten OS RS en SS
☐ SS083 Algemeen PVE Secundaire Installaties OS RS en SS
☐ SS084 Modulebeschrijving Transformatornuimte
☐ SS085 Modulebeschrijving MS-schakelruimte
☐ SS086 Modulebeschrijving Secundaire ruimte
☐ SS087 Modulebeschrijving Accu-ruimte
☐ SS088 Modulebeschrijving LS-ruimte
☐ SS090 Modulebeschrijving Kabels
☐ SB010 Algemeen PVE Terreinen en Gebouwen OS RS en SS
☐ SB011 Modulebeschrijving Gebouw Terrein en GGD
☒ SS091 - Ontwerpkeuringen modulair bouwen station

Bij te werken onderdelen:

Ontwerpdocument
150[110]/10 kV: 01.02.D13, 01.02.D14, 02.00.D18, 02.00.D19, 02.02.D13, 02.03.D15, 02.04.D43, 02.04.D44, 02.05.D17, 02.05.D18, 02.05.D19, 06.01.D57, 06.01.D58, 08.01.D59, 08.01.D60, 08.01.D61, 08.03.D18, 08.03.D19, 08.04.D15, 08.04.D16, 09.02.D41, 09.02.D42, 09.03.D11, 09.03.D12.

150[110]/20 kV: 3D Revit. in PDF
afdrukken op: 02.00.D01, 02.00.D02, 02.01.D01, 02.01.D02, 02.01.D03, 02.02.D01, 02.03.D01, 02.04.D01, 02.04.D02, 02.05.D00, 02.05.D01, 02.05.D01, 02.05.D02, 08.01.D13, 08.01.D25, 08.01.D27, 08.03.D02, 08.04.D02, 09.02.D17, 09.03.D01, 09.03.D02.

150[110]/10 kV gesloten
transformatorruimte: 02.00.D20, 02.00.D21, 02.02.D14, 02.03.D16, 02.04.D49, 02.04.D50, 02.05.D20, 08.01.D62, 08.01.D68, 08.03.D20, 09.02.D43.

4.2 Overige beleids- of standaardisatiedocumenten in de AM Showroom

☐ Ja, namelijk ....
☒ Niet van toepassing
4.3 Beheer (o.a. BEI, documentatie)
☐ Ja, namelijk ....
☒ Niet van toepassing

4.4 Projecten
☐ In de PV fase, namelijk ....
☐ In de BO fase, namelijk ....
☐ In de DO fase, namelijk ....
☐ In de REALISATIE fase, namelijk ....
☐ Gerealiseerde projecten
☐ Anders, namelijk ....
☒ Geen van bovenstaande

4.5 Trainingen
☐ Ja, namelijk ....
☒ Niet van toepassing

4.6 Eventuele opmerkingen
☐ Ja, namelijk ....
☒ Geen opmerkingen
5. Financiële impact

Hieronder volgt een grove, maar wel onderbouwde indicatie, van de financiële impact van het wijzigingsvoorstel.

### 5.1 Eenmalige kosten

<table>
<thead>
<tr>
<th>Nr.</th>
<th>Onderdeel</th>
<th>Kosten ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Bijwerken van de standaarden: tekeningen 2D: 24 x 0,5 uur = 12 uur á €90,-</td>
<td>€ 1.080,-</td>
</tr>
<tr>
<td></td>
<td>Bijwerken van de standaarden: tekeningen 3D: 1x 4 uur = 4 uur á €90,- + PDF 4 uur á €90,-</td>
<td>€ 720,-</td>
</tr>
<tr>
<td></td>
<td>Bijwerken van standaarden: tekeningen gesloten ruimte: 11 x 0,5 uur = 5,5 uur á €90,-</td>
<td>€ 495,-</td>
</tr>
<tr>
<td></td>
<td>Ontwerp sterpoint 8 uur á €90,-</td>
<td>€ 720,-</td>
</tr>
<tr>
<td></td>
<td>Review door kernteam 8 uur á €90,-</td>
<td>€ 720,-</td>
</tr>
<tr>
<td></td>
<td>Totaal</td>
<td>€ 3.735,-</td>
</tr>
</tbody>
</table>

### 5.2 Wederkerende kosten per project

<table>
<thead>
<tr>
<th>Nr.</th>
<th>Onderdeel</th>
<th>Kosten ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Effect op engineeringuren Afstemming TenneT i.v.m. afwijking standaard -8 uur á €90,-</td>
<td>€ 720,-</td>
</tr>
<tr>
<td>2</td>
<td>Effect op realisatie uren</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Effect op test-/inbedrijfstellingsuren</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Effect op beheer/ storing zoeken</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Materialen/ diensten</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Effect op materialkosten (installaties, transformatoren, bouwkundig, overige materialen)</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Effect op ruimtebeslag</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Totaal</td>
<td>€ 720,-</td>
</tr>
</tbody>
</table>
## Appendix C: SHACL constraint types overview

<table>
<thead>
<tr>
<th>Constraint Type</th>
<th>Name</th>
<th>Description</th>
<th>Form</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Value type constraints</strong></td>
<td>sh:class</td>
<td>The condition specified by sh:class is that each value node is a SHACL instance of a given type.</td>
<td>x = of class y</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sh:datatype</td>
<td>Specifies a condition to be satisfied with regards to the datatype of each value node.</td>
<td>x = of datatype y, string, integer, etc.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sh:nodeKind</td>
<td>Specifies a condition to be satisfied by the RDF node kind of each value node.</td>
<td>x = of nodekind IRI, BlankNode, Literal, BlankNodeOrIRI, BlankNodeOrLiteral, IRIOrLiteral</td>
<td></td>
</tr>
<tr>
<td><strong>Cardinality constraints</strong></td>
<td>sh:minCount</td>
<td>Specifies the minimum number of value nodes that satisfy the condition.</td>
<td>x ≥ y amount of nodes z</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sh:maxCount</td>
<td>Specifies the maximum number of value nodes that satisfy the condition.</td>
<td>x ≤ y amount of nodes z</td>
<td></td>
</tr>
<tr>
<td><strong>Value range constraints</strong></td>
<td>sh:minExclusive</td>
<td>Specify value range conditions to be satisfied by value nodes that are comparable via operators such as &lt;, &lt;=, &gt;, and &gt;=.</td>
<td>value x &gt; y, y = integer</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sh:maxExclusive</td>
<td>Specify value range conditions to be satisfied by value nodes that are comparable via operators such as &lt;, &lt;=, &gt;, and &gt;=.</td>
<td>value x &lt; y, y = integer</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sh:minInclusive</td>
<td>Specify value range conditions to be satisfied by value nodes that are comparable via operators such as &lt;, &lt;=, &gt;, and &gt;=.</td>
<td>value x ≥ y, y = integer</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sh:maxInclusive</td>
<td>Specify value range conditions to be satisfied by value nodes that are comparable via operators such as &lt;, &lt;=, &gt;, and &gt;=.</td>
<td>value x ≤ y, y = integer</td>
<td></td>
</tr>
<tr>
<td><strong>String-based constraints</strong></td>
<td>sh:minLength</td>
<td>Specifies the minimum string length of each value node that satisfies the condition.</td>
<td>length string x &gt; y, y = integer</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sh:maxLength</td>
<td>Specifies the maximum string length of each value node that satisfies the condition.</td>
<td>length string x &lt; y, y = integer</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sh:pattern</td>
<td>Specifies a regular expression that each value node matches to satisfy the condition.</td>
<td>string follows pattern, y = Regex</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sh:languageIn</td>
<td>The condition specified by sh:languageIn is that the allowed language tags for each value node are limited by a given list of language tags.</td>
<td>value x = language y, BCP47</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sh:uniqueLang</td>
<td>The property sh:uniqueLang can be set to true to specify that no pair of value nodes may use the same language tag.</td>
<td>value x = unique language for y, BCP47</td>
<td></td>
</tr>
<tr>
<td><strong>Property pair constraints</strong></td>
<td>sh:equals</td>
<td>Specifies the condition that the set of all value nodes is equal to the set of objects of the triples that have the focus node as subject and the value of sh:equals as predicate.</td>
<td>x = y and z = y</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sh:disjoint</td>
<td>Specifies the condition that the set of value nodes is disjoint with the set of objects of the triples that have the focus node as subject and the value of sh:disjoint as predicate.</td>
<td>if x = y then z ≠ y</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sh:lessThan</td>
<td>Specifies the condition that each value node is smaller than all the objects of the triples that have the focus node as subject and the value of sh:lessThan as predicate.</td>
<td>value x &lt; value y, x and y are same datatype</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sh:lessThanOrEquals</td>
<td>Specifies the condition that each value node is smaller than or equal to all the objects of the triples that have the focus node as subject and the value of sh:lessThanOrEquals as predicate.</td>
<td>value x ≤ value y, x and y are same datatype</td>
<td></td>
</tr>
<tr>
<td>Logical constraints</td>
<td>Constraint</td>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------------------</td>
<td>------------</td>
<td>-------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sh:not</td>
<td>Specifies the condition that each value node cannot conform to a given shape.</td>
<td>x should not conform to shape y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sh:and</td>
<td>Specifies the condition that each value node conforms to all provided shapes.</td>
<td>x should conform to shape y and z</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sh:or</td>
<td>Specifies the condition that each value node conforms to at least one of the provided shapes.</td>
<td>x should conform to shape y or z</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sh:xone</td>
<td>Specifies the condition that each value node conforms to exactly one of the provided shapes.</td>
<td>x conforms to one of the shapes x or y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Shape-based constraints</td>
<td>Constraint</td>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sh:node</td>
<td>Specifies the condition that each value node conforms to the given node shape.</td>
<td>x conforms to node shape y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sh:property</td>
<td>Can be used to specify that each value node has a given property shape.</td>
<td>x conforms to property shape y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sh:qualifiedValueShape</td>
<td>Specifies the condition that a specified number of value nodes conforms to the given shape</td>
<td>an amount of x should conform to shape y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sh:qualifiedMinCount</td>
<td>Each sh:qualifiedValueShape can have: one value for sh:qualifiedMinCount, one value for sh:qualifiedMaxCount or, one value for each</td>
<td>a minimum amount of x conforms to shape y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sh:qualifiedMaxCount</td>
<td>Each sh:qualifiedValueShape can have: one value for sh:qualifiedMinCount, one value for sh:qualifiedMaxCount or, one value for each</td>
<td>a maximum amount of x conforms to shape y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Other constraints</td>
<td>Constraint</td>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sh:closed</td>
<td>Can be used to specify the condition that each value node has values only for those properties that have been explicitly enumerated via the property shapes specified for the shape via sh:property.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sh:ignoredProperties</td>
<td>Specifies the condition that at least one value node is equal to the given RDF term.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sh:in</td>
<td>Specifies the condition that each value node is a member of a provided SHACL list.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Appendix D: Prototype data

This file contains the data that is used for the final prototype test of the rule checking system for the thesis of Ellen van den Berkel. It describes a part of the converted original IFI dataset, including the objects and object relations that were mapped to the desired vocabulary as used in the NIF/TransformerOIL. Beware of the fact that this is dummydata, which is manually made to indicate how it should look like when this process is fully automated. The implementation of this automation is not within the scope of this project.

```
DataTransformerSpace_001 a owl:transformerspace;
owl:isConnectedTo DataLightingStriker_001;
owl:isConnectedTo DataLightingStriker_002;
owl:isConnectedTo DataLightingStriker_003;
owl:isConnectedTo DataLightingStriker_004;
owl:isConnectedTo DataBuildingRelatedInstallation_001.

DataBuildingRelatedInstallation_001 a owl:BuildingRelatedInstallation;
owl:hasPart DataEmergencyLighting_001;
owl:hasPart DataEmergencyLighting_002;
owl:hasPart DataEmergencyLighting_003;
owl:hasPart DataEmergencyLighting_004;
owl:hasPart DataCableCarrier_001;
owl:hasPart DataCableCarrier_002;
owl:hasPart DataBuildingRelatedInstallation_002.

DataLightingStriker_001 a owl:LightingStriker;
owl:isConnectedTo owl:TransformerSpace_001.

DataLightingStriker_002 a owl:LightingStriker;
owl:isConnectedTo owl:TransformerSpace_001.

DataLightingStriker_003 a owl:LightingStriker;
owl:isConnectedTo owl:TransformerSpace_001.

DataLightingStriker_004 a owl:LightingStriker;
owl:isConnectedTo owl:TransformerSpace_001.

DataEmergencyLighting_001 a owl:EmergencyLighting;
owl:isPartOf DataBuildingRelatedInstallation_001.

DataEmergencyLighting_002 a owl:EmergencyLighting;
owl:isPartOf DataBuildingRelatedInstallation_001.

DataLighting_001 a owl:Lighting;
owl:isPartOf DataBuildingRelatedInstallation_001.
```

Data:Lighting_002 a Qtkl:Lighting ;
  Qtkl:IsPartOf Data:BuildingRelatedInstallation_001 .

Data:Lighting_003 a Qtkl:Lighting ;
  Qtkl:IsPartOf Data:BuildingRelatedInstallation_001 .

Data:Lighting_004 a Qtkl:Lighting ;
  Qtkl:IsPartOf Data:BuildingRelatedInstallation_001 .

Data:CableCarrier_001 a Qtkl:CableCarrier ;
  Qtkl:IsPartOf Data:BuildingRelatedInstallation_001 .

Data:CableCarrier_002 a Qtkl:CableCarrier ;
  Qtkl:IsPartOf Data:BuildingRelatedInstallation_001 .

Data:CableCarrier_003 a Qtkl:CableCarrier ;
  Qtkl:IsPartOf Data:BuildingRelatedInstallation_001 .

Data:ServicePowerOutlet_001 a Qtkl:ServicePowerOutlet ;
  Qtkl:IsPartOf Data:BuildingRelatedInstallation_001 .

Data:ServicePowerOutlet_002 a Qtkl:ServicePowerOutlet ;
  Qtkl:IsPartOf Data:BuildingRelatedInstallation_001 .

Qtkl:EmergencyLighting a rdfs:Class ;
  rdfs:subClassOf Qtkl:Lighting .

Qtkl:Lighting a rdfs:Class .

Qtkl:ServicePowerOutlet a rdfs:Class .

Qtkl:CableCarrier a rdfs:Class .
Data:TennetGroundingTransformer_001 a Qotl:TennetGroundingTransformer ;
Qotl:IsConnectedTo Data:SurgeProtector_001 ;
Qotl:IsConnectedTo Data:SurgeProtector_002 ;
Qotl:IsConnectedTo Data:SurgeProtector_003 ;
Qotl:IsConnectedTo Data:Pothead_001 ;
Qotl:IsConnectedTo Data:Pothead_002 ;
Qotl:IsConnectedTo Data:Pothead_003 .

Data:SurgeProtector_001 a Qotl:SurgeProtector ;
Qotl:IsConnectedTo Data:TennetGroundingTransformer_001 ;
Qotl:HasDistanceToSP_002 [ qudt:unit qudt:MilliM ;
    qudt:value "2250.0"^^xsd:float] .

Data:SurgeProtector_002 a Qotl:SurgeProtector ;
Qotl:IsConnectedTo Data:TennetGroundingTransformer_001 ;
Qotl:HasDistanceToSP_001 [ qudt:unit qudt:MilliM ;
    qudt:value "2250.0"^^xsd:float] .

Data:SurgeProtector_003 a Qotl:SurgeProtector ;
Qotl:IsConnectedTo Data:TennetGroundingTransformer_001 ;
Qotl:HasDistanceToSP_003 [ qudt:unit qudt:MilliM ;
    qudt:value "2250.0"^^xsd:float] .

Data:Pothead_001 a Qotl:Pothead ;
Qotl:IsConnectedTo Data:TennetGroundingTransformer_001 .

Data:Pothead_002 a Qotl:Pothead ;
Qotl:IsConnectedTo Data:TennetGroundingTransformer_001 .

Data:Pothead_003 a Qotl:Pothead ;
Qotl:IsConnectedTo Data:TennetGroundingTransformer_001 .

Qotl:TennetGroundingTransformer a rdfs:Class .
Qotl:SurgeProtector a rdfs:Class .
Qotl:Pothead a rdfs:Class .
Qot1:BuildingRelatedInstallation a rdfs:Class.
Qot1:TransformerSpace a rdfs:Class.
Qot1:LightningStriken a rdfs:Class.
 Data:WiringHarness_001 a Qot1:WiringHarness;
   Qot1:HasPart Data:CableCoil_001;
   Qot1:HasPart Data:CableCoil_002;
   Qot1:HasPart Data:CableCoil_003;
   Qot1:HasPart Data:CCV_001;
   Qot1:HasPart Data:CCV_002;
   Qot1:HasPart Data:CCF_001;
   Qot1:HasPart Data:CCF_002.
 Data:CableCoil_001 a Qot1:CableCoil;
   Qot1:IsPartOf Data:WiringHarness_001.
 Data:CableCoil_002 a Qot1:CableCoil;
   Qot1:IsPartOf Data:WiringHarness_001.
 Data:CableCoil_003 a Qot1:CableCoil;
   Qot1:IsPartOf Data:WiringHarness_001.
 Data:CCV_001 a Qot1:CopperConnectionVane;
   Qot1:IsPartOf Data:WiringHarness_001.
 Data:CCV_002 a Qot1:CopperConnectionVane;
   Qot1:IsPartOf Data:WiringHarness_001.
 Data:CCF_001 a Qot1:CopperConnectionFork;
   Qot1:IsPartOf Data:WiringHarness_001.
 Data:CCF_002 a Qot1:CopperConnectionFork;
   Qot1:IsPartOf Data:WiringHarness_001.
Qot1:WiringHarness a rdfs:Class.
Qot1:CableCoil a rdfs:Class.
Qot1:CopperConnectionVane a rdfs:Class.
Qot1:CopperConnectionFork a rdfs:Class.
 Data:EarthingBall_001 a Qot1:EarthingBall;
   Qot1:HasDiameter [ qudt:unit qudt:MilliM;
      qudt:value "25.0"^^xsd:float].
Qot1:EarthingBall a rdfs:Class.