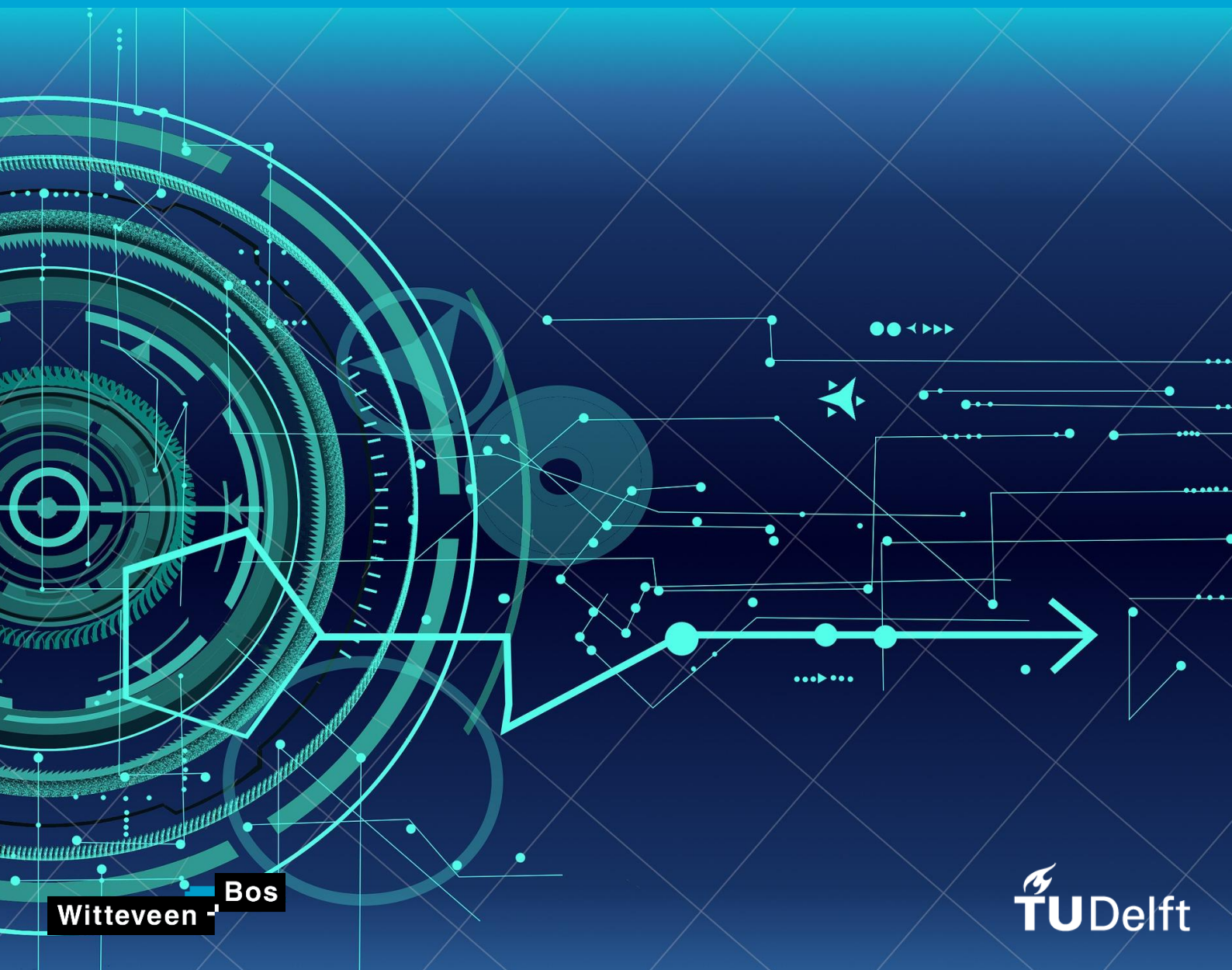


# Exploring the Process of Semi-Automated Requirement Verification within a BIM Model

Master Thesis Report

B. Overbeek - 5866464





# Exploring the Process of Semi-Automated Requirement Verification within a BIM Model

Master Thesis Report

by

B. (Brian) Overbeek

to obtain the degree of Master of Science in Construction Management and Engineering  
at the Delft University of Technology

June 26, 2025

**Dr. ir. G.A. (Sander) van Nederveen**

Faculty of Civil Engineering and Geosciences  
Materials, Mechanics, Management & Design (3MD)  
*Chair supervisor*

**Dr. ir. T. (Tong) Wang**

Faculty of Architecture and the Built Environment  
Design & Construction Management (D&CM)  
*First supervisor*

**Dr. ir. R. (Ranjith) Kuttanharappel Soman**

Faculty of Civil Engineering and Geosciences  
Materials, Mechanics, Management & Design (3MD)  
*Second supervisor*

**ing. D. (Dies) Flikweert**

Witteveen+Bos  
*Company supervisor*



# Preface

This thesis named “Exploring the Process of Semi-Automated Requirement Verification within a BIM Model” was written as part of the Master's program in Construction Management and Engineering at Delft University of Technology. It represents the final step in completing my studies and reflects both my academic journey and professional interests. The graduation topic aligned well with my academic interests in BIM and digital construction, as well as with the broader industry shift towards more intelligent and data-driven processes. The research was carried out in collaboration with Witteveen+Bos from early February 2025 until the end of June 2025.

I would like to thank my TU Delft supervisors Sander van Nederveen, Tong Wang, and Ranjith Kuttanharappel Soman for their valuable feedback and guidance during the research process. I also want to thank my company supervisor, Dies Flikweert, for his efforts and guidance during my graduation period. In addition, I want to thank the colleagues at Witteveen+Bos who were involved in my graduation project and gave me the opportunity to carry out my research within the company.

Also, I want to thank my fellow students. It was helpful to be able to exchange ideas and discuss challenges together. A special thanks to Emilio Murillo Sierra for his valuable insight on the IFC file format. Furthermore, I am thankful to my parents for their support, patience and belief in me throughout this journey. Their encouragement played an important role in helping me reach this point. I would also like to thank my family and friends for their support along the way. Finally, I want to thank my girlfriend for her continuous support and belief in me during my study time. Her presence made a big difference. I would also like to express my gratitude to her parents for their support along the way, especially her father whose insights proved particularly valuable along the way.

Looking back, I feel grateful for my time at TU Delft. Having lived in Delft my whole life, I have always looked up to the university. What once felt like a distant goal gradually became a reality. After completing my Bachelor's degree in Civil Engineering at a University of Applied Sciences (HBO) and the TU Delft pre-master program, reaching this point has been a rewarding journey in both academic and personal ways. I am grateful for the experiences I gained during my time as a student and look forward to the next step in my career and personal life.

*Brian Overbeek  
Delft, June 2025*

# Summary

Digital technologies have increasingly shaped the global construction industry with the fourth industrial revolution (Industry 4.0) emerging as a widely discussed concept. In this context, Construction 4.0 represents the sector's response to Industry 4.0 principles, particularly through innovative technologies such as Building Information Modeling (BIM). Accurately defining and verifying requirements is essential for project quality and safety, but complex projects often involve thousands of requirements that must be manually checked. As projects grow more complex, there is a clear need for automated verification. This research explores the process of semi-automated requirement verification. The main research question is: *"How can a (semi-) automated process be developed to validate and verify requirements within a 3D model?"*. This is supported by four sub-questions exploring current limitations, existing tools and methods, and key development stages of automated requirement verification.

A literature review highlights that project requirements are typically derived from the client's vision and form the basis of contractual obligations. Successfully extracting these requirements is important for defining the project scope and achieving client satisfaction. The literature study also explores various international and national BIM standards, which are adopted to different extents across countries worldwide. Several verification tools were evaluated such as Solibri, Navisworks, BIM Assure, SMARTreview and Verifi3D. From the comparison, Solibri and Navisworks emerged as the tools that provide the most comprehensive functionalities. In terms of requirement structuring methods, the RASE language was identified as the most promising approach due to its clarity and expressiveness.

The research used the Oosterweel project as a case study that is being developed in Antwerp (Belgium) with the aim of completing the city's highway ring road. The verification process of requirements is broken down in five key steps. The first step focused on the classification of requirements into geometric and non-geometric requirements using a machine learning model in Python. The second step involved manual rule interpretation using the RASE mark-up language. The third step focused on preparing the model data by filtering the model and performing semantic enrichment. Step four looked at the actual execution of the verifications where different types of requirements were distinguished in classes based on their computational complexity. The last step focused on reporting of the verification results in a Common Data Environment.

Implementation challenges during this process included ambiguous requirement texts, inconsistencies in model data and a fragmented workflow. The insights from the literature review, findings and interviews were combined into guidelines which are divided into three levels: strategic, operational and BIM level. The strategic level focuses on guidelines from the organizational point of view. The operational level looks at how strategic intentions can be translated into practical actions. The BIM level addresses guidelines that relate to the practical use of BIM. The guidelines are validated through interviews with professionals.

The results of this research show that a semi-automated approach to requirement verification within the BIM environment is feasible and significantly more efficient than current manual methods. Research limitations include the limited scope of one case study and the internal validation of the guidelines. Based on the research, multiple recommendations were given for future research and practice. Practical recommendations include using a structured approach and the use of open standards. Recommendations for future research include the further investigation of AI beyond the current machine learning model for requirement verification and the investigation of the impact of the automated verification process.

# Table of Contents

1. Introduction.....	1
1.1 Context.....	1
1.2 Problem statement .....	1
1.3 Research framework .....	2
1.3.1 Research objective .....	2
1.3.2 Research gap.....	3
1.3.3 Research scope .....	3
1.3.4 Research questions.....	3
1.3.5 Research methodology .....	4
1.3.6 Research relevance .....	4
1.3.7 Research outline .....	5
2. Literature study.....	6
2.1 Requirements in construction projects.....	6
2.2 Building Information Modeling.....	7
2.2.1 The evolution of BIM .....	7
2.2.2 Standards and regulations in BIM .....	8
2.2.2.1 International standards .....	8
2.2.2.2 National standards .....	12
2.2.2.3 BIM Basis ILS .....	13
2.2.2.4 The adoption of BIM standards.....	14
2.2.3 BIM maturity .....	15
2.2.4 Future trends and emerging technologies.....	17
2.2.4.1 Artificial intelligence and subfields .....	17
2.2.4.2 Digital twins.....	18
2.2.4.3 Virtual reality and augmented reality .....	19
2.2.4.4 Internet of things and cloud computing.....	19
2.2.4.5 Future perspective .....	20
2.3 Automated model verification tools.....	20
2.3.1 Evolution of automated verification tools .....	21
2.3.2 Current automated compliance checking software .....	22
2.3.3 Comparison of the different tools .....	24

2.3.4 Challenges with automated requirement checking .....	26
2.3.5 Findings.....	29
2.4 Transforming requirements into computer code .....	29
2.4.1 Natural Language Processing .....	30
2.4.2 RASE language.....	33
2.4.3 BIMRL .....	34
2.4.4 BERA language .....	34
2.4.5 Comparison of the different methods .....	34
2.4.5.1 Standard for rule definition .....	35
2.4.5.2 User-friendliness and accessibility .....	35
2.4.5.3 Language expressiveness.....	36
2.4.5.4 Efficiency and scalability .....	37
2.4.5.5 Level of maturity.....	37
2.4.5.6 Openness and interoperability .....	38
2.4.6 Findings .....	38
3. Development phase .....	40
3.1 Case study .....	40
3.2 Implementation.....	41
3.2.1 Classification of requirements.....	41
3.2.2 Manual rule interpretation .....	45
3.2.3 Preparation of the building model data .....	46
3.2.4 Execution of the verifications .....	47
3.2.4.1 Verifications of Class 1 .....	48
3.2.4.2 Verifications of Class 2 .....	50
3.2.4.3 Verifications of Class 3 .....	54
3.2.4.4 Verification of Class 4.....	57
3.2.5 Reporting of the results.....	58
3.3 Implementation challenges .....	61
3.3.1 Ambiguity in requirement texts .....	61
3.3.2 Hybrid requirements.....	61
3.3.3 Inconsistent model data and parameters .....	61
3.3.4 Classification of requirements .....	61
3.3.5 Linking model elements to requirements.....	62
3.3.6 Fragmented workflow .....	62



3.4 Workflow .....	62
3.5 Efficiency metrics .....	63
4. Guidelines .....	66
4.1 Creating guidelines .....	66
4.1.1 Strategic level .....	66
4.1.2 Operational level .....	69
4.1.3 BIM level .....	71
4.2 Validation through interview .....	73
5. Discussion .....	76
5.1 Validity of the research .....	76
5.2 Result interpretation .....	76
5.3 Research limitations .....	77
5.4 Research implications .....	77
6. Conclusion .....	78
6.1 Sub-research questions .....	78
6.2 Main research question .....	79
7. Recommendations .....	81
7.1 Recommendations for practice .....	81
7.2 Recommendations for future research .....	82
References .....	84
Appendices .....	101
Appendix A: Interview protocol and insights .....	102
Appendix B: Requirement classification script .....	107
Appendix C: Verification of pipe diameter (Class 1) .....	116
Appendix D: Verification of diaphragm walls (Class 1) .....	120
Appendix E: Verification of element depth (Class 2) .....	122
Appendix F: Verification of tunnel equipment (Class 2) .....	127
Appendix G: Verification of tunnel compartments (Class 3) .....	129
Appendix H: Verification of escape ducts (Class 4) .....	135
Appendix I: Efficiency metrics .....	138
Appendix J: openBIM approach .....	140

# List of Abbreviations

The table below provides definitions for the abbreviations used throughout this report. The corresponding page number indicates where each abbreviation first appears.

Abbreviation	Meaning	Page
2D	Two dimensional	12
3D	Three dimensional	1
4D	Four dimensional	17
5D	Five dimensional	17
AEC	Architecture, Engineering, and Construction	33
AI	Artificial Intelligence	8
AIoT	Artificial Intelligence Internet of Things	19
AISC	American Institute of Steel Construction	21
API	Application Programming Interface	9
APR	Automated Plan Review	23
AR	Augmented Reality	19
ACC	Autodesk Construction Cloud	58
ACCC	Automated Code Compliance Checking	9
BCF	BIM Collaboration Format	8
BDA	Building Design Advisor	7
BDR	Bouwdigitaliseringsraad	15
BDS	Building Description System	7
BERA	Building Environment Rule and Analysis language	34
BIM	Building Information Modeling	1
BIM Basis ILS	Basis Informatieleveringsspecificatie	13
BIMRL	BIM Rule Language	34
BOM	BERA Object Model	34
bSDD	buildingSMART Data Dictionary	70
bSI	buildingSMART International	73
CAD	Computer-Aided Design	7
CDE	Common Data Environments	10
CMO	Coordination Model	46
CORENET	Construction and Real Estate NETwork	21
CSV	Comma Separated Values	24
DEME	Dredging, Environmental and Marine Engineering	40
DL	Deep Learning	18
DO	Preliminary Design	104
DT	Digital Twin	18
DSL	Domain Specific Language	35
EU	European Union	41
FN	False Negative	43
FP	False Positive	43
GPT	Generative Pre-training Transformer	32
GLIDE	Graphical Language for Interactive Design	7
GPS	Global Positioning Systems	19
ICC	International Code Council	22
IBC	International Building Code	24
IDS	Information Delivery Specification	9
IDM	Information Delivery Manual	10
IFD	International Framework for Dictionaries	12
IFC	Industry Foundation Classes	7

<b>Abbreviation</b>	<b>Meaning</b>	<b>Page</b>
IE	Information Extraction	30
IoT	Internet of Things	19
IPM	Integral Project Management	40
IR	Information Retrieval	30
ISO	International Organization for Standardization	11
Lantis	Leefbaar Antwerpen door Innovatie en Samenwerking	40
MAD	Mean Absolute Deviation	139
ML	Machine Learning	18
MQ	Main Question	3
NEN	Royal Netherlands Standardization Institute	13
NEC	New Engineering Contract	41
NLCS	Nederlandse CAD Standaard	12
NLG	Natural Language Generation	32
NLI	Natural Language Interpretation	32
NLP	Natural Language Processing	30
NLU	Natural Language Understanding	32
NT	Non-Temporary	51
OBS	Object Breakdown Structure	73
OTL	Object Type Library	73
PDF	Portable Document Format	9
PhD	Doctor of Philosophy	37
PTC	Parametric Technology Corporation	7
PVR	Clearance profile (Profiel van Vrije Ruimte)	45
RASE	Requirement, Applicability, Specification, and Exception	30
RMS	Requirement Management System	7
ROCO	Respect, Oplossingsgericht, Connectie, Onderscheidend	40
SICAD	Standard Interface for Computer Aided Design	21
SMART	Specific, Measurable, Assignable, Realistic and Time-bound	28
SMC	Solibri Model Checker	22
SQL	Structured Query Language	37
SQ	Sub Question	3
TF-IDF	Term Frequency-Inverse Document Frequency	107
TN	True Negative	43
TP	True Positive	43
TT	Temporary Condition	51
UAV-GC	Uniform Administrative Conditions for Integrated Contracts	104
UK	United Kingdom	15
UO	Detailed Design	105
V&V	Verification and Validation	6
VR	Virtual Reality	8
XHTML	EXtensible HyperText Markup Language	38
XML	Extensible Markup Language	9
XLS	Microsoft Excel Spreadsheet	24

# 1. Introduction

## 1.1 Context

Over the past decade, the use of digital technologies in the construction industry has significantly increased across various sectors worldwide with the fourth industrial revolution (Industry 4.0) emerging as a widely discussed concept (Elnadi & Abdallah, 2024). At the beginning of the fourth industrial revolution, the rise of digitalization and innovative technologies began transforming the construction industry. Construction 4.0 represents the construction sector's adaptation to the Industry 4.0 principles like Building Information Modeling (BIM), Artificial Intelligence and automation (Casini, 2021). Adopting such advanced technologies can shorten construction timelines, reduce life-cycle costs, improve quality, safety and sustainability (Casini, 2021). Moreover, according to the World Economic Forum (2016), full-scale digitalization in non-residential construction could generate annual global cost savings of \$0.7-1.2 trillion in engineering and construction within 10 years.

Similarly, in the construction and infrastructure sector, accurately defining and verifying requirements is an indispensable step to ensure the quality and safety of projects. These requirements form the basis of project standards and specifications and play a central role in all phases of a project, from design and planning to final delivery. More specifically, requirements ensure that essential aspects such as safety, functionality, sustainability and legal requirements are met. Without a well-structured and effective requirements verification system, a project runs the risk of quality losses, delays, increased costs and even safety risks (Teyseyre, 2002).

In complex projects, requirements often include hundreds, if not thousands, of specifications all of which need to be accurately verified and tracked. Traditionally, this verification is performed manually by comparing textual requirements with technical drawings. However, this approach is very time consuming and prone to human error, especially given the size and complexity of modern construction projects. Errors in the verification process can only come to light in later project phases when corrections are not only costly but can also lead to disruptions in planning and execution (Ghannad et al., 2019).

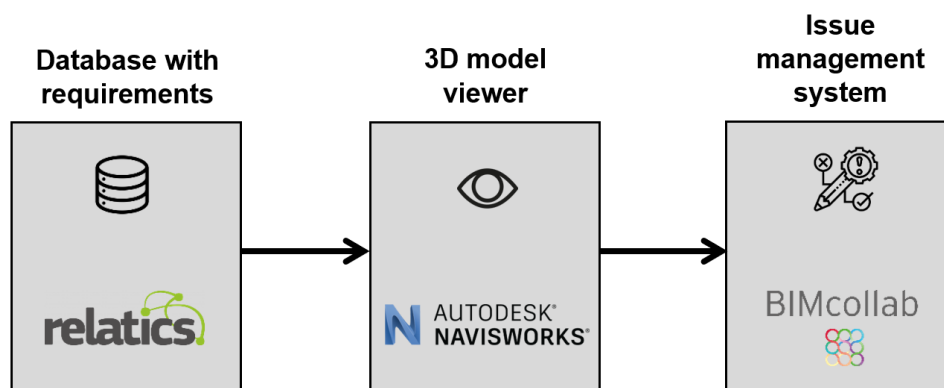
With the emergence of new digital technologies there is a growing need for innovative methods to make this process more efficient and accurate. By linking requirements directly to objects with a 3D modelling environment and using automated processes, verification and compliance with requirements can be optimised. Instead of manual checks an automated process provides a systematic approach that detects conflicts faster. This allows project teams to identify and resolve issues early leading to greater efficiency and accuracy in requirements compliance. By investing in streamlining the requirement verification process the construction industry cannot only improve the consistency and reliability of projects, but also contribute to cost savings, reduce risks and increase overall project quality. Ultimately, this progress will not only benefit individual projects but take the entire industry to a higher level.

## 1.2 Problem statement

In the construction industry, effective and efficient verification of project requirements is essential to ensure quality and compliance with standards in a 3D design. Many organizations, including Witteveen+Bos, work with different systems to verify requirements through structural calculations, visual inspections and links to specifications from Relatics. However, the alignment between the BIM model and the requirement specifications is often not optimal, leading to

inconsistencies such as the use of different names for the same element. This lack of consistency makes verification error-prone, time consuming and inefficient.

There is a clear gap between the literature, which frequently promotes the integration of systems and automated processes, and practice, where the use of multiple systems and software tools such as Relatics, Autodesk Navisworks and BIMcollab leads to fragmentation and inefficiencies (see **Figure 1**). Although technology in the sector is rapidly evolving and new tools such as Autodesk Construction Cloud provide clash detection, integration between the systems remains limited. Working with non-integrated systems complicates collaboration and hinders automation which results in a lack of transparency and an increased risk of human error. These practical differences form the basis of the gap between literature and practice.



**Figure 1.** Current workflow of Witteveen+Bos for the manual requirement verification process.

## 1.3 Research framework

This section discusses the research framework of this study which includes the objective, research gap, scope and research questions. These will be central throughout the duration of this research.

### 1.3.1 Research objective

With the increasing complexity of construction projects and the growing volume and variety of data, the need for automated processes in the construction sector is becoming increasingly clear. In the traditional approach, requirements are manually checked and compared with BIM models, resulting in inefficiencies, errors and delays. This research aims to explore the process of semi-automated requirement verification within the BIM environment.

The primary goal of this research is to develop a (semi-)automated verification process that can effectively integrate requirements recorded in Relatics with 3D models created in BIM software. This process will utilize advanced technologies like data integration, Artificial Intelligence and automated checks to reduce the risk of human errors and improve efficiency of the verification process. The objective is to create a systematic approach that accelerates verification and identifies conflicts more quickly enabling project teams to respond faster and more effectively.

Additionally, the research focuses on identifying the standards and measurable criteria necessary for accurate and efficient requirement verification. It will also analyse obstacles associated with integrating different systems (such as Relatics and Autodesk software) and investigate how these barriers can be overcome to achieve an automated workflow.

### 1.3.2 Research gap

The identified research gap lies in the integration of systems used to verify project requirements within the BIM environment. While the literature often advocates for the integration of systems and automation to improve the efficiency and accuracy of requirement verification, the reality in the construction industry is fragmented. Companies like Witteveen+Bos rely on multiple non-integrated tools to verify requirements. This disjointed approach creates inefficiencies, increases the risk of errors and complicates collaboration among stakeholders. The lack of connection between the different software tools leads to discrepancies in terminology and misalignments between the 3D model and requirement specifications. This also results in time consuming and error prone verification processes which hinder the automation of requirement verification.

The gap between the theoretical benefits of automated systems as highlighted in academic research and the practical challenges faced by the industry show the need for solutions that bridge this gap. This research focuses on addressing this gap by providing insight into how systems can be better integrated within the BIM environment. By looking into the possibilities to develop a (semi-)automated process this research aims to streamline the verification of requirements and improve the efficiency. The research also seeks to overcome the current limitations to enhance the overall productivity of construction projects.

### 1.3.3 Research scope

The scope of this research is limited to developing a ‘proof of concept’ rather than a fully functional solution. A single case study is used to test certain aspects in practice and to identify the key areas within automated code compliance that offer the greatest potential benefits. Based on these findings, guidelines will be developed for implementation within Witteveen+Bos which will be validated through interviews with the company's employees, as they are the primary stakeholders. The interview protocol is presented in Appendix A1.

### 1.3.4 Research questions

To guide the research towards achieving its objective it is essential to answer the main research question. In addition to the main research question there are four sub-questions formulated. These sub-questions have been designed to break down this central question into more focused and manageable components. Answering these sub-questions systematically will enable the research to meet its objective. The main question that is formulated for this research is:

#### **Main question (MQ)**

*“How can a (semi-)automated process be developed to validate and verify requirements within a 3D model?”*

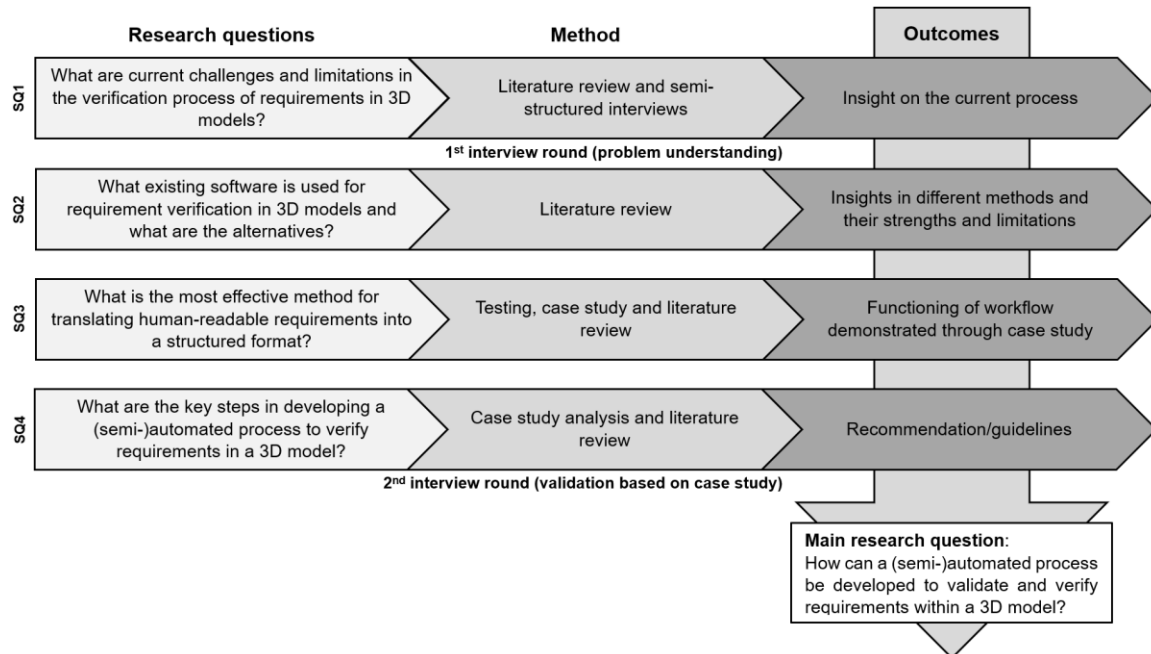
#### **Sub-questions (SQ)**

The sub-questions that help in guiding this research are:

1. *“What are current challenges and limitations in the verification process of requirements in 3D models?”*
2. *“What existing software is used for requirement verification in 3D models and what are the alternatives?”*
3. *“What is the most effective method for translating human-readable requirements into a structured format?”*
4. *“What are the key steps in developing a (semi-)automated process to verify requirements in a 3D model?”*

### 1.3.5 Research methodology

This subsection outlines the research methods used to achieve the objectives of this research. The research methods are displayed together with the research questions and the outcomes in **Figure 2**. The research method ensures a structured and systematic investigation and provides a solid foundation for the analysis of the findings.



**Figure 2.** Overview of research methodology including research questions and outcomes.

### 1.3.6 Research relevance

This subsection discusses the relevance of the research in both the scientific and practical field.

#### Scientific relevance

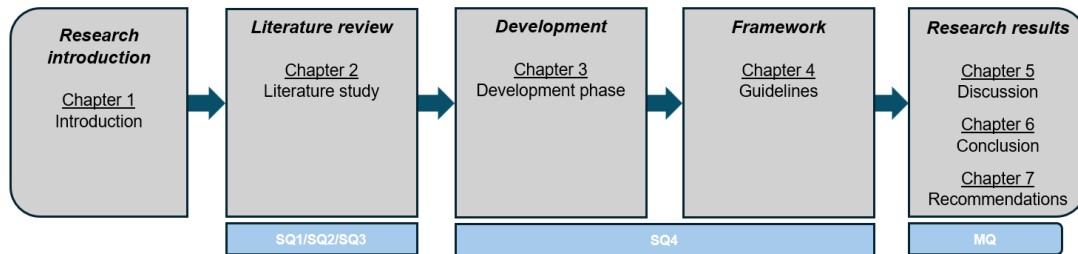
The scientific relevance of this research lies in its contribution to integrating automated processes within the construction industry specifically in the verification of requirements in the BIM environment. By addressing the fragmentation of tools used for requirement verification this research explores how the integration of different systems can enhance the accuracy and efficiency of the verification process. This research contributes to the theoretical development of data verification and the efficient use of digital tools in construction project management. It investigates how data from various BIM systems can be structured and processed to optimise requirement verification. This work highlights the potential for a more efficient workflow, reduced fragmentation and improved quality control in construction projects.

#### Practical relevance

The practical relevance of this research lies in its direct applicability to construction companies like Witteveen+Bos which seek to improve their verification process of requirements. The findings can offer practical solutions for integrating systems, optimizing requirement verification and enhancing efficiency which benefit the sector.

### 1.3.7 Research outline

This subsection provides an overview of the structure of the thesis and highlights the progression of the research. An overview is included to visually represent the research process (see **Figure 3**), illustrating how the chapters are connected and how each chapter contributes to achieving the research objective.



**Figure 3.** Outline of the thesis.

The thesis is structured into seven chapters where each chapter addresses different components of the research process. Chapter 1 introduces the context and problem definition. Chapter 2 presents a literature review that discusses the current challenges and limitations of verifying requirements within BIM models (SQ1). It also examines the existing software used for requirement verification and the most effective methods for requirement formatting, answering sub-questions 2 and 3. Chapter 3 focuses on the development phase and the case study, where sub-question 4 will be answered. In Chapter 4, guidelines will be created based on the findings from the previous stages. Chapter 5 includes a discussion of the findings. Chapter 6 presents the conclusion and the thesis concludes with Chapter 7, which provides recommendations for future work.



## 2. Literature study

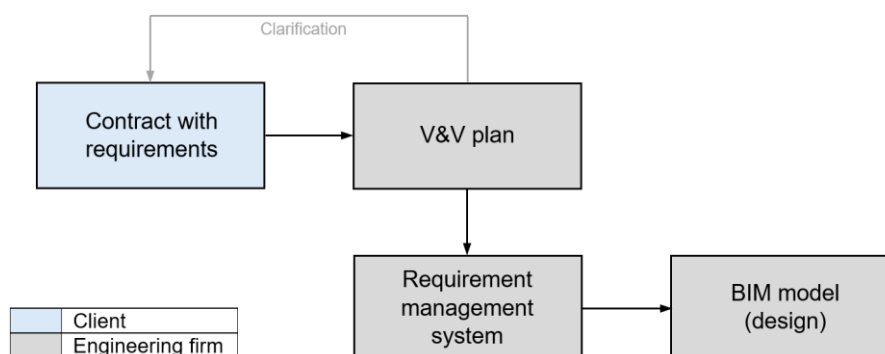
This chapter introduces the concepts of Building Information Modeling and project requirements. Section 2.1 explores the role of requirements in construction projects and discusses common challenges such as misinterpretation. In Section 2.2, the evolution of BIM will be discussed along with its role in modern construction, relevant standards and emerging technologies that influence the integration of requirement verification. Section 2.3 looks at automated model verification tools and compares their strengths and limitations. Finally, Section 2.4 will look at different languages for transforming textual requirements into a structured format. Through this literature review, the chapter lays the foundation for understanding the current state of requirement verification in BIM and the potential for improvement.

### 2.1 Requirements in construction projects

Throughout construction history numerous projects have fallen short of meeting the client expectations due to a lack of precise understanding of the project requirements at an early stage (Karim Jallow et al., 2014). This has led to clients now having higher demands which requires extra effort to stay within the project constraints (Hassan & Le, 2020). Conflicting requirements are a major cause of disputes which often result in costly delays if not addressed early (Cheung & Pang, 2013). Also, due to the increasing complexity of construction contracts, it can occur that some requirements are missed which can influence the understanding of the project scope and obligations associated (Walsh, 2017). This can lead to various issues in the project such as design changes, waste of resources, legal conflicts and higher risks of accidents because of construction errors (Moon et al., 2022).

Project requirements refer to the obligations outlined by the client within the contract documents (Hassan & Le, 2020). The requirements outlined in the contract originate from the client's design vision and the associated specifications. Effectively identifying and extracting the requirements from the contract is essential for accurately defining the project scope and to ensure the projects successful delivery while fully meeting client expectations (Hassan & Le, 2020).

Based on the project requirements from the contract, the engineering firm prepares a verification and validation (V&V) plan (see **Figure 4**). The V&V-plan describes how requirements need to be verified within a project to ensure the design meets the functional, legal and technical specifications. Customer requirements are translated into engineering specifications and expressed using clear technical terminology. During this process, some requirements may be omitted or misinterpreted (Shabi et al., 2017). Therefore, it is advisable to implement continuous verification and validation throughout the project lifecycle starting as early as possible (Sage & Rouse, 2011).



**Figure 4.** Verification and validation (V&V) procedure from contract to design (simplified).

The paper by Lake (1999) defines verification and validation within the context of Systems Engineering. Verification involves assessing a completed project to confirm that it meets the specified technical requirements (Lake, 1999). On the other hand, validation refers to evaluating a product against the defined client requirements to determine whether it satisfies stakeholder expectations (Shabi et al., 2017). Once the verification and validation plan is approved by the client, the requirements can be documented in a requirement management system (RMS) such as Relatics. Based on the specified requirements within the RMS, a design can be developed, which is then continuously evaluated against the V&V-plan.

## 2.2 Building Information Modeling

### 2.2.1 The evolution of BIM

Building Information Modeling has evolved significantly since its inception. The evolution of BIM has been progressing for many years and continues to develop. It is difficult to pinpoint the exact origin of the BIM concept (Borkowski, 2023). In 1957 Pronto was created by Dr. P.J. Hanratty which was the first commercial software for computer-aided manufacturing (Elanchezhian et al., 2007). This technology, initially focused on numerical control machining, eventually evolved into modern computer-aided manufacturing (Khochare & Waghmare, 2018). The first computer-aided design (CAD) software with a graphical user interface known as Sketchpad was developed in 1964 by Dr. I.E. Sutherland (Sutherland, 1964). This was the first instance of a user interacting with a computer to design objects through a graphical interface (Fay, 2020).

In 1975 the concept of Building Description System (BDS) was introduced by C.M. Eastman (Eastman, 1976). This early prototype discussed the principles of parametric design and detailed 3D representations and described the foundational ideas of what is now recognized as Building Information Modeling (Khochare & Waghmare, 2018). Shortly after that, Eastman developed GLIDE (Graphical Language for Interactive Design) in 1977 (Eastman & Henrion, 1977). This language for design information systems showcased many of the features found in modern BIM platforms (Khochare & Waghmare, 2018).

In the 1980s the development of digital modelling and the exchange of project data in the construction sector started (Laakso & Kiviniemi, 2012). Also, numerous systems were being developed and gained traction within the industry with some even being implemented in construction projects (Khochare & Waghmare, 2018). The development of ArchiCAD began in 1982 led by Gabor Bojar who also released Radar CH in 1984 which used similar technology to BDS (Khochare & Waghmare, 2018). Radar CH was reintroduced as ArchiCAD in 1987 making it the first BIM software accessible on a personal computer (Khochare & Waghmare, 2018). During that period, Vectorworks was developed by Diehl Graphsoft in 1985, which was one of the first 3D modelling software programs (Khochare & Waghmare, 2018). In the same year (1985), Parametric Technology Corporation (PTC) was founded which is known as the company that launched Pro/ENGINEER in 1988, which was considered to be the first parametric modelling software in BIM history (Gobesz, 2020; Khochare & Waghmare, 2018).

In 1992, the term 'Building Information Model' was first documented in a paper by G.A. van Nederveen and F. Tolman (van Nederveen & Tolman, 1992; Sakib, 2021). A year later (in 1993), the Building Design Advisor (BDA) was developed, offering software that ran simulations and proposed solutions based on a model (Papamichael et al., 1997). After that miniCAD was created by Mapsoft in 1994 that added the feature of CAD surveying (Sakib, 2021). During that period, a significant shift towards BIM as it is known today started (Bloor & Owen, 1995). In 1995, the Industry Foundation Classes (IFC) were introduced to provide a standardized data model and to

enable interoperability between different BIM applications (Laakso & Kiviniemi, 2012). Over the years IFC has undergone various updates and evolved from simple data exchange mechanisms to a framework that supports a wide range of construction processes (Eastman, 1999).

In 2000, two former PTC employees founded Charles River Software with the aim to create an architectural version of Pro/ENGINEER capable of managing more complex projects than ArchiCAD, which led to the creation of Revit (Sakib, 2021). A year later (2001), Navisworks was developed which allowed 3D navigation and collaboration in a model with varying file formats (Khochare & Waghmare, 2018; Sakib, 2021). The BIM Collaboration Format (BCF) was created in 2010 which enabled different BIM applications to exchange model based issues (Borkowski, 2023; buildingSMART, n.d.-e). Also, Augmented Reality (AR) and Virtual Reality (VR) began to be integrated into BIM which improved visualisation and led to more interactive collaboration (Matthys et al., 2021).

The openBIM concept was introduced in 2012 which is an international standard that enables open collaboration among all project stakeholders (Borkowski, 2023; buildingSMART, n.d.-a). In 2013, the concept of Internet of Things (IoT) began to emerge which enabled the connection with real-time data of various devices (Borkowski, 2023; Tang et al., 2019). The blockchain technology emerged in 2015, which could lead to benefits in construction management (Borkowski, 2023; Nawari & Ravindran, 2019). In 2018, machine learning was introduced which can be used with BIM to analyse data for example (Zabin et al., 2022). Besides, Artificial Intelligence (AI) has emerged which has the potential to address existing limitations of BIM applications (Borkowski, 2023).

## **2.2.2 Standards and regulations in BIM**

Standardizing information is crucial for effectively using data within a BIM project. For information to be valuable, it must be understandable and accessible to all users and not just those who initially entered the information into the model (Weygant, 2011). The adoption of standards helps reduce the risk of misinterpretations and difficulties in implementation (Shehzad et al., 2021). The international organization buildingSMART developed the open standard known as openBIM but there are also standards on a national level.

### **2.2.2.1 International standards**

The role of international standards in BIM is very important to ensure effective data sharing and collaboration across all project stakeholders. BuildingSMART is dedicated to the adoption of open standards that promote the digital transformation of the construction sector. OpenBIM is an approach for designing, constructing and managing construction projects using open standards. The openBIM methodology enables clear and open collaboration among all project stakeholders regardless of the software they use (buildingSMART, n.d.-a).

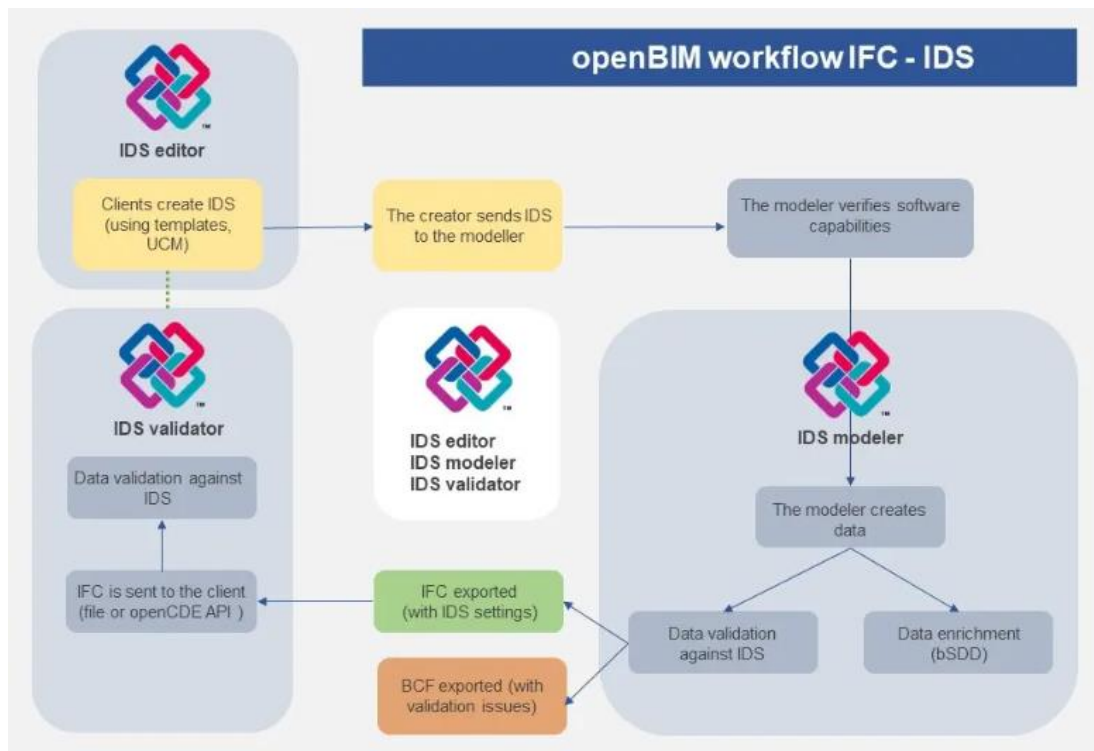
#### **BuildingSMART standards**

OpenBIM is known for its collaborative approach to enhance the accessibility, usability and sustainability of digital data within the construction sector. By supporting sharable project information, it helps with interoperability and ensures smooth collaboration throughout the lifecycle of a project (buildingSMART, n.d.-b). The buildingSMART standards, which are a core part of the openBIM initiative, consist of five basic standards that focus on data sharing (buildingSMART, n.d.-c). These standards are described below and displayed in **Table 1** among with their function.

### Information Delivery Specification

The Information Delivery Specification (IDS) is designed to define information requirements in a format that is both human readable and machine interpretable (buildingSMART, n.d.-d). It aims to improve communication among stakeholders by providing a more structured approach to specify requirements (buildingSMART, n.d.-d). In the openBIM workflow the client uses the IDS editor to create IDS files and sends this to the IDS modeler (see **Figure 5**). The modeler verifies the software capabilities and arranges the data. The integration of IDS with automated code compliance checking (ACCC) is a significant advancement in making sure construction projects follow standards and regulations, improving efficiency in the digital construction industry (de Marco et al., 2024).

Generally, information regarding requirements is shared in formats that are not computer interpretable like Excel or PDF which are hard to process automatically (buildingSMART, n.d.-d). In the construction sector, users often understand the type of information they need for a specific task but may struggle to clearly define it. This can significantly affect processes such as automated code compliance checking, which demands different data than automated cost estimation for example (buildingSMART, n.d.-d). The IDS addresses this by working best with structured data according to the IFC standard, which enables automated validation of the required project information (buildingSMART, n.d.-d; Eichler et al., 2023). Each IDS file contains specifications representing specific information requirements (Bigai & Santos, 2024).



**Figure 5.** openBIM workflow. Adapted from BibLus (2023).

### BIM Collaboration Format

The BIM Collaboration Format (BCF) enables different BIM applications to exchange model based issues by using IFC models shared among project collaborators (buildingSMART, n.d.-e). The BCF uses two main approaches for sharing data (Jiang et al., 2019). One method involves a file based workflow where issues are stored in a zip file using the BCF XML schema. The other method makes use of a server based approach using the BCF API specification for data transfer (Schulz et al., 2021). A BCF file can be exported from the IDS modeler together with an overview of the issues (see **Figure 5**).

Originally, BCF relied on a XML format that allowed for round-tripping which makes it possible for the same file to be extended with additional issues or to be modified. This avoids that the issues are stored over multiple files, so a clear structure can be maintained (Schulz et al., 2021). The BCF API was created to replace the file based method of exchanging issues in building processes (Schulz et al., 2021).

### *Industry Foundation Classes*

The Industry Foundation Classes (IFC) standard is used to describe building information model data. It establishes a standardized method for representing and storing data to enable various software applications to import and export building data in the same format (Jiang et al., 2019). IFC enables seamless sharing and interpretation of project data across different BIM software applications throughout the project lifecycle. The IFC standard improves interoperability and makes it easier for project stakeholders to work together (BIMcollab, 2024). The IFC2x3 TC1 version is currently the most widely adopted, with IFC4 closely following behind. Due to their similar structure, both IFC versions are supported by many applications (digiGO, 2023a). An IFC file can be exported from the IDS modeler and sent to the client for data validation against the IDS (see **Figure 5**). This export can be carried out using a file or through the openCDE API, which is explained in more detail in the next subsection.

### *Common Data Environment*

In the construction sector, Common Data Environments (CDEs) serve as centralised platforms that provide stakeholders with a unified source of project information to bring structure to the project and make communication between stakeholders easier. BuildingSMART offers a set of APIs known as openCDE APIs, which make the functionalities of these CDEs more accessible (Schulz & Beetz, 2024). The openCDE initiative consists of a collection of API standards (buildingSMART, n.d.-b). An Application Programming Interface (API) is an interface that facilitates standardised data exchange between different applications and software components. By following a defined syntax, APIs ensure that data and commands are transferred across an organization wide environment (Bucher & Hall, 2020). Also, with the growing trend of cloud-based BIM services, there is an increasing need to integrate BIM with web service APIs to facilitate web-based data exchange (Afsari et al., 2017; Wu & Issa, 2012).

### *Information Delivery Manual*

The Information Delivery Manual (IDM) is a methodology designed to define and document business processes and data requirements to both new and existing processes (buildingSMART, n.d.-c). This approach details the information that must be exchanged between various parties and can be used to create a more detailed specification that can form the basis for a software development process (buildingSMART, n.d.-f). The main purpose of an IDM is to ensure that data is communicated in a way that can be interpreted by the receiving software (buildingSMART, n.d.-f; Jiang et al., 2019). An IDM captures the entire process and information flow throughout the project lifecycle to promote efficient data exchange (Jiang et al., 2019). Given the multiple stakeholders in a construction project, it is important to know what and when information needs to be communicated among them (Jiang et al., 2019).

**Table 1.** Overview of the five buildingSMART standards and their corresponding ISO standards.

Name	Function	ISO standard
<b>IDS</b> Information Delivery Specification	Specifies data requirements	-
<b>BCF</b> BIM Collaboration Format	Facilitates model-based communication	-
<b>IFC</b> Industry Foundation Classes	Defines data models	ISO 16739-1:2024
<b>CDE</b> Common Data Environment	Standardizes data access	-
<b>IDM</b> Information Delivery Manual	Documents processes and requirements	ISO 29481-1:2010

Industry Foundation Classes stand out as the most used initiative among the various buildingSMART standards, such as IDM, BCF, et cetera (Hooper, 2015). With the growing adoption of BIM applications within the construction industry, IFC has gained prominence as an open format standard and is recognized as an ISO standard (Diara & Rinaudo, 2020). In general, as a solution for addressing interoperability and data sharing challenges, openBIM has significantly improved collaboration in construction projects. With the ongoing evolution of openBIM it is expected to offer even more comprehensive solutions to the challenges currently faced in the industry (Jiang et al., 2019).

### ISO-standards

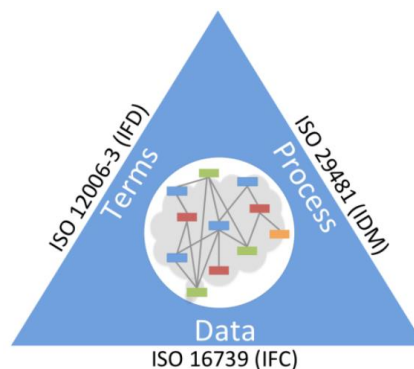
In addition to the standards established by buildingSMART, there are also ISO standards set by the International Organization for Standardization. These ISO standards are internationally agreed upon by experts to ensure a global consensus on best practices in BIM implementation (ISO, n.d.). For some standards, both ISO and buildingSMART have the right to publish them (see **Table 1**) due to a copyright agreement signed between these parties (Poljanšek, 2017). This subsection discusses the most important ISO standards for the construction industry regarding the use of BIM.

The ISO 19650 standard is an international standard that facilitates the management of information throughout the lifecycle of a project (Pan et al., 2024). This standard has been developed over several years to improve digital information sharing across the value chain of assets. Moreover, the ISO 19650 standard connects information management with the United Nation's sustainable development goals and consists of six parts (Bjørn et al., 2022).

The first part outlines the core concepts and principles of information management (ISO, 2018a). Part 2 specifies the requirements for managing information during the delivery phase of a project (ISO, 2018b). The third part of the ISO 19650 standard focuses on the operational phase by specifying the requirements for maintaining the project after its completion (ISO, 2020a). Part 4 outlines the principles for information exchange between parties and promotes data interoperability throughout the lifecycle of a project (ISO, 2022). The fifth part of the standard addresses the steps required to create a security-minded approach to keep information secured across different organizations (ISO, 2020b). Lastly, part 6 emphasizes the integration of health and safety information but this is still under development and yet to be published (ISO, 2025).



In addition to ISO 19650, there are three standards that are part of the ISO standard triangle (see **Figure 6**). This conceptual framework illustrates the integration of data, processes and terms within the BIM environment. Each side of the triangle represents a critical component that ensures seamless data exchange and collaboration in construction projects (Laakso & Kiviniemi, 2012). At the core of the data component is the IFC standard (ISO 16739), which enables the exchanges and sharing of information across different software platforms as discussed earlier (buildingSMART, n.d.-g). The process component is represented by the IDM standard under ISO 29481 which also is a buildingSMART standard. This standard about IDM provides a method for defining the processes of information exchange (buildingSMART, n.d.-f). The third component is associated with the International Framework for Dictionaries (IFD) covered by ISO 12006. The IFD standard provides standardised terminology for the construction industry (Cho et al., 2011).






**Figure 6.** The triangle with ISO standards. Adapted from Laakso and Kiviniemi (2012).

### 2.2.2.2 National standards

The paper by van Nederveen et al. (2010) describes that a remaining issue is the use of open standards for BIM. The process of standardization is often perceived as slow and challenging, which can hinder broader adoption and integration efforts (van Nederveen et al., 2010). The organisation digiGO, formerly known as BIM Locket, is a Dutch initiative that aims at advancing the digital transformation in the construction industry for open BIM standards (digiGO, 2023b). DigiGO aims to make maximum use of international standards but also develops its own. DigiGO is the owner of three open standards: NLCS, NL-SfB, and VISI (digiGO, 2023c), as can be seen in **Table 2**.

**Table 2.** Dutch open standards by digiGO.

Open standard	Description
	NLCS is the 2D drawing standard within the Dutch construction sector. Having a CAD standard is essential to ensure consistency across technical drawings and improve the sharing of digital files. This open standard includes guidelines for managing data, the visual presentation of drawings, and the system for organizing layers within the drawings.
	The open standard NL-SfB is the most commonly used classification for building elements. NL-SfB has been used for many years to code layers and objects in BIM and CAD systems. Several initiatives from the Netherlands have been brought together to link NL-SfB with IFC in order to achieve consistency in this approach.
	VISI is an open standard for digital communication which is crucial for the success of complex construction projects. It assists in determining when, who, what and to whom information should be provided.

## Dutch NEN-standards

Besides the open standards from digiGO there are also national standards developed by the Royal Netherlands Standardization Institute (NEN). An important national standard is NEN-EN ISO 19650 which descends from the international ISO 19650 standard discussed in Subsection 2.2.2.1. Since the implementation of this standard, adaption or updating of existing protocols to ISO and NEN standards is expected (Bruggeman, 2020). In the Netherlands, the IFC standard is broadly acknowledged as the leading standard for BIM which also is part of the NEN-EN ISO 19650 (van Nederveen et al., 2010).

### 2.2.2.3 BIM Basis ILS

The BIM Base IDS (in Dutch: BIM Basis ILS) is a Dutch standard that provides guidelines for the exchange of information in BIM projects. The Information Delivery Specification (IDS) is covered in more detail in Subsection 2.2.2.1 as part of the buildingSMART standards. However, the BIM Base IDS is specifically tailored to the Dutch construction sector. Effective cooperation within a project is enhanced when the information used is exchangeable, structured, accurate and reusable (digiGO, 2023a). The BIM Base IDS serves as an essential first step in achieving this (digiGO, 2023a). The BIM Base IDS is supported by digiGO which provides access to all relevant documents. The adoption within the Dutch industry is widespread, with over 500 companies recognized as partners of the BIM Base IDS (digiGO, 2023a). The BIM Base IDS consists of four parts or 'questions' that one can ask themselves to ensure effective information management, namely: 1) Why we exchange information, 2) How we exchange information, 3) What we agree on to enable collaboration and 4) Which information is required in one of the aspect models (see Figure 7).

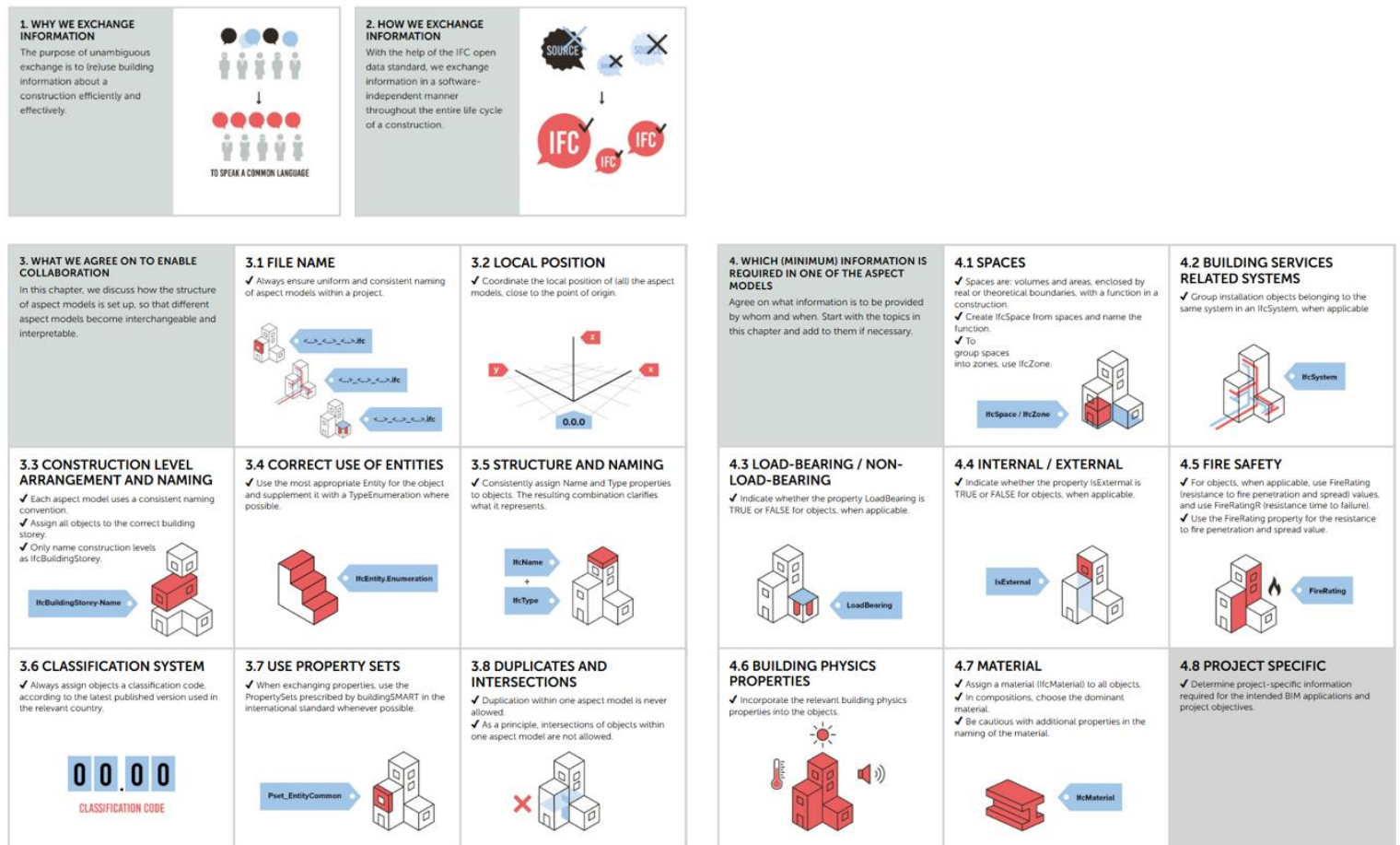


Figure 7. Overview of BIM Base IDS parts 1-4. Adapted from digiGO (2023a).



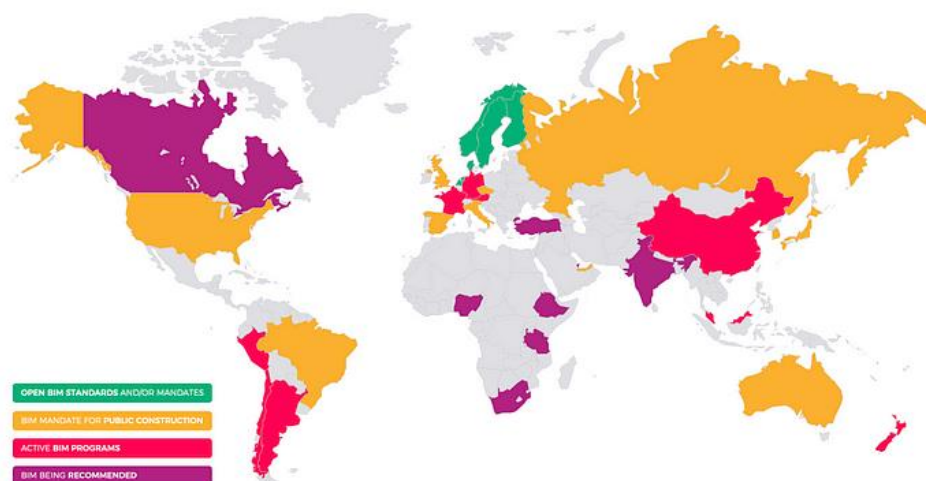
The first and the second part of the BIM Base IDS are a bit more general than the third and the fourth part. The first part has the purpose to effectively reuse building information throughout the construction process. It is a set of agreements that define the requirements that the integrated building information model must meet (digiGO, 2023a). The second part discusses how information is exchanged using the IFC standard. Part 3 looks at enabling collaboration by agreeing on the structure of aspect models to ensure they are interchangeable. This includes file naming, local positioning among others as can be seen in **Figure 7**. The fourth part of the BIM Base IDS discusses the minimum information that is required.

#### 2.2.2.4 The adoption of BIM standards

Countries around the world have invested significant effort in developing distinct standards independently to effectively implement BIM in projects (Ganah & Lea, 2021). Improving the existing BIM standards is necessary to allow the integration of BIM with other technologies for managing structured and unstructured data (Godager et al., 2021). BIM standards have a strong effect on whether companies and individuals adopt BIM in projects (Xu et al., 2014). One of the major barriers to BIM adoption within the construction industry is the lack of BIM standards (Chan et al., 2019). The ongoing advancements in BIM standards continue to play a crucial role in facilitating seamless communication and collaboration across the diverse stakeholders involved in construction projects (Kiviniemi et al., 2008).

The use of BIM standards is essential for the broad adoption and implementation of BIM. However, many countries such as China, India, France, Germany, the Netherlands, Norway and Brazil, do not provide project stakeholders with any form of guidelines and standards (Ganah & Lea, 2021). The United States has developed the most BIM standards, but these standards are state led which has resulted in duplicated efforts and fragmented standardisation (Ganah & Lea, 2021).

Among all continents, North America, Europe, Asia and Oceania are progressing quickly towards reaching a high level of maturity (introduced in following subsection). On the other hand, Africa and South America are still in the early stages of BIM implementation as can be seen in **Figure 8** (Jung & Lee, 2015). Europe has a greater number of standards, guidelines and templates than all other continents combined with 5,2 times as many BIM standards and document relations as the second ranking continent, Asia (Ganah & Lea, 2021). Also, Europe holds more than three times the number of relationships between BIM standards compared to all other continents. In particular, these relationships between standards ensure a consistent workflow at the full length of any project (Ganah & Lea, 2021).



**Figure 8.** Adoption of BIM standards worldwide. Adapted from Eischet (2022).

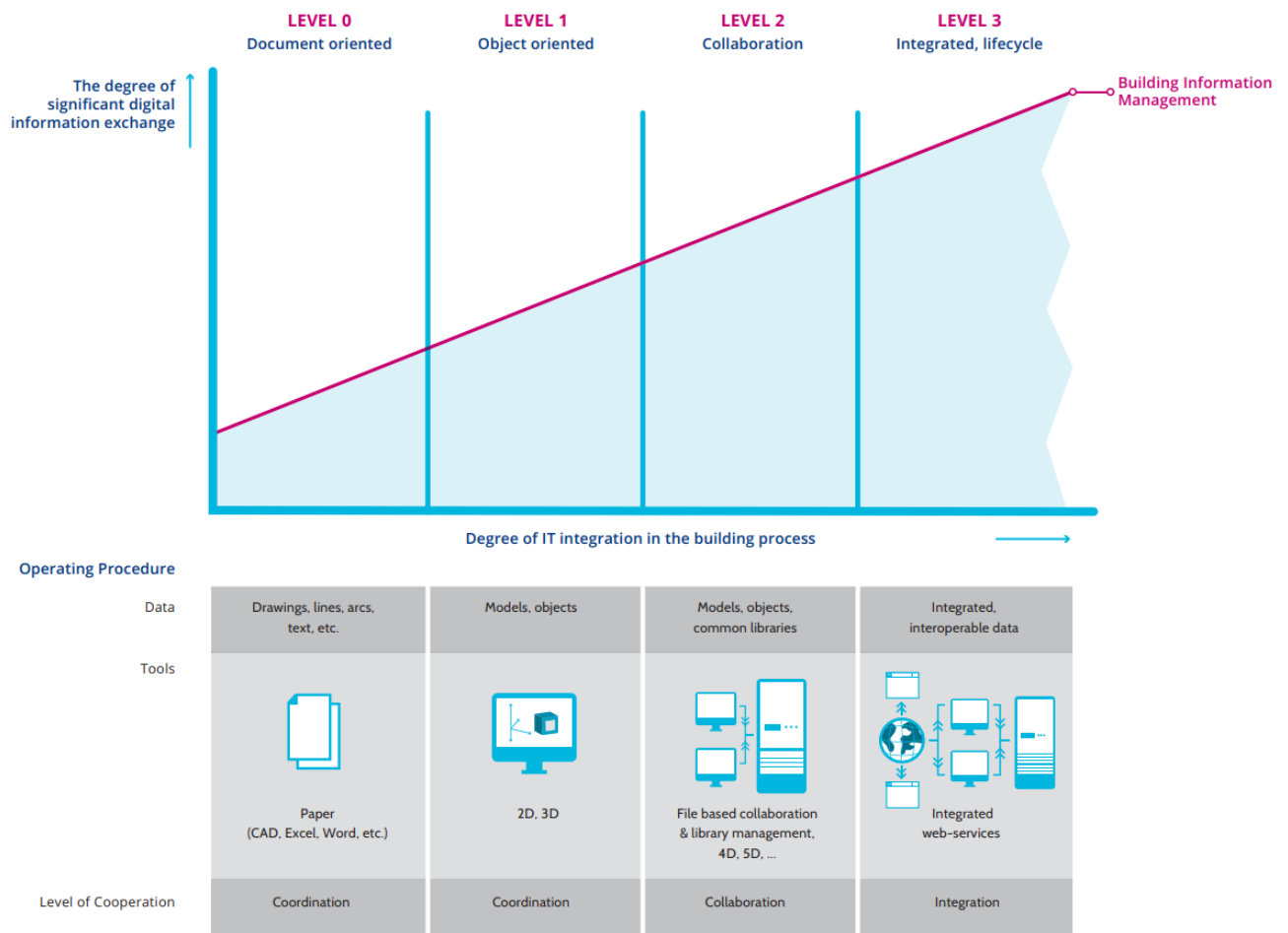
In recent years, public sector organizations in the Netherlands (e.g. Rijkswaterstaat) have significantly expanded the use of BIM technology (Cheng & Lu, 2015). Although there is no official legislation, clients frequently require the use of BIM in projects (Panteli et al., 2020). Significant efforts have been undertaken in the Netherlands (as seen in Subsection 2.2.2.2) to establish standardised BIM methodologies and tools for documenting the information requirements (Ganah & Lea, 2021). The public sector is crucial in driving the construction industry's transition to BIM adoption (Cheng & Lu, 2015). Lately, there has been a notable rise in BIM implementation as numerous government agencies have integrated BIM into their projects and introduced diverse BIM standards (Cheng & Lu, 2015).

Overall, BIM standards are still underdeveloped and have not yet achieved full implementation and widespread adoption in many countries. Numerous countries have yet to begin adopting BIM, while others have adopted BIM standards from other countries rather than developing their own (Ganah & Lea, 2021). The adoption and development of BIM have been driven by the need for greater efficiency and collaboration within construction projects (Laakso & Kiviniemi, 2012).

### 2.2.3 BIM maturity

BIM technology is mainly used in new construction projects. However, a significant obstacle to its adoption for older or existing projects that were not originally designed with modelling, lies in acquiring precise building data and transforming it into a BIM model (Megahed & Hassan, 2022). The construction industry cannot achieve the massive shift to fully digitized model-based workflows all at once. Rather, a more suitable approach is to gradually introduce the new technologies and the associated changes in steps (Borrmann et al., 2018). The range of steps that a BIM model can achieve are referred to as maturity levels (Dakhil et al., 2015). BIM maturity reflects the quality, repeatability and excellence achieved in the production and delivery of a BIM model (Succar, 2010). An increasing number of evaluation models have been developed to assess BIM maturity (Dakhil et al., 2015). The United Kingdom (UK) BIM framework is among the most frequently referenced models for assessing BIM maturity (Adekunle et al., 2022).

The UK BIM maturity model, initially developed by Bew and Richard (2008) as the foundational framework, was later advanced by the UK BIM Taskforce for the British government and is increasingly being adopted across Europe (BDR, 2014; Dakhil et al., 2015). The model can easily be understood by stakeholders and measures maturity based on adherence to its specifications (Dakhil et al., 2015). However, this model is restricted to the UK as its maturity levels are specifically aligned with local standards (Dakhil et al., 2015). As a result, the Bouwdigitaliseringsraad (BDR) developed a Dutch BIM maturity model based on the UK BIM maturity model (see **Figure 9**). This model is a growth framework for BIM and does not say anything about the organization's maturity (BDR, 2014).



**Figure 9.** The Dutch BIM maturity levels. Adapted from BDR (2014).

The BDR encourages the adoption of BIM with the maturity levels being a helpful tool (BDR, 2014). The BDR is a collaboration between the government, clients, contractors and knowledge institutions with the focus on accelerating BIM adoption and digitalisation in the construction sector (digiGO, 2023b). The line in **Figure 9** illustrates the extent of information and integration across the various stages of a project lifecycle (BDR, 2014). The Dutch BIM maturity model uses four distinct levels ranging from 0 to 3 (see **Table 3**). Each level correlates with specific strategies, including data types, software tools, collaboration methods and cultural approaches (BDR, 2014). In relation to automated requirement verification, its applicability becomes relevant at the higher BIM maturity levels. The lower levels lack the necessary information structure to enable automated compliance checking. At the higher maturity levels, the prerequisites for automated requirement verification, such as interoperable data and integrated web services, are more fully established (see **Figure 9**). Advancement to a higher level requires the completion of the preceding BIM maturity levels (BDR, 2014).

**Table 3.** Description of the Dutch BIM maturity levels. Based on BDR (2014).

BIM level	Description
Level 0	Level 0 uses text, lines, arcs, etc. These are digitally processed at document level such as drawings in CAD software. This concerns non-intelligent information where no digital objects are applied.
Level 1	The first step in implementing BIM involves working with objects (either 2D or 3D). A key characteristic of level 1 is the use of specific objects that can be linked to information. At this stage there is no integration between different aspects, meaning there is no connection of the 3D model with for example a requirement database.
Level 2	At this level the object models developed in level 1 can be shared. Collaboration occurs using a federation of databases where each participant has its own model and all these models are combined into a single view model to collaborate on a project in a file based manner. Also, applications like planning (4D BIM) and cost calculations (5D BIM) can be linked to the model in this maturity level.
Level 3	Level 3 involves using interoperable open BIM standards to share information among various parties. This is done in an integrated web services environment, shifting from file-based to object-based exchanges. The construction process is fully integrated into the chain, and by the end of this level, lifecycle information is shared within the integrated environment.

The Dutch BIM maturity level is already far advanced (Papadonikolaki, 2018). This has been achieved independently of any obligatory regulations or external pressures from the Dutch government (Kassem et al., 2015). Rather, there are numerous initiatives in the Netherlands aimed at promoting BIM adoption as different companies create standards to facilitate BIM implementation (Papadonikolaki, 2018). As BIM technology has matured, it has transitioned from a digital tool for visual representation to an integrated system that supports the entire building lifecycle (Laakso & Kiviniemi, 2012).

## 2.2.4 Future trends and emerging technologies

The construction industry has been known for its slow adoption of innovative technologies. The use of digital technologies was often viewed as an unrealistic or improbable approach (Sepasgozar et al., 2023). The reason that the construction industry lags in adopting new technologies is because of interdependencies and the specialized nature of its projects (Khan et al., 2021). With the ongoing advancements in technology, it becomes important to explore the future of BIM and its potential influence on the construction industry (Altieri, 2024). This subsection outlines the most discussed future trends and emerging technologies in the field of BIM as highlighted in recent studies (Cassandro et al., 2024; Kim & Kim, 2022; Mahajan & Narkhede, 2024).

### 2.2.4.1 Artificial intelligence and subfields

In recent years, Artificial Intelligence (AI) has rapidly advanced demonstrating its effectiveness in processing vast amounts of data in complex and uncertain conditions (Pan & Zhang, 2023). Artificial intelligence is designed to use the same human-like abilities, including logical thinking, problem solving and reasoning (Heidari et al., 2023). As the construction industry moves toward increased digitalisation it is crucial to uncover the connection between BIM and AI to support the intelligent development of civil engineering (Pan & Zhang, 2023). Integration of AI technologies with BIM can benefit the construction industry by improving planning, construction, maintenance and operations through digitalisation (Huang et al., 2021).

In comparison to traditional analytical methods, AI offers a more reliable and automated solution to address complex real world problems (Pan & Zhang, 2023). It is helpful for making decisions more easily with less reliance on human knowledge and increases the chances of successfully achieving the project goals (Hu et al., 2022). The characteristics of AI make it an effective tool for managing the complex and often repetitive challenges found in construction projects (Pan & Zhang, 2023). Because of this integrating different AI technologies is anticipated to provide sustainable advantages in automation, risk reduction, enhanced efficiency and safety within the construction sector (Pan & Zhang, 2021).

While the growing use of AI in the construction industry has driven research efforts, the field remains in its early stages with significant growth expected in the coming years (Pan & Zhang, 2023). The limited adoption of AI in the industry is likely due to factors such as high costs, lack of trust, security concerns, a shortage of skilled people, the uniqueness of projects and other challenges (Mahajan & Narkhede, 2024; Zabin et al., 2022).

Machine learning (ML) is a subfield of artificial intelligence (Yussuf & Asfour, 2024). Machine learning in AI plays an important role in analysing large amounts of data from different sources and using insights from both structured and unstructured data to make intelligent decisions (Heidari et al., 2023). Deep learning (DL) is a branch of ML that uses multiple layers of computing elements to process data (Baduge et al., 2022). This technology can be used to improve the analysis of construction projects by providing visual insights. For example, infrared and video-based systems are being more widely adopted in deep learning algorithms by using various types of cameras to gather image and video data (Baduge et al., 2022).

#### **2.2.4.2 Digital twins**

A Digital Twin (DT) is a virtual replica of a physical asset or object such as a viaduct which can continuously adapt over time as additional data becomes available throughout the project's lifecycle (Baduge et al., 2022; Boje et al., 2020). A DT is typically developed using data gathered from sensors and other sources including BIM models and is used to reflect the current condition of the object (Sepasgozar, 2020; Sepasgozar et al., 2023). Artificial intelligence technology can be integrated with digital twins to enhance the precision of these models and progressively refine them using the vast amount of data gathered during the construction process and the buildings remaining lifecycle (Baduge et al., 2022; Madubuike et al., 2022).

Despite increasing BIM adoption in the construction sector, progress toward DTs remains limited with minimal reliance on IFC, as the technology is still in its early stages but holds significant potential for impactful applications (Boje et al., 2020; Madubuike et al., 2022). Digital twin technology has shown considerable potential in the construction sector by allowing stakeholders throughout the value chain to interact, collaborate and simulate actions on the physical asset in a virtual setting (Zhu et al., 2024). This leads to valuable information that benefits the project lifecycle while also enabling effective data management (Madubuike et al., 2022; Zhu et al., 2024).

The development of an effective DT involves complex and technical steps (Madubuike et al., 2022). The challenges of developing a DT in combination with the unpredictable nature of the construction industry make it a difficult process to implement DTs in the construction sector (Brilakis et al., 2019). The absence of standards for developing a DT in the industry complicate the adoption as the construction sector would need tailored standards for different types of projects (Madubuike et al., 2022). Given the challenges in creating a DT and the complexity of the construction industry, it would be harder to convince stakeholders about the feasibility of a DT because of the limited data available on its implementation (Madubuike et al., 2022).

The emergence of advanced technologies like BIM, augmented reality, artificial intelligence, machine learning and sensors has made the implementation of DTs in the construction industry more achievable (Madubuike et al., 2022). Due to the complexities in the construction sector, there is a need to implement DT applications to solve the ongoing issues in the industry (Madubuike et al., 2022). For DT technology to realise its full potential, all stakeholders in the industry must be willing to share data and adopt standardised tools and methods. This collaborative approach allows everyone to access and contribute to an unified platform, which in the end reduces fragmentation within the construction sector (Moshood et al., 2024). The construction industry should tailor best practices for DTs from other sectors to suit its specific needs while establishing industry standards that promote technology integration and enhance collaboration among stakeholders (Zhu et al., 2024).

#### **2.2.4.3 Virtual reality and augmented reality**

Virtual and Augmented Reality (VR/AR) are innovative technological developments that connect the physical world with the virtual world (Pan & Zhang, 2023). Virtual reality (VR) creates a full simulation of a scenario, while augmented reality (AR) integrates the information about real entities with computer generated elements (Pan & Zhang, 2021). Given the characteristics of BIM and VR/AR, their combined use is emerging as a promising development which offers significant benefits during both the design and construction phases (Pan & Zhang, 2023). The construction industry is progressively looking for methods that help clients with incomplete knowledge of a project with a clearer understanding of the final result. Virtual reality has the potential to close the gap between clients and construction professionals by using the data already integrated into BIM models (Asgari & Rahimian, 2017).

In the recent years, the use of VR/AR has rapidly increased across various industries driven by their strengths in 3D visualisation, interactive experiences, among other benefits (Sidani et al., 2021). The capability to virtually experience a physically non-existent space with participants located in different parts of the world represents a shift in perception rather than just technology within the construction sector (Asgari & Rahimian, 2017). Virtual reality technology can also help improve design reviews of a BIM model by allowing issues to be identified early in the design process to prevent delays later on (Asgari & Rahimian, 2017). Given the vast amount of geometric and semantic data embedded in BIM for virtual simulation and information sharing, it is practical to integrate VR/AR technologies with the BIM environment (Sidani et al., 2021).

#### **2.2.4.4 Internet of things and cloud computing**

The Internet of Things (IoT) lacks a universally accepted definition but is widely viewed as an interconnected network of physical devices such as drones, sensors, Extended Reality (XR), global positioning systems (GPS) and other tools for communication (Ghosh et al., 2021). By integrating with other technologies, IoT transforms the internet and serves as a potential bridge to merge physical and virtual worlds (Sepasgozar et al., 2023). The adoption of IoT in the construction industry offers opportunities for economic growth and supports the development of a larger data environment enabling future insights powered by big data (Bilal et al., 2016). The IoT technology will enable rapid reporting which can lower communication costs and minimize the risk of human errors (Ghosh et al., 2021). Also, it will enhance process control and optimization by using advanced algorithms and AI which can interpret the data rather than just analysing it (Al-Ali et al., 2017).

Artificial Intelligence Internet of Things (AIoT) is the new generation of IoT that integrates AI technologies into IoT systems to enhance operational efficiency and data analysis (Pan & Zhang, 2021). While various studies have applied IoT-based sensing technologies in projects, the integration of AI can improve decision making and predictions for construction activities such as



project planning and maintenance (Baduge et al., 2022; Sepasgozar et al., 2023). Despite its potential, the practical application of AIoT is still in its early stages as technology faces challenges such as edge computing issues and security concerns among others (Pan & Zhang, 2021).

The concept of (A)IoT revolves around the integration of sensors into a network that facilitates seamless information sharing through the internet (Tang et al., 2019). Cloud computing refers to hosting computing services online to allow IoT devices to integrate with the current internet network (Khalid et al., 2017). Cloud computing systems simplify the processing of large datasets and enable informed decision making through ML and big data analytics (Baduge et al., 2022). The construction industry has broadly adopted cloud computing due to its ability to support various BIM tool applications and data storage (Tang et al., 2019).

An example is cloud VR/AR which refers to the use of VR/AR technologies hosted in the cloud to enable information sharing across different devices (Baduge et al., 2022). These cloud based solutions are becoming essential in applications like clash detection, model improvements and education (Baduge et al., 2022). Also, the increased adoption of technologies like IoT and cloud computing is expected to drive the expansion of the digital twin market (Singh et al., 2021). In addition, the combination of AIoT and BIM will play an important role in realizing smart cities and infrastructure within the construction sector in the future (Mannino et al., 2021).

#### **2.2.4.5 Future perspective**

The integration of BIM and AI is expected to remain a prominent focus of research in the coming years (Pan & Zhang, 2023). Many AI applications in the construction industry are still in the early stage with a handful of organisations providing commercial solutions (Baduge et al., 2022). Given the benefits of the described technologies the integration is considered an interesting area for exploration. BIM can serve as a digital foundation that collaborates with different techniques like AI and cloud computing to improve intelligence in project management across the entire project lifecycle (Pan & Zhang, 2023).

### **2.3 Automated model verification tools**

In the traditional process, building authorities manually review 2D drawings to ensure they align with the codes and regulations (Preidel & Borrmann, 2018). Any modifications to the design that follow at a later stage, require the affected elements to be checked again (Preidel & Borrmann, 2018). Manually verifying designs for compliance with requirements is a complex task that carries the risk of human error and can lead to high costs (Tan et al., 2010). The use of digital technologies like BIM along with the creation of data standards for digital construction models (e.g. IFC), has introduced new tools that offer a highly effective foundation to enhance the validation process (Preidel & Borrmann, 2018).

That is why automated model verification tools are becoming an important part of new technology in the BIM sector. These tools help improve compliance checks and requirement verification and make project coordination more efficient (Eastman et al., 2018). Automated code compliance checking (ACCC) is said to offer significant advantages for both designers and consultants as well as certifiers, regulatory authorities and construction professionals (Tan et al., 2010). Various companies around the world have launched initiatives to create automated systems for checking designs (Eastman et al., 2009). Companies like Solibri and SmartReview have created tools that automatically check whether models follow certain rules and codes using IFC files. These tools reduce the time spent on manual reviews, which allows for quicker identification of issues and better project timelines (Eastman et al., 2018).

Automated compliance checking relies on the use of software tools that are typically international while requirements are often local and specific. Most compliance checking software has focused on converting BIM models into the international IFC format and creating custom rules to apply on these models (Greenwood et al., 2010). This approach faces challenges because international tools often fail to include all necessary data in the IFC models. As a result, for accurate compliance checking, additional data must be added as a separate task (Greenwood et al., 2010).

### 2.3.1 Evolution of automated verification tools

The first successful attempt at automating design compliance in the construction industry was done by Fenves (1966), who explored the use of decision tables to represent the American Institute of Steel Construction (AISC) standard specifications (Nawari, 2018). He noted that decision tables, a new programming method using if-then statements, could be used to clearly represent design rules (Ilal & Günaydin, 2017). The research of Lopez et al. (1989) expanded on Fenves's work by developing the Standard Interface for Computer Aided Design (SICAD) system. This software program was developed to demonstrate how designed components could be checked for compliance with the requirements (Nawari, 2018).

Automated compliance checking has been a subject of ongoing research for over 50 years (Zou et al., 2023). Research on frameworks for automating code compliance checking through the representation and processing of design standards began two decades ago (Nawari, 2018). Back then building models and rule checking methods were developed, but fully functional computable code systems have only recently started to appear (Nawari, 2018). Significant progress in automated code compliance checking occurred after the introduction of the IFC standard for BIM in the late 1990s (Han et al., 1998). Some examples of such automated code compliance checking tools are SMARTcodes, AutoCodes, ePlanCheck, and DesignCheck (see Figure 10).

In 2000, ePlanCheck was developed in Singapore as part of the Construction and Real Estate NETwork (CORENET) project (Malsane et al., 2015). This system enabled online submission and approval of building plans, combining 2D drawings with additional IFC-based data (Malsane et al., 2015). The tool ePlanCheck used an early version of IFC as an open standard, but various limitations hindered its widespread adoption so the project was stopped (Solihin et al., 2020). DesignCheck was developed in Australia in 2005 and was one of the first ACCC tools in the world, but it was not used in the industry and it is no longer being developed (Zou et al., 2023). DesignCheck had the benefit to assess compliance at different stages of the design process, but it lacked the ability to view 3D models and all reports were text-based (Malsane et al., 2015).



Figure 10. Origin of automated compliance checking tools.



The International Code Council (ICC) located in the US, developed the SMARTcodes system which used Solibri Model Checker to evaluate the accuracy of the code compliance results (Eastman et al., 2009). This project has primarily concentrated on the problem of converting human written codes into rules that can be understood by machines (Greenwood et al., 2010). The ICC ended the developments in 2010 due to insufficient funding, but the company AEC3 has continued to develop the mark-up language originally used by SMARTcodes (Dimyadi & Amor, 2013; Preidel & Borrmann, 2018). Another effort to automate code checking is the AutoCodes project initiated by the organization Fiatech based in the US (Eid et al., 2020). The project was launched in 2011 aiming to accelerate and standardize the regulatory process by using technology to automate code checks for BIM models and to make the review process more efficient (Eid et al., 2020). The AutoCodes project was discontinued by Fiatech in 2017 after failing to gain traction (Khemlani, 2018).

### 2.3.2 Current automated compliance checking software

This subsection discusses the current automated code compliance checking software.

#### **Solibri Model Checker**

Solibri Model Checker is developed by the Finish company Solibri and was released in the year 2000 (Preidel & Borrmann, 2018). The company is acquired in 2015 by the German firm Nemetschek AG (Solibri, 2015). Solibri Model Checker (SMC) is a widely used BIM tool that helps designers identify and address potential issues with a model before and during the construction process (Ismail et al., 2017). The tool offers automated checking including highlighting potential design problems, identifying design deficiencies and missing components (Ismail et al., 2017). Every process within SMC relies on data from IFC files, which are integrated into an internal data structure (Preidel & Borrmann, 2018). While SMC interacts directly with building model data in IFC format, it selectively extracts only necessary objects for the compliance check (Malsane et al., 2015). Even though the IFC format is standardised, different BIM software may export IFC data in their own unique ways. As a result, SMC uses custom routines to adjust to these different export methods which allows SMC to read the IFC files from various BIM applications (Preidel & Borrmann, 2018).

Solibri Model Checker has a rule manager component that offers a library of rules and guidelines that allow users to choose a customized review process based on their requirements (Eastman et al., 2009). These predefined templates, representing a standard checking procedure, can only be modified by a few parameters (Preidel & Borrmann, 2018). Apart from basic rules that check the quality of the imported IFC model, the manager component primarily offers geometry oriented rules such as those related to accessibility which are hard-coded (Preidel & Borrmann, 2015). This means that the information is accessed via the data model through a programming interface which is not visible to the users (Ismail et al., 2017). Adding new rules in SMC is hard as they can only be custom created by software developers from Solibri (Greenwood et al., 2010). This results in the fact that most of the users rely on the predefined rules by SMC and focus on basic checks like information completeness or verifying clashes which are the intersections of construction elements (Preidel & Borrmann, 2018).

As SMC is designed for checking the quality of model data it is not intended for editing or creating IFC data, but the results from the checking routines can be exported as PDF or BCF reports (Preidel & Borrmann, 2018). For the Dutch industry, digiGO provides a guide to check whether BIM models of version IFC2x3 comply with the BIM Base IDS by importing them into SMC (digiGO, 2020). The Solibri Model Checker uses strict guidelines on model structure requirements, such as ensuring the space layout does not intersect and attribute names are consistent with the spaces (Ismail et al., 2017). Also, SMC offers a 3D modelling engine that, combined with its ability

to read IFC files, enables clear visual reports of rule violations. Besides, its built-in rule library includes validation rules for the data model before checking, which adds extra value to the tool (Malsane et al., 2015).

## **Navisworks**

Navisworks is a software program developed in the United Kingdom (later acquired by Autodesk), and has been used for over a decade for clash detection and model reviewing and does support multiple file formats (Hjelseth, 2015; Thapa, 2019). Navisworks allows users to visualise and assess multiple BIM models within a single environment (Abdalthameed, 2023). Navisworks is used a lot in infrastructure projects because of its ability to handle and import various file formats (Hjelseth, 2015). The advanced features for model review make Navisworks one of the best visualisation tools on the BIM market (Thapa, 2019). Navisworks allows users to conduct virtual inspections by doing walkthroughs and visualise issues and non-compliances within the digital model (Paskoff et al., 2024).

Navisworks stands out from other 3D model coordination software by offering a simulation feature which makes 4D BIM (time included) and 5D BIM (costs included) implementation possible (Thapa, 2019). While Navisworks supports various file formats, the checking capabilities are somewhat limited (Paskoff et al., 2024). Navisworks lacks BCF management and monitoring capabilities, but the BIMcollab plug-in enables collaboration with the stakeholders (Thapa, 2019).

## **BIM Assure**

In 2016 the company Invicara launched BIM Assure which is an online platform designed for model checking (Preidel & Borrmann, 2018). Invicara was involved as a BIM application development company within Singapore's Building Construction Authority, which helped in gaining valuable insights in code checking and automation (Eastman et al., 2018). The required model data is obtained by BIM Assure through a plug-in for Autodesk Revit (Preidel & Borrmann, 2018). As a cloud-based application, BIM Assure handles rule checks and focuses particularly on rule checking throughout the design phase (Eastman et al., 2018). BIM Assure provides users with visibility into construction data through features like 2D/3D model viewing in the cloud by supporting the IFC format (Parsamehr et al., 2023). The tool enhances workflows by ensuring data quality and helps in minimising model errors and the need for rework (Eastman et al., 2018).

A process called normalization is used to align the data from the model with elements and object classes (Eastman et al., 2018). Elements that remain undetected after this process can be modified manually (Preidel & Borrmann, 2018). Users can create a customized checking process by selecting templates with adjustable parameters, while non-geometric issues like incorrect attributes can be solved quickly within BIM Assure (Preidel & Borrmann, 2018). BIM Assure can be used with minimal programming and domain knowledge and makes rule definition easy (Parsamehr et al., 2023). As the BIM model data and check routines are stored in the cloud, rule execution can be done without local hardware or software and even automated as part of a workflow (Preidel & Borrmann, 2018).

## **SMARTreview**

SMARTreview Automated Plan Review (APR) is an innovative rule checking solution designed to evaluate models for compliance with construction codes (Eastman et al., 2018). SMARTreview is a plug-in for Autodesk's Revit BIM authoring tool which is designed to assist compliance checks for specific aspects of the International Building Code (Clayton et al., 2013). Autodesk Revit is one of the most well-known BIM software programmes worldwide (Waas, 2022). SMARTreview functions by integrating a limited amount of data within the BIM environment. It then extracts

relevant quantities and relationships to analyse them against the rules outlined in the International Building Code (Clayton et al., 2013).

The International Building Code (IBC) describes the base requirements which are applied throughout the United States and various international governing bodies (Eastman et al., 2018). SMARTreview checks the Revit model by collecting data and forwarding it to a cloud based system to analyse it (Eastman et al., 2018). SMARTreview integrates directly with the BIM authoring tool and in that way designers can perform plan reviews as they work on the design (Clayton et al., 2013). Storing the rules in the cloud helps reduce processing work and allows users to receive new rules quickly (Eastman et al., 2018). The software generates a report which is linked with the Revit model, that highlights compliant and non-compliant elements by displaying the results through both drawings and text (Clayton et al., 2013; Eastman et al., 2018).

## Verifi3D

Verifi3D is a cloud based platform, developed by the Dutch company Xinaps, that performs real time checks and verifications of the designed model (Autodesk, 2025; Xinaps, n.d.-a). Verifi3D offers integration with various Common Data Environments (e.g. Autodesk Construction Cloud) and multiple issue trackers like BIMcollab (Xinaps, n.d.-a). The platform automates model checking and design coordination according to ISO 19650 standards (Autodesk, 2025). Verifi3D allows users to upload Revit models and IFC files to a browser for automated compliance checking (Autodesk, 2025). The platform offers a customisable reporting feature that compiles results from the verification tests into a report. Users can define the report format to suit their needs with export options including BCF, PDF, CSV and XLS (Day, 2019). In the beginning of 2025, Xinaps was acquired by Solibri and the platform was renamed to Solibri CheckPoint (Solibri, 2025).

### 2.3.3 Comparison of the different tools

The different software tools are compared based on five criteria, namely: 1) IFC compatibility, 2) Report generation, 3) Merge BIM files, 4) Direct edit possibility and 5) Rule-based checks. The comparison of the various compliance checking tools was conducted based on several sources. The insights gained from the exploratory interviews helped to establish the criteria used to compare the tools. The significance of these criteria for automated verification of requirements is further explained below.

The *IFC compatibility* is essential to ensure that models from different sources can be integrated to allow for an effective verification process. By supporting an open standard, IFC-compliant software enables interoperability between different BIM platforms to facilitate accurate and efficient automated requirement verification. Furthermore, *report generation* has been selected as a criterion because it plays a critical role in documenting and communicating verification results in a structured manner. Clear and well-structured reports help to ensure transparency and avoid misunderstandings (see Appendix A2.12).

The ability to *merge BIM files* is another important criterion, as large projects often involve multiple discipline specific models such as structural, electrical and mechanical models that are combined into a coordination model (see Appendix A2.11). Software that can merge multiple BIM files enables more advanced verification processes by allowing cross-disciplinary checks. In addition, the *direct edit possibility* criterion evaluates whether the software allows direct modifications to the model after an issue has been identified during verification. This feature helps designers to incorporate feedback efficiently and improves the overall workflow. Also, the ability to directly edit the model can facilitate the resolution of design changes, which often lead to rework (see Appendix A2.2).

Finally, the *rule-based check* criterion assesses the extent to which verification rules can be structured and implemented within the software. This capability is essential for automating the requirement verification process and to ensure that compliance checks are performed consistently. The results of the comparison of the different tools are shown in **Table 4** and further explanation is provided below. The colours in the table indicate if the software tools meet the respective criteria: the tool meets the criteria (✓), does not meet them (✗) or meets them under certain conditions (✓<sup>1</sup>).

**Table 4.** Comparison of different compliance checking software. Based on Kincelova et al. (2019). The numbers in parentheses correspond to the reference list.

	Solibri	Navisworks	BIM Assure	SMARTreview	Verifi3D
IFC compatibility	✓ (1)	✓ (2)	✓ (3)	✓ (4)	✓ (5)
Report generation	✓ (6)	✓ (7)	✗ (8)	✓ (9)	✓ (10)
Merge BIM files	✓ (11)	✓ (12)	✗ (8)	✗ (9)	✗ (10)
Direct edit possibility	✗ (13)	✗ (14)	✓ <sup>1</sup> (13)	✗ (9)	✗ (10)
Rule-based check	✓ (15)	✓ <sup>2</sup> (14)	✓ (13)	✓ <sup>3</sup> (9)	✓ (5)

<sup>1</sup>Only possible for non-geometrical parameters, <sup>2</sup>only clash detection possible and <sup>3</sup>possible with plug-in.

All of the tools are compatible with the IFC openBIM standard and allow these files to be imported (Autodesk, 2024; Invicara, 2017; Solibri, 2024; Solihin et al., 2020; Xinaps, n.d.-a). In terms of report generation, Solibri Model Checker, Navisworks, SMARTreview and Verifi3D can export a report that exchanges model based issues which can be shared among project collaborators (Autodesk, n.d.-a; SMARTreview, n.d.; Solibri, 2024; Xinaps, n.d.-b). This is something that is not possible with BIM Assure (Invicara, n.d.). With Solibri Model Checker and Navisworks, BIM files can be merged into one file, which is not possible for the other tools that were analysed (Autodesk, n.d.-b; Invicara, n.d.; SMARTreview, n.d.; Solibri, n.d.-a; Xinaps, n.d.-b).

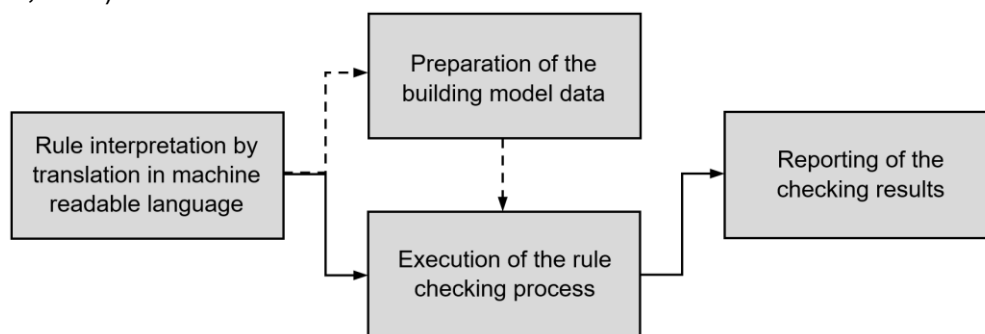
If a code checking tool detects non-compliance, direct modification of the model's properties based on feedback can enable faster corrections (Kincelova et al., 2019). The possibilities to do such a thing are very limited, but BIM Assure allows the direct modification of non-geometrical parameters, for example material properties (Invicara, n.d.; Kincelova et al., 2019). The other tools that were analysed do not provide the option to directly edit the model after detecting non-compliance with the requirements (Autodesk, n.d.-c; Preidel & Borrmann, 2018; SMARTreview, n.d.; Xinaps, n.d.-b).

Finally, automated rule checking refers to software that evaluates a design without changing it (Eastman et al., 2009). Design rules are initially defined by people and expressed in human readable formats such as written text (Eastman et al., 2009). Solibri Model Checker makes it difficult to create custom rules so most users depend on predefined rules provided by Solibri (Solibri, n.d.-b). BIM Assure executes rule-based checks throughout the design phase in the cloud (Preidel & Borrmann, 2018). With Verifi3D, design checks can be carried out according to ISO 19650 standards (Xinaps, n.d.-a). However, SMARTreview and Navisworks allow for limited rule-based checks. Navisworks focuses mainly on clash detection and model coordination, but its

checking capabilities are not that advanced as those in SMC (Autodesk, [n.d.-c](#)). SMARTreview has limited checking capabilities since it requires an add-on for Revit (SMARTreview, [n.d.](#)).

### 2.3.4 Challenges with automated requirement checking

The use of automated compliance checking not only improves compliance with regulations but also minimises the effort that is needed for verification (Preidel & Borrmann, [2018](#)). Automated code compliance checking offers significant potential benefits, but it also presents many challenges which make it a highly investigated topic (Nawari, [2018](#)). The process of automated compliance checking can be divided into four main parts (see **Figure 11**): 1) Rule interpretation by translation in machine readable language, 2) Preparation of the building model data, 3) Rule checking execution and 4) Reporting of the checking results (Eastman et al., [2009](#); Preidel & Borrmann, [2018](#)).

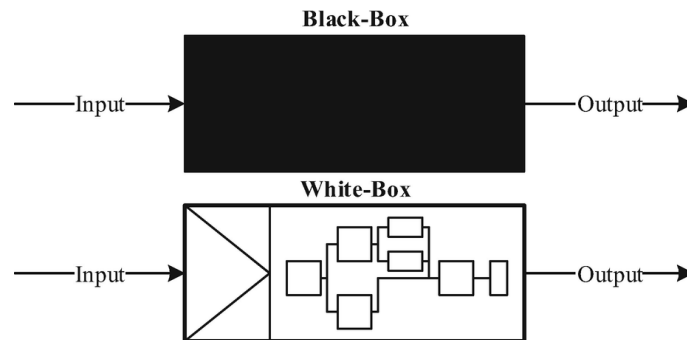


**Figure 11.** Simplified process of automated compliance checking. Based on Eastman et al. (2009).

#### Rule interpretation by translation in machine readable language

One of the most difficult challenges is converting human-readable text into code that computers can interpret (Preidel & Borrmann, [2018](#)). This converting process is the first essential step of any compliance review (Preidel & Borrmann, [2015](#)). Design regulations are initially defined by humans and expressed in written language and hold legal authority in building codes (Eastman et al., [2009](#)). The concept of converting oral or written language into digital form has been present since the beginning of computer science and remains a relevant subject across various fields (Preidel & Borrmann, [2015](#)). The aim is to accurately convert the content of standards and guidelines into binary code, but the varying ways in which information is presented, make standardising this process a significant challenge (Preidel & Borrmann, [2015](#)). There are two methods for the conversion of the contents that help with rule interpretation (Preidel & Borrmann, [2018](#)).

The simplest approach of conversion involves directly embedding the checking process into a predefined program routine (Preidel & Borrmann, [2018](#)). The translation here focuses on defining machine readable algorithms which remain hidden from the user (Preidel & Borrmann, [2018](#)). This approach emphasizes the automated checking process but fails to consider the involvement of the user (Eastman et al., [2009](#)). The checking process is hidden from the user with only the input and output data visible (Preidel & Borrmann, [2018](#)). This is referred to as the Black-Box method from the General System Theory by Von Bertalanffy ([1972](#)) as shown in **Figure 12**. The main benefit of this method is its low error rate which results from its closed nature and direct access to the code checking system's internal data (Preidel & Borrmann, [2018](#)).



**Figure 12.** Visualisation of Black-Box and White-Box approach. Adapted from Preidel and Borrmann (2015).

On the other hand, there is a method that makes the checking process itself visible for the user (Preidel & Borrmann, 2018). This is called the White-Box method where the user is incorporated into the processing, as can be seen in **Figure 12** (Preidel & Borrmann, 2015). Many current research projects adopt the White-Box approach by creating a computer language that aligns with the domain concepts and that is easy to use to allow it to manually translate regulations into a computer readable format (Preidel & Bormann, 2018). For transparency associated with the White-Box approach, the translated code should be understandable by both the machine and the user by converting the rules using a specific language (Preidel & Bormann, 2018). While more effort is needed to develop a system like this compared to the Black-Box approach, it offers benefits by keeping control over the outcome of each process step (Preidel & Bormann, 2018).

### **Preparation of the building model data**

Often the BIM model requires pre-processing because some information, needed to check regulations, is not directly included in the model and must be derived first (Preidel & Borrmann, 2018). The preparation of the model is necessary to ensure that it contains the required properties such as names and objects needed (Eastman et al., 2009). Before generating new model views, pre-checking is carried out to ensure that the required data for verification is present in the model (Eastman et al., 2009). It is important for the building model to be accurate for the checking process to produce reliable results (Preidel & Borrmann, 2018). Achieving full correctness of data standards requires the use of formally defined data structure (Beetz et al., 2009). It is challenging to formulate an universally applicable checking process for a specific data standard and this can only be accomplished through a pre-processing procedure that verifies and prepares the model data (Preidel & Borrmann, 2018). However, executing rules directly can introduce significant errors due to inconsistencies, missing information, contradictions or inaccuracies within the building model (Beetz et al., 2009).

### **Rule checking execution**

After the preparation of the building model, the rule checking process can be done. In the execution step, the rules are interpreted and applied based on the information provided by the digital building model (Preidel & Borrmann, 2015). The rule checking becomes relatively simple when the rules are translated in computer readable form with functions and that the required functions correspond with the building model's available data (Eastman et al., 2009).

### **Reporting of the checking results**

After the checking process, the outcomes need to be documented in a way that allows the user to understand the identified issue and take corrective action (Preidel & Borrmann, 2018). Also, the design parameters that comply with the required standards must be documented to ensure the check's completeness (Eastman et al., 2009). The identified issues can be shared through the BIM Collaboration Format or in a report and in modern practices the results are displayed graphically (Preidel & Borrmann, 2015; Preidel & Borrmann, 2018).

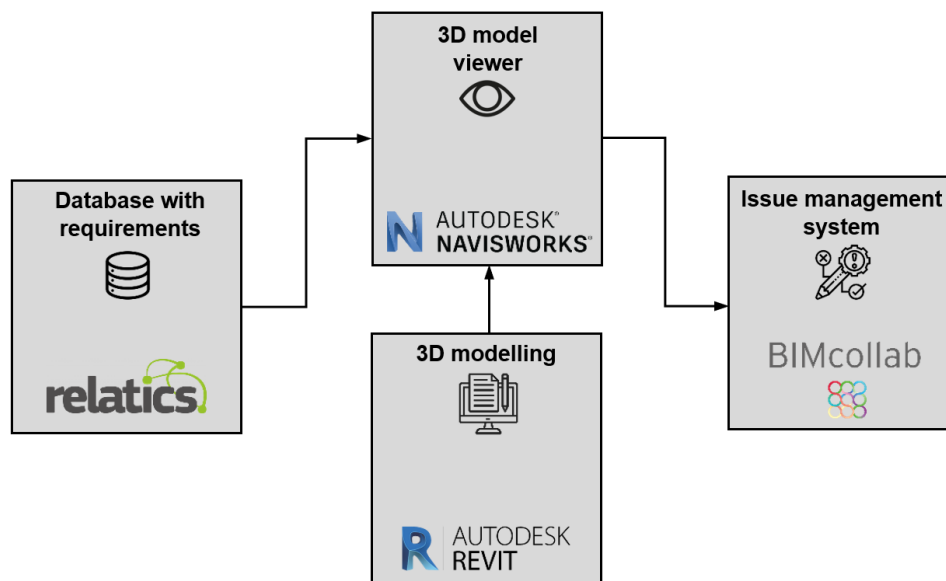


## Organizational challenges and limitations

Several of these challenges are also recognized within Witteveen+Bos. The paper by Preidel & Borrmann (2018) mentioned that one of the most difficult challenges for automated code compliance checking is converting written text into computer readable language. At Witteveen+Bos, efforts are being made to make requirements SMART, as some requirements are unclear or ambiguous to interpret. However, no specific mark-up language is used for this process (see Appendix A2.12 and A2.13).

Project requirements are provided by the client. Witteveen+Bos develops a verification & validation plan, which specifies how the requirements for the design will be verified. Not all requirements can be visually checked in a BIM model and some are, for example, verified through structural calculations. The verification of requirements therefore depends on a classification step, where a selection is made regarding which requirements can be checked within the BIM model (see Appendix A2.7 and A2.14).

Another challenge in automating the requirement verification is the fragmented workflow (see **Figure 13**). At Witteveen+Bos, the program Navisworks by Autodesk is used for verifying requirements within the BIM environment. The requirements are recorded in Relatics, which serves as a database for project requirements. In Relatics, requirements are stored as text in human written language, with the possibility of linking additional information such as the responsible person and verification method. Additionally, Witteveen+Bos uses BIMcollab as an issue management system, where issues identified during requirement verification in the BIM model can be logged (see Appendix A2.10 and A2.11).



**Figure 13.** Simplified workflow of Witteveen+Bos for manual requirement verification using BIM.

A further challenge in this process is that after visually verifying a requirement in the model, a design change might occur. As a result, a requirement that was initially met may no longer be satisfied within the design. Manually checking requirements is highly labour intensive, especially if design changes occur frequently. Currently, automation is used to a limited extent at Witteveen+Bos for requirement verification. This is mainly restricted to visual checks in Navisworks, with small-scale use of Dynamo scripts in the Revit design. With Dynamo, certain actions can be automated where possible. Creating a Dynamo script does require some time, but it can provide benefits if the tool can be applied multiple times within a project. However, such a tool is project specific and cannot be directly re-used in another project (see Appendix A2.9).

### 2.3.5 Findings

Various software tools exist for verifying requirements in BIM models with each offering different functionalities and facing diverse limitations. Navisworks and Solibri Model Checker are among the most commonly used tools for compliance checking. These tools assist in automating model verification by enabling clash detection and rule based checks. Solibri Model Checker is widely adopted due to its predefined rule sets, while Navisworks is often used for visualisation and coordination rather than advanced compliance checking. BIM Assure leverages cloud based rule execution, SMARTreview integrates compliance verification within Autodesk Revit and Verifi3D provides real-time model validation. Despite these advancements, most tools rely heavily on predefined rule sets that are difficult to modify, limiting their adaptability for project specific requirements. Additionally, interoperability challenges persist due to variations in how different platforms interpret IFC data.

The verification of requirements in a BIM model remains a complex and evolving process. One of the primary challenges is the translation of human-readable requirements into structured machine-readable formats. Existing compliance tools are generally limited to predefined rule sets, making it difficult to accommodate unique project specifications without extensive customisation. Another issue lies in the preparation of BIM models, as many lack the necessary attributes required for accurate verification. Missing or inconsistent data can lead to unreliable compliance results, making it essential to pre-process models before rule execution.

Beyond data-related challenges, fragmented workflows present significant inefficiencies. At Witteveen+Bos, requirement verification relies on multiple disconnected systems: Navisworks is used for visual verification, Relatics serves as a database for requirements, Revit is used as 3D modelling software and BIMcollab functions as an issue management platform. This separation creates inefficiencies and increases the need for manual coordination. Furthermore, the iterative nature of design verification adds to the complexity. When design changes occur, previously verified requirements may no longer be met, requiring additional checks which are very time consuming. While some automation exists through tools like Dynamo scripts, the project specific nature makes it difficult to scale or re-use across different projects.

## 2.4 Transforming requirements into computer code

According to Ghannad et al. (2019) the effective formalization of various norms, guidelines and requirements into a machine readable format is key for making the implementation of compliance checking software successful. The challenge is converting human readable requirements into a format that can be processed by computers in a way that enables their validation within a BIM model (Ghannad et al., 2019). Recent research has primarily focused on the application of Natural Language Processing (NLP) and Artificial Intelligence techniques to automate this translation process (Ghannad et al., 2019; Salama & El-Gohary, 2016; Zhang & El-Gohary, 2015).

Requirements often consist of complex and interconnected legal documents which can include contradictions and are frequently poorly organised. These documents also undergo regular revisions and updates, making their automated conversion into computable formats challenging. While NLP and AI techniques can help maintain an updated computable representation, the lack of an official data exchange standard remains a critical gap that needs to be addressed (Dimyadi & Amor, 2013).



Many studies have worked on creating a computer language to help experts turn legal requirement texts into rules that computers can understand (Ghannad et al., 2019). One example is the SMARTcodes project by the International Code Council in 2005. Later, AEC3 introduced a new method called RASE which organizes legal texts into rule objects (Ghannad et al., 2019). This section will discuss these methods among others and will look into language-based approaches to transform requirement texts into a structured format.

### 2.4.1 Natural Language Processing

Natural Language Processing (NLP) is a branch of Artificial Intelligence (AI) focused at making natural language text understandable for a computer so that the text can be processed by computers to make it easier for people to interact with them (Cherapas, 1992). In the field of infrastructure construction NLP has demonstrated its value by automatically extracting text data and transforming it into a structured format (Cai et al., 2020). NLP uses Machine Learning and Deep Learning to analyse human language by training models on large datasets to recognize patterns.

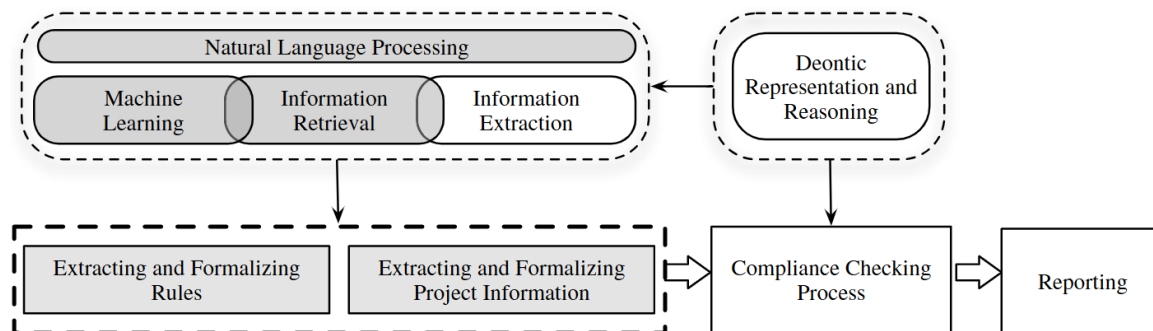
The NLP techniques can be categorized into shallow and deep methods. Shallow NLP involves partial sentence analysis or extracting specific information, whereas deep NLP focuses on analysing entire sentences to fully understand their meaning (Zouaq, 2011). Because of this difference in information extraction, shallow NLP tasks have achieved reasonable performances while deep NLP tasks continue to present challenges (Tierney, 2012). Although many studies have focused on NLP, it is clear that the full potential of the existing techniques has yet to be fully realised (Zouaq, 2011).

Natural Language Processing tasks can be approached in two ways, namely through a machine learning approach or by using a rule-based approach. An ML-based method employs machine learning algorithms to process text, while a rule-based method relies on manually defined rules (Zhang & El-Gohary, 2015). Rule-based methods demand more human input for creating rules but often result in better text processing performance (Crowston et al., 2010). The NLP technology can be used in combination with language-based approaches to transform requirements into a structured format. However, NLP should be seen more as a method and therefore cannot be compared to the languages covered in the following subsections (RASE, BIMRL and BERA).

#### Information retrieval

Information retrieval (IR) is one of the two primary mechanisms in NLP along with information extraction (IE). Information retrieval is described as the automated process of locating and retrieving relevant information within documents within a large collection of data (Manning et al., 2009). **Figure 14** shows the compliance checking processes with the extracting of regulatory documents into logical sentences, gathering project data, performing compliance checks and reporting the results (Salama & El-Gohary, 2016).

One of the simplest tasks in IR is document retrieval. In an IR process the user's information need is expressed through keywords which are submitted to the system. The system then searches its database to retrieve the most relevant results and returns them to the user (Li & Lu, 2016). More complex IR tasks include answering questions based on documents, databases or knowledge bases. Recent advances in deep learning have introduced new opportunities for IR. A key challenge in the field is integrating deep learning with traditional processing (Li & Lu, 2016).



**Figure 14.** Compliance checking processes. Adapted from Salama and El-Gohary (2016).

## Text classification

Text classification is a part of NLP that involves sorting text into one or more preset classes (Hassan & Le, 2020). Text classification is recognized as a subfield of information retrieval (Manning et al., 2009). The aim of text classification is to classify parts of documents into predefined labels for easier retrieval and further information extraction (Salama & El-Gohary, 2016). Also, text classification helps in identifying the categories to which a text belongs. Categories, often referred to as concepts or classes, are represented by labels (Manning et al., 2009).

Text classification can be divided into single-label text and multi-label classification where labels are assigned to text units after determining their relevance to particular categories (Hassan & Le, 2020; Salama & El-Gohary, 2016). Single label classification assigns one label per text sample whereas multi label classification is able to assign multiple labels to a single text (Hassan & Le, 2020). Single label classification can be further categorised into binary text classification which involves only two classes and multiclass text classification where more than two classes are included (Hotho et al., 2005).

## Information extraction

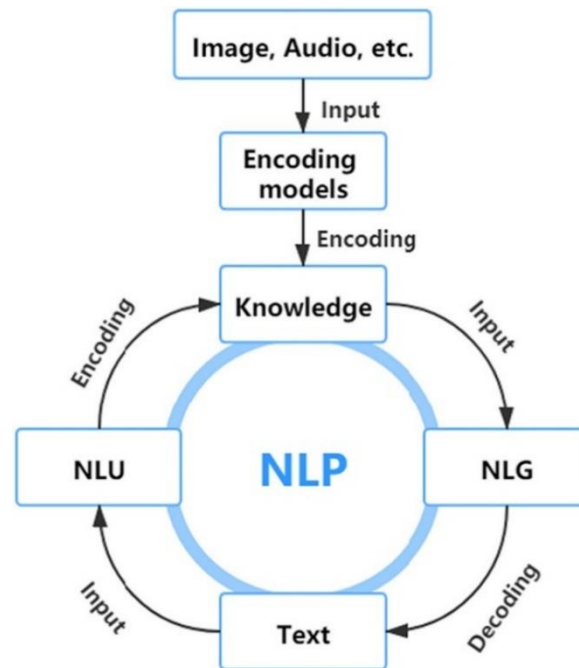
Information extraction is a core NLP mechanism focused on identifying and extracting data from unstructured sources (such as text, images, etc.) and organise it into a structured format (Salama & El-Gohary, 2016). This technology emerged to address the growing need for efficient processing of texts in specialized domains (Hobbs & Riloff, 2010). Information extraction systems are typically domain specific and focus on retrieving relevant information to a user's interest (Hobbs & Riloff, 2010). In the construction sector, where much of the data exists as unstructured text, IE plays a critical role in enhancing efficiency. Advances in NLP techniques over recent decades have significantly improved these capabilities (Wu et al., 2022).

Information extraction approaches can be categorized into rule-based methods and machine learning based methods (Kim & Chi, 2019). Rule-based systems rely on manually crafted patterns to extract specific information which offers high accuracy but requires substantial manual effort. Conversely, machine learning methods make it possible to automatically analyse text data by training algorithms to learn extraction tasks when sufficient training data is available (Kim & Chi, 2019).

## Subdomains of NLP

Pre-trained language models have made significant advancements in NLP tasks, but applying them to specialized domains remains a challenge (Liu et al., 2023). There are various subdomains of NLP as illustrated in **Figure 15**. This figure also shows the relationships between these subdomains.

One such subdomain, Natural Language Generation (NLG), focuses on producing natural language output from non-linguistic inputs (Mehra, 2024). The primary goal of NLG is to explore how computer programs can generate natural language texts from internal data (Semaan, 2012). Natural Language Generation is the opposite of Natural Language Understanding (NLU) or Natural Language Interpretation (NLI) as it translates meaning to text whereas NLU interprets text to derive the meaning (Semaan, 2012). The complexity of NLG is relatively easier in comparison to NLU, because it deals with the unpredictable nature of input language structure (Semaan, 2012).



**Figure 15.** Relation between NLP, NLG, and NLU. Adapted from Ding et al. (2022).

Natural Language Inference which is also referred to as textual entailment, is a key aspect of NLP that involves identifying the logical connection between two text segments (Mehra, 2024). This technique can be used to identify logical relationships between project documents, such as contracts and design specifications to detect any contradictions between them.

Within the construction industry, NLP applications, such as automatic compliance checking, have been gaining popularity in BIM (Ding et al., 2022). Despite this, the presence of NLG in the construction sector remains relatively limited compared to NLU (Ding et al., 2022). An example of NLG is its ability to automatically generate compliance reports from construction data, while NLU can for example interpret unstructured text from project documents to verify if design plans meet the required standards.

## Ethics in NLP

The paper by Strube and Pan (2023) describes that large pre-trained language models, such as GPT-3, serve as the foundation for numerous applications that are designed to process natural language. As these models predominantly rely on datasets created by humans the models often reflect human biases and prejudices (Strube & Pan, 2023). Natural language processing is a critical area within AI focused on transforming human language into a format that is understandable by machines to enable the analysis and processing of human linguistic data (Rafsanjani & Nabizadeh, 2023).





Given the information intensive nature of the architecture, engineering, and construction (AEC) industry, NLP has significant potential to extract valuable insights from various forms of human language. Application ranges from understanding stakeholder perspectives to verifying the compliance of designs with requirements of building codes (Rafsanjani & Nabizadeh, 2023). Natural language processing has evolved rapidly from a specialised discipline into a field of considerable industrial significance (Hovy & Prabhumoye, 2021). Despite its rapid adoption, the understanding of its unintended consequences has not kept pace with its widespread application (Hovy & Prabhumoye, 2021).

Research highlights the potential misuse of NLP technologies for harmful purposes, such as breaching privacy, suppressing dissent or profiling (Hovy & Prabhumoye, 2021). Incorporating human centred AI into the AEC industry requires good security measures. Advanced technologies in this field are attractive targets for cyberattacks and unauthorized access to sensitive data can harm privacy, reduce trust and make it harder to use these technologies in the construction industry (Hovy & Prabhumoye, 2021).

### 2.4.2 RASE language

The RASE mark-up language, originally developed as part of the SMARTcodes project, is currently supported by the company AEC3 (Dimyadi & Amor, 2013). The initiative was designed to create computable rules through user-friendly tools to enable individuals without programming experience to efficiently tag and organize requirement texts (Ilal & Günaydın, 2017). The RASE framework is a semantic driven methodology to convert human written text into a computer readable format that can be integrated into BIM models using the IFC format (Hjelseth & Nisbet, 2011). The RASE methodology is built upon a mark-up system that uses four operators to extract the meaning of the requirement texts (Hjelseth & Nisbet, 2011; Ilal & Günaydın, 2017). These operators include Requirement, which specifies the conditions that must be met; Applicability, which identifies the components to which the requirements apply to; Selection, which defines criteria for applying the rule to particular elements; and Exception, which outlines the conditions where the requirement does not apply (Ilal & Günaydın, 2017).

This approach allows even complex requirement text to be structured and broken down into simple components (Preidel & Borrmann, 2018). The RASE framework is used for the interpretation of normative text by having three methods (Hjelseth & Nisbet, 2011). First, clear text can be directly translated into RASE operators. Second, vague text needs to be transformed by rewriting it to clarify its meaning. Finally, overly general statements that cannot be modelled accurately are flagged for manual interpretation by experts. By using this approach RASE improves efficiency by reducing the time needed to process requirements (Hjelseth & Nisbet, 2011). The model checker demands the applied requirements to be interpreted in an unambiguous manner (Hjelseth & Nisbet, 2011). A color-coded system within the mark-up language highlights the connection between the operators and the requirement text, as can be seen in **Figure 16** (Hjelseth & Nisbet, 2011).

			
Requirement (blue)	Applicability (green)	Selection (red)	Exception (orange)

**Figure 16.** RASE mark-up language operators. Based on Hjelseth and Nisbet (2011).

### 2.4.3 BIMRL

The BIM Rule Language (BIMRL) is a specialised language designed for automating the process of rule checking in BIM (Solihin & Eastman, 2016). BIMRL was created to simplify and standardise the process of rule checking with a language-based approach, but it became clear that the challenges involved are more complex than simply formulating the rules (Dimyadi et al., 2016). The BIM Rule Language supports spatial operation in a building model with geometry and offers a complete environment for data, rule definition and execution (Solihin & Eastman, 2016). The rule language offers several common functions (e.g. clash detection) that can be used to define BIM rules (Solihin, 2016).

BIMRL supports extensions through plug-ins that make it possible to add new functions to the language without changing it (Solihin, 2016). Besides, BIMRL has been developed to provide efficient querying of IFC data to support multiple geometric representations (Dimyadi et al., 2016). Highly complex rules can be structured as a sequence of interconnected rules (Solihin, 2016). The language can function as an independent IFC engine to support a variety of software interfaces that need BIM data for any application (Dimyadi et al., 2018). The BIM Rule Language is created to manage BIM-related rules and requires a basic programming knowledge (Solihin et al., 2019).

### 2.4.4 BERA language

The Building Environment Rule and Analysis (BERA) language is developed to interact with BIM models intuitively in order to define and evaluate rules for assessing the arrangement of building elements (Lee, 2011). The language is known for its user friendly readability and the ability to directly access information within a BIM model (Preidel & Borrmann, 2018). Focusing on the creation of building objects, the BERA language is developed to incorporate the properties and relationships between them (Lee et al., 2015). BERA is a domain specific language that is designed to handle BIM models in a way to effectively define, analyse and verify rules (Nawari, 2018). BERA language is designed to be user friendly which allows users to use it without knowledge of general purpose languages that are typically used in the development of BIM software (Lee et al, 2015).

The BERA language is closely related to building information models by using the IFC format (Lee et al., 2015). Since the BERA language functions within a building model, the initial step is defining the BERA Object Model (BOM), followed by creating rules using the defined objects (Lee et al., 2015). The BERA language makes use of BERA code to write the rules (Lee, 2011). From there the BERA Listener processes input and transforms it into a defined data structure format (Lee et al., 2015). This initial stage of language recognition is managed by the BERA listener (Lee, 2011). The BERA Object Model is a semantic set of data that is interpreted and processed through predefined rules (Lee et al., 2015). It holds the details of a building model, enabling more efficient analysis and execution. While both IFC and BOM contain similar building model information, the BERA Object Model's format is more suitable for rule-based checking compared to IFC (Lee et al., 2015).

### 2.4.5 Comparison of the different methods

In Subsection 2.3.4, it is explained that the rule checking process is divided into four parts, where this subsection focuses specifically on the first step: rule interpretation through translation into machine-readable language. As a rule checking system needs a method to define rules, it follows that such a system will be language based (Solihin et al., 2019). The methods described in the previous subsection are compared here to see how they relate to each other.

Although comparing the different languages can be challenging or even impossible due to the lack of examples for direct comparisons, available references are used to classify each language appropriately (Solihin et al., 2019). For instance, one such reference comes from the Regulatory Room initiated by buildingSMART on automated rule checking (buildingSMART, 2017). Specifically, this initiative seeks to create a platform for open discussion among users, with the aim of promoting an open BIM-based approach that will result in shared frameworks or templates (Preidel & Borrmann, 2018). The buildingSMART platform applies various methods for automated compliance checking to a part of the Korean Building Act providing a useful comparison (Solihin et al., 2019).

Due to the complexity of the rules, it is challenging for an application to offer a solution. A well-defined language-based method could create a platform for combining building blocks into more complex rules, but the complexity of the language may reduce its ability to handle simpler rule definitions (Solihin et al., 2019). This concept is referred to as Domain Specific Language (DSL). In particular, a DSL typically simplifies the translation of codes by offering a more structured approach to make it easier to read compared to the lower level languages used by computers to check a code (Dimyadi et al., 2020). BIM rule languages are usually classified as DSLs since they are designed specifically for building information and related requirements (Solihin et al., 2019).

To compare the different languages six criteria are used, namely: 1) Standard for rule definition, 2) User-friendliness and accessibility, 3) Language expressiveness, 4) Efficiency and scalability, 5) Level of maturity and 6) Openness and interoperability. The significance of these criteria for automated verification of requirements is further explained below. Each method has been assigned a score of low, medium, high or not applicable for each criterion. The aim of comparing the different methods is to find out which of these is the most effective for translating human-readable requirements into a structured format. The comparison results for each language and criterion are shown in

**Table 5.**

### **2.4.5.1 Standard for rule definition**

The existence of a standard schema for defining rules is a standardized way to represent computable forms of requirements (Ghannad et al., 2019; Solihin et al., 2019). A well-defined standard ensures that requirements can be interpreted consistently by different software tools. This reduces the risk of misinterpretation or implementation errors in automated compliance checking (see Appendix A2.7). In the context of automated requirement verification, having a standard for rule definition enables these rules to be represented in a machine-readable format that can be systematically processed. This improves repeatability and accuracy during the verification of requirements in BIM models. Not having a standard schema can result in inconsistent rule definitions across different projects which creates inefficiencies when embedding automated verification processes in the BIM workflow (see Appendix A2.12).

Among the compared rule languages, only the RASE language meets this criteria, as it uses a formal schema specifically for BIM rule definition. The RASE language structures requirement texts using four operators: Requirement, Applicability, Selection and Exception to ensure that requirements are precisely processed in a logical manner (Hjelseth & Nisbet, 2011). In contrast, both the BIMRL and BERA languages do not use a standardised schema for rule definition making this criteria not applicable for them (Lee, 2011; Solihin, 2016; Solihin et al., 2019).

### **2.4.5.2 User-friendliness and accessibility**



















Given the varying complexity of BIM rules, some form of programming is necessary to address the complex rule requirements (Solihin et al., 2019). The criteria *user-friendliness and accessibility*







looks at how easily requirements can be translated into computer code in a way that is intuitive for the user. This is important for the automation of requirement verification as it enhances usability and minimises errors.

The RASE language uses a color-coded system to visually link operators with the requirement text, making their connection clear and structured (Hjelseth & Nisbet, 2011). AEC3, the company behind RASE, has developed a checking engine that directly processes the RASE content and eliminates the need for any intermediary format (buildingSMART, 2017). Similarly, the BERA language is relatively easy readable by humans and offers direct access to the BIM contents (Preidel & Borrmann, 2018). This allows users to work effectively without needing in-depth knowledge of general purpose languages that are typically used in BIM software development (buildingSMART, 2017; Lee, et al., 2015). In addition, the BIMRL language is developed to work with BIM rules while requiring minimal programming knowledge (Solihin, 2016). The language is both highly expressive and user-friendly (Solihin et al., 2019).

**Table 5.** Comparison of languages. Based on Solihin et al. (2019).

Languages	Standard for rule definition	User-friendliness and accessibility	Language expressiveness	Efficiency and scalability	Level of maturity	Openness and interoperability
RASE						
BIMRL						
BERA						

 Low
  Medium
  High
  Not applicable

### 2.4.5.3 Language expressiveness

The level of expressiveness of a language is important to determine how well it can define a wide range of rules. A less expressive language may be difficult to use and become impractical for handling complex requirements (Solihin et al., 2019). In the process of automated requirement verification, a highly expressive language ensures that rules can accurately represent practical requirements, which reduces the risk of misinterpretation and non-compliance. Not all requirements are always easy to interpret, which can lead to misinterpretations (see Appendix A2.7 and A2.12). This shows the importance of *language expressiveness*, as a more expressive language helps minimize ambiguity and ensures that requirements are clearly understood.

Looking at RASE, the automatic creation of logical statements is reliable, despite the fact that badly written requirements need extra attention (Hjelseth & Nisbet, 2011). The RASE language allows any requirement to be structured using its standard schema, which makes its language expressiveness high (buildingSMART, 2017). Moreover, BIMRL is developed to handle BIM specific rules with minimal programming skills required (Solihin et al., 2019). Its ability to process a broad range of rules makes it both very flexible and compact (buildingSMART, 2017; Solihin et al., 2019). On the other hand, the BERA language is capable of expressing a wide range of rules and conditions through its semantic structure (Lee, 2011). It supports several commands that

enables users to easily perform rule-checking on specific components (Lee et al., 2014). However, its level of expressiveness is moderate, as it lacks the advanced flexibility and integration capabilities seen in more mature languages.

#### **2.4.5.4 Efficiency and scalability**

The *efficiency and scalability* evaluates how efficient each method translates human readable requirements into a structured format. It also considers how well the method handles increased volume or complexity to ensure consistent performance as the task grows. This criteria is selected given the importance of it for requirement verification, a process that can take up to one hour per requirement manually (see Appendix A2.18).

The RASE language has proven to be an effective method for capturing requirements, demonstrating up to 15 times greater efficiency compared to procedural coding, according to an analysis by a client of AEC3 (buildingSMART, 2017). The development of rules is based on the interpretation of normative text, an area in which RASE excels and provides efficient support (Hjelseth & Nisbet, 2011). The BIM Rule Language uses a relational database schema to enable efficient access to building data using the SQL standard (buildingSMART, 2017; Dimyadi et al., 2016). The language's adaptable and expandable nature allows for a wide range of rules to be created (buildingSMART, 2017). Considering BERA, that uses scripting language, enables rule verification in an efficient and flexible manner (Lee et al., 2014; Solihin, 2016). It has the potential for extensibility, but this has not yet been fully developed (Solihin, 2016).

#### **2.4.5.5 Level of maturity**

The maturity level is assessed based on how long the language exists, its widespread adoption, the frequency of updates and maintenance and the number of implementations it has been used in (Solihin et al., 2019). The *level of maturity* was chosen as a criteria because the effectiveness of rule language depends on its reliability and ongoing development. The criteria contributes to requirement verification by ensuring the language is stable and continuously updated to make it more dependable for automating the verification process.

### **RASE**

The RASE language originates from the SMARTcodes project, which was initiated by the International Code Council (ICC) in 2006. At that time, the mark-up language was far ahead of any other system (buildingSMART, 2017). However, after the ICC stopped its development in 2010 due to a lack of funding, the company AEC3 took over and continued to develop the RASE language (Dimyadi & Amor, 2013; Preidel & Borrmann, 2018). Despite this, the RASE language continues to be used as demonstrated by Al-Turki et al. (2024), who explored the automation of RASE for regulatory compliance using Large Language Models. Similarly, Mendonça et al. (2020) examined the potential of integrating RASE with Solibri Model Checker. Further efforts to advance the language include a presentation at the European Conference on Computing in Construction last year (Nisbet et al., 2024). The continued development, application and integration of RASE in various studies over nearly two decades demonstrate its high level of maturity within the field of automated compliance checking.

### **BIMRL**

In 2015 the BIM Rule Language (BIMRL) was developed as part of a PhD thesis at the Georgia Institute of Technology (Solihin, 2016). The development of ePlanCheck (see Subsection 2.3.1) has had a major impact on shaping the concept of BIMRL (buildingSMART, 2017). The language is specifically designed to automate requirement verification in the BIM environment (Solihin & Eastman, 2016). BIMRL is developed using Structured Query Language (SQL) which is a programming language designed for managing data within a database management system



(Solihin, 2016). The BIM Rule Language is referenced on a small scale, indicating that its widespread adoption remains limited. Examples include the paper by Dimyadi et al. (2016) which explores integration of BIMRL into a computer-aided compliance audit framework and the study by Parsamehr et al. (2023), which discusses BIM-based model checking to enhance building occupant safety. While its potential is clear, the relatively limited references and adoption suggest that BIMRL is still in the process of gaining broader implementation, positioning it at a medium maturity level.

## **BERA**

The BERA language was developed in 2010, also as part of a PhD dissertation at the Georgia Institute of Technology (Lee, 2011). Originally designed as a domain-specific programming language for model checking, it has received minimal support and updates (Sydora & Stroulia, 2020). The usage of BERA is poorly documented, and most papers discussing the language, such as Nawari (2018) and Preidel & Borrmann (2018), focus on comparisons rather than actual implementation. Detailed insights into BERA mainly come from its developer, including a paper by Lee et al. (2015) on the implementation process and another by Lee & Eastman (2019) on offsite construction design analysis using BERA. Despite these efforts, its practical implementation remains limited, highlighting its low maturity in the field of automated compliance checking.

### **2.4.5.6 Openness and interoperability**

The *openness and interoperability* of a language is important as it allows the use of available tools and ensures it will last over time because of the open specifications (Solihin et al., 2019). In terms of requirement verification, the openness and interoperability of a language makes it easier to process requirements automatically. The process of requirement verification in BIM is fragmented with different software used throughout the process. A BIM rule language with high openness and interoperability contributes to making the process easier.

Looking at the RASE language, it ensures interoperability with any XML-based technology by using XHTML for its marked-up documents. This allows integration with different regulatory and compliance systems and demonstrates a high level of openness (buildingSMART, 2017). Similarly, BIMRL is built based on IFC and SQL and is open source which promotes both interoperability and openness (buildingSMART, 2017). Finally, the BERA language is developed to be open-ended so it can cover the entire set of building objects (Lee, 2011). However, the language is largely based on Java, which can be challenging for those without programming experience and may make it less transparent (Sydora & Stroulia, 2020).

## **2.4.6 Findings**

When comparing the various languages for translating written requirements into a structured format, it is important to evaluate them based on the defined criteria. Each language has its strengths and weaknesses in these areas, which must be considered when determining the most effective method. The RASE language stands out in terms of standardization, language expressiveness, and interoperability. It uses a well-defined schema and has been shown to efficiently process requirements through its color-coded system, making it easy for users to understand and apply. Furthermore, its XML-based format ensures high openness with other systems.

BIMRL also demonstrates strong performance in terms of efficiency, language expressiveness, and scalability, thanks to its relational database-based structure and open source nature. The BIM Rule Language is user-friendly, requiring minimal programming knowledge and has potential for further development in automated compliance checking. However, its relatively limited

adoption indicates that it is still gaining traction within the industry. BERA, although open-ended and capable of expressing a broad range of rules, lacks widespread adoption and support. Its reliance on Java programming can make it less accessible for users without programming experience which hinders its transparency and ease of use.

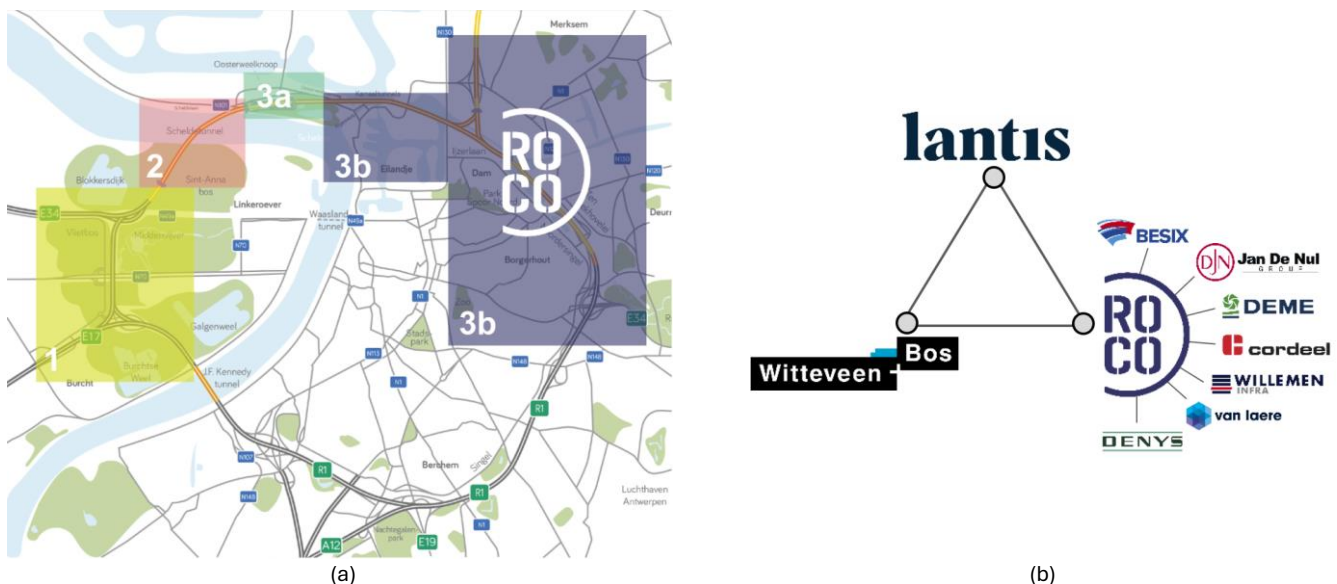
In conclusion, the most effective method for translating human readable requirements into a structured format among the analysed languages is the RASE language. Its strong standardisation, user-friendly design, high language expressiveness and interoperability make it a good choice for automated compliance checking. However, BIMRL also shows promise and may become more effective as it matures, while BERA still faces significant hurdles in terms of efficiency, scalability and user accessibility.

# 3. Development phase

## 3.1 Case study

The case study central to this thesis is the Oosterweel connection project that is being developed in Antwerp, Belgium. The aim of the Oosterweel connection is to complete the highway around the city of Antwerp and turn it into a ring road (Oosterweelverbinding, [n.d.](#)). Parts of the project involve the construction of tunnels and the partial underground placement of the ring road in order to create harmony between the ring and its surroundings (Oosterweelverbinding, [n.d.](#)). In this way, road traffic is moved underground, freeing up significant space for the development of parks, (play)squares and cycle paths (Lantis, [n.d.](#)). The goal of the project is to contribute to a healthier and more pleasant city and region for living, working and recreation (ROCO, [n.d.](#)). In addition, the Oosterweel project is intended to reduce traffic congestion and accidents by diverting port traffic away from the ring road (De Grote Verbinding, [n.d.](#)).

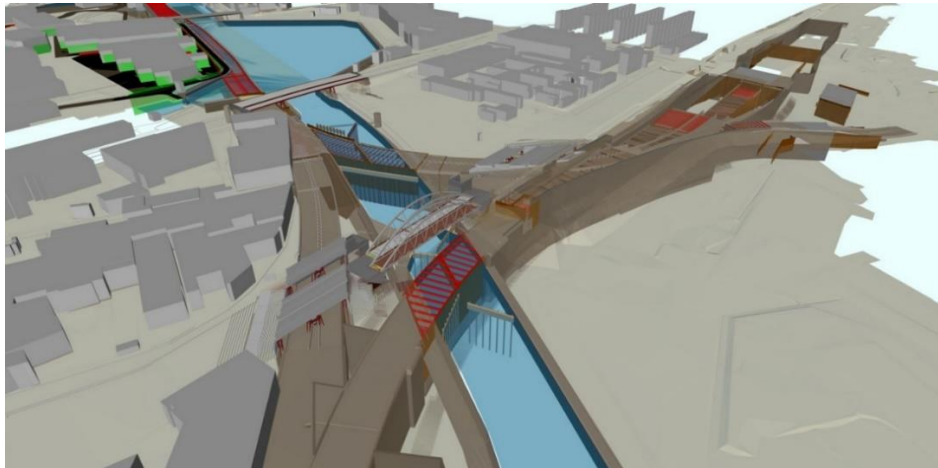
Given the complexity and large scale of the Oosterweel connection, the project has been divided into four different subprojects (see **Figure 17a**). The contractor ROCO is responsible for the largest section (section 3b) on the right bank of the city of Antwerp. The work on this section includes the construction of the Kanaalzone tunnels to complete the ring and the demolition of the Merksem viaduct to create a highway with a lowered road profile. The contractor consortium ROCO consists of companies such as Besix, Jan de Nul and DEME among others, as shown in **Figure 17b**. The project is being carried out on behalf of the client Lantis and Witteveen+Bos has an advisory role and is responsible for the engineering work of the project (Oosterweelverbinding, [n.d.](#); Witteveen+Bos, [n.d.](#)).



**Figure 17.** (a) Overview of Oosterweelverbinding project and (b) involved project parties. Based on Oosterweel ([n.d.](#)), Witteveen+Bos ([n.d.](#)) and ROCO ([n.d.](#)).

Due to the size and technical complexity of project area 3b, it has been divided into seven subareas: 1) Kanaalzone tunnels, 2) Vork, 3) R1-North, 4) R1-East, 5) Underground crossing of the Albert Canal, 6) Bypass and 7) Schijnkoker (ROCO, [n.d.](#)). The case study is framed to the Vork subproject as part of project area 3b of the Oosterweelproject (see **Figure 18**). Dividing the project into the seven subareas allows for a phased execution of the work. To effectively manage the complexity of processes within the project, the Integral Project Management (IPM) model is

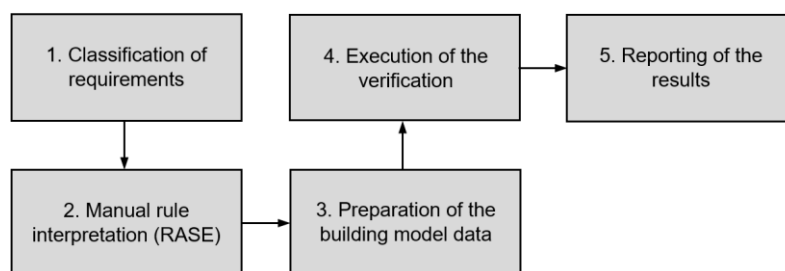
applied. This means that the project team consists of five key components: project management, project control, stakeholder management, technical management and contract management (ROCO, [n.d.](#)). Moreover, the Oosterweelverbinding project uses the innovative NEC4 contract form, which is highly focused on collaboration (see Appendix A2.6). Originally developed in the UK, this type of contract allows for an integrated and solution-oriented approach within the project. The NEC4 contract has been rarely used within the European Union (EU), making its implementation in Belgium a first (Appendix A2.6; ROCO, [n.d.](#)).



**Figure 18.** Partial overview of subproject the 'Vork' with crossing of the Albert canal. Adapted from COB (2018).

## 3.2 Implementation

The workflow for automated requirements verification consists of several fragmented steps. This section looks in detail at the steps required to automate requirement verification in a BIM model where the case study will be used as a focus point. The process is distinguished into five steps: classification of requirements, manual rule interpretation, preparation of the building model data, execution of the verification and the reporting of the results (see **Figure 19**). These steps in the process are structured with the subsections below. The steps also correspond with the workflow described in Section 3.4.



**Figure 19.** Simplified workflow of the semi-automated requirement verification steps.

### 3.2.1 Classification of requirements

The classification step consists of identifying and distinguishing which requirements can be verified in a BIM model and for which requirements this is not possible. For example, some requirements must be verified through structural calculations by a structural engineer, while other requirements can be verified in a BIM model with a visual inspection (see Appendix A2.8). The contractor has the complete freedom to determine the verification method. However, if the client suggests a specific verification method, it is not recommended but mandatory to verify the requirement that way (see Appendix A2.1). The classification step is important for the contractor to determine which requirements can be verified in a BIM model. In **Table 6**, two requirements are given as an indication to show what kind of requirements are geometrical requirements and

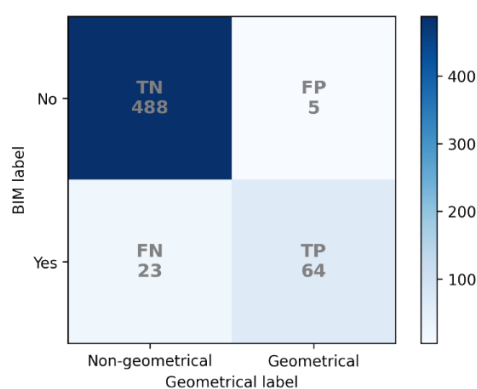
which are not. Both the original requirement text in Dutch and the direct translation into English are included in the table. Verification of geometrical requirements is generally possible directly within a BIM model. Verification of non-geometrical requirements typically requires additional calculations or analyses that go beyond pure geometry. However, if BIM is defined broadly to include integrated simulation and analytical tools, some non-geometrical requirements can also be verified within the BIM environment.

**Table 6.** Example of geometrical and non-geometrical requirement from the case study.

	Geometrical requirement	Non-geometrical requirement
English	All foundation bases must be located at least 1 m below the ground surface to prevent any adverse effects from frost heave.	For structural works, the minimum concrete strength class required is C30/37 and maximum C90/105.
Dutch	Alle funderingsaanzetten dienen zich tenminste 1 m onder het maaiveld te bevinden om geen nadelige gevolgen van opvriezing te ondervinden.	Voor kunstwerken is de minimale sterkteklasse C30/37 en maximaal C90/105.

The given example of the geometrical requirement can be verified in a BIM model by checking that the distance between the foundation bases and ground level is at least 1 meter. However, it is important here that the requirement is interpreted correctly (more on this in the following subsection). For example, it should be clear what the foundation bases are which can be done by linking the objects to the requirement in the model directly (see Subsection 3.2.3). On the contrary, the given example of the non-geometrical requirement cannot be verified in a BIM model but must rather be demonstrated through a structural calculation. The screening between geometrical requirement and non-geometrical requirement is a step that is performed based on the requirement package from the contract. This analysis requires a lot of manual work and has to be performed again for each project.

As a result, the possibilities for automating this process were explored. A Python script is created using Natural Language Processing to classify textual requirements into two categories: geometrical requirements (1) and non-geometrical requirements (0). A machine learning model was trained using the scikit-learn package to make these predictions (see Appendix B). The input data consisted of various geometrical and non-geometrical project requirements. The model was tested on more than 600 requirements and the results were compared to their actual categories as defined by the company for validation. The results of the classification of the dataset are shown through a confusion matrix in **Figure 20**. The figure provides insight into the performance of the model by showing the number of correct and incorrect predictions for each requirement.



**Figure 20.** Confusion matrix of DataFrame with project requirements.

The confusion matrix distinguishes between true negatives (TN), true positives (TP), false negatives (FN) and false positives (FP). The plot shows that the model correctly classified 488 requirements as non-geometrical, representing the true negatives. Additionally, 64 requirements were accurately identified as geometrical requirements, which are the true positives. However, the model also made some errors where 5 non-geometrical requirements were incorrectly predicted as geometrical, referred to as false positives. Also, 23 geometrical requirements were mistakenly predicted as non-geometrical, known as false negatives. Overall, the model achieves an accuracy of 0.95 which means that it correctly categorizes 95% of all requirements. The requirements that have expired or are no longer applicable to the project have been filtered out and therefore do not contribute to the accuracy score.

Further analysis showed that the deviations that occurred can be categorised into three groups. For the first group, the verification method from the requirement database is stated as ‘document inspection’, where the word BIM was not mentioned in the requirement which could have misled the model. The second group did not mention any specific verification method and the word BIM was not used in the requirement text. In these cases, the model failed to classify the requirement correctly as geometrical or non-geometrical. For the latter group, the requirement was no longer applicable for the project (based on the verification method in the database) and has therefore been removed from the system engineering package. The geometrical requirements from the second group were included in the subsequent steps.

### Requirement categorisation by rule classes

Looking at the verification of requirements, depending solely on visual inspection is no longer an effective method to ensure the compliance of BIM models (Solihin & Eastman, 2015). There is a growing need for automated checking processes that use clearly defined requirements and require little user involvement. As simpler requirements become automated, experts will increasingly focus on more complex tasks, which involve requirements that are harder to define and cannot be easily automated (Nawari, 2012). Since the RASE methodology, earlier identified as the most effective approach for translating requirements, focuses only on the semantic structure, it must be combined with an additional method for executing the rules (Solihin & Eastman, 2016). Not all requirements can be verified in a BIM model using the same approach. According to Solihin and Eastman (2015), requirements for BIM verification can be grouped into four classes based on their computational complexity as can be seen in **Table 7** (Hjelseth, 2016; Solihin et al., 2020).

**Table 7.** Different classes for automated BIM rule verification. Based on Solihin and Eastman (2015).

Rule classification	Description
<b>Class 1</b> Rules based on a single or small number of explicit data	<ul style="list-style-type: none"> <li>General BIM file structure rules: “Model should have components” checks ifcElement</li> </ul>
<b>Class 2</b> Rules that require simple derived attribute values	<ul style="list-style-type: none"> <li>Component properties rules: “free area in front of components rules” checks</li> <li>Clash detection: finds overlapping elements using basic geometry</li> </ul>
<b>Class 3</b> Rules that require extended data structure	<ul style="list-style-type: none"> <li>Checks that require extensive analysis of spatial relationships between elements</li> </ul>
<b>Class 4</b> Rules that require a ‘proof of solution’	<ul style="list-style-type: none"> <li>Used when the model must demonstrate a solution</li> <li>Simulation based solutions</li> </ul>



The categorisation from this framework is also applied in this research to provide a targeted approach for carrying out the verification within the BIM model. The rule classes do not reflect their importance, as each class serves a valid purpose in the process of automated requirement verification, though they differ in application and scope (Solihin & Eastman, 2015). The four classes that are part of the framework are as follows: Class 1 includes checks based on explicit data; Class 2 involves checks based on simple derived attribute values; Class 3 consists of checks based on an extended data structure and Class 4 covers checks and corrective actions or solutions (Hjelseth, 2016). The difference between the classes and examples of their application are further explained below.

The first class involves verifying requirements by using clearly defined attributes and element references that are directly available within the BIM model data (Solihin & Eastman, 2015). Common applications of this class of requirements are entity attribute checks and basic requirement verifications. At this level, the required information can be directly accessed from the BIM model, either through the entities or through their properties which allows simple navigation of the relationships needed for requirement verification. When looking at Class 2, verifications are based on a single value or a limited number of derived values without generating new data structures. For this class the verification of requirements often relies on implicit relationships, which are typically not explicitly represented in the model. Therefore, the values need to be derived from the basic BIM model data and relationships within the model. An example of Class 2 checks is clash detection where conflicts between different elements in the BIM model are identified (Solihin & Eastman, 2015).

Class 3 of requirements involves expanding the data structure to represent higher level semantic conditions of building data (Solihin & Eastman, 2015). This type of requirements is commonly used when checking regulations that contain more complex requirements. Finally, Class 4 can be seen as a collection of requirements that are not limited to determining compliance or non-compliance, but instead focus on demonstrating that a valid solution has been achieved. In other words, the requirements focus on how the building model demonstrates compliance rather than just meeting specific criteria. Class 4 does not introduce complexity but requires more semantic information compared to the other three classes (Solihin & Eastman, 2016). Furthermore, requirements that are usually assigned to a specific class may shift to a different one when the problem is viewed from another perspective. This is particularly the case for Class 3 requirements which can involve into Class 4 (Solihin & Eastman, 2015).

Based on the requirement classes, four examples from the dataset are given for each class. **Table 8** shows these examples with requirement text in combination with the corresponding RASE mark-up (more on RASE in Subsection 3.2.2). The first example in the table aligns with Class 1, as the requirement is based on a small set of explicit data. The minimum pipe diameter is specified as 200 mm which represents a clear threshold and in combination with the slope, the requirement can be verified in the model directly. The example of the Class 2 requirements also includes a clear threshold (at least 1 meter below ground level), but also demands semantic understanding of the requirement's underlying purpose. This requirement can be categorised in Class 2, as it is based on derived attribute values and can be verified through a geometric analysis looking at the dimensions within the model.

Moreover, the example of Class 3 involves extended data structure and requires detailed information beyond simple numeric checks. Specifically, the requirement refers to an attached document which suggests that interpretation of the design and spatial relationships are necessary. This makes it harder to automate as it demands a more complex understanding. The last example is categorised as Class 4 because it suggests corrective actions to ensure a conflict-

free space. It does not provide clear, measurable thresholds but implies that human intervention may be needed to resolve conflicts or to come up with a solution.

**Table 8.** Requirement text classification using rule classes by Solihin and Eastman (2015) for automated BIM checking with RASE mark-up.

	Type	Requirement texts
Class 1	Dutch	De buisdiameter van de ingestorte riolering in de verkeerskokers dient afgestemd te zijn op de langshelling en de gevraagde transportcapaciteit, maar is minimaal 200 mm.
	English	The pipe diameter of the embedded sewer system in the traffic ducts must be adjusted according to the longitudinal slope and the required transport capacity, but it is at least 200 mm.
	RASE	<R>The <a>pipe diameter</a> of the <a>embedded sewer system</a> in the <a>traffic ducts</a> must <r>be adjusted according to the longitudinal slope and the required transport capacity</r>, but <r>it is at least 200 mm.</r></R>
Class 2	Dutch	Alle funderingsaanzetten dienen zich tenminste 1 m onder het maaiveld te bevinden om geen nadelige gevolgen van opvriezing te ondervinden.
	English	All foundation bases must be located at least 1 m below the ground surface to prevent any adverse effects from frost heave.
	RASE	<R>All <a>foundation bases</a> must <r>be located at least 1 m below the ground surface</r> to <r>prevent any adverse effects from frost heave.</r></R>
Class 3	Dutch	Een tunnelconstructie dient de van toepassing zijnde verkeerskoker-, vluchtkoker- en dienstkoker-ruimtes conform gekoppeld document te realiseren.
	English	A tunnel construction must realize the applicable traffic, escape- and service tube spaces in accordance with the linked document.
	RASE	<R>A <a>tunnel construction</a> <r>must realize</r> the <a>applicable traffic</a>, <a>escape-</a> and <a>service tube spaces</a> <s>in accordance with the linked document</s>.</R>
Class 4	Dutch	Een vluchtkoker in een tunnelconstructie dient conflictvrij ruimte te bieden aan de van toepassing zijnde PVR, afbouwelementen, installaties en alle daarbij benodigde vrije ruimtes.
	English	An escape duct in a tunnel construction must provide conflict-free space for the applicable PVR, finishing elements, installations, and all necessary free spaces.
	RASE	<R>An <a>escape duct</a> in a <a>tunnel construction</a> must <r>provide conflict-free space</r> for the <s>applicable PVR</s>, <s>finishing elements</s>, <s>installations</s>, and <s>all necessary free spaces</s>.</R>

Requirement
Applicability
Selection
Exception

### 3.2.2 Manual rule interpretation

The interpretation of rules in automated code compliance checking goes two ways. On one hand, requirements must be correctly interpreted from the contract. On the other hand, it is crucial that the computer accurately interprets these requirements. The literature review showed that translating human-readable text into code that a computer can interpret, is one of the most challenging tasks, yet it is an essential step in the process (Preidel & Borrmann, 2015; Preidel & Borrmann, 2018). Additionally, correctly interpreting requirements within a BIM model is the most error-prone aspect of this process. It is also essential to ensure that verification results clearly demonstrate whether the requirement has been met (see Appendix A2.17).









To ensure accurate requirement interpretation, the SMART methodology is applied within the company. However, even though some requirements are SMART, they are not always specifiable within a BIM model (see Appendix A2.24). Furthermore, certain requirements are difficult to measure because they have not been made SMART enough (see Appendix A2.7). At Witteveen+Bos no specific markup language is used besides the SMART methodology, but a review process is in place to verify that requirements are clear and leave no room for misinterpretation (see Appendix A2.12). The literature review indicates that the RASE language is the most effective method for translating human-readable requirements into a structured format among the analysed languages. Mainly because of the strong standardisation, user-friendliness, and high language expressiveness, the RASE language is a suitable approach for automated compliance checking.

The RASE language was developed to efficiently tag and organize requirement texts. It uses four operators to extract the meaning of the requirement text: Requirement, which specifies the conditions that must be met; Applicability, that identifies which components the requirements apply to; Selection, which specifies criteria for applying the rule to particular elements; and Exception, which outlines the conditions where the requirement does not apply (Ilal & Günaydın, 2017). The RASE language has been applied to the project requirements from the case study (see **Table 8**). It is worth noting that no exception (E) operator is visible in the markup of the requirements, which means that there are no exceptions to the requirement texts. The requirement text examples are manually marked and to automate this process, the RASE mark-up tool developed by La Salle University for the EU ACCORD project has been explored (Accord, n.d.).

### 3.2.3 Preparation of the building model data

After the requirements have been interpreted using the RASE language, the next step is to prepare the building model data. Verifying requirements in a BIM model is not always straightforward, as there is sometimes insufficient information in the BIM model to perform the check (see Appendix A2.14). An example of this could be the missing material type for an object in the model. Additionally, one of the challenges is linking the requirements to the different objects in the model. A requirement concerns an object within the project, but this link is not always properly established or the way the requirement is formulated may not align well with the BIM model (see Appendix A2.23). It would be helpful if the requirements were written in a way that could be directly linked to elements in the BIM model (see Appendix A2.19).

The BIM model is composed of various sub-models which all are integrated into a single coordination model (CMO). The structure of the BIM model () is illustrated in the *Selection Tree* shown in **Figure 21a** from Navisworks. The model consists of a layer () that is named T.A.W. which refers to the reference level used for height measurements in Belgium. Under this layer several collections () are organised (e.g. Floors and Generic Models), each containing a series of items in the model. These collections are further subdivided into additional collections representing groupings of building elements, such as Multilayer pipelines. This collection is then structured into groups () and instanced groups () under which individual items are placed () ). The model contains properties which are assigned at the group level for each building element within the model. These properties include various attributes such as the element name, a unique identifier and an object code, as shown in **Figure 21b**.

In this research, four requirement classes were used based on their computational complexity and the nature of the verification process. Class 1 requirements involve basic checks based on explicit data, such as verifying the presence of components in the IFC model. These checks are generally straightforward and do not require any additional processing. Class 2 requirements include verifications that depend on simple derived properties, such as clearance distances or basic clash detection. These require some interpretation of geometric relationships and often rely on checking tools such as Navisworks.

For Class 3 and Class 4 requirements, the complexity increases. Class 3 verifications require an extended understanding of spatial relationships within the model, such as verifying spatial hierarchy. These rules often demand semantic enrichment where additional properties, data structuring or classification is needed. Class 4 requirements represent the most complex category, which typically requires simulation based verifications or a ‘proof of solution’.

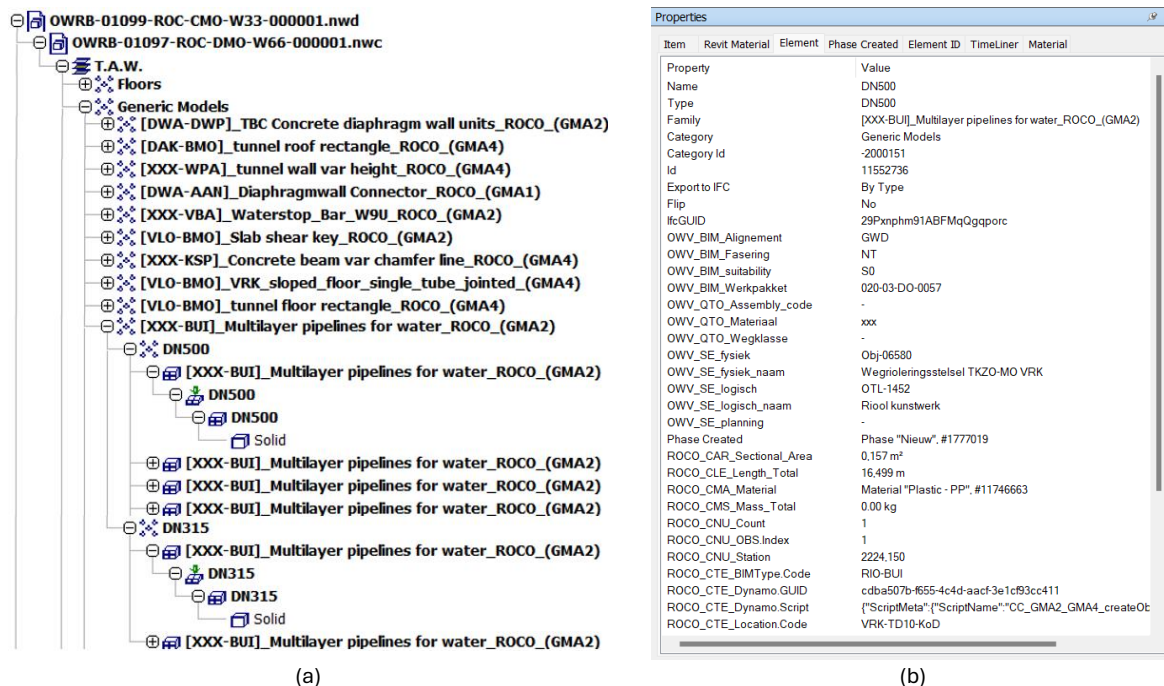


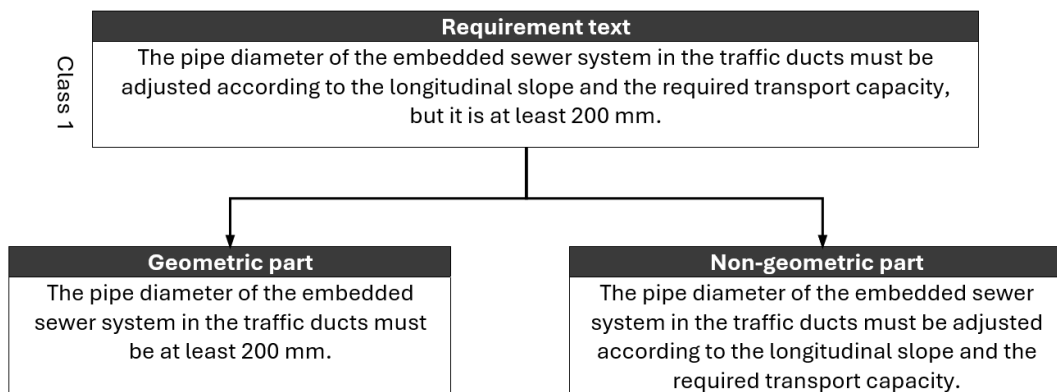
Figure 21. (a) Example of Selection Tree and (b) Model properties from Autodesk Navisworks.

### 3.2.4 Execution of the verifications

Once the preparation of the building model data is complete, the next step is to focus on the execution of the verification. However, even when a requirement can be verified in a BIM model, the verification method itself can still present challenges. Often, verifications in a BIM model require a significant amount of manual work. On average, verifying a requirement in a BIM model takes about one hour per requirement (see Appendix A2.18). The verification process can be (semi-)automated using scripts. Within Witteveen+Bos, this is only used to a limited extent. For example, there is a script that can group clashes in Navisworks. These scripts are often very project-specific, making them applicable only in certain situations (see Appendix A2.9). This subsection focuses on automated requirement verification where a distinction between the different requirement classes is made. The verifications will be conducted using the requirement text examples previously discussed.

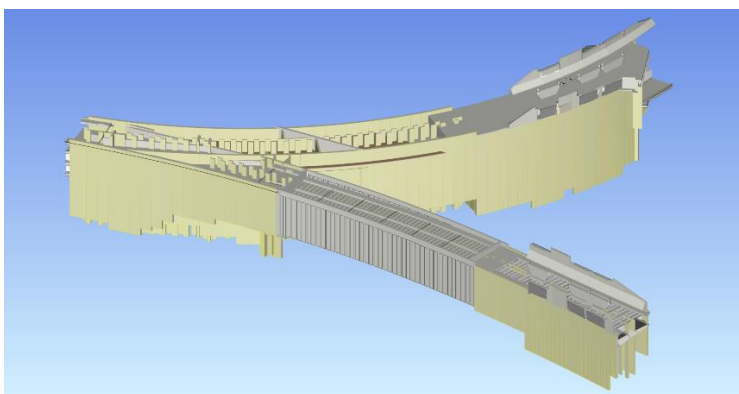
### 3.2.4.1 Verifications of Class 1

A requirement from Class 1 can be verified based on a limited amount of data. The earlier given example of a Class 1 requirement is presented in **Figure 22**. Although this requirement is classified by the model in the classification step as a geometrical requirement, it is in fact a twofold requirement. The required transport capacity and the corresponding slope of the pipe must be verified through calculations, which fall outside the scope of the BIM model. However, the minimum pipe diameter of 200 mm can be verified within the BIM model. The requirement text can be broken down to emphasize which part of the requirement is geometric and which part is not (see **Figure 22**). Such requirements will be referred to from now on as *hybrid* requirements, since they are partly geometric and partly non-geometric. In practice, both parts will need to be verified simultaneously given their dependency on each other. It can therefore be argued that the full hybrid requirement might rather be considered a Class 2 requirement.

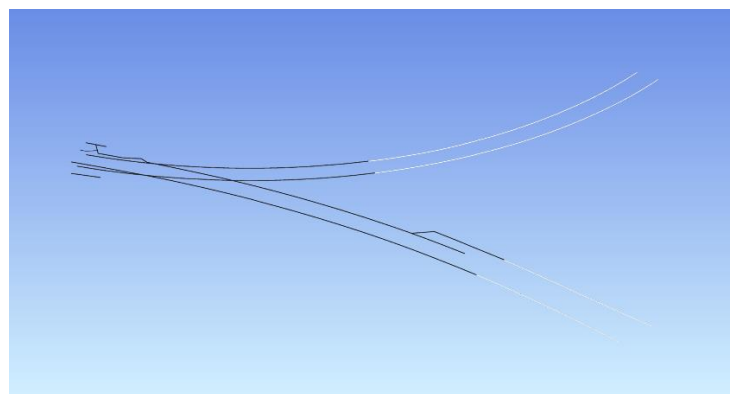


**Figure 22.** Example of a Class 1 requirement text with distinction between geometric and non-geometric part.

Given that the requirement text has been made more specific for the verification process within the BIM model, its complexity has been reduced as only the pipe diameter needs to be analysed. To make the verification of the requirement possible, it is necessary to clearly identify the specific part of the project to which the requirement applies. The requirement text has been linked in Relatics to object codes which can be traced back to the models. **Figure 23a** presents an overview of the tunnel elements associated with the requirement. Since this requirement relates to the road drainage system, the pipelines within the models of these tunnel elements can be isolated (see **Figure 23b**).



(a)



(b)

**Figure 23.** (a) Overview of tunnel elements related to the verification of the requirement and (b) the isolated pipelines that need to be verified based on the requirement text.


For the automated execution of this verification within the BIM model, the Python module IfcOpenShell was used. This open-source library enables the reading of the IFC file format to extract building information from the model. Within the company the IFC file format is used as the standard for exchanging building model data. To prevent information loss, it is essential that IFC files are exported correctly from Autodesk Revit. This requires selecting the most comprehensive export settings when generating the IFC model file.

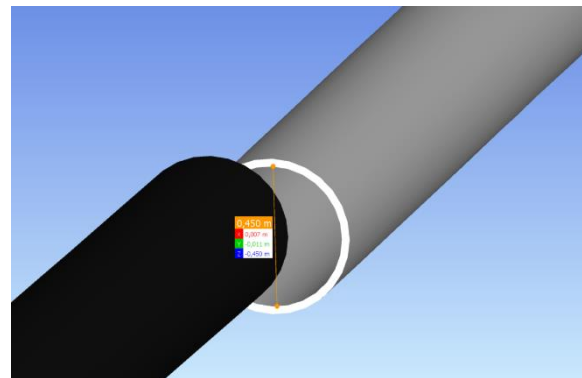
In the script, a similar filtering approach was applied to isolate the pipeline elements for the verification as illustrated in **Figure 23b**. The script extracts specific properties, such as the element name, identifier and diameter. It is capable of verifying, for each pipeline element in the model, whether the diameter is larger than or equal to the 200 millimeter specified in the requirement. A code snippet of the script along with a more detailed explanation and reporting of the results is provided in Appendix C.

In total 138 elements in the BIM models were verified to assess the pipe diameter. By using a count function in the script, it was determined that verifying 138 elements within the models required 80 seconds. The generated output presents whether the pipe diameter meets the specified requirement for each unique building element in the model (see **Figure 24a**). The results show that all elements in the model comply with the minimum pipe diameter requirement of 200 millimeters. Additionally, a manual verification was conducted to validate the script. Based on measurements in the model (see **Figure 24b**), it was confirmed that the automated verification of the pipe diameters was performed correctly. The manual verification led to the same conclusion that all the elements in the model meet the pipe diameter requirement. In contrast to the 80 seconds needed for the execution of the automated verification, the manual verification required approximately 120 minutes to verify the same number of building elements in the model. This time difference shows the potential of automated requirement verification.

**Element name**  
[XXX-BUI]\_Multilayer pipelines for  
water\_ROCO\_(GMA2):DN500:11552736

**GlobalId**  
29Pxnphm91ABFMqQgqporc

**Verification**  
 $d = 0.45 \geq 0.20$  



(a)

(b)

**Figure 24.** (a) Extraction of verification output (shortened) and (b) manual measurement of pipe diameter in the BIM model.

### Verification of Class 1 (Example 2)

This subsection looks at another verification of a Class 1 requirement which is shown in **Table 9**. The requirement text specifically pertains to the diaphragm walls that must be verified against the groundwater level. **Figure 25a** gives an overview of the tunnel elements relevant to the verification process. Since the ground level mentioned in the requirement text is not explicitly defined, it must first be determined. Based on project documentation, the groundwater level during the construction works is determined to be +3,00 m T.A.W. The groundwater level after the construction works is set at +4,75 m T.A.W. This represents an exceptionally high groundwater level that corresponds to an extreme event expected to occur once during the 100 year design life of the tunnel construction.

**Table 9.** Second example of a Class 1 requirement from the case study.

Class 1	Requirement text
	The top of the diaphragm wall must be at a height that prevents groundwater from flowing in during and after the works.

To verify the requirement, the diaphragm wall mentioned in the requirement text must be linked to the corresponding elements within the BIM model. A Python script was used to filter the IFC files of the models related to this requirement (see Appendix D). Only the elements associated with the diaphragm walls were extracted which resulted in the situation shown in **Figure 25a**, where the ground level is also shown as the green plane. This figure demonstrates that the top of the diaphragm walls prevent groundwater from flowing into the excavation pit during the construction works.

Given that a different groundwater level is applied after the construction works, the requirement must be verified twice for the two distinct situations. However, the major difference in the situation after the works is that the tunnel structure is fully completed. **Figure 25b** shows the tunnel structure with a groundwater level of +4,75 m T.A.W. It can be seen that the diaphragm walls do not prevent groundwater from all parts of the tunnel structure as some parts are located below the groundwater level. Nevertheless, it can be concluded that after construction, the tunnel is equipped with a roof structure which ensures that groundwater cannot enter the tunnel.



(a)



(b)

**Figure 25.** (a) Overview of tunnel elements related to the verification of the requirement in relation to the ground level during the works and (b) an overview of the tunnel elements in relation to the ground level after the works.

### 3.2.4.2 Verifications of Class 2

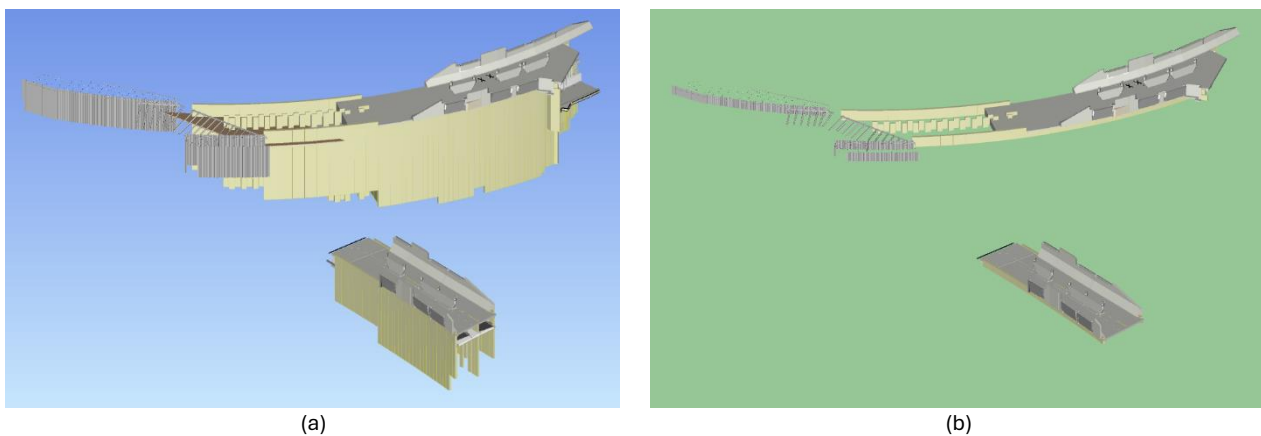
A Class 2 requirement can be verified based on simple derived attribute values. The example of a Class 2 requirement provided earlier is presented again in **Table 10**. In **Figure 26a** an overview is provided of the tunnel elements that must be verified against this requirement. The requirement text refers to a ground surface level which is not further specified. For the purpose of verification, a reference level relative to T.A.W. has been used (see **Figure 26b**). Based on project documentation, the ground surface level has been established at +5.5 m T.A.W. According to the requirement text, the foundation bases must be located at least one meter below the ground surface which corresponds to +4,5 m T.A.W. No foundation elements should therefore be present above this level in order to prevent frost heave. In addition to the ambiguity regarding the ground surface level in the requirement text, it is also unclear to which specific elements are referred to as foundation bases.



**Table 10.** Example of a Class 2 requirement from the case study.

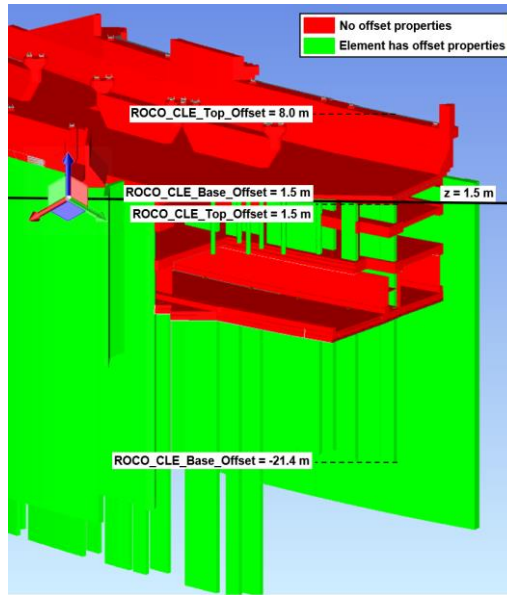
Class 2	Requirement text
	All foundation bases must be located at least 1 m below the ground surface to prevent any adverse effects from frost heave.

To verify the requirement, the ‘foundation bases’ mentioned in the requirement text must be traced down to the element level within the BIM model. For this purpose, a manual selection was made for the elements in the model based on their Family Name and Workset, which are relevant to the requirement (see Appendix E1). After this step, a Python script was used to filter the IFC files of the models related to the requirement (see Appendix E1). Only the elements associated with foundation bases which are relevant for the verification of the requirement were retained. Additionally, the remaining elements in the models were filtered based on the BIM phasing parameter (OWV\_BIM\_Fasering), distinguishing between Temporary Condition (TT) and Non-Temporary (NT). Temporary elements in the model, such as sheet piles, will be removed after construction and will not be part of the permanent structure. Therefore, temporary construction elements are not relevant for the verification of the requirement in terms of frost heave and have been excluded from the BIM model. This filtering process ensures that only the non-temporary elements that are associated with foundation bases are verified in the model.



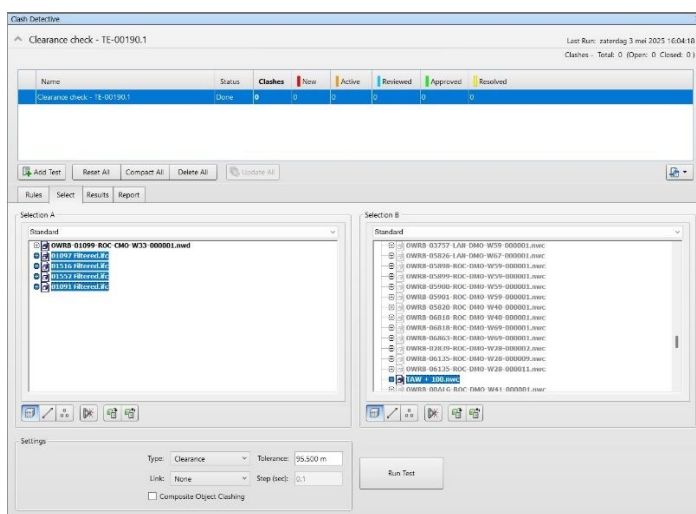
**Figure 26.** (a) Overview of tunnel elements related to the verification of the requirement and (b) overview of tunnel elements with plane layer at T.A.W. level.

After filtering the models, the verification process was examined using the IfcOpenShell module in Python. For each element in the model, the cartesian coordinates can be extracted which define the spatial position of the element. The requirement states that no foundation base elements may be present one meter below ground level. Within the model, a reference plane can be used to identify which elements are located above this plane based on their vertical position (Z-coordinate). However, elements may share the same Z-coordinate even if they are oriented differently within the model (e.g., facing upward or downward). The exact spatial extent of an element relative to the plane can be determined using the *Top Offset* and *Base Offset* properties in combination with the Z-coordinate. Nevertheless, not all elements in the model contain these properties. **Figure 27** illustrates a specific instance of this issue, indicating which elements have these properties and which do not. As a result, this verification approach based on the cartesian coordinates combined with the IfcOpenShell module is not feasible in this case. This highlights the importance of using consistent parameters when performing automated requirement verification in the BIM models.

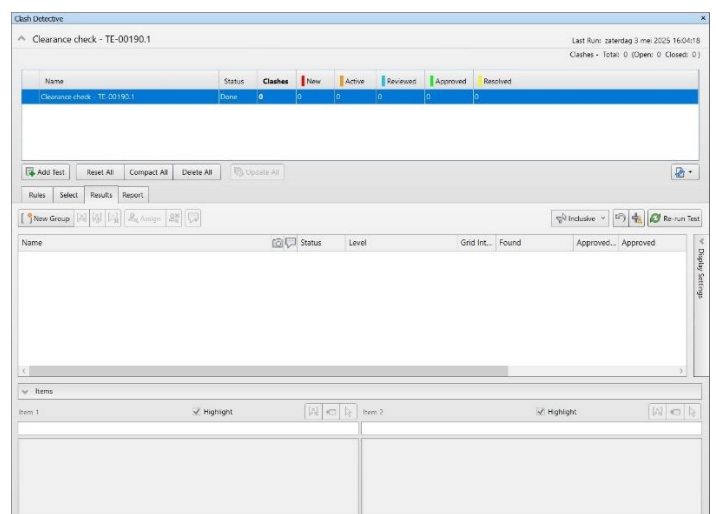


**Figure 27.** Indication of Offset properties in a model instance.

A different verification approach is explored using Navisworks where the filtered files from Python are used specifically targeting elements related to ‘foundation bases’. To verify the requirement, a clearance check is done between the model elements and a reference layer. A clearance check is a type of clash detection that checks whether there is sufficient space between objects in a model. The model includes a layer at +100 m T.A.W. which is used for conducting the clearance check. Since the requirement states that no elements should be present at least 1 meter below ground level (previously determined at +5,5 m T.A.W.), this translates to a tolerance of 95,5 meters relative to the layer at +100 m T.A.W. (see **Figure 28a**). The clearance check shows no clashes between the model and the layer, indicating that no foundation base elements are located 1 meter below ground level that could be adversely affected by frost (see **Figure 28b**). A clearance check with a tolerance of 96 meters results in 4 clashes, which indicates that these elements are located 1,5 meters below the ground level (see Appendix E2). This falls within the margin of the requirement, which verifies that all foundation base elements are at least 1 meter below the ground level to prevent freezing. This demonstrates that all foundation base elements meet the specified requirement.



(a)



(b)

**Figure 28.** (a) Clearance check with tolerance of 95,5 meter and (b) results of the performed clearance check.

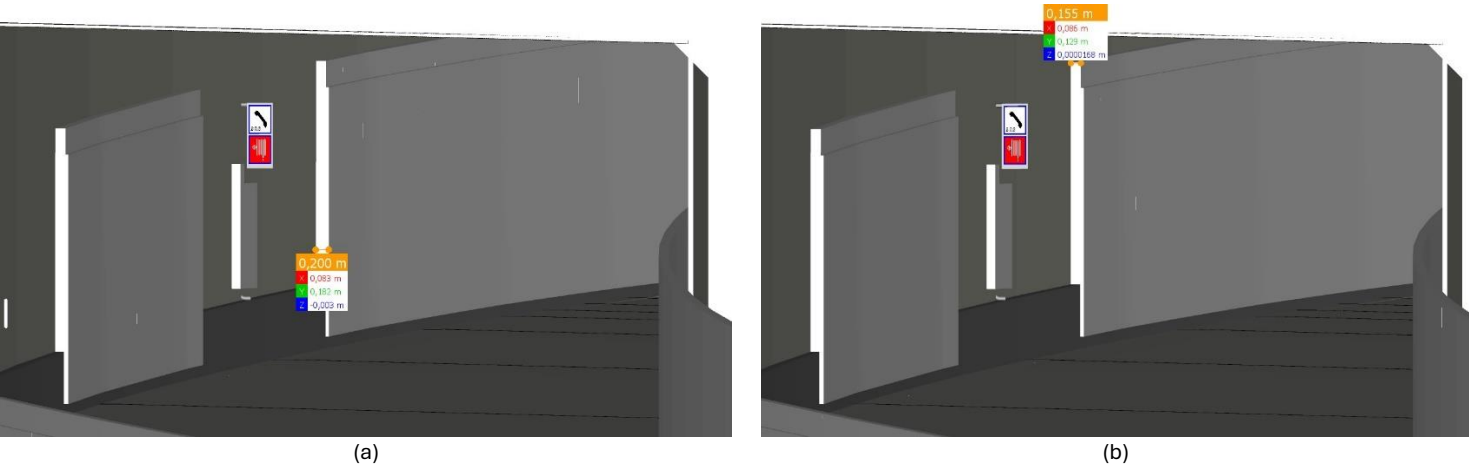


Verification of Class 2 (Example 2)

This subsection presents the verification of another Class 2 requirement which is shown in **Table 11**. The requirement concerns the minimum clearance between the road and the wall of the tunnel structure which must remain unobstructed. **Figure 29** illustrates both the tunnel configuration and the clearance profile (PVR) used for the verification. Specifically, **Figure 29a** indicates that the clearance profile has a width of 200 mm measured from the top of the barrier, while **Figure 29b** shows a width of 155 mm at the top of the clearance profile. These clearance profiles extend along the entire length of the tunnel structure.

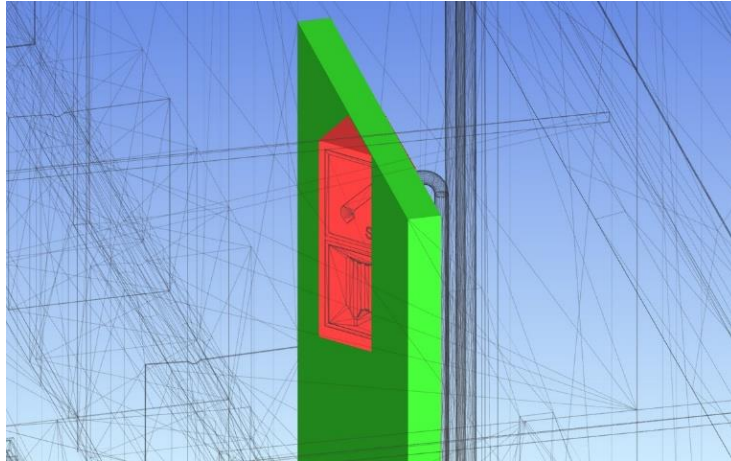
Class 2	<b>Requirement text</b>
	A tunnel structure should keep the space between the road and the finished wall, from the top of the barrier to the top of the PVR completely clear. This PVR should also be at least 200 mm wide at the top of escape pictograms along the entire length of the tunnel and at least 155 mm wide at the top of pictograms above the auxiliary station boxes along the entire length of the tunnel.

To verify this requirement, the clearance profile was used to perform a clash detection within the BIM model (see Appendix F). The clash detection was conducted between the clearance profiles in the tunnel structure and the escape pictograms and auxiliary station boxes along the entire length of the tunnel. This requirement ensures that sufficient space remains between these elements and the roadway used by traffic. The clash detection revealed that nearly all clashes occurred at the lower part of the clearance profile. An example is shown in **Figure 30**, where the clash detection revealed that the escape pictogram extends more than 200 millimeters into the roadway from the wall of the tunnel structure. In total, 75 clashes were detected in relation to this specific requirement. Consequently, compliance with the requirement could not be confirmed due to these identified clashes.



**Figure 29.** (a) Tunnel structure with clearance profile shown with a top width of 200 mm and (b) tunnel structure with clearance profile with a bottom width of 155 mm.

Although clash detection is useful for identifying spatial conflicts in the model, a detected clash does not necessarily indicate non-compliance with the requirement. Each clash must be reviewed in context, considering its relation to the clearance profile. Automating this evaluation process is difficult, as it often requires visual inspection to judge whether a clash presents a real design issue or remains within acceptable limits. Based on this review, the design may need adjustments.



**Figure 30.** Clash between signing and wall element.

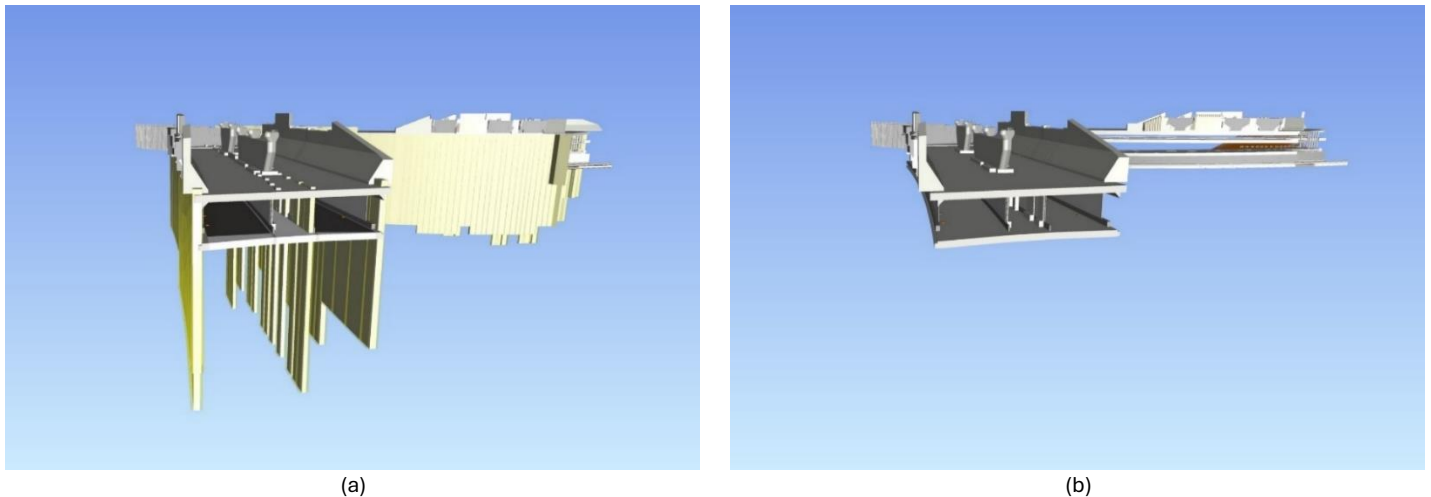
### 3.2.4.3 Verifications of Class 3

This subsection discusses the verification of Class 3 requirements, which necessitate an extended data structure. Class 3 requirements involve checks that require an in-depth analysis of spatial relationships between elements. The previously introduced example of a Class 3 requirement is show again in **Table 12**. Also, **Figure 31a** provides an overview of the tunnel elements that must be verified against this requirement. Prior to the verification, it is necessary to determine which elements in the models correspond to the traffic, escape and service tube spaces. This classification must take into account the information provided in the linked documents to this requirement and mentioned in the requirement text. These documents include technical drawings of the alignment and specify the various tunnel sections (see Appendix G1). According to these documents, the identified spaces must remain free of obstructions in the model.

**Table 12.** Example of an Class 3 requirement from the case study.

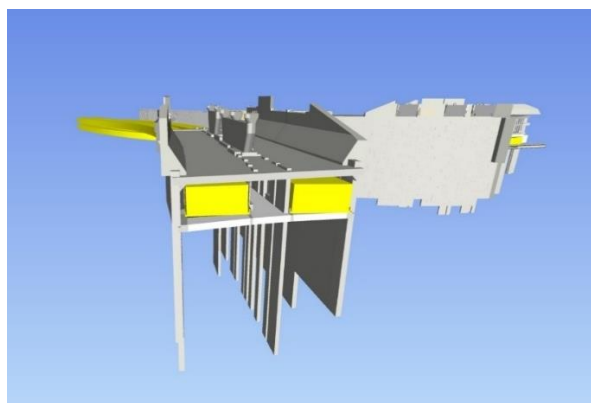
Class 3	Requirement text
	A tunnel construction must realize the applicable traffic, escape- and service tube spaces in accordance with the linked document.

In order to verify this requirement, the traffic, escape and service tube spaces referenced in the requirement text must be identified at the element level within the BIM model. This process began with a manual selection of relevant elements based on their Family Name to ensure alignment with the specified spatial zones. A Python script was used to process the corresponding IFC files to isolate only those elements situated within the tunnel tube spaces and considered relevant for the verification (see Appendix G1). Also, a filtering was done based on the BIM phasing parameter which distinguishes between elements in temporary condition and those classified as non-temporary. This ensured that the model contains only the permanent elements that are relevant to the requirement (see **Figure 31b**).



**Figure 31.** (a) Overview of tunnel elements related to the verification of the requirement and (b) overview of filtered tunnel elements.

To carry out the verification, a clearance profile (in Dutch: *profiel van vrije ruimte* or PVR) was used to assess whether the traffic, escape- and service spaces within the tunnels are free from obstructions. This clearance profile extends along the entire alignment of the tunnel elements and was incorporated into the model to enable the verification process (see **Figure 32**). The clash detection identified 366 clashes in the models and the clearance profile by using a tolerance of 0 millimeters. These clashes were grouped by workset and revealed that over half of the total number of clashes occur in the workset *general water management*. A more detailed overview of the clash detection results is provided in Appendix G2. When the clash detection tolerance was increased to 0,05 meters, the total number of clashes was reduced to 18. This reduction shows the sensitivity of clash detection outcomes to the selected tolerance level and emphasizes the importance of defining context appropriate threshold to distinguish between critical and non-critical clashes in the model.



**Figure 32.** Impression of clearance profile in the model.

Compliance with the requirement cannot be confirmed due to the clashes identified during the verification process using the clearance profile. Although clash detection provides valuable insights into potential spatial conflicts, the presence of a clash does not automatically indicate non-compliance. Each clash must be evaluated based on its severity, location and relevance to the clearance profile. This evaluation is difficult to automate and typically relies on visual inspection of the clash context to determine whether the clash constitutes a design issue or falls within acceptable tolerances. Based on the outcome of this assessment, the design and the BIM model must be adjusted accordingly to ensure compliance with the requirement. Subsection 3.2.5 provides a more detailed discussion on reporting the results from the verification.

### Verification of Class 3 (Example 2)

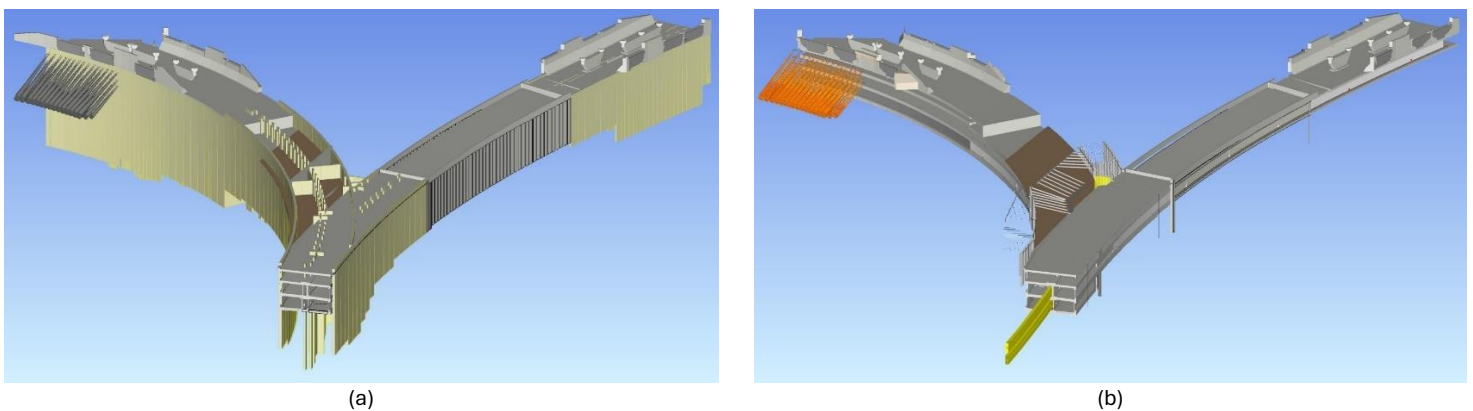
This second example of a Class 3 requirement closely resembles the previous one. In this case, the requirement text presented in **Table 13** is used. Additionally, **Figure 33a** provides an overview of all tunnel elements that must be verified in accordance with this requirement. To conduct the verification, it is essential to identify which elements in the model correspond to the escape, service and cable ducts. Moreover, it is important to incorporate the information from the linked documents into the verification process. These documents include technical drawings of the alignment, with sectional views of the tunnel segments, indicating the specific elements that are subject to verification. According to these documents, the identified spaces must remain free from obstructions. This approach, which uses a clearance profile, is consistent with the previous Class 3 verification process, which also uses a Python script to filter the model.

**Table 13.** Second example of a Class 3 requirement from the case study.

Class 3	Requirement text
	A tunnel structure shall realize the spaces in escape shafts, service shafts and cable ducts in accordance with linked documents.

To perform the verification, a clearance profile was used to assess whether the spaces within the escape, service and cable ducts are unobstructed. This clearance profile is embedded across all tunnel elements throughout the entire modelled alignment which enables the verification process (see **Figure 33b**). A clearance check revealed 333 clashes between the model and the clearance profile. As a result, compliance with the requirement could not be confirmed due to these identified clashes.

While clash detection is a useful method for identifying potential spatial interferences within a model, the detection of a clash does not inherently signify that the requirement has been violated. Each identified clash needs to be carefully reviewed, taking into account its significance and its relationship to the defined clearance profile. The review process can be carried out within the CDE which is used for documenting and tracking issues. However, automating such an evaluation process proves to be challenging as it typically depends on visual inspection to understand the context of the clash and to judge whether it represents an actual design problem or remains within acceptable limits. Depending on the findings of this review, necessary modifications must be made to both the design and the BIM model to ensure that the requirement is met.



**Figure 33.** (a) Overview of tunnel elements related to the verification of the requirement and (b) overview of filtered tunnel elements with clearance profile.

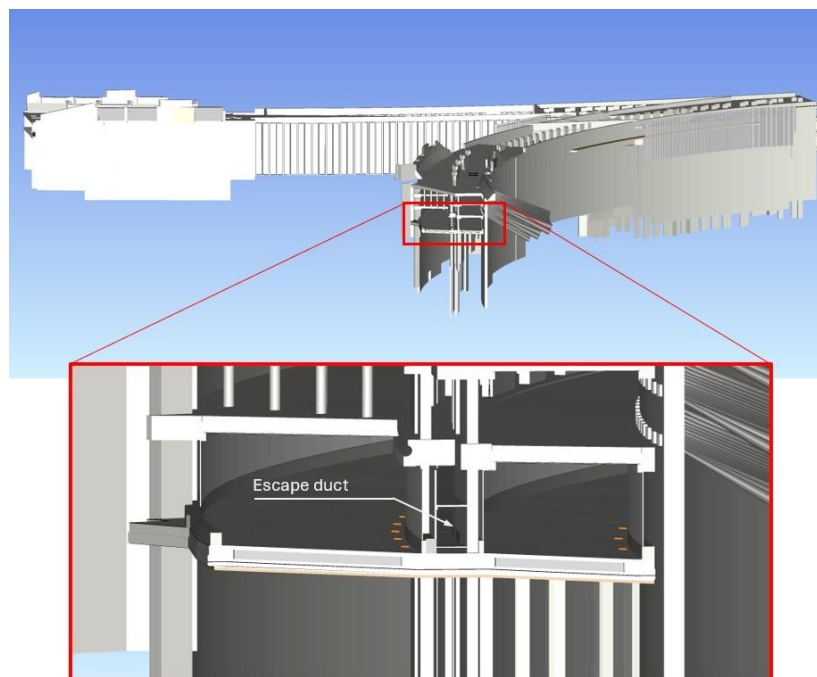
### 3.2.4.4 Verification of Class 4

This subsection focuses on the verification example of a Class 4 requirement. The example of the Class 4 requirement provided earlier is presented again in **Table 14**. The requirement is categorised in this class since it suggests corrective actions to ensure a conflict-free space. The requirement does not provide a clear measurable threshold but it rather implies human intervention is needed to resolve conflicts. This class of requirement is not very common, so no other example is available from the requirement dataset. **Figure 34** and **Figure 35a** show an overview of the tunnel elements that must be verified against this requirement. The figure also illustrates what is referred to as the escape duct within the tunnel structures. Additional information is linked to the requirement regarding the minimum clearance profile (PVR) dimensions for the escape ducts in the tunnel tube. For the *Vork*, these minimum dimensions are specified as 1,7 meters in width and 2,1 meters in height. These clearance profile dimensions must be achieved along the entire length of the section to comply with the requirement.

**Table 14.** Example of a Class 4 requirement from the case study.

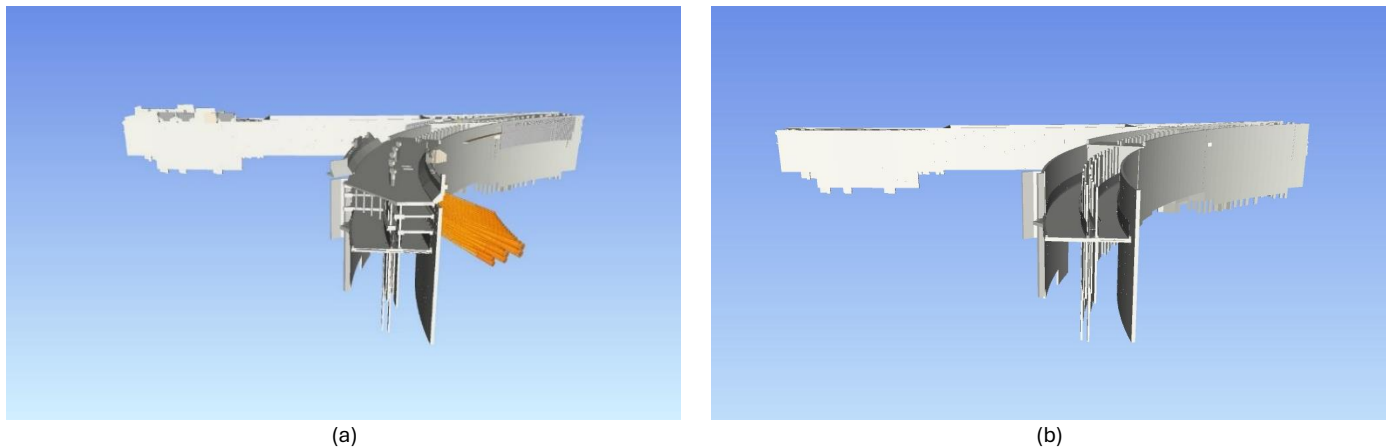
Class 4	Requirement text
	An escape duct in a tunnel construction must provide conflict-free space for the applicable PVR, finishing elements, installations, and all necessary free spaces.

To perform the verification, a selection is made within the BIM model of the elements related to the escape ducts that are relevant to the requirement. The filtering of the BIM model was carried out using the Python module IfcOpenShell. The building elements are filtered based on the parameter `OWV_SE_fysiek_naam`, which contains the physical names of the elements in the model. This filtering process isolated and retained elements with names such as ‘espace door’ and ‘escape duct floor’. Additionally, a filter was applied using the BIM phasing parameter to remove temporary construction elements from the model. These temporary components are dismantled after construction and are therefore not directly relevant for verifying the requirement. As a result, only the building elements necessary for the verification of the requirement remain in the BIM model (see **Figure 35b**).



**Figure 34.** Tunnel close-up of escape duct.

Following the selection of the relevant building elements for verifying the requirement using Python, the requirement verification can be carried out. A clearance profile within the model was used for this purpose, representing the escape ducts of the tunnel structure. The clearance profile spans the entire length of the tunnel to verify whether the escape ducts maintain a conflict-free space. The clash detection process in Navisworks identified 340 clashes between the model elements and the clearance profile. Appendix H provides a more detailed explanation of the filtering of building elements for the requirement verification and the implementation of the verification process in Navisworks. Since a Class 4 requirement is rare due to its computational complexity, the requirement discussed above is the only one of this class that appears in the filtered dataframe.



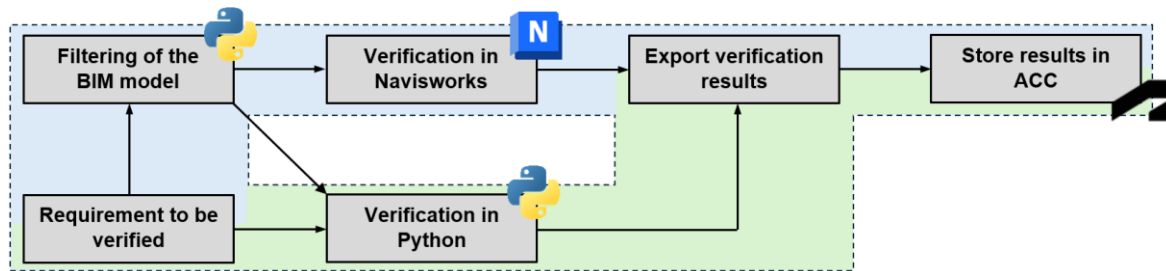
**Figure 35.** (a) BIM model with all the elements in place and (b) filtered model focusing on tunnel escape ducts.

### 3.2.5 Reporting of the results

After verifying the BIM models against the specified requirements, the outcomes must be documented in a way that enables the users to understand the identified issues and take appropriate corrective actions. To manage the issues, Autodesk Construction Cloud (ACC) is chosen as the issue management system. In addition to issue tracking, ACC offers functionalities for project management, model coordination and document control. Autodesk Construction Cloud serves as a Common Data Environment (CDE) by centralizing project data across all construction phases and allows for real-time collaboration. The use of a CDE aligns with the buildingSMART standard and complies with ISO 19650 (see Subsection 2.2.2). According to ISO 19650 clause 3.3.15, a CDE is the agreed source of information for a given project used to collect, manage and share each information container through a managed process (ISO, 2018a).

**Figure 36** presents a simplified representation of the requirement verification process and the use of the Common Data Environment (discussed in detail in Section 3.4). The figure shows two verification approaches: the first approach involves directly verifying the requirement in Python by importing the IFC file of the model, while the second involves filtering the BIM model prior to performing the verification in Navisworks. In both cases, the verification results are exported to Autodesk Construction Cloud which serves as the CDE in this process.





**Figure 36.** Process of BIM requirement verification and reporting of results to the Common Data Environment.

An overview of how issues are displayed within Autodesk Construction Cloud is shown in **Figure 37a**. The issues stored in ACC are based on IFC files derived from the BIM models. These IFC files enhance interoperability and facilitate collaboration among project stakeholders. Moreover, their use aligns with the buildingSMART IFC standard and the related ISO 16739 standard. **Figure 37b** illustrates how issue details are presented within Autodesk Construction Cloud. The issue details shown in this figure originate from the verification of the Class 4 requirement discussed in Subsection 3.2.4.4. It indicates which elements constitute the issue, the associated requirement, the responsible party, the relevant model and other contextual information. Within the Model Coordination environment of ACC, users have direct access to the most recent version of the model with the related issue. This setup with using a CDE enables cloud-based collaboration among various disciplines involved in requirement verification.

The issues stored in ACC can be exported into the BIM Collaboration Format (BCF). This is another international openBIM standard, which enables different BIM applications to exchange model-based issues using IFC files of the models. Additionally, ACC also makes it possible to generate issue reports from the issue overview shown in **Figure 37a**. Besides recording issues in ACC, it is also important to document the approved requirement verifications. An example of this is shown in **Figure 37c**, where the example corresponds to the verification of the Class 1 requirement discussed in Subsection 3.2.4.1. It is important to note that this illustrates how results from both requirements verified using Python with the IfcOpenShell module and those verified using clash detection in Navisworks can be integrated into the Common Data Environment.





### 3.3 Implementation challenges

This section addresses the implementation challenges encountered during the execution of the verification process. Although the verifications carried out show potential for automated requirement verification, several difficulties arose during the practical implementation of the requirement checks. The order in which the challenges are presented does not reflect their relative importance.

#### 3.3.1 Ambiguity in requirement texts

One of the main challenges in automating the verification of requirements in this research lies in the frequent ambiguity and incompleteness of those requirements. Often formulated in natural language, requirements tend to be imprecise, open to interpretation or lacking crucial information. As a result, manual interpretation or clarification is frequently required before automated verification can even begin. This dependence on human input reduces the efficiency of automated approaches and shows the need for more structured requirement definitions.

#### 3.3.2 Hybrid requirements

In Subsection 3.2.4.1, the term *hybrid requirement* was introduced for the first time referring to a requirement that is partially geometric and partially non-geometric. In the given example, one part of the requirement text concerned a geometric property (pipe diameter) which could be verified within the BIM model. However, the other part related to a non-geometric aspect (transport capacity), necessitated a structural calculation in addition to verification within the BIM model. Verifying such hybrid requirements presents additional challenges, as it demands increased coordination between the various disciplines involved in the verification process.

#### 3.3.3 Inconsistent model data and parameters

Inconsistencies in the BIM model's data and parameter structure limited the effectiveness of automated verification. The lack of standardised element properties led to missing or incomplete information across various model building elements. This issue was particularly problematic for special checks as seen in Subsection 3.2.4.2, where required property data was not consistently available making it difficult to verify the requirements. In addition to hindering automation, the inconsistent use of parameters also reduces the reusability of verification procedures across different BIM models. This shows the need for a consistent use of parameters and building information within the process of automated requirement verification.

#### 3.3.4 Classification of requirements

The classification step within the process of automated requirement verification focuses on identifying which requirements can be verified within the BIM model (geometric) and which cannot (non-geometric requirements). To facilitate this classification, a machine learning model was developed and trained using Python to predict the category for each requirement as described in Subsection 3.2.1. The model achieved a classification accuracy of 95%, successfully categorizing the majority of the requirements. However, the remaining 5% were misclassified, indicating that full automation is not yet feasible. This shows that human intervention is still necessary to review and validate the outcomes of the classification step. It ensures that requirements are correctly classified before the verification process continues.

### 3.3.5 Linking model elements to requirements

In addition to the fact that requirement texts are often ambiguous and open to interpretation, the requirement texts are not directly linked to building elements within the model. The requirement text from the contract is associated with an object code that is related to the BIM model. However, this means that the building elements are not linked to the requirement text at an element level. As a result, it is often unclear which specific elements in the BIM model need to be verified against the requirement. The requirement text is reviewed from a Systems Engineering perspective to ensure it is SMART, but the focus does not extend to linking the requirement text to elements in the model. The absence of a direct connection between individual building elements and the requirement text unnecessarily complicates the process of automated requirement verification.

### 3.3.6 Fragmented workflow

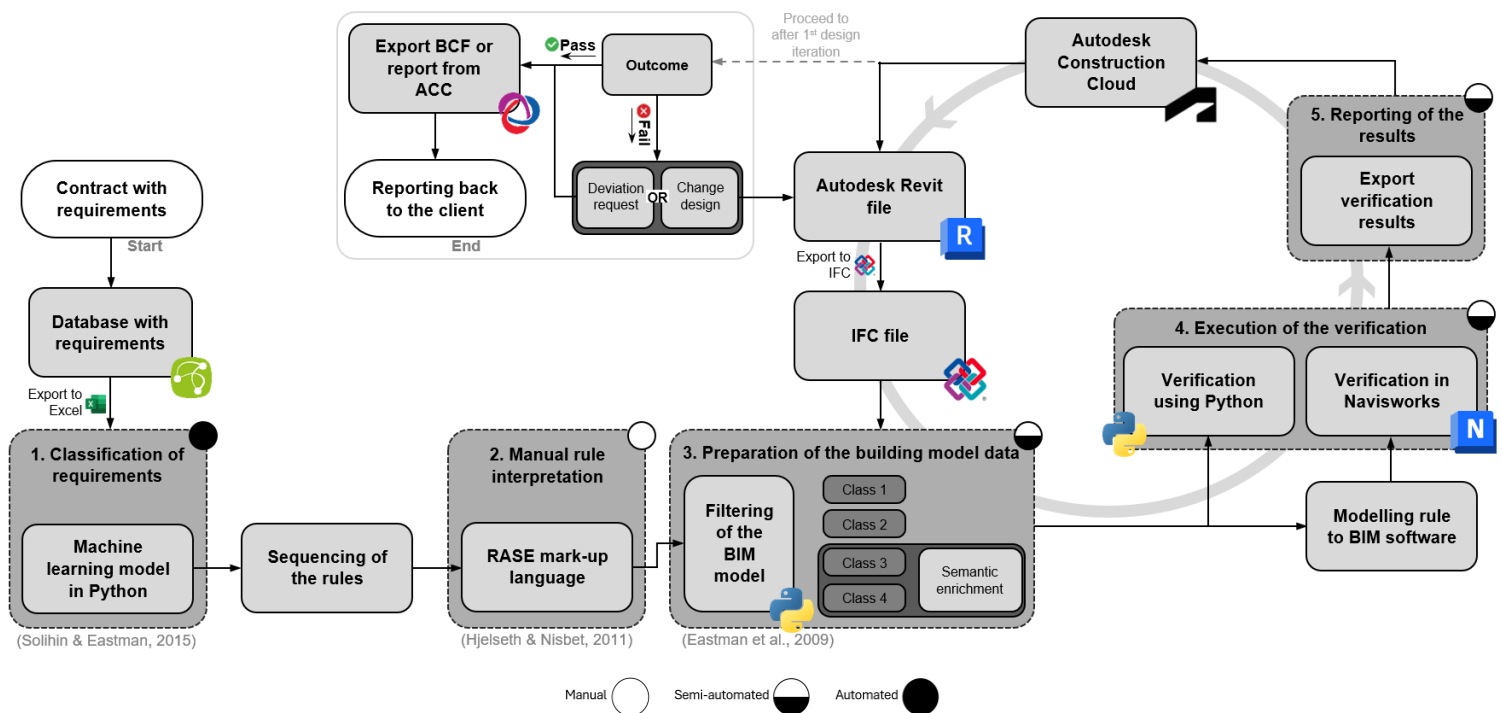
The case study demonstrates that the automated requirement verification process involves a fragmented workflow. This is partly unavoidable due to the structure of the process which necessitates manual steps. For example, requirements from the Relatics database must first be processed into geometrical and non-geometrical categories to see which requirements can be verified using a BIM model. This filtering was performed using Python in order to reduce the need for manual selection during the automated verification process. Nevertheless, this step, along with the ambiguity in requirement texts and the inconsistent use of model data, contributes to the fragmentation of the workflow.

A more structured approach to project (model) data and the formulation of clear, unambiguous requirements that are directly linked to the BIM model can improve the efficiency of the workflow. While the workflow for automated requirement verification will continue to depend on multiple tools, the findings demonstrate significant potential for improving the efficiency of their integration. The workflow used in this research is explained in the following section. In addition, guidelines were developed based on the implementation challenges, which are discussed in detail in Chapter 4.

## 3.4 Workflow

This section focuses on the workflow throughout the entire process of automated requirement verification, starting from the contractual requirements to the reporting of the verification results. At the initiation of the project, the contractual requirements are first entered into a requirement database which is Relatics in this case. According to Eastman et al. (2009), the process of automated requirement verification can be divided into four main parts. The workflow described in this report broadly corresponds to this structure. However, an additional fifth part has been included in the workflow which addresses the classification of the requirements. The complete workflow of the process is presented in **Figure 38**.

After the requirements have been entered into Relatics, they can be exported into an Excel file. This data is then processed in the classification step (see part 1 in **Figure 38**), where a machine learning model developed in Python distinguishes between geometrical and non-geometrical requirements. Geometrical requirements can be verified using a BIM model, whereas non-geometrical requirements must be verified in another way (e.g. calculation). In addition to this categorisation, the requirements are also classified into four classes during the classification step. These classes differentiate how the requirements can be verified within the BIM model, based on their computational complexity (Solihin & Eastman, 2015). After that, the sequencing of the rules must be considered, focusing on requirements that depend on the outcome of other requirements.



**Figure 38.** Workflow of the automated requirement verification process.

Following the classification of the requirements, the next step involves manual rule interpretation (see part 2 in **Figure 38**). This is done using the RASE mark-up language which is a method that converts human-readable requirements into a machine-readable format that is compatible with BIM models using the IFC standard (Hjelseth & Nisbet, 2011). Once the requirements have been interpreted, the building model data is prepared (see part 3). This preparation includes filtering the BIM model to isolate elements relevant for verifying specific requirements. The process is done in Python using IFC files derived from the latest BIM models, which can be obtained from Autodesk Construction Cloud by exporting Revit files to IFC format. Also, depending on the requirement class, it may be necessary to semantically enrich the model to ensure that missing or implicit information is added for the verification. The resulting IFC file can then be used for verification in Navisworks or the verification can be done directly in Python using the IFC file (see part 4).

Part 5 focuses on the reporting of verification results, where the outcomes are exported to Autodesk Construction Cloud which functions as a CDE within the workflow. From ACC, a verification report is generated if the requirement is met. If the requirement is not met, either a deviation request is filed or a design change is initiated. In the latter case, the verification process needs to be repeated. This iterative nature of the process is reflected in the cyclic representation of the design process in **Figure 38**.

### 3.5 Efficiency metrics

This section looks at efficiency metrics to evaluate the performance of the proposed (semi-)automated approach in comparison to the traditional method. A before-and-after comparison is done to evaluate changes in the semi-automated requirement verification process. The findings from the case study are applied using an empirical approach to provide insight into the effectiveness of the implementation. The comparison uses the following metrics to evaluate the effectiveness: 1) time savings, 2) repeatability and 3) the quality of the result. The *time savings* metric evaluates the reduction in time when using the automated approach compared to the manual method. The *repeatability* metric examines the extent to which the verification can be

consistently replicated and still get the same results. Lastly, the *quality of the result* metric assesses the accuracy and precision of the output. The evaluation of the performance of the proposed approach was conducted by assessing the five main steps as presented in the workflow (see **Figure 38**). A table is used in this section to facilitate a quantitative comparison of the metrics used in the evaluation (see **Table 15**).

### **Efficiency of the classification**

The classification of requirements in the semi-automated process involves categorizing the requirements as either geometrical or non-geometrical depending on whether they can be verified within the BIM model. This classification step, performed manually, takes approximately 139 minutes for the full dataset of the case study consisting of 614 requirements. To automate this classification step, a machine learning model was developed in Python to classify the requirements into the same categories. The model completed the classification in 42 seconds. This shows a time reduction of 99.5% for this specific dataset (see Appendix I).

In terms of repeatability, the manual approach may produce inconsistent results as the classification of requirements is influenced by subjective human judgment. It is therefore likely that, when repeating the classification step, not all requirements will be categorized consistently as either geometrical or non-geometrical. In contrast, the repeatability of the automated approach is significantly higher as the classification is performed using a consistent algorithm. Regarding the quality of the results, the manual approach aligns with the repeatability metric in the sense that the output may vary. As such, it is difficult to assign an exact performance value to the manual method. However, the machine learning model used in the automation approach achieved an accuracy of 0.95 which indicates that it correctly categorized 95% of the requirements (see Appendix I).









### **Efficiency of the rule interpretation**





For the rule interpretation step of the process, no exact efficiency metric can be provided as the RASE language was not applied in the traditional approach. Moreover, because the RASE mark-up is performed manually, a direct comparison between the two approaches is not possible. Regarding the repeatability of the RASE mark-up, it can be considered moderate. While the standardised algorithm provides a consistent framework for marking up requirements, the manual nature of its application might introduce some variability. In terms of quality, the results produced using RASE are likely to be high, as the structured approach improves accuracy relative to the manual approach.

### **Efficiency of the preparation**

Although exact time savings were not measured, the semi-automated preparation of building model data improved the efficiency. Automating element filtering with scripts using IfcOpenShell reduces manual effort which suggests time savings. The approach also enhances repeatability by applying consistent filtering and eliminating variability in the manual approach. This consistency is important for Class 3 and 4 requirements that need semantic data. The quality improves as automated filtering reduces errors which results in more accurate results.

**Table 15.** Efficiency metrics of the proposed semi-automated approach relative to the manual approach for the steps in the workflow of automated compliance checking.

	Automation	Time savings	Repeatability	Quality of the result
1. Classification of the requirement		99,5%	High	95,0%
2. Interpretation of the requirements			Moderate	High
3. Preparation of the building model data			Moderate	High
4. Execution of the verification		98,9%	High	High
5. Reporting of the results			High	High

Manual  Semi-automated  Automated  Not quantifiable 

### Efficiency of the verification process

The execution phase of the verification process showed improvements in efficiency when comparing the manual and semi-automated approaches. In terms of time savings, the Class 1 requirement example involving the verification of pipe diameters served as a quantifiable benchmark. The manual verification within the BIM model required 120 minutes to assess compliance with the specified requirement. In contrast, the automated approach, using the Python module lfcOpenShell, completed the verification of the same number of elements in 80 seconds. This shows a reduction of 98.9% in terms of time for this specific case. While exact time measurement could not be directly compared for the other requirement examples, the verification workflows were largely consistent. In all cases, the use of Python for filtering reduced manual navigation time within the model.

In terms of repeatability, the manual approach relies on measurements within the BIM model, which may vary upon repetition. In contrast, the automated verification produces consistent results that show a high degree of repeatability. However, it is essential to ensure that the same parameters are used during each verification to maintain this level of repeatability. The final metric examines the quality of the result by evaluating the precision of the output. As previously noted, the repeatability of the manual approach may vary which can affect the precision of the results. To assess this, the precision was measured based on ten verification attempts. The absolute deviation of these observations was found to be 0.0011 meters which corresponded to a 0.24% deviation. The automated approach showed no variation in the results, leading to a deviation of zero percent (see Appendix I).

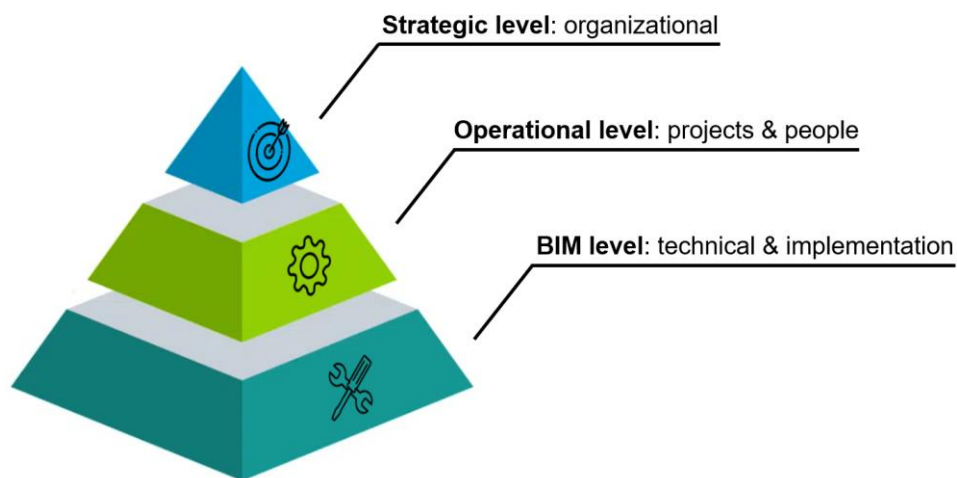
### Efficiency of reporting the results

The final step of the process, the reporting of the results, cannot be directly compared in the before-and-after analysis and therefore no exact value can be provided for the efficiency metrics (see **Table 15**). However, it can be stated that the outcomes of both the manual and semi-automated processes are expected to be comparable when the same verifications are repeated resulting in a high degree of repeatability. This is because each approach employs a consistent method for reporting the results which will give similar results upon repetition. Regarding the quality of the reported results, the two approaches cannot be quantitatively compared. Nonetheless, it is reasonable to assume that both the manual and semi-automated methods produce outputs of similar quality. However, the semi-automated approach does offer improved traceability as it is better integrated within the CDE instead of relying on another software tool.

# 4. Guidelines

## 4.1 Creating guidelines

For creating the guidelines, a distinction has been made between three levels, namely: a strategic level, an operational level and a BIM level (see **Figure 39**). The strategic level concerns the vision, policy and culture of the organisation. The operational level looks at the translation of strategy into daily practice. The BIM level focuses on concrete applications of tools, formats, standards and models. The Oosterweel case study project serves as the scope for developing the guidelines. The insights from the literature review and the conducted interviews were also used in the development of the guidelines.



**Figure 39.** Distinction between different levels for developing guidelines.

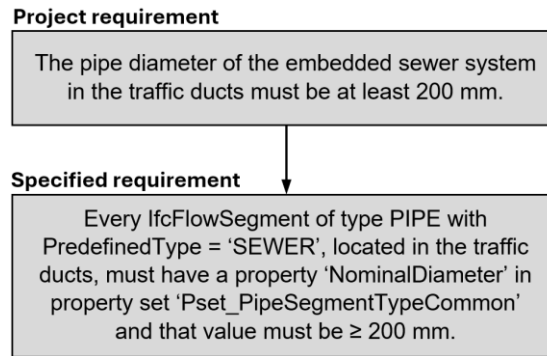
### 4.1.1 Strategic level

At the strategic level, making automated requirement verification a standard part of BIM processes requires clear decisions and actions from the top of an organisation. This means deliberate changes are needed in procurement practices and engagement with the client from the start of the project. If these changes are not made at the higher, strategic level, then it becomes much harder to implement automated verification properly in the project. Also, without a top down push, implementation at the operational or BIM level will remain fragmented and inconsistent.

#### **Ambiguity in requirement texts**

During the implementation of the case study, it became clear that ambiguity in requirement texts presents a significant barrier to the automation of the verification process (see Subsection 3.3). This issue must be addressed at a strategic level, beginning in the early stages of the project. Contracts should clearly define the expectations regarding the automated verification of requirements within BIM models. This includes the specification of required formats, deliverables and the allocation of responsibilities. Establishing clear and computer interpretable requirements during the procurement phase sends a strong signal to both clients and the market that verification within the BIM process should transition from a reactive task to a proactive responsibility shared among all stakeholders.

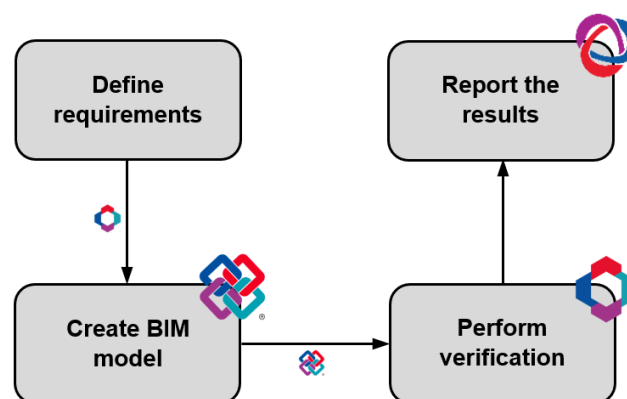




**Figure 40.** Computer interpretable requirement text using the IDS standard.

The requirements used in the case study were insufficiently specific which hindered the automation of the verification process. To address this, the RASE mark-up language was applied to structure the requirement texts. For automated requirement verification to be effective, requirements must be computer-interpretable and explicitly linked to the corresponding building elements within the BIM model in accordance with the IFC/IDS bSI standards and ISO 16739 (IFC). As an illustration of this approach, a requirement from the case study was reformulated and specifically linked to model elements following these standards (see **Figure 40**). To test the full use of the bSI standards using the IDS, the verification of this same requirement is included in Appendix J.

To prevent ambiguity in requirement texts during their formulation, a guideline has been formulated (see S1 in **Table 16**). Once a requirement is made specific, as illustrated in the provided example, it becomes significantly easier to automate its verification within the IFC model. This leads to both improved quality control and data fidelity. The verification results can be generated using the BIM Collaboration Format which enables the linking of verification information directly to building elements in the model in accordance with the bSI BCF standard. This approach aligns with ISO 19650 which outlines a framework for the structured exchange of information across disciplines throughout the entire project duration. **Figure 41** provides a simplified overview of the complete process, from clearly defining project requirements to avoid ambiguity to reporting verification results using BCF. If this approach and guideline S1 had been followed from the start, the process of automated verification would have been much simpler in execution.



**Figure 41.** Simplified verification process using bSI standards.

## Hybrid requirements

The use of hybrid requirements was identified in Subsection 3.3 as an implementation challenge in the process of automated requirement verification. The presence of both a geometric and a non-geometric component within a single requirement complicates automation as it necessitates manual interpretation to determine which parts can be verified within the BIM model

and which require alternative validation methods. Verifying a hybrid requirement also demands increased coordination between disciplines as the verification process becomes interdependent. Moreover, hybrid requirements introduce uncertainty when the boundary between geometric and non-geometric aspects is unclear. Therefore, the guideline presented in **Table 16** (see S2) recommends avoiding the use of hybrid requirements, if the geometric property can be individually verified. If hybrid requirements were avoided in the case study by following guideline S2, the process of automated requirement verification would require significantly less coordination.

When hybrid requirements are avoided, the original requirement is divided into a geometric and a non-geometric component. The handling of these interrelated parts is closely aligned with the bSI IDM standard and ISO 29481 (IDM). These standards offer a structured methodology for specifying who is responsible for delivering particular information and at what point in the process it must be provided. This approach facilitates a clear allocation of responsibilities for verifying the geometric and non-geometric aspects of a requirement. Furthermore, ISO 19650 supports this by providing a framework for managing information throughout the project lifecycle to ensure that these verifications remain coordinated and traceable through the use of a CDE.

**Table 16.** Guideline overview with related standards.

Guidelines		Standards							
		bSI					ISO		
		IDS	BCF	IFC	CDE	IDM	19650 (BIM)	12006 (IFD)	29481 (IDM)
<b>Strategic level</b>									
S1	Address ambiguity by formulating clear and machine-interpretable requirements to improve automated verification using bSI standards.	X	X	X			X		X
S2	Avoid hybrid requirements by separating geometric requirements that can be verified in the BIM model from those requiring alternative methods, if the geometric property can be individually verified.				X	X	X		X
<b>Operational level</b>									
O1	Establish a structured Common Data Environment for consistent storage of verification results and BIM model data.		X	X	X		X		X
O2	Use the BIM Collaboration Format for issue tracking during the verification process and integrate it with the CDE.		X	X	X		X		X
O3	Use standardised terminology from the Information Framework Dictionary (IFD) in all BIM model data to ensure consistency, interoperability and automation.			X				X	X
<b>BIM level</b>									
B1	Standardize the use of properties and data structures across BIM models to support consistent requirement verification.	X		X			X		X
B2	Establish a direct and traceable link between requirement texts and the corresponding building elements in the BIM model.			X			X	X	X
B3	Use the latest available IFC version to ensure long-term interoperability, enhance data exchange, and improve the efficiency of the automated requirement verification process.			X			X		X

### 4.1.2 Operational level

The operational level focuses on translating strategic intentions into practical actions that can be consistently implemented across projects. While strategic decisions define the overall direction and priorities, such as the adoption of automated requirement verification, the operational level ensures these goals are embedded in the daily workflows. This subsection addresses guidelines that focusing on how processes are carried out in practice.

#### Common Data Environment

To support a reliable automated requirement verification process, it is important to use a well-organised Common Data Environment. The CDE should serve as the central platform where all relevant model files, verification data and reports are stored (see **Figure 42**). A cloud-based CDE solution like Autodesk Construction Cloud used in the case study implementation, enables real-time collaboration between disciplines and ensure all users are working with the most current information. This not only reduces the risk of working with outdated files but also ensures that the automated requirement verifications are aligned with the most recent design iterations.

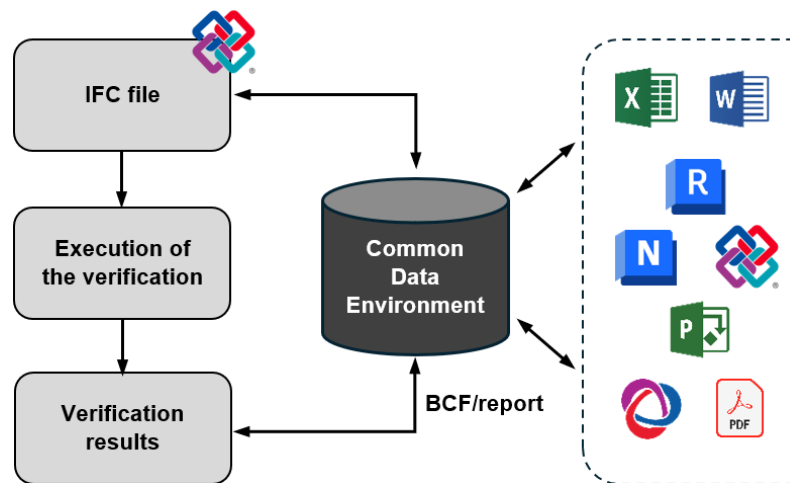
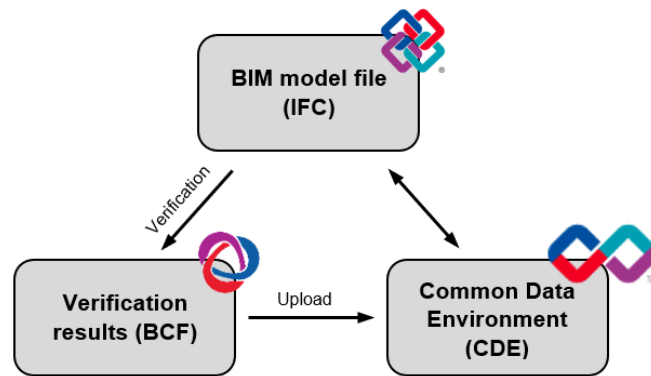


Figure 42. Visualisation of Common Data Environment.

The resulting guideline is presented in **Table 16** (see O1), which also includes the relevant standards. The use of a CDE aligns with the ISO 19650 standard for information management and the bSI standard in that area. It also connects with the bSI standards for IFC and BCF, including the ISO 16739 standard on IFC. The IFC file format facilitates automated requirement verification by supporting data exchange across different software environments. The BIM Collaboration Format further supports this process by enabling structured issue tracking and communication among project stakeholders. The use of a CDE is beneficial given that a fragmented workflow was identified as one of the implementation challenges during the practical execution of the verifications within the case study (see Subsection 3.3).

#### BIM Collaboration Format for issues

A fragmented workflow, characterized by disconnected communication between stakeholders and systems, remains a challenge which results in inefficiencies in the automated requirement verification process (see Subsection 3.3). The use of the BIM Collaboration Format facilitates structured issue tracking and communication throughout the project lifecycle. This ensures that issues are clearly documented and linked to specific elements in the BIM model. The BCF is designed for model-based communication which means that, in addition to text and images, references to objects can be included for easier navigation to the location in the BIM model. The use of BCF is presented in a guideline in **Table 16** (see O2) which also shows the associated BIM standards.



**Figure 43.** BIM Collaboration Format and relationship with IFC and CDE.

The use of BCF is related to the bSI IFC standard and ISO 16739 (IFC), as IFC provides an open data model for describing building elements and their interrelationships. BCF enables the linking of issues in the BIM models to specific elements which enhances data traceability and consistency. This is important in the process of automated requirement verification, where BCF information can be linked to model data stored in the IFC. Furthermore, integrating BCF with a CDE facilitates centralised data storage which supports data exchange and automation. **Figure 43** shows the relationship between BCF, IFC and CDE. This approach aligns with ISO 19650 which promotes collaboration and the use of standardised processes.

### Standardised terminology

Given the substantial amount of data embedded within a BIM model, it is essential to ensure consistent use of terminology. This becomes especially important with implementing an automated requirement verification process where adherence to uniform terms simplifies the execution. The case study revealed inconsistencies in the naming of data used within the model. For example, the Dutch term *dak platen* (roof panels) was applied to a workset, while *tunnel roof* was used as a family name for the same element type. Such variation introduces an additional interpretative layer which complicates the automation of requirement checks. To address this, a guideline has been developed to promote the use of standardised terminology throughout the model (see O3 in **Table 16**).

The International Framework Dictionary (IFD) enhances the interpretability of data exchanged via the IFC standard by providing contextual meaning. While IFC offers a foundational structure for describing model elements, it lacks detailed semantic definitions. The IFD library provides this by assigning precise meanings to object-related information. By linking IFC-based models to databases containing project specific data, the IFD framework introduces a level of flexibility in information handling. This aligns with the ISO 12006 (IFD) standard and the buildingSMART Data Dictionary (bSDD) which implements the concepts covered in ISO 12006. For example, within the dictionary there are definitions of what a Foundation and NetArea are linked to defining the requirements (see **Figure 44**). Furthermore, the IFD library can state that a foundation must be modelled with an *IfcSpace* and the NetArea must be expressed as a positive number in square meters. The adoption of standardised terminology as facilitated by IFD and bSDD support the automation process of requirement verification.

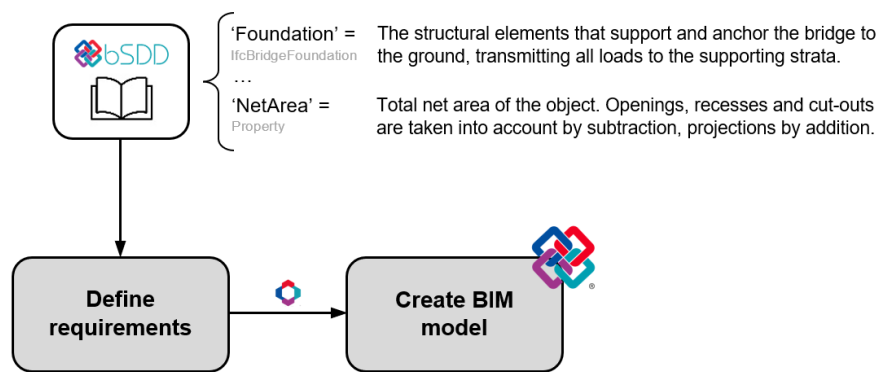


Figure 44. Dictionary based on ISO 12006 (IFD) and bSDD.

### 4.1.3 BIM level

The BIM level addresses guidelines that relate to the practical use of Building Information Modeling in the context of automated requirement verification. This level focuses on model specific aspects such as data structuring and the application of relevant BIM formats. The guidelines discussed in this subsection translate higher level intentions into specific actions within the BIM environment based on the insights gained from the implementation of the case study.

#### Inconsistent model data

The case study revealed inconsistencies in the parameter structure of the BIM model data. This issue is also identified as one of the implementation challenges for automated requirement verification (see Subsection 3.3). The IFC format uses properties to define custom data linked to building elements within the model. The IFC schema includes predefined properties organised into property sets associated with specific entities. Each entity contains multiple properties grouped together within these sets.

The IFC files from the case study contain a mixture of standard and custom IFC properties with the majority being custom properties. While the models include standard IFC property sets such as Pset\_QuantityTakeOff, Pset\_SlabCommon and Pset\_Dimensions, most of the property sets are custom like for example Identity Data, Constraints and others. These custom property sets also contain custom properties in the model, such as ROCO\_Local\_Width and ROCO\_CLE\_Base\_Offset. However, these custom properties were inconsistently available across the different building elements within the model making it difficult to verify the requirements. It is advisable to structure the BIM model using buildingSMART standardised properties before introducing custom ones. In addition to complicating the automation of requirement verification, the inconsistent use of parameters also diminishes the reusability of verification procedures across different BIM models and future projects. This emphasizes the need for consistent use of parameters and building information within the process of automated requirement verification.

This leads to the guideline presented in **Table 16** (see B1), where the use of standardisation of parameters is promoted. The use of standardised IFC properties defined by buildingSMART supports reliable automated requirement verification and aligns with both the bSI IFC standard and ISO 16739. Additionally, the IDS standard by bSI and the national BIM Base IDS are closely related to this guideline. The IDS defines the required information for the model and prescribes how it should be structured. For instance, it may specify that every IfcBridgePart must include a LoadBearingCapacity property within Pset\_BridgePartCommon. Implementing IDS enforces standardised parameter structures within the IFC schema which ensures that each building element contains the correct and consistent properties to simplify automated requirement

verification. If more standardisation of properties according to guideline B1 had been applied in the case study, the automation process could have been easier. Additionally, by using property standardisation, the reusability of verification procedures across different models and projects would have been significantly increased.

### **Linking model elements to requirements**

During the implementation of the case study, it became evident that there is no direct link between the requirement texts and the building elements within the BIM models. Although a Systems Engineer reviews the requirement text according to the SMART principle, this specific connection to the BIM model is lacking. This represents one of the main challenges hindering the process of automated requirement verification (see Subsection 3.3). While requirements are associated with object codes that are indirectly related to the BIM model, they are not explicitly connected to individual model elements. Furthermore, the requirement texts are often vague and ambiguous which leaves room for interpretation regarding which specific elements in the BIM model must be verified. As a result, manual selection of the relevant building elements is required for verification against each requirement. Therefore, the absence of a direct link between the requirement text and the model elements can be considered a significant barrier to the efficiency of automated requirement verification.

To address this issue, a guideline has been formulated recommending the establishment of a direct link between requirement texts and building elements (see B2 in **Table 16**). This connection can be achieved by associating each requirement text with individual objects in the BIM model at the element level. This approach aligns with ISO 16739 (IFC) and the bSI IFC standard, as it allows requirements to be linked to model elements via their unique identifier (GlobalID). Furthermore, it is advisable to use standardised terminology for both the requirement texts and BIM elements. This is in accordance with ISO 12006 (IFD), which supports the use of a shared and consistent vocabulary. Applying standard terms for the IFD reduces ambiguity and enhances the accuracy of automated requirement verification. Linking model elements to requirements is also consistent with ISO 19650, as this standard promotes the use of structured information and facilitates traceability. It ensures that data is properly managed and exchanged between the project stakeholders. If guideline B2 had been followed from the start, the implementation of the case study with using automated requirement verifications would have been easier due to the presence of direct links between the building elements in the model and the requirement.

### **IFC version**

The version of IFC is crucial for automated requirement verification, as the verification process relies on the IFC files of the BIM model. The most recent official version of IFC is 4.3.2.0, commonly referred to as IFC 4.3. This version has been published by ISO as the official ISO 16739-1:2024 standard (buildingSMART, [n.d.-g](#)). In addition, the next generation of IFC, version IFC 5.0, is currently under development by buildingSMART. The project from the case study uses the version IFC2x3, which is a widely adopted version. Many applications are capable of supporting both IFC2x3 and IFC4, as these versions share a similar structure (digiGO, [2023a](#)).

Despite the continued use of older IFC versions, there are notable differences in properties and property sets across various versions. For example, `IfcBuildingElement` (IFC2x3) was renamed to `IfcBuiltElement` in IFC4. As previously mentioned, properties are used in automated requirement verification by linking them to the IFC model data. It is therefore recommended to always use the latest IFC version to ensure the efficiency of the automated verification process and maintain interoperability with other tools. Using the latest IFC version enhances interoperability which is essential for exchanging model data between different stakeholders (aligns with ISO 19650). This leads to a guideline presented in **Table 16** (see B3). Each new IFC version introduces additional



features that are also aligned with other standards, such as bSI CDE, BCF and IDS. Because of this using the latest version and following guideline B3 positively contributes to the automation of the requirement verification process.

## 4.2 Validation through interview

To evaluate the applicability and practicality of the proposed guidelines for the process of automated requirement verification in the BIM environment, a series of interviews were conducted with professionals from the industry (see Appendix A3). The interviewees were selected based on their diverse roles within the organization in order to align with the strategic, operational and BIM levels at which the guidelines are categorised. This section is structured by briefly restating each guideline along with its corresponding level, followed by the findings based on the results of the interviews.

Strategic level	
S1	Address ambiguity by formulating clear and machine-interpretable requirements to improve automated verification using bSI standards.

It was highlighted that making requirements unambiguous and machine-interpretable from the start of a project is both necessary and challenging. A commonly cited method is to formulate requirements in a SMART manner tailored to BIM model elements. The use of Object Type Libraries (OTLs) was also mentioned which were also used within the case study. The OTL codes are used to structure the model elements according to the Object Breakdown Structure (OBS). However, a direct link between individual requirements and the corresponding OTL elements is lacking. This is why the guideline promotes the use of buildingSMART standards, such as the IDS which facilitates the definition of requirements in a machine-interpretable format. The interviewees indicated that implementing such structured approaches would positively impact the automation of the requirement verification process. Ideally, a standard IDS should be adopted industry-wide to reduce the need for custom coding. Also, it was mentioned that IDS offers a way to bridge the gap by linking requirements more directly to IFC model data. It was also mentioned that integrating tools like Relatics with the IDS format would be a major step. Nonetheless, concerns were raised about the fact that, in the early stages of a project, a BIM model is not always yet available. This might make it challenging to link requirements to the model in such a structured manner.

Strategic level	
S2	Avoid hybrid requirements by separating geometric requirements that can be verified in the BIM model from those requiring alternative methods, if the geometric property can be individually verified.

Although a screening of the requirements is initially done during the early stages of the project, where the SMART criteria is applied, the case study reveals that certain requirements may still be classified as hybrid requirements after this process. The interview results indicated that these hybrid requirements are sometimes mistakenly categorised as geometrical which prevents their complete verification within the BIM model. Consequently, a note is included to the verification output stating that the requirement also contains a non-geometrical component that cannot be verified in the BIM model. This necessitates coordination as the requirement cannot yet be considered resolved and the verification of its non-geometrical part may influence the results of the geometrical verification. Therefore, it is advisable to avoid the use of hybrid requirements as much as possible.



Operational level	
O1	Establish a structured Common Data Environment for consistent storage of verification results and BIM model data.

Using a Common Data Environment is essential for ensuring a reliable automated requirement verification process. The CDE can be used to store BIM models, as well as issues from the verifications or other project data. Currently, the company uses BIMcollab to manage issues arising from the BIM model. However, the implementation of a CDE, such as Autodesk Construction Cloud, can provide a central platform for storing both BIM model data and verification results. The interviews confirmed the importance of a CDE, highlighting its role in improving accessibility and coordination across different disciplines. Additionally, it was noted that the use of a CDE aligns with ISO 19650, which addresses information management throughout the project lifecycle. This further indicates that the guideline is well aligned with the proposed workflow outlined in Section 3.4. However, some interviewees acknowledged that there is often resistance to adopting new tools or systems, which can slow down the transition towards a centralised environment.

Operational level	
O2	Use the BIM Collaboration Format for issue tracking during the verification process and integrate it with the CDE.

The BIM Collaboration Format can be used to exchange model-based issues derived from IFC models which can be further integrated with a CDE. The issue management system BIMcollab, which is currently used within the company, also supports exporting issues in BCF. However, the integration with a CDE is lacking. By linking issues using BCF to a CDE it becomes possible to review issues at the element level directly within the model. Although interviewees indicated that a viewpoint as a graphical representation of the issue is often sufficient, this additional step demonstrates that such integration is feasible. Linking issues in a model-based manner using BCF and a CDE enhances the accessibility and traceability of the automated requirement verification process for the users. Moreover, integrating BCF with a CDE helps reduce workflow fragmentation.

Operational level	
O3	Use standardised terminology from the Information Framework Dictionary (IFD) in all BIM model data to ensure consistency, interoperability and automation.

Given the large volumes of data handled in a BIM model, it is essential to use terminology consistently. This is particularly important when implementing an automated requirement verification approach, as the use of uniform terminology simplifies the verification process. The Oosterweel case study revealed inconsistencies in the naming of data within the model. The interviews confirmed that terminology standardization is recognized as important factor for automating requirement verification. However, at present, no comprehensive dictionary is being used. It was noted during the interviews that Object Type Libraries are applied, which rely on standardized data structures to document information about objects. Nevertheless, the use of terminology standards such as ISO 12006 (IFD) or the buildingSMART Data Dictionary, which uses similar concepts, is currently not adopted in practice. The standardisation of terminology in accordance with these frameworks could substantially enhance the efficiency and reliability of the automation process.

BIM level	
B1	Standardize the use of properties and data structures across BIM models to support consistent requirement verification.

A BIM model contains properties that are used to define custom data linked to building elements. These properties are essential for performing automated requirement verifications as they hold the element-specific information necessary for executing the checks. The IFC models from the case study revealed limited use of standardised properties. According to the interviews, the frequent reliance on custom properties was primarily motivated by the need to track additional performance indicators, such as linking elements to quantities for financial purposes. Nevertheless, the use of standardised properties enables the reuse of verification approaches across different models and thereby supports the scalability and consistency in the verification process.

BIM level	
B2	Establish a direct and traceable link between requirement texts and the corresponding building elements in the BIM model.

In the process of automated requirement verification, it is beneficial if the requirement text contains a direct link to the building elements in the BIM model that are subject to the verification. Nevertheless, during the implementation of the case study it became clear that none of the requirement texts were directly associated with the building elements within the BIM model. However, the requirements were exclusively associated with object codes in the model. The interviews indicated that alternative methods are frequently used to match requirement texts with the relevant model elements. These methods include filtering based on physical names, OTL codes or worksets. Furthermore, interviewees indicated that establishing a direct connection between the requirement text and building elements using a unique identifier is not considered practical. This is due to the possibility that design changes may result in altered identifiers which could disturb this link. However, it is possible to use the Information Delivery Specification to explicitly link requirement texts to IFC model data to facilitate the verification of model elements against the specified requirements. This approach is more structured and addresses the previously mentioned practical challenge. Therefore, the existence of a clear link between the requirement text and the corresponding building elements is essential for the effective automation of the requirement verification process.

BIM level	
B3	Use the latest available IFC version to ensure long-term interoperability, enhance data exchange, and improve the efficiency of the automated requirement verification process.

To promote interoperability and improve the efficiency of automated requirement verification, it is recommended to use the most recent available IFC version. The BIM models used in the case study were based on IFC2x3 version, which remains widely adopted in current practice. However, different IFC versions define varying naming for property sets which can impact consistency and interpretation. The interview results indicated that the continued reliance on older versions (e.g. IFC2x3), is often driven by client preferences and established workflows. Although newer versions like IFC4 provide enhanced data structures and more comprehensive property definitions, their adoption in practice remains limited. This suggests that while the guideline supports technological advancements in data exchange and automation, its practical implementation depends on project specific requirements and the willingness of stakeholders to adopt a more recent version of the standard. Promoting the use of updated IFC versions among clients and project teams is therefore essential to further realise the benefits of automated requirement verification.

# 5. Discussion

This chapter discusses the validity of the research, followed by an interpretation of the results. It also addresses the limitations of the study and explores the implications for practice and future research.

## 5.1 Validity of the research

The validity of this research is influenced by the methodological choices made throughout the research process. The internal validity was supported by the use of the Oosterweel project which provided a relevant context for testing the proposed verification workflow. The case study allowed the semi-automated requirement verification process to be implemented effectively, covering classification, rule interpretation, model preparation, execution and reporting of the results. This ensured that the research outcomes were based on practical challenges and real data.

To ensure the research approach aligned with the practical needs, two round of interviews were conducted with the company experts. The first round aimed to gain a deeper understanding of the existing requirement verification process, including the challenges encountered in practice and user experiences of the different software tools. These insights directly contributed to the design of the semi-automated verification workflow by highlighting issues and areas for improvement. A second round of interviews was conducted after the guidelines had been developed to validate their relevance. These interviews provided useful feedback on the proposed recommendations, helping to refine them to the company's context.

However, external validity is more limited. The study is based on a single case study and all interviews were conducted within Witteveen+Bos. Although this approach ensured internal alignment and contextual relevance from the engineering firms perspective, it limits how generalisable the findings are to other projects and organisations. Furthermore, the use of industry standards such as IFC, ISO 19650 and the openBIM method ensured that the workflow applied in this research reflected common practices in the construction sector. However, the implementation still revealed several challenges such as for example the need for manual intervention for requirement classification. This indicates that while the approach is suitable to the case study, further development and testing across different projects is necessary to improve its reliability and broader applicability.

## 5.2 Result interpretation

The results of this research show that a semi-automated approach to requirement verification within the BIM environment is feasible and significantly more efficient than current manual methods. Integrating a machine learning model, Python scripts and Autodesk tools such as Navisworks and Autodesk Construction Cloud optimised the verification process. Furthermore, the RASE language was found to be the most effective method for structuring requirements due to its standardisation, readability and expressiveness. The language was also well aligned with industry standards which support compatibility with IFC based model checking and strengthen the automation potential.

## 5.3 Research limitations

While this research provides insights into the development of a semi-automated requirement verification process within the BIM environment, it is important to recognize the limitations of the study. Understanding these limitations is essential to accurately interpret the results and identify areas for improvement in the future. The key limitations to this research are listed below:

- **Scope limited to one case study**

The research used a single infrastructure project as a case study which limits the generalizability of the findings. While the case was useful for testing the feasibility of the proposed workflow, other projects may involve other types of requirements and workflows not addressed in this study.

- **Only internal validation**

Both rounds of interviews and all testing was done through employees from Witteveen+Bos. While this helped to ensure relevance to the company's internal processes, it introduces potential bias and reduces the external validity of the findings. Broader validation across multiple project stakeholders would provide a more generalizable conclusion.

- **Fragmented use of tools**

The development process relied on a combination of individual tools such as Relatics, Autodesk Revit/Navisworks, Autodesk Construction Cloud and Python. Each tool was used for a different task, but they lacked overall integration. Consequently, manual effort was needed to transfer data between tools to align information and maintain consistency across the different systems. For this process to be practical it is necessary to integrate these steps into a more unified and automated platform.

- **No full automation of the process**

The research focused on developing and testing a proof of concept to demonstrate the feasibility of automating requirement verification in the BIM environment. While the results confirmed that several steps such as classification of requirements and verifications, can be partially automated, the overall process is not fully automated. This means the full workflow has not yet been integrated into a scalable solution. This restricts the direct use of the approach and highlights the need for further development, research and testing to achieve full automation in practice.

## 5.4 Research implications

From a practical perspective, the findings show that automation can improve the efficiency of requirement verification processes, even when full automation has not yet been achieved. However, the fragmented nature of the tools combined with the need for manual coordination shows the importance of further integration. In terms of research implications, this study helps to bridge the gap between the theoretical approaches proposed in academic literature and the practical implementation within the construction industry. While many studies promote the use of automation, it is not always shown how it can be applied in the BIM environment using multiple software tools. By basing the research on a case study and validating the findings through interviews, this study improves the understanding of how a semi-automated verification process can be designed.

# 6. Conclusion

This chapter presents a conclusion by answering the research questions formulated at the beginning of this report. The objective of this research was to develop a (semi-)automated process that optimises the verification of requirements within the BIM environment. By exploring the possibilities of developing such a process, this research aimed to optimize requirement verification and improve efficiency by overcoming current limitations. To achieve this objective, a main research question was formulated, supported by four sub-research questions. The conclusion is drawn by first addressing the sub-questions followed by answering the main research question.

## 6.1 Sub-research questions

### **SQ1: What are current challenges and limitations in the verification process of requirements in 3D models?**

While automated requirement verification shows considerable potential, the process is still subject to several challenges and limitations, making it an ongoing topic of academic and industry interest. One of the most fundamental and complex challenges identified in the literature is the translation of human-readable texts into a structured computer interpretable format. The difficulty of converting requirement texts also emerged as a significant challenge during the execution of this research. A closely related problem is the inherent ambiguity in many requirement texts which often leaves room for subjective interpretation. As a result, human clarification remains necessary, which limits the efficiency of the automated approach. It can also be concluded that human input is required when processing hybrid requirements, which consist of both geometric and non-geometric parts.

Additionally, there are also some more practical limitations to the process of automated requirement verification. The inconsistent use of element properties within BIM models hinders automation and considerably reduces the reusability of verification rules across different projects and models. Furthermore, the absence of explicit allocation of requirements to specific elements in the model often makes it unclear which components need to be verified. Finally, it can be concluded that the overall workflow of automated requirement verification remains fragmented as it typically depends on multiple tools. Nevertheless, the findings suggest that there is significant potential for improving the efficiency of their integration.

### **SQ2: What existing software is used for requirement verification in 3D models and what are the alternatives?**

The integration of BIM technology with standardised digital construction models has introduced tools that support the verification process. These tools can help with automated requirement verification and compliance checks. A variety of software tools are available for requirement verification with each offering different functionalities. Among the analysed tools, SMC, Navisworks, BIM Assure, SMARTreview and Verifi3D, all support IFC compatibility and rule-based verification to some extent. However, they vary significantly in features such as report generation, the ability to merge BIM files, direct edit capabilities and the depth of rule-based checks. Solibri Model Checker and Navisworks provide the most comprehensive functionalities, particularly in merging models and generating reports while BIM Assure stands out for allowing to edit non-geometrical parameters. Alternatives like Verifi3D and SMARTreview offer cloud-based or plug-in

solutions with varying levels of automation and flexibility. For the purpose of this research, Navisworks was used to conduct requirement verifications from the analysed software tools. This was primarily based on the fact that Navisworks offers one of the most extensive sets of functionalities among the analysed tools. Moreover, it already is in use within the company which makes it the preferred option from a licensing and practical implementation perspective.

**SQ3: What is the most effective method for translating human-readable requirements into a structured format?**

To enable automated requirement verification, human-readable requirements must first be translated into a structured machine readable format. Various methods have been developed to support this translation process, including RASE, BIMRL and BERA. The BIM Rule Language shows strong integration with IFC models and spatial reasoning, while BERA emphasizes domain specific handling of building elements. However, it can be concluded that RASE stands out as the most effective method due to its use of a standardised schema, clear semantic structure based on four operators and its accessibility for users without programming expertise. This allows complex requirement texts to be consistently interpreted which makes RASE the most suitable approach for converting requirement texts into a structured format ready for automated requirement verification.

**SQ4: What are the key steps in developing a (semi-)automated process to verify requirements in a 3D model?**

There are several essential steps for developing a (semi-)automated process for requirement verification within a BIM model. Based on the findings of this research, the process can be divided into five key phases. The first phase concerns the classification of requirements to determine whether a requirement can be verified using a BIM model or whether an alternative verification method is necessary. For this categorization, a machine learning model was developed in Python. The interpretation of the requirements can be regarded as the key step in the process. The case study revealed that many requirements are ambiguous and open to multiple interpretations, which significantly complicates the automation process. To address this, the RASE mark-up language was applied to enable effective tagging and structuring of the requirement texts. From the case study it could be concluded that clearly formulated requirements, explicitly linked to elements in the model, could significantly reduce the time needed for interpretation. Once the requirements have been interpreted, the subsequent verification and reporting phases can be executed in a relatively straightforward manner.

## **6.2 Main research question**

**MQ: How can a (semi-)automated process be developed to validate and verify requirements within a 3D model?**

To answer the main research question, the process of automated requirement verification was systematically mapped out. A case study was used to test the theory in a practical context. The requirement classification step proved to be a crucial foundation for the process of semi-automated requirement verification for the proposed approach in this research. It involved differentiating between requirements that could be verified within a BIM model (geometrical) and those that cannot (non-geometrical). A Python script was developed to classify textual requirements accordingly. The model was tested on a dataset of 614 requirements and its predictions were compared against their actual classifications. It was found that the model

correctly identified 488 requirements as non-geometrical (true negatives) and 64 requirements as geometrical (true positives). Some misclassifications were observed with the model occasionally predicting geometrical requirements as non-geometrical and vice versa. However, these errors accounted for less than 5% of the total dataset. The model achieved an overall accuracy of 0.95 which indicates that 95% of the requirements were classified correctly. It was observed that misclassifications often occurred when the term “BIM” or related terminology was absent from the requirement text which potentially misled the model.

The second step in the process of automated requirement verification focused on manual rule interpretation. It was found that, despite the requirements being formulated according to the SMART criteria at the start of the project, the requirement texts remained open to interpretation. Therefore, the RASE mark-up language was applied, as it appeared to be the most effective method for translating human-readable requirements into a structured format based on the literature review. However, despite the use of RASE, the absence of a direct link to specific elements within the model made the interpretation process more complex. Literature indicated that accurately interpreting requirements within a BIM model is the most error-prone aspect of the verification process. Due to the absence of a direct link between the requirement text and the model elements, significant effort was required to prepare the building model data (step 3 in the process). This involved identifying which model elements were referenced in the requirement texts and needed to be verified. It was found that the steps of rule interpretation and the preparation of the building model data was very time consuming and inefficient within the case study which complicated the process of automated requirement verification.

In contrast, the execution phase of the verification process showed promising outcomes after the requirements were interpreted correctly. It was found that the semi-automated execution of the verifications showed a significant time reduction, although the exact time savings were not always directly quantifiable. Finally, the results were documented in a Common Data Environment where non-compliances were marked as issues. Once the design is revised, re-verification can be done against the specified requirement. Overall, it was found that ambiguity in requirement texts, inconsistent use of parameters, and the absence of explicit links to model elements made the process of automated requirement verification unnecessarily complex. Based on theoretical insights and the case study findings, a set of guidelines was developed to support the verification process. These guidelines were validated through expert interviews, where practitioners generally affirmed their relevance across strategic, operational and BIM level guidelines.



# 7. Recommendations

This chapter presents the recommendations derived from the findings of this research. Based on the development and evaluation of a (semi-)automated process for requirement verification valuable insights were obtained. The implementation of an automated process shows potential for increasing efficiency, reducing errors and enhancing collaboration in construction projects. However, challenges related to system integration, data consistency and requirement interpretation remain and require further attention.

To address these insights, the recommendations are divided into two sections. Section 7.1 provides recommendations for practice which aim at enhancing the efficiency, accuracy and integration of requirement verification processes in a professional setting. Section 7.2 focuses on directions for future research by identifying areas where further investigation is needed. The recommendations aim to address the disconnect between academic research and practical implementation while supporting further development of automated requirement verification.

## 7.1 Recommendations for practice

The following recommendations focus on improving the process of automated requirement verification in practice:

1. The case study showed that project requirements from the contract are often formulated ambiguously which leads to challenges in their interpretation and verification. To address this issue, it is recommended that project requirements are expressed in a computer interpretable format such as RASE and the IDS standard. By linking the model elements and the corresponding requirement texts, the risk for misinterpretation can be significantly reduced. Furthermore, it is advisable to encourage clients to explicitly define within contractual documentation, how requirement verification is to be performed in the BIM environment. This should include verification methods, data formats and responsibilities. These measures help to reduce ambiguity and ensure consistent and verifiable requirements throughout the project and set clear expectations from the start.
2. In addition to making requirements computer interpretable, it is recommended to avoid hybrid requirements by clearly separating geometric and non-geometric components of the requirements during the formulation stage. This approach reduces the required coordination between disciplines and minimizes manual interpretation which enhances the potential for automation.
3. It is recommended to further explore the adoption of open standards that are not yet actively applied in current practice within the company. Relevant examples include the Information Delivery Specification (IDS), the BIM Collaboration Format (BCF), the Information Delivery Manual (IDM) and the International Framework for Dictionaries (IFD). The IDS makes it possible to define requirements in a computer interpretable format. BCF enables users of different BIM applications to communicate clearly about issues related to IFC models. The IDM standard provides a structured approach for documenting responsibilities for delivering specific information. The IFD offers standardised terminology to ensure consistent data exchange. Together, these standards support interoperability and provide a more reliable foundation for automating the verification of project requirements.

4. To support efficient collaboration and clear documentation, it is recommended to use a Common Data Environment as a centralised place for storing project documents and BIM models. Also, it is recommended to use the CDE, Autodesk Construction Cloud in this case, as an issue management system to further integrate the verification process. The identified issues from the verifications can be stored in the CDE by either using the BIM Collaboration Format or a verification report. This ensures all relevant information remains accessible and well-coordinated.
5. To improve consistency and reduce manual effort it is recommended to integrate the requirement management system (Relatics in this case), directly with the modelling software. This integration helps bridge the gap between the textual requirements and visual model elements reducing errors and disconnected systems which improves the verification process. For example, Autodesk Docs can be used for integration with Relatics. Autodesk Docs is a cloud-based solution for document and data management within Autodesk Construction Cloud which is used as CDE in this research.
6. To improve the effectiveness and scalability of automated requirement verification, it is recommended to adopt a consistent and standardised property structure within the BIM models. If possible, the use of predefined IFC property sets is advisable before introducing custom properties.

## 7.2 Recommendations for future research

The following recommendations are intended to support future research on automated requirement verification:

1. This research was limited to a single case study which focused on the Oosterweel project. To assess the generalizability of the proposed process, future research should apply it to multiple case studies across different disciplines. This would help to identify potential limitations and necessary refinements for broader implementation.
2. While this research developed a semi-automated verification process, some tasks may be fully automatable especially those relying on geometry. Future research should explore which requirement types are suitable for complete automation and under what conditions this is possible.
3. It is recommended to investigate the machine learning model to improve the classification of requirements, especially those that are hybrid or ambiguous. Future research should focus on analysing the misclassifications made by the model to determine whether these can be addressed to increase its accuracy.
4. Future research should look at the expanded application of AI beyond the current machine learning model used for requirement classification within the automated requirement verification process. It is advisable to investigate how AI can enhance classification accuracy, interpret (ambiguous) requirements and support decision-making to further improve the efficiency and reliability of the automated requirement verification process.
5. It is recommended to link the verification process to a Digital Twin to enable continuous validation throughout the operation phase. Future research should study how the automated verification process can be extended beyond the design phase by incorporating sensor data and real-time monitoring to ensure ongoing compliance with

the requirements. Also, the use of AR/VR technology should be explored to visualize the BIM model and verification results.

6. It is advisable that future research investigates the impact of automated requirement verification on project outcomes such as cost savings, error reduction, rework and efficiency. Quantifying these effects across multiple projects can provide clearer insight into the practical value of implementing an automated verification process.
7. It is recommended to investigate how automated requirement verification aligns with existing legal and contractual frameworks. This includes assessing whether automated verifications can be considered equivalent to manual inspections and clarifying the liabilities when using automated verification software for regulatory compliance.

# References

- Abdalhameed, B. F. (2023). Based BIM techniques to clash detection for construction projects. *Periodicals of Engineering and Natural Sciences*, 11(1), 239-245. <https://doi.org/10.21533/pen.v11.i1.93>
- Accord. (n.d.). Digital building permit and compliance verification. Retrieved January 29, 2025, from <https://accordproject.eu/>
- Adekunle, S. A., Aigbavboa, C., Ejohwomu, O., Ikuabe, M., & Ogunbayo, B. (2022). A critical review of maturity model development in the digitisation era. *Buildings*, 12(6), 858. <https://doi.org/10.3390/buildings12060858>
- Afsari, K., Eastman, C., & Shelden, D. (2017). Building Information Modeling data interoperability for cloud-based collaboration: Limitations and opportunities. *International Journal of Architectural Computing*, 15(3), 187-202. <https://doi.org/10.1177/1478077117731174>
- Al-Ali, A.R., Zuolkernan, I.A., Rashid, M., Gupta, R. & Alikarar, M. (2017). A smart home energy management system using IoT and big data analytics approach. *IEEE Transactions on Consumer Electronics*, 63(4), 426-434. <https://doi.org/10.1109/TCE.2017.015014>
- Al-Turki, D., Hettiarachchi, H., Gaber, M. M., Abdelsamea, M. M., Basurra, S., Iranmanesh, S., Saadany, H. & Vakaj, E. (2024). Human-in-the-Loop Learning with LLMs for Efficient RASE Tagging in Building Compliance Regulations. *IEEE Access*, 12, 185291-185306. <https://doi.org/10.1109/ACCESS.2024.3512434>
- Altieri, A. (2024, August 16). *The future of BIM: Predictions & expectations*. Vectorworks. Retrieved January 24, 2025, from <https://www.vectorworks.net/en-US/newsroom/the-future-of-bim>
- Asgari, Z., & Rahimian, F. P. (2017). Advanced virtual reality applications and intelligent agents for construction process optimisation and defect prevention. *Procedia Engineering*, 196, 1130-1137. <https://doi.org/10.1016/j.proeng.2017.08.070>
- (2) Autodesk. (2024, October 31). *Supported file formats and applications for Autodesk Navisworks*. Retrieved April 1, 2025, from <https://www.autodesk.com/support/technical/article/caas/sfdcarticles/sfdcarticles/Supported-file-formats-and-applications-for-Autodesk-Navisworks-2009-to-2012.html>
- Autodesk. (2025). *Verifi3D by Xinaps*. Retrieved January 31, 2025, from <https://apps.autodesk.com/BIM360/en/Detail/Index?id=2053598109832362600&appLang=en&os=Web>
- (7) Autodesk. (n.d.-a). *To create a clash report*. Retrieved April 1, 2025, from <https://help.autodesk.com/view/NAV/2022/ENU/?guid=GUID-3D392AAA-EB37-452A-B621-72D6E129C9D0>

- (12) Autodesk. (n.d.-b). *Merge files*. Retrieved April 1, 2025, from <https://help.autodesk.com/view/NAV/2023/ENU/?guid=GUID-79DE7F56-59CF-4786-A31D-170E11BF89EC>
- (14) Autodesk. (n.d.-c). *Welcome to the Navisworks 2025 help*. Retrieved April 1, 2025, from <https://help.autodesk.com/view/NAV/2025/ENU/>
- Baduge, S. K., Thilakarathna, S., Perera, J. S., Arashpour, M., Sharafi, P., Teodosio, B., Shringi, A. & Mendis, P. (2022). Artificial intelligence and smart vision for building and construction 4.0: Machine and deep learning methods and applications. *Automation in Construction*, 141, 104440. <https://doi.org/10.1016/j.autcon.2022.104440>
- Beetz, J., Van Leeuwen, J., & De Vries, B. (2009). IfcOWL: A case of transforming EXPRESS schemas into ontologies. *AI Edam*, 23(1), 89-101. <https://doi.org/10.1017/S0890060409000122>
- Bouwdigitaliseringsraad. (2014, November). BIM Kenniskaarten & poster. *Kenniskaart 1: Nederlandse BIM levels*. [https://www.bimloket.nl/documents/Kenniskaart\\_1\\_-\\_Nederlandse\\_BIM\\_Levels.pdf](https://www.bimloket.nl/documents/Kenniskaart_1_-_Nederlandse_BIM_Levels.pdf)
- BibLus. (2023, 27 February). *What information can be managed with the IDS standard?* Retrieved January 29, 2025, from <https://biblus.accasoftware.com/en/what-information-can-be-managed-with-the-ids-standard/>
- Bigai, S., & Santos, E. T. (2024). Implementing Information Delivery Specification (IDS) encoding for a BIM object standard based on model uses. *Proceedings of the 41st International Conference of CIB W78*, Marrakech, Morocco. <http://itc.scix.net/paper/w78-2024-85>
- BIMcollab (2024, September 10). *What is IFC in BIM? Key Benefits for Collaboration*. BIMcollab. Retrieved 29 January, 2025, from <https://www.bimcollab.com/en/resources/blog/what-is-ifc/>
- Bilal, M., Oyedele, L. O., Qadir, J., Munir, K., Ajayi, S.O., Akinade, O.O., Owolabi, H.A., Alaka, H.A. & Pasha, M. (2016). Big Data in the construction industry: A review of present status, opportunities, and future trends. *Advanced Engineering Informatics*, 30(3), 500-521. <https://dx.doi.org/10.1016/j.aei.2016.07.001>
- Bjørn, G., Knud, M., Christoph, M., Klakegg, O. J., & Lizhen, H. (2022). Towards an improved framework for enterprise BIM: the role of ISO 19650. *ITcon*, 27, 1075-1103, <https://doi.org/10.36680/j.itcon.2022.053>
- Bloor M., & Owen J. (1995). *Product data exchange*. UCL Press, London, pp. 262
- Boje, C., Guerriero, A., Kubicki, S., & Rezgui, Y. (2020). Towards a semantic Construction Digital Twin: Directions for future research. *Automation in construction*, 114, 103179. <https://doi.org/10.1016/j.autcon.2020.103179>
- Borkowski, A. S. (2023). Evolution of BIM: epistemology, genesis and division into periods. *Journal of Information Technology in Construction*, 28, 646-661. <https://doi.org/10.36680/j.itcon.2023.034>

- Borrmann, A., König, M., Koch, C., & Beetz, J. (2018). Building Information Modeling: Why? What? How?. In: Borrmann, A., König, M., Koch, C., Beetz, J. (Eds.), *Building Information Modeling*. Springer. [https://doi.org/10.1007/978-3-319-92862-3\\_1](https://doi.org/10.1007/978-3-319-92862-3_1)
- Brilakis, I., Pan, Y., Borrmann, A., Mayer, H. G., Rhein, F., Vos, C., Pettinato, E., & Wagner, S. (2019). Built environment digital twinning. *International Workshop on Built Environment Digital Twinning presented by TUM Institute for Advanced Study and Siemens AG*. <https://doi.org/10.17863/CAM.65445>
- Bruggeman, E. M. (2020). Legal aspects of Building Information Modelling: The ‘Dutch approach’: An overview. *OffSite Manufacture & the future of the Construction Industry*, 22.
- Bucher, D., & Hall, D. (2020). Common Data Environment within the AEC Ecosystem: moving collaborative platforms beyond the open versus closed dichotomy. In *EG-ICE 2020 Proceedings: Workshop on Intelligent Computing in Engineering* (pp. 491-500). Universitätsverlag der TU Berlin. <https://doi.org/10.3929/ethz-b-000447240>
- buildingSMART. (2017, April). *Regulatory Room Working Group: Report on Open Standards for Regulations, Requirements and Recommendations Content*. Retrieved February 18, 2025, from <https://buildingsmart.org/wp-content/uploads/2017/11/17-11-08-Open-Standards-for-Regulation.pdf>
- buildingSMART. (n.d.-a). *About buildingSMART International*. buildingSMART International. Retrieved February 18, 2025, from <https://www.buildingsmart.org/>
- buildingSMART. (n.d.-b). *openBIM®*. buildingSMART International. Retrieved February 18, 2025, from <https://www.buildingsmart.org/about/openbim/>
- buildingSMART. (n.d.-c). *Scope of buildingSMART standards*. buildingSMART International. Retrieved February 18, 2025, from <https://technical.buildingsmart.org/>
- buildingSMART. (n.d.-d). *What is Information Delivery Specification (IDS)*. buildingSMART International. Retrieved February 18, 2025, from <https://www.buildingsmart.org/what-is-information-delivery-specification-ids/>
- buildingSMART. (n.d.-e). *BIM Collaboration Format (BCF)*. buildingSMART International. Retrieved February 18, 2025, from <https://technical.buildingsmart.org/standards/bcf/>
- buildingSMART. (n.d.-f). *Information Delivery Manual (IDM)*. buildingSMART International. Retrieved February 18, 2025, from <https://technical.buildingsmart.org/standards/information-delivery-manual/>
- buildingSMART. (n.d.-g). *Industry Foundation Classes (IFC)*. buildingSMART International. Retrieved on February 18, 2025, from <https://technical.buildingsmart.org/standards/ifc/>
- Cai, H., Jeon, J., Xu, X., Zhang, Y., & Yang, L. (2020). Automating the generation of construction checklists. *Joint Transportation Research Program Publication*. Purdue University. <https://doi.org/10.5703/1288284317273>

- Casini, M. (2021). *Construction 4.0: Advanced Technology, Tools and Materials for the Digital Transformation of the Construction Industry* (1<sup>st</sup> ed.). Woodhead Publishing.
- Cassandro, J., Mirarchi, C., Gholamzadehmehr, M., & Pavan, A. (2024). Advancements and prospects in building information modeling (BIM) for construction: A review. *Engineering, Construction and Architectural Management*.  
<https://doi.org/10.1108/ECAM-04-2024-0435>
- Chan, D. W., Olawumi, T. O., & Ho, A. M. (2019). Perceived benefits of and barriers to Building Information Modelling (BIM) implementation in construction: The case of Hong Kong. *Journal of Building Engineering*, 25, 100764.  
<https://doi.org/10.1016/j.jobbe.2019.100764>
- Cheng, J. C., & Lu, Q. (2015). A review of the efforts and roles of the public sector for BIM adoption worldwide. *Journal of Information Technology in Construction (ITcon)*, 20(27), 442-478. <https://www.itcon.org/2015/27>
- Cherpa, C. (1992). Natural language processing, pragmatics, and verbal behavior. *The Analysis of verbal behavior*, 10, 135-147. <https://doi.org/10.1007/BF03392880>
- Cheung, S. O., & Pang, K. H. Y. (2013). Anatomy of construction disputes. *Journal of construction engineering and management*, 139(1), 15-23.  
[https://doi.org/10.1061/\(ASCE\)CO.1943-7862.0000532](https://doi.org/10.1061/(ASCE)CO.1943-7862.0000532)
- Cho, D. W., Kim, I. H., Seo, J. C., & Kim, J. H. (2011). A Study on Usage of IFD of Open BIM-Based Library. *Korean Journal of Computational Design and Engineering*, 16(2), 137-145.
- Clayton, M., Fudge, P., & Thompson, J. (2013, July). Automated plan review for building code compliance using BIM. In *Proceedings of 20th International Workshop: Intelligent Computing in Engineering (EG-ICE 2013)* (pp. 1-10).
- COB (2018). *Oosterweelverbinding: grensverleggend ontwerpen in BIM*. Nederlands kenniscentrum voor ondergronds bouwen en ondergronds ruimtegebruik.  
<https://www.cob.nl/magazines-brochures-en-nieuws/verdieping/verdieping-jan2018/digi-taal/oosterweelverbinding-grensverleggend-ontwerpen-in-bim/>
- Crowston, K., Liu, X., & Allen, E. E. (2010). Machine learning and rule-based automated coding of qualitative data. *Proceedings of the American Society for Information Science and Technology*, 47(1), 1-2. <https://doi.org/10.1002/meet.14504701328>
- Dakhil, A., Alshawi, M., & Underwood, J. (2015, June). BIM client maturity: Literature review. In *Proceedings of the 12th International Post-Graduate Research Conference* (pp. 1-12).
- Day, M. (2019, October 8). *Review: Xinaps Verifi3D*. AEC Magazine. Retrieved January 31, 2025, from <https://aecmag.com/technology/review-xinaps-verifi3d/>
- De Grote Verbinding (n.d.). Oosterweelverbinding. Retrieved February 26, 2025, from <https://www.degroteverbinding.be/nl/projecten/oosterweel/over>



- de Marco, G., Slongo, C., & Siegele, D. (2024). Enriching Building Information Modeling Models through Information Delivery Specification. *Buildings*, 14(7), 2206. <https://doi.org/10.3390/buildings14072206>
- Diara, F., & Rinaudo, F. (2020). IFC Classification for FOSS HBIM: Open Issues and a Schema Proposal for Cultural Heritage Assets. *Applied Sciences*, 10(23), 8320. <https://doi.org/10.3390/app10238320>
- digiGO. (2020, September 20). *Controleren op de BIM Basis ILS* (Version 1.0). Retrieved February 29, 2025, from <https://www.digigo.nu/ilsen-en-richtlijnen/bim-basis-ils/handleidingen/>
- digiGO. (2023a). *BIM Base IDS*. Retrieved February 18, 2025, from <https://www.digigo.nu/en/ilsen-en-richtlijnen/bim-base-ids/>
- digiGO. (2023b). *Wat is de Bouwdigitaliseringsraad (BDR)?* Retrieved February 18, 2025, from <https://www.digigo.nu/wat-is-digigo/bouwdigitaliseringsraad/>
- digiGO (2023c). *Open standaarden*. Retrieved January 10, 2025, from <https://www.digigo.nu/standaarden/wat-zijn-open-standaarden/>
- Dimyadi, J., & Amor, R. (2013). Automated building code compliance checking—where is it at. *Proceedings of CIB WBC*, 6(1).
- Dimyadi, J., Solihin, W., Eastman, C., & Amor, R. (2016, October). Integrating the BIM Rule Language into Compliant Design Audit Processes. In *Proceedings of the 33th CIB W78 international conference* (pp. 1-10).
- Dimyadi, J., Solihin, W., Amor, R. (2018). Using IFC to Support Enclosure Fire Dynamics Simulation. In: Smith, I., Domer, B. (Eds.), *Advanced Computing Strategies for Engineering: EG-ICE 201* (Lecture Notes in Computer Science, Vol. 10864). Springer. [https://doi.org/10.1007/978-3-319-91638-5\\_19](https://doi.org/10.1007/978-3-319-91638-5_19)
- Dimyadi, J., Fernando, S., Davies, K., & Amor, R. (2020, February). Computerising the New Zealand building code for automated compliance audit. New Zealand Built Environment Research Symposium (NZBERS).
- Ding, Y., Ma, J., & Luo, X. (2022). Applications of natural language processing in construction. *Automation in Construction*, 136. <https://doi.org/10.1016/j.autcon.2022.104169>
- Eastman, C. (1976). General purpose building description systems. *Computer-Aided Design*, 8(1), 17-26. [https://doi.org/10.1016/0010-4485\(76\)90005-1](https://doi.org/10.1016/0010-4485(76)90005-1)
- Eastman, C., & Henrion, M. (1977). GLIDE: A language for design information systems. *ACM SIGGRAPH Computer Graphics*, 11(2), 24-33. <https://doi.org/10.1145/965141.563863>
- Eastman, C.M. (1999). *Building Product Models: Computer Environments Supporting Design and Construction*. CRC Press. <https://doi.org/10.1201/9781315138671>
- Eastman, C., Lee, J. M., Jeong, Y. S., & Lee, J. K. (2009). Automatic rule-based checking of building designs. *Automation in construction*, 18(8), 1011-1033. <https://doi.org/10.1016/j.autcon.2009.07.002>

- Eastman, C., Teicholz, P., Sacks, R., & Liston, K. (2018). *BIM handbook: A guide to building information modeling for owners, managers, designers, engineers, and contractors* (2nd ed., 381-382). John Wiley & Sons.
- Eichler, C. C., Schranz, C., Krischmann, T., & Urban, H. (2023). BIMcert Handbook: Basic Knowledge openBIM. Edition 2023. <https://doi.org/10.34726/5383>
- Eid, A. S., Aboulnaga, M. M., & Mahmoud, A. H. (2020). Future Cities for Climate Action: Automated Code Compliance Checking in Reference to Energy Efficiency Building Regulations. In *Green Buildings and Renewable Energy: Med Green Forum 2019-Part of World Renewable Energy Congress and Network* (pp. 71-86). Springer International Publishing.
- Eischet, O. (2022, 13 October). BIM adoption across the world: A Global Outlook - Specter Automation Insights - Medium. Retrieved from <https://medium.com/specter-automation-insights/bim-adoption-across-the-world-a-global-outlook-1b3879f23bc6>
- Elanchezhian, C., Selwyn, T. S. & Sundar, G. S. (2007). *Computer aided manufacturing*. Firewall Media. Second edition.
- Elnadi, M., Abdallah, Y.O (2024). Industry 4.0: critical investigations and synthesis of key findings. *Manag Review Quarterly*, 74, 711–744. <https://doi.org/10.1007/s11301-022-00314-4>
- Fay, C.D. (2020). Computer-Aided Design and Manufacturing (CAD/CAM) for Bioprinting. In: Crook, J.M. (Eds.) *3D Bioprinting. Methods in Molecular Biology*, vol 2140. Humana, New York, NY. [https://doi.org/10.1007/978-1-0716-0520-2\\_3](https://doi.org/10.1007/978-1-0716-0520-2_3)
- Ganah, A., & Lea, G. (2021). A global analysis of BIM standards across the globe: A critical review. *Journal Of Project Management Practice (JPMP)*, 1(1), 52-60. <https://doi.org/10.22452/jpmp.vol1no1.4>
- Ghannad, P., Lee, Y.-C., Dimyadi, J., & Solihin, W. (2019). Automated BIM data validation integrating open-standard schema with visual programming language. *Advanced Engineering Informatics*, 40, 14–28. <https://doi.org/10.1016/j.aei.2019.01.006>
- Ghosh, A., Edwards, D. J., & Hosseini, M. R. (2021). Patterns and trends in Internet of Things (IoT) research: future applications in the construction industry. *Engineering, Construction and Architectural Management*, 28(2), 457-481.
- Gobesz, F. Z. (2020). The roots of BIM. *Papers on Technical Science*, 12, 42-49. <https://doi.org/10.33894/mtk-2020.12.06>
- Godager, B., Onstein, E., & Huang, L. (2021). The Concept Of enterprise BIM: Current Research Practice and Future Trends. *IEEE Access*, 9, 42265-42290. <https://doi.org/10.1109/ACCESS.2021.3065116>

- Greenwood, D., Lockley, S., Malsane, S., & Matthews, J. (2010). Automated compliance checking using building information models. In *The Construction, Building and Real Estate Research Conference of the Royal Institution of Chartered Surveyors, Paris 2nd-3rd September*. RICS.
- Guo, D., Onstein, E., & La Rosa, A. D. (2021). A Semantic Approach for Automated Rule Compliance Checking in Construction Industry. *IEEE Access*, 9, 129648-129660. <https://doi.org/10.1109/ACCESS.2021.3108226>
- Han, C. S., Kunz, J. C., & Law, K. H. (1998). A Hybrid Prescriptive/Performance Based Approach to Automated Building Code Checking. In *International Computing Congress*, 537-548.
- Hassan, F. U., & Le, T. (2020). Automated Requirements Identification from Construction Contract Documents Using Natural Language Processing. *Journal of Legal Affairs and Dispute Resolution in Engineering and Construction*, 12(2), 04520009. [https://doi.org/10.1061/\(ASCE\)LA.1943-4170.0000379](https://doi.org/10.1061/(ASCE)LA.1943-4170.0000379)
- Heidari, A., Peyvastehgar, Y., & Amanzadegan, M. (2023). A systematic review of the BIM in construction: from smart building management to interoperability of BIM & AI. *Architectural Science Review*, 67(3), 237-254. <https://doi.org/10.1080/00038628.2023.2243247>
- Hjelseth, E., & Nisbet, N. (2011, October). Capturing normative constraints by use of the semantic mark-up RASE methodology. In *Proceedings of CIB W78-W102 conference*, 1-10.
- Hjelseth, E. (2015). BIM-based model checking (BMC). *Building Information Modeling–Applications and Practices*, 33-61.
- Hjelseth, E. (2016). Classification of BIM-based model checking concepts. *Journal of Information Technology in Construction (ITcon)*, 21, 354-369, <http://www.itcon.org/2016/23>
- Hobbs, J. R., & Riloff, E. (2010). Information Extraction. *Handbook of natural language processing*, 2.
- Hooper, M. (2015). BIM standardisation efforts - the case of Sweden. *Journal of Information Technology in Construction*, 20, 332-346. <http://www.itcon.org/2015/21>
- Hotho, A., Nürnberger, A., & Paaß, G. (2005). A Brief Survey of Text Mining. *Journal for Language Technology and Computational Linguistics*, 20(1), 19-62. <https://doi.org/10.21248/jlcl.20.2005.68>
- Hovy, D., & Prabhumoye, S. (2021). Five sources of bias in natural language processing. *Language and Linguistics compass*, 15(8). <https://doi.org/10.1111/lnc3.12432>
- Hu, Z. Z., Leng, S., Lin, J. R., Li, S. W., & Xiao, Y. Q. (2022). Knowledge Extraction and Discovery Based on BIM: A Critical Review and Future Directions. *Archives of Computational Methods in Engineering*, 29(1), 335-356. <https://doi.org/10.1007/s11831-021-09576-9>

- Huang, M. Q., Ninić, J., & Zhang, Q. (2021). BIM, machine learning and computer vision techniques in underground construction: Current status and future perspectives. *Tunnelling and Underground Space Technology*, 108, 103677. <https://doi.org/10.1016/j.tust.2020.103677>
- Ilal, S. M., & Günaydin, H. M. (2017). Computer representation of building codes for automated compliance checking. *Automation in Construction*, 82, 43-58. <https://doi.org/10.1016/j.autcon.2017.06.018>
- International Organization for Standardization (ISO). (n.d.). Standards. Retrieved January 10, 2025, from <https://www.iso.org/standards.html>
- (3) Invicara. (2017, September 12). *How BIM Assure 1.3 Benefits Owners and FM: Q&A with Scott Mollon*. Retrieved April 1, 2025, from <https://invicara.com/resources/blog/bim-assure-benefits-owners-fm>
- (8) Invicara. (n.d.). *Build Your BIM Workflows With Trusted Data*. Retrieved April 1, 2025, from <https://invicara.com/technology/bim-assure>
- Ismail, A. S., Ali, K. N., & Iahad, N. A. (2017, July). A Review on BIM-Based Automated Code Compliance Checking System. In 2017 international conference on research and innovation in information systems (icriis) (pp. 1-6). IEEE.
- International Organization for Standardization (ISO). (2018a). Organization and digitization of information about buildings and civil engineering works, including building information modelling (BIM) — Information management using building information modelling (ISO Standard No. 19650-1:2018). <https://www.iso.org/standard/68078.html>
- International Organization for Standardization (ISO). (2018b). Organization and digitization of information about buildings and civil engineering works, including building information modelling (BIM) — Information management using building information modelling (ISO Standard No. 19650-2:2018). <https://www.iso.org/standard/68080.html>
- International Organization for Standardization (ISO). (2020a). Organization and digitization of information about buildings and civil engineering works, including building information modelling (BIM) — Information management using building information modelling (ISO Standard No. 19650-3:2020). <https://www.iso.org/standard/75109.html>
- International Organization for Standardization (ISO). (2020b). Organization and digitization of information about buildings and civil engineering works, including building information modelling (BIM) — Information management using building information modelling (ISO Standard No. 19650-5:2020). <https://www.iso.org/standard/74206.html>
- International Organization for Standardization (ISO). (2022). Organization and digitization of information about buildings and civil engineering works, including building information modelling (BIM) — Information management using building information modelling (ISO Standard No. 19650-5:2020). <https://www.iso.org/standard/74206.html>
- International Organization for Standardization (ISO). (2022). Organization and digitization of information about buildings and civil engineering works, including building information modelling (BIM) — Information management using building information modelling (ISO Standard No. 19650-4:2022). <https://www.iso.org/standard/78246.html>

- International Organization for Standardization (ISO). (2025). Organization and digitization of information about buildings and civil engineering works, including building information modelling (BIM) — Information management using building information modelling (ISO Standard No. 19650-4:2022). <https://www.iso.org/standard/78246.html>
- International Organization for Standardization (ISO). (2025). Organization and digitization of information about buildings and civil engineering works, including building information modelling (BIM) — Information management using building information modelling (ISO Standard No. 19650-6:2025). <https://www.iso.org/standard/82705.html>
- Jiang, S., Jiang, L., Han, Y., Wu, Z., & Wang, N. (2019). OpenBIM: An Enabling Solution for Information Interoperability. *Applied Sciences*, 9(24), 5358. <https://doi.org/10.3390/app9245358>
- Jung, W., & Lee, G. (2015). The Status of BIM Adoption on Six Continents. *International Journal of Civil, Environmental, Structural, Construction and Architectural Engineering*, 9(5), 444-448. <https://doi.org/10.5281/zenodo.1100430>
- Karim Jallow, A., Demian, P., N. Baldwin, A. and Anumba, C. (2014), An empirical study of the complexity of requirements management in construction projects, *Engineering, Construction and Architectural Management*, 21(5), 505-531. <https://doi.org/10.1108/ECAM-09-2013-0084>
- Kassem, M., Succar, B., & Dawood, N. (2015). Building Information Modeling: Analyzing Noteworthy Publications of Eight Countries Using a Knowledge Content Taxonomy. In *Building information modeling: Applications and practices*.
- Khalid, M. U., Bashir, M. K., & Newport, D. (2017). Development of a Building Information Modelling (BIM)-Based Real-Time Data Integration System Using a Building Management System (BMS). In *Building Information Modelling, Building Performance, Design and Smart Construction*, 93-104. Cham: Springer International Publishing.
- Khan, A., Sepasgozar, S., Liu, T., & Yu, R. (2021). Integration of BIM and Immersive Technologies for AEC: A Scientometric-SWOT Analysis and Critical Content Review. *Buildings*, 11(3), 126. <https://doi.org/10.3390/buildings11030126>
- Khemlani, L. (August 30, 2018). *Automated Code Compliance Updates, 2018*: AECBytes feature. Retrieved January 29, 2025, from <https://aecbytes.com/feature/2018/CodeCheckingUpdates2018.html>
- Khochare, S. D., & Waghmare, A. P. (2018). 3D, 4D and 5D Building Information Modeling for Commercial Building Projects. *International Research Journal of Engineering and Technology (IRJET)*, 5(1), 132-138.
- Kim, T., & Chi, S. (2019). Accident Case Retrieval and Analyses: Using Natural Language Processing in the Construction Industry. *Journal of Construction Engineering and Management*, 145(3). [https://doi.org/10.1061/\(ASCE\)CO.1943-7862.0001625](https://doi.org/10.1061/(ASCE)CO.1943-7862.0001625)
- Kim, I., & Kim, J. I. (2022). Special Issue on BIM and Its Integration with Emerging Technologies. *Applied Sciences*, 12(11), 5368. <https://doi.org/10.3390/app12115368>

- Kincelova, K., Botton, C., Blanchet, P., & Dagenais, C. (2019). BIM-based code compliance checking for fire safety in timber buildings: A comparison of existing tools. <https://espace2.etsmtl.ca/id/eprint/18892>
- Kiviniemi, A., Karlshøj, J., Tarandi, V., Bell, H., & Karud, O. J. (2008). Review of the Development and Implementation of IFC compatible BIM.
- Laakso, M., & Kiviniemi, A. O. (2012). The IFC standard: A review of history, development, and standardization, information technology. *ITcon*, 17(9), 134-161.
- Lake, J.G. (1999). 4 V & V in Plain English. *INCOSE International Symposium*, 9(1). 1134-1140. <https://doi.org/10.1002/j.2334-5837.1999.tb00284.x>
- Lantis (n.d.). De Oosterweelverbinding. Retrieved February 26, 2025, from <https://www.lantis.be/projecten/de-oosterweelverbinding>
- Lee, J. K. (2011). Building environment rule and analysis (BERA) language (Doctoral dissertation, Georgia Institute of Technology).
- Lee, H., Kim, J., Shin, M., Kim, I., & Lee, J. K. (2014). A Demonstration of BIM-enabled Quantitative Circulation Analysis using BERA Language. In ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction, 31(1). IAARC Publications.
- Lee, J. K., Eastman, C. M., & Lee, Y. C. (2015). Implementation of a BIM Domain-specific Language for the Building Environment Rule and Analysis. *Journal of Intelligent & Robotic Systems*, 79, 507-522.
- Lee, J. K., & Eastman, C. M. (2019). Demonstration of BERA language-based approach to offsite construction design analysis. In *Offsite Production and Manufacturing for Innovative Construction*, 129-162. Routledge.
- Li, H., & Lu, Z. (2016, July). Deep Learning for Information Retrieval. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, 1203-1206. <https://doi.org/10.1145/2911451.2914800>
- Li, S., Wang, J., & Xu, Z. (2024). Automated compliance checking for BIM models based on Chinese-NLP and knowledge graph: an integrative conceptual framework. *Engineering, Construction and Architectural Management*.
- Liu, F., Liu, Q., Bannur, S., Pérez-García, F., Usuyama, N., Zhang, S., Naumann, T., Nori, A., Poon, H., Alvarez-Valle, J., Oktay, O. & Hyland, S. L. (2023). Compositional Zero-shot Domain Transfer with Text-to-Text Models. *Transactions of the Association for Computational Linguistics*, 11, 1097-1113. [https://doi.org/10.1162/tacl\\_a\\_00585](https://doi.org/10.1162/tacl_a_00585)
- Lombardo, A. (2023, 21 May). *Digital twins, un'opportunità per il magazzino*. Logistica. <https://www.logisticanews.it/digital-twins-unopportunita-per-il-magazzino/>
- Lopez, L. A., Elam, S., & Reed, K. (1989). Software concept for checking engineering designs for conformance with codes and standards. *Engineering with Computers*, 5, 63-78. <https://doi.org/10.1007/BF01199070>



- Madubuike, O. C., Anumba, C. J., & Khallaf, R. (2022). A review of digital twin applications in construction. *Journal of Information Technology in Construction (ITcon)*, 27, 145-172. <https://www.10.36680/j.itcon.2022.008>
- Mahajan, G., & Narkhede, P. (2024). Integrating BIM with Digital Technology Trends in the Construction Industry: Implementation Insights for 2023. *Library of Progress-Library Science, Information Technology & Computer*, 44(3).
- Malsane, S., Matthews, J., Lockley, S., Love, P. E., & Greenwood, D. (2015). Development of an object model for automated compliance checking. *Automation in construction*, 49, 51-58. <https://doi.org/10.1016/j.autcon.2014.10.004>
- Manning, C., Raghavan, P., and Shutze, H. (2009). *An introduction to information retrieval*, Cambridge University Press, Cambridge, U.K.
- Mannino, A., Dejacco, M. C., & Re Cecconi, F. (2021). Building information modelling and internet of things integration for facility management—Literature review and future needs. *Applied Sciences*, 11(7), 3062. <https://doi.org/10.3390/app11073062>
- Matthys, M., De Cock, L., Vermaut, J., Van de Weghe, N., & De Maeyer, P. (2021). An “Animated Spatial Time Machine” in Co-Creation: Reconstructing History Using Gamification Integrated into 3D City Modelling, 4D Web and Transmedia Storytelling. *ISPRS International Journal of Geo-Information*, 10(7), 460. <https://doi.org/10.3390/ijgi10070460>
- Megahed, N. A., & Hassan, A. M. (2022). Evolution of BIM to DTs: A Paradigm Shift for the Post-Pandemic AECO Industry. *Urban Science*, 6(4), 67. <https://doi.org/10.3390/urbansci6040067>
- Mehra, S. (2024). Unravelling the Tapestry of Natural Language Processing: A Journey from Past to Present and Beyond, *Insights2Techinfo*.
- Mendonça, E. A., Manzione, L., & Hjelseth, E. (2020). Converting Brazilian accessibility standard for BIM-based code checking using RASE and SMC. In *7th CIB W78 Information Technology for Construction Conference (CIB W78)*, Sao Paulo (pp. 291-307).
- Moon, S., Lee, G., & Chi, S. (2022). Automated system for construction specification review using natural language processing. *Advanced Engineering Informatics*, 51, 101495. <https://doi.org/10.1016/j.aei.2021.101495>
- Moshood, T. D., Rotimi, J. O., Shahzad, W., & Bamgbade, J. A. (2024). Infrastructure digital twin technology: A new paradigm for future construction industry. *Technology in Society*, 77, 102519. <https://doi.org/10.1016/j.techsoc.2024.102519>
- Nawari, N.O. (2012). The challenge of computerizing building codes in a BIM environment. In *Computing in Civil Engineering*, 285-292. <https://doi.org/10.1061/9780784412343.0036>
- Nawari, N.O. (2018). *Building Information Modeling: Automated Code Checking and Compliance Processes* (1st ed.). CRC Press. <https://doi.org/10.1201/9781351200998>



- Nawari, N. O., & Ravindran, S. (2019). Blockchain technology and BIM process: review and potential applications. *Journal of Information Technology in Construction (ITcon)*, 24, 209-238. <http://www.itcon.org/2019/12>
- van Nederveen, G. A., & Tolman, F. P. (1992). Modelling multiple views on buildings. *Automation in Construction*, 1(3), 215-224. [https://doi.org/10.1016/0926-5805\(92\)90014-B](https://doi.org/10.1016/0926-5805(92)90014-B)
- van Nederveen, S., Beheshti, R., & Willems, P. (2010). Building Information Modelling in the Netherlands: A Status Report. In *W078-Special Track 18th CIB World Building Congress May 2010 Salford, United Kingdom* (p. 28). Citeseer.
- Nisbet, N., Zhang, Z., & Cidik, S. (2024, July). Real-Time Assessment of Regulatory Compliance of Construction Sites. In EC3 Conference 2024 (Vol. 5). *European Council on Computing in Construction*. <http://www.doi.org/10.35490/EC3.2024.215>
- Oosterweelverbinding. (n.d.). Het project. Accessed on February 26, 2025, from <https://www.oosterweelverbinding.be/het-project>
- Pan, Y., & Zhang, L. (2021). Roles of artificial intelligence in construction engineering and management: A critical review and future trends. *Automation in Construction*, 122, 103517. <https://doi.org/10.1016/j.autcon.2020.103517>
- Pan, Y., & Zhang, L. (2023). Integrating BIM and AI for Smart Construction Management: Current Status and Future Directions. *Archives of Computational Methods in Engineering*, 30(2), 1081-1110.
- Pan, X., Khan, A. M., Eldin, S. M., Aslam, F., Rehman, S. K. U., & Jameel, M. (2024). BIM adoption in sustainability, energy modelling and implementing using ISO 19650: A review. *Ain Shams Engineering Journal*, 15(1), 102252. <https://doi.org/10.1016/j.asej.2023.102252>
- Panda, S., & Kaur, N. (2023). Enhancing User Experience and Accessibility in Digital Libraries through Emerging Technologies. In K.P. Sinhamahapatra [et al.] (Eds.), *Digital Libraries: Sustainable Development in Education, Presented on 21 November 2023 at the International Symposium on Digital Libraries: Sustainable Development in Education, Indian Institute of Technology Kharagpur*, 676-703.
- Panteli, C., Polycarpou, K., Morsink-Georgalli, F. Z., Stasiuliene, L., Pupeikis, D., Jurelionis, A., & Fokaides, P. A. (2020). Overview of BIM integration into the Construction Sector in European Member States and European Union Acquis. In *IOP Conference Series: Earth and Environmental Science*, 410(1), 012073. IOP Publishing. <https://doi.org/10.1088/1755-1315/410/1/012073>
- Papadonikolaki, E. (2018). Loosely coupled systems of innovation: Aligning BIM adoption with implementation in Dutch construction. *Journal of management in engineering*, 34(6), 05018009. [https://doi.org/10.1061/\(ASCE\)ME.1943-5479.0000644](https://doi.org/10.1061/(ASCE)ME.1943-5479.0000644)
- Papamichael, K., LaPorta, J., & Chauvet, H. (1997). Building Design Advisor: automated integration of multiple simulation tools. *Automation in construction*, 6(4), 341-352. [https://doi.org/10.1016/S0926-5805\(97\)00043-5](https://doi.org/10.1016/S0926-5805(97)00043-5)

- Parsamehr, M., Dodanwala, T. C., Perera, P., & Ruparathna, R. (2023). Building information modeling (BIM)-based model checking to ensure occupant safety in institutional buildings. *Innovative Infrastructure Solutions*, 8(6), 174. <https://doi.org/10.1007/s41062-023-01141-6>
- Paskoff, C., Botton, C., Blanchet, P. (2024). Comparative Analyses of Four BIM-Based Compliance Checking Tools. In *Proceedings of the Canadian Society for Civil Engineering Annual Conference 2023, Volume 5. CSCE 2023. Lecture Notes in Civil Engineering (LNCE, vol. 499)*. Springer, Cham. [https://doi.org/10.1007/978-3-031-61503-0\\_34](https://doi.org/10.1007/978-3-031-61503-0_34)
- Poljanšek, M. (2017). Building Information Modelling (BIM) standardization. *European Commission*. <https://dx.doi.org/10.2760/36471>
- Preidel, C., & Borrmann, A. (2015). Automated code compliance checking based on a visual language and building information modeling. In *Proc. of the 32nd ISARC 2015*.
- (13) Preidel, C., Borrmann, A. (2018). BIM-Based Code Compliance Checking. In *Borrmann, A., König, M., Koch, C., Beetz, J. (eds) Building Information Modeling*. Springer, Cham. [https://doi.org/10.1007/978-3-319-92862-3\\_22](https://doi.org/10.1007/978-3-319-92862-3_22)
- Rafsanjani, H. N., & Nabizadeh, A. H. (2023). Towards human-centered artificial intelligence (AI) in architecture, engineering, and construction (AEC) industry. *Computers in Human Behavior Reports*, 11. <https://doi.org/10.1016/j.chbr.2023.100319>
- ROCO (n.d.). *Over ons*. Retrieved February 26, 2025, from <https://roco.be/over-ons/>
- Sage, A. P., & Rouse, W. B. (2011). Handbook of systems engineering and management. John Wiley & Sons.
- Sakib, S. N. (2021). Strategies, potentials and uses of BIM. *Preprint*.
- Salama, D. M., & El-Gohary, N. M. (2016). Semantic text classification for supporting automated compliance checking in construction. *Journal of Computing in Civil Engineering*, 30(1), 04014106. [https://doi.org/10.1061/\(ASCE\)CP.1943-5487.0000301](https://doi.org/10.1061/(ASCE)CP.1943-5487.0000301)
- Schulz, O., Oraskari, J., & Beetz, J. (2021). bcfOWL: A BIM collaboration ontology. In *LDAC* (pp. 142-153).
- Schulz, O., & Beetz, J. (2024). Aligning openCDE APIs with Linked Building Data through Constrained Containers in Common Data Environments. *Universitätsbibliothek der RWTH Aachen*.
- Semaan, P. (2012). Natural language generation: an overview. *Journal of Computer Science & Research (JCSCR)*, 1(3), 50-57. <http://www.lacsc.org/papers/PaperA6.pdf>
- Sepasgozar, S. M. E. (2020). Digital Twin and Web-Based Virtual Gaming Technologies for Online Education: A Case of Construction Management and Engineering. *Applied Sciences*, 10(13), 4678. <https://doi.org/10.3390/app10134678>

- Sepasgozar, S. M., Khan, A. A., Smith, K., Romero, J. G., Shen, X., Shirowzhan, S., Li, H. & Tahmasebinia, F. (2023). BIM and Digital Twin for Developing Convergence Technologies as Future of Digital Construction. *Buildings*, 13(2), 441. <https://doi.org/10.3390/buildings13020441>
- Shabi, J., Reich, Y., & Diamant, R. (2017). Planning the verification, validation, and testing process: a case study demonstrating a decision support model. *Journal of Engineering Design*, 28(3), 171–204. <https://doi.org/10.1080/09544828.2016.1274964>
- Shehzad, H. M. F., Ibrahim, R. B., Yusof, A. F., Khaidzir, K. A. M., Iqbal, M., & Razzaq, S. (2021). The role of interoperability dimensions in building information modelling. *Computers in Industry*, 129, 103444. <https://doi.org/10.1016/j.compind.2021.103444>
- Sidani, A., Dinis, F. M., Sanhudo, L., Duarte, J., Santos Baptista, J., Pocas Martins, J., & Soeiro, A. (2021). Recent Tools and Techniques of BIM-based Virtual Reality: A Systematic Review. *Archives of Computational Methods in Engineering*, 28(2), 449–462. <https://doi.org/10.1007/s11831-019-09386-0>
- Singh, M., Fuenmayor, E., Hinchy, E. P., Qiao, Y., Murray, N., & Devine, D. (2021). Digital twin: Origin to future. *Applied System Innovation*, 4(2), 36. <https://doi.org/10.3390/asi4020036>
- (9) SMARTreview. (n.d.). *SMARTreview APR*. Retrieved April 1, 2025, from <https://smartreview.biz/home>
- Solibri. (December 18, 2015). Solibri acquired by Nemetschek: Allowing exponential growth in Quality Assurance and Quality Control. Retrieved January 29, 2025, from <https://www.solibri.com/articles/solibri-acquired-by-nemetschek-allowing-exponential-growth-in-quality-assurance-and-quality-control>
- (1) Solibri. (2024). *Supported File Formats*. Retrieved April 1, 2025, from <https://help.solibri.com/hc/en-us/articles/1500003935681-Supported-File-Formats#:~:text=Solibri's%20products%20support%20IFC%20R1,Solibri's%20IFC%20R2>.
- (6) Solibri. (2024). *Creating Reports from Checking Results*. Retrieved April 1, 2025, from <https://help.solibri.com/hc/en-us/articles/1500004671222-Creating-Reports-from-Checking-Results#:~:text=This%20article%20describes%20how%20you%20can%20create%20reports,stages%20or%20periodic%20times%2C%20depending%20on%20your%20requirements>.
- Solibri. (January 7, 2025). Introducing Solibri CheckPoint: Business success delivered with a new cloud-based model checking solution. Retrieved January 31, 2025, from <https://www.solibri.com/articles/introducing-solibri-checkpoint-business-success-delivered-with-a-new-cloud-based-model-checking-solution>

- (11) Solibri. (n.d.-a). *Enforcing the need for Quality Assurance in all BIM Projects*. Retrieved April 1, 2025, from <https://www.solibri.com/enforcing-quality-assurance#:~:text=Starting%20from%20the%20release%209.12.0%2C%20Solibri%20Anywhere%20will,individual%20IFC%20files%20can%20be%20opened%20or%20viewed>
- (15) Solibri. (n.d.-b). *Understanding Checking*. Retrieved April 1, 2025, from <https://help.solibri.com/hc/en-us/articles/1500005009042-Understanding-Checking>
- Solihin, W., & Eastman, C. (2015). Classification of rules for automated BIM rule checking development. *Automation in construction*, 53, 69-82. <https://doi.org/10.1016/j.autcon.2015.03.003>
- Solihin, W. (2016). A simplified BIM data representation using a relational database schema for an efficient rule checking system and its associated rule checking language. Georgia Institute of Technology.
- Solihin, W., & Eastman, C. M. (2016). A knowledge representation approach in BIM rule requirement analysis using the conceptual graph. *Journal of Information Technology in Construction (ITcon)*, 21, 370-401. <https://www.itcon.org/2016/24>
- Solihin, W., Dimyadi, J., Lee, YC. (2019). In Search of Open and Practical Language-Driven BIM-Based Automated Rule Checking Systems. In *Advances in Informatics and Computing in Civil and Construction Engineering*. Springer, Cham. [https://doi.org/10.1007/978-3-030-00220-6\\_69](https://doi.org/10.1007/978-3-030-00220-6_69)
- (4) Solihin, W., Dimyadi, J., Lee, Y. C., Eastman, C., & Amor, R. (2020). Simplified schema queries for supporting BIM-based rule-checking applications. *Automation in Construction*, 117, 103248. <https://doi.org/10.1016/j.autcon.2020.103248>
- Strube, M., & Pan, S. (2023). Ethics in nlp: Bias and dual use. *Science*, 356, 183-186.
- Succar, B. (2010). Building information modelling maturity matrix. In *Handbook of research on building information modeling and construction informatics: Concepts and technologies*, 65-103. IGI Global. <https://doi.org/10.4018/978-1-60566-928-1.ch004>
- Sutherland, I. E. (1964, January). Sketch pad a man-machine graphical communication system. In *Proceedings of the SHARE design automation workshop*, 6-329.
- Sydora, C., & Stroulia, E. (2020). Rule-based compliance checking and generative design for building interiors using BIM. *Automation in Construction*, 120, 103368. <https://doi.org/10.1016/j.autcon.2020.103368>
- Tan, X., Hammad, A., & Fazio, P. (2010). Automated Code Compliance Checking for Building Envelope Design. *Journal of Computing in Civil Engineering*, 24(2), 203-211. [https://doi.org/10.1061/\(ASCE\)0887-3801\(2010\)24:2\(203\)](https://doi.org/10.1061/(ASCE)0887-3801(2010)24:2(203))
- Tang, S., Shelden, D. R., Eastman, C. M., Pishdad-Bozorgi, P., & Gao, X. (2019). A review of building information modeling (BIM) and the internet of things (IoT) devices integration: Present status and future trends. *Automation in construction*, 101, 127-139. <https://doi.org/10.1016/j.autcon.2019.01.020>

- Teyseyre, A. R. (2002). A 3d visualization approach to validate requirements. In *VIII Congreso Argentino de Ciencias de la Computación*, 938-949.
- Thapa, P. (2019). BIM Model Coordination Software Testing and Evaluating in Construction Project.
- Tierney, P. J. (2012). A Qualitative Analysis Framework Using Natural Language Processing and Graph Theory. *International Review of Research in Open and Distributed Learning*, 13(5), 173-189. <https://doi.org/10.19173/irrodl.v13i5.1240>
- Von Bertalanffy, L. (1972). The History and Status of General Systems Theory. *Academy of Management Journal*, 15(4), 407-426. <https://doi.org/10.5465/255139>
- Waas, L. (2022). Review of BIM-Based Software in Architectural Design: Graphisoft Archicad VS Autodesk Revit. *Journal of Artificial Intelligence in Architecture*, 1(2), 14-22. <https://doi.org/10.24002/jarina.v1i2.6016>
- Walsh, K. P. (2017). Identifying and mitigating the risks created by problematic clauses in construction contracts. *Journal of Legal Affairs and Dispute Resolution in Engineering and Construction*, 9(3), 03717001. [https://doi.org/10.1061/\(ASCE\)LA.1943-4170.0000225](https://doi.org/10.1061/(ASCE)LA.1943-4170.0000225)
- Weygant, R. S. (2011). *BIM content development: standards, strategies, and best practices*. John Wiley & Sons.
- Witteveen+Bos. (n.d.). Oosterweelverbinding. Retrieved February 26, 2025, from <https://www.witteveenbos.com/nl/projects/oosterweel-link>
- World Economic Forum. (2016, May 4). *Shaping the future of construction: A breakthrough in mindset and technology*. In collaboration with The Boston Consulting Group. [https://web-assets.bcg.com/img-src/Shaping\\_the\\_Future\\_of\\_Construction\\_may\\_2016\\_tcm9-59165.pdf](https://web-assets.bcg.com/img-src/Shaping_the_Future_of_Construction_may_2016_tcm9-59165.pdf)
- Wu, W., & Issa, R. R. (2012). Leveraging cloud-BIM for LEED automation. *Journal of Information Technology in Construction (ITcon)*, 17(24), 367-384. <http://www.itcon.org/2012/24>
- Wu, C., Li, X., Guo, Y., Wang, J., Ren, Z., Wang, M., & Yang, Z. (2022). Natural language processing for smart construction: Current status and future directions. *Automation in Construction*, 134, 104059. <https://doi.org/10.1016/j.autcon.2021.104059>
- (5) Xinaps. (n.d.-a). *Introducing Solibri CheckPoint*. Retrieved January 31, 2025, from <https://verifi3d.xinaps.com/>
- (10) Xinaps. (n.d.-b). *About*. Retrieved April 1, 2025, from <https://verifi3d.xinaps.com/about-us/>
- Xu, H., Feng, J., & Li, S. (2014). Users-orientated evaluation of building information model in the Chinese construction industry. *Automation in Construction*, 39, 32-46. <https://doi.org/10.1016/j.autcon.2013.12.004>

- Yussuf, R. O., & Asfour, O. S. (2024). Applications of artificial intelligence for energy efficiency throughout the building lifecycle: An overview. *Energy and Buildings*, 113903. <https://doi.org/10.1016/j.enbuild.2024.113903>
- Zabin, A., González, V. A., Zou, Y., & Amor, R. (2022). Applications of machine learning to BIM: A systematic literature review. *Advanced Engineering Informatics*, 51, 101474. <https://doi.org/10.1016/j.aei.2021.101474>
- Zhang, J., & El-Gohary, N. M. (2013). Semantic NLP-based information extraction from construction regulatory documents for automated compliance checking. *Journal of Computing in Civil Engineering*, 30(2). [https://doi.org/10.1061/\(ASCE\)CP.1943-5487.0000346](https://doi.org/10.1061/(ASCE)CP.1943-5487.0000346)
- Zhang, J., & El-Gohary, N. M. (2015). Automated Information Transformation for Automated Regulatory Compliance Checking in Construction. *Journal of Computing in Civil Engineering*, 29(4), B4015001. [https://doi.org/10.1061/\(ASCE\)CP.1943-5487.0000427](https://doi.org/10.1061/(ASCE)CP.1943-5487.0000427)
- Zhang, R., & El-Gohary, N. (2022). Building information modeling, natural language processing, and artificial intelligence for automated compliance checking. In *Research Companion to Building Information Modeling*, 248-267. Edward Elgar Publishing. <https://doi.org/10.4337/9781839105524.00022>
- Zhou, Y. C., Zheng, Z., Lin, J. R., & Lu, X. Z. (2022). Integrating NLP and context-free grammar for complex rule interpretation towards automated compliance checking. *Computers in Industry*, 142, 103746. <https://doi.org/10.1016/j.compind.2022.103746>
- Zhu, H., Hwang, B. G., Tan, Y. Z., & Wei, F. (2024). Building on Digital Twin: Overcoming Barriers and Unlocking Success in the Construction Industry. *Journal of Construction Engineering and Management*, 150(10), 04024142. <https://doi.org/10.1061/JCEMD4.COENG-13991>
- Zou, Y., Guo, B. H., Papadonikolaki, E., Dimyadi, J., & Hou, L. (2023). Lessons learned on adopting automated compliance checking in the AEC industry: A global study. *Journal of Management in Engineering*, 39(4), 04023019. <https://doi.org/10.1061/JMENEA.MEENG-5051>
- Zouaq, A. (2011). An Overview of Shallow and Deep Natural Language Processing for Ontology Learning. *Ontology learning and knowledge discovery using the web: Challenges and recent advances*, 16-37. <https://doi.org/10.4018/978-1-60960-625-1.ch002>

# Appendices

Appendix A: Interview protocol and insights.....	102
Appendix B: Requirement classification script.....	107
Appendix C: Verification of pipe diameter (Class 1).....	116
Appendix D: Verification of diaphragm walls (Class 1).....	120
Appendix E: Verification of element depth (Class 2).....	122
Appendix F: Verification of tunnel equipment (Class 2).....	127
Appendix G: Verification of tunnel compartments (Class 3).....	129
Appendix H: Verification of escape ducts (Class 4).....	135
Appendix I: Efficiency metrics.....	138
Appendix J: openBIM approach.....	140



## Appendix A: Interview protocol and insights

This appendix outlines the interview protocol used in this research and presents the insights gained from the interviews. Appendix A1 describes the interview approach, Appendix A2 contains the statements from the exploratory interviews and Appendix A3 presents the statements from the validation interviews. These statements are referenced at several points throughout the report. The interviews were conducted among employees of Witteveen+Bos, but the statements have been anonymized for privacy reasons.

### Appendix A1: Interview protocol

#### Purpose of the interviews

Two rounds of semi-structured interviews were conducted as part of this research. The interviews were intended to serve the following purposes:

**Table A1.1.** Interview objectives.

Round	Purpose
Round 1	To identify practical challenges and needs related to (automated) requirement verification.
Round 2	To assess the clarity, feasibility and applicability of the proposed guidelines for automated requirement verification.

#### Interview approach

The general interview approach used in this research is shown in the table below.

**Table A1.2.** Interview approach.

Item	Description
Interview type	Semi-structured interviews
Format	In-person or online (based on availability)
Duration	30-60 minutes
Recording method	Notes (anonymized)
Number of rounds	2

#### Participant selection

To ensure both a thorough exploration of the problem and a well-informed validation of the proposed guidelines, professionals with diverse roles and expertise related to BIM were interviewed. In the first round, participants were selected to represent a range of perspectives from strategic, operational and BIM level in order to gain a comprehensive understanding of the challenges associated with requirement verification in practice. In the second round, the same layered approach was used to validate the developed guidelines across different organisational levels.

#### Interview objectives

The interview objectives of each round are presented in the following table.

**Table A1.3.** Interview objectives.

Round	Objective
Round 1	Understand the current requirement verification workflows, problems and information gaps.
Round 2	Validate and evaluate the practical applicability of the proposed guidelines.

## Interview questions

The questions used during the interview rounds are shown in the table below.

**Table A1.4.** Interview questions.

	Question
Round 1	1.1 What are the current challenges and limitations in the requirement verification process within 3D models?
	1.2 How are requirements currently verified within the BIM environment?
	1.3 How do you ensure that requirements are correctly interpreted and applied in the BIM model?
	1.4 Are requirements processed according to a specific standard methodology (e.g. SMART) before they are validated within the BIM model?
	1.5 To what extent is automation currently used in the verification process and what limitations are encountered?
	1.6 What are the most common errors or problems you observe during the verification process?
	1.7 How is collaboration between different disciplines (e.g. structural engineers, BIM modelers, et cetera) organised during the requirement verification process in BIM?
	1.8 Not all requirements can be verified within the BIM model and are sometimes verified through, for example structural calculations. Who determines which verification method is used for each requirement?
	1.9 On average, how much time does it take to verify a single requirement in a BIM model and which step in the process is the most prone to errors?
	1.10 What improvements could make the verification process more efficient?
	1.11 What existing software is used for requirement verification in 3D models and what are the alternatives?
	1.12 Which software tools are currently used for verifying requirements in 3D models?
	1.13 What are the main limitations of the software you use?
	1.14 Are manual methods or workarounds applied to compensate for software limitations?
	1.15 Which standards are applied when verifying requirements in BIM models?
	1.16 To what extent are scripts used to automate parts of the verification process?
Round 2	2.1 To what extent do you agree that ambiguity in requirement texts hinders automation in BIM verification?
	2.2 Have you seen or used computer-interpretable requirement formats like RASE or IDS in practice? If so, what was your experience?
	2.3 How realistic is it to demand machine-readable requirement texts during procurement or tendering phases?
	2.4 Do you think clients are ready to include requirements for automated verification in contracts?
	2.5 What is your experience with hybrid requirements?
	2.6 Do you think splitting hybrid requirements into separate checks is practical and realistic?
	2.7 How do teams in your experience handle these types of requirements in current projects?
	2.8 Do you work with a Common Data Environment? If so, how effectively does it support requirement verification workflows?
	2.9 Have you used BCF for issue tracking in BIM? If so, does it improve coordination for requirement verification?
	2.10 Is linking BCF issues to IFC elements a common practice in your project environment?
	2.11 How often do you encounter inconsistent terminology across your BIM models?
	2.12 Do you see value in using standardised terminology like IFD or bSDD?
	2.13 What obstacles might prevent teams from adopting these dictionaries?
	2.14 To what extent are BIM models standardised in terms of property usage and data structure?
	2.15 Would standardising properties improve requirement verification in your view?
	2.16 Do you currently link requirement texts directly to model elements?
	2.17 What tools or processes have you used to support traceability for requirement to model object?
	2.18 Are you working with the latest IFC versions (e.g. IFC4.3)?
	2.19 Do you believe using newer IFC versions improves automation and interoperability in your workflow?

## Ethical considerations

In accordance with the TU Delft Regulations on Human Trials, any research involving human participants must be approved by the Human Research Ethics Committee (HREC). Prior to this research, a HREC application procedure was initiated. As part of this process, a completed HREC Checklist, informed consent forms, and a Data Management Plan were created and approved by the faculty's data steward. These documents outline several key ethical safeguards including the anonymization of interview statements and the requirement that participants give verbal consent for their input to be used for academic research purposes.

## Appendix A2: Exploratory interview statements (round 1)

**Table A2.** Overview of statements made during the exploratory interviews.

Nr.	Statement
1	The contractor has full freedom to determine the verification method. However, if the client suggests a specific method, it is not a recommendation but it becomes a mandatory requirement.
2	Ideally, you have a complete set of requirements before starting the design. However, what often happens is that a Preliminary Design (DO) is made that meets all requirements, but then the client suddenly introduces changes, leading to rework.
3	If Relatics states that a BIM model is the verification method, Witteveen+Bos adds a clash detection report. If a clash is detected, it is documented and a conclusion is drawn on whether the requirement is met.
4	In clash detection reports, ambiguous sentences like '...the requirement is considered verified' should be avoided. A requirement should either be verified and met or not met but there should be no room for interpretation. Precise language is crucial in requirement verification to ensure clarity and prevent misunderstandings.
5	UAV-GC is the most commonly used contract form for large projects in the Netherlands. It allows the contractor to design based on a set of requirements, maintaining flexibility and room for innovation.
6	The Oosterweelverbinding project uses a NEC4 contract, which is highly focused on collaboration. NEC4 originates from the UK and is only now being applied in a few European projects, making it a relatively new contract form in the region.
7	Some requirements are not easily measurable, meaning they have not been made sufficiently SMART.
8	A verification plan has been established that determines whether a requirement should be demonstrated through calculations by a structural engineer or validated using the BIM model. This plan is leading because it has been agreed upon with the client.
9	At the moment, not much scripting is used. Some tools have been developed in Navisworks to group clashes, making the process more efficient, but there is no significant scripting involved. However, these scripts are project specific and can therefore only be used once.
10	We have a BIMcollab environment where we can assign issues to people.
11	We primarily use Navisworks for clash detection, which I believe is still the most powerful tool for this purpose. Our coordination model (CMO) is also a Navisworks model, where we store all relevant information.
12	There is no specific language used for defining requirements. There is a review process to check if requirements are clear and to ensure there is no room for misinterpretation.
13	The SMART methodology is used as a standard to describe requirements properly.
14	Verifying a requirement in the BIM model is not always straightforward because there may not be enough information in the BIM model to perform the check.
15	If the requirement can be validated in the BIM model the verification method itself is still a challenge. Sometimes verifying requirements involves a lot of manual work.
16	Clash detection is performed within Navisworks to check for conflicts.

17	The most error-prone aspect of verifying requirements in a BIM model is interpreting the requirements correctly and ensuring that the verification results truly confirm that the requirements have been met.
18	Verifying requirements in a BIM model requires about one hour per requirement on average.
19	The way requirements are formulated does not align well with the BIM model. It would be helpful if requirements were written in a way that directly links elements in the text to elements in the BIM model.
20	A contractor must carry out all activities in such a way that the results explicitly and objectively comply with the project requirements.
21	Verification must be done during the Preliminary Design (DO) phase and again in the Detailed Design (UO) phase. Some requirements are so specific or critical that they might also be verified upon completion of construction which is called an inspection. This is sometimes done for large projects.
22	Sometimes projects fail to meet client expectations due to a lack of early stage insight into project requirements. Ideally, you have a complete set of requirements before starting the design, so you can verify them during the DO and UO phases. What often happens is that a DO is made that meets all requirements, but then there are changes introduced.
23	The main challenges or limitations that are faced is linking the requirements to the different components in the model. A requirement concerns an object within your project, but this linkage is not always good.
24	Even though requirements are SMART, they are not always specifiable in the BIM model.
25	From the contract system requirements without translation only 30% of the total can be verified. There is a wide range of requirements that cannot be properly verified without someone manually checking them.
26	Maybe the way of writing contracts needs to change and explicitly state what is needed in terms of requirement verification in the BIM model. It can also be a signal to the market and clients that the way of thinking needs to change from their point of view instead from those verifying.

## Appendix A3: Validation interview statements (round 2)

**Table A3.** Overview of statements made during the validation interviews.

Nr.	Statement
1	Ideally, there would be a standard template for IDS that every company uses, instead of writing custom codes themselves.
2	A link from Relatics to the IDS format would be a significant step forward.
3	There is value in breaking down hybrid requirements. You can extract the information that can be interpreted and verified in the BIM model and return the information that cannot be verified within the model.
4	The use of bSDD can be very valuable for drafting requirements.
5	The question is why the client wants to use custom properties instead of standard property sets. By using standard property sets, they can likely still request the same information.
6	Using custom properties quickly undermines standardization.
7	There is often resistance to implementing new tools or software.
8	The IDS could help bridge that gap by linking requirements to IFC data more reliably.
9	In early project phases, there's usually no complete BIM model yet, which makes linking requirements at that stage difficult, but it's still crucial for automation.
10	The IDS looks promising for structuring requirements in a way that software can understand them.
11	We need to flag requirements that can't be entirely verified in the model, so it's clear that manual calculations are still needed.
12	Storing verification results and BIM files in one platform would really improve team coordination and tracking progress.
13	The process of requirement verification is about traceability. ISO 19650 emphasizes this kind of structured information management.

14	We track model issues in BIMcollab and export BCFs, but there's no direct link between that and the rest of our project data.
15	A viewpoint in the BCF is usually enough, but if we could link it to the CDE, that would really help streamline the process.
16	We do use Object Type Libraries (OTLs), but there's no universal dictionary being followed.
17	An international standard like the IFD would help reduce confusion and make requirement verification more reliable.
18	The client wants a lot of custom properties just to track things like costs or quantities. That's why standard properties are not used so much in practice.
19	The downside of custom properties is that we can't reuse scripts or tools between projects so it is always starting from scratch.
20	Right now, requirements aren't really linked directly to the BIM model. We rely on worksets or OTL codes to find matching elements.
21	A unique requirement ID linked to the model elements sounds good in theory, but design changes can break those links.

## Appendix B: Requirement classification script

This appendix explains the Python script that is created to categorize requirements into two categories: *geometrical requirements* (1) and *non-geometrical requirements* (0). The model uses the pandas library to organize the dataset, the re module for regular expressions, nltk for text processing and the Scikit-learn (or sklearn) library for machine learning tasks. The script uses NLTK's stopwords from Dutch and downloads a predefined set of common words that don't contribute to the meaning of the text e.g. 'the' (in Dutch 'de') and removes them from the text during processing. The script loads the Excel export file from Relatics into a DataFrame (see **Figure B1**).

EisID	Requirement title	Requirement text	Verification method client	Object ID	Object title
E-00952.1	Sloop Leidingkoker Noorderlaanbrug	De bestaande leidingkoker onder het Albertkanaal thv oostzijde Noorderlaanbrug dient in het tracé van de tunnel verwijderd te worden.	Obj-01539	Tunneldeel 10	Splittingsconsti
E-03843.1	Tunneldelen Kanaalkruisingen Top-down	Tunneldelen 9, 10, 11, 12, 13 en 16 van de Vork dienen met een top-down bouwmethode gerealiseerd te worden.	Obj-01091	Tunneldeel 11	Kruising Alberti
TE-00058.1	Buisdiameter riolering verkeerskokers	De buisdiameter van de ingestorte riolering in de verkeerskokers dient afgestemd te zijn op de langshelling en de gevraagde tr	Obj-01539	Tunneldeel 10	Splittingsconsti
TE-00060.1	Voorkomen bezwijken vloer verkeerskoker	Indien een afvoersysteem met rioolbuizen in verkeerskoker zonder ballastbeton wordt toegepast, dient er te worden aangetoo	Obj-01091	Tunneldeel 11	Kruising Alberti
TE-00190.1	Kunstwerk, minimale diepte fundering	Alle funderingsaanzetten dienen zich tenminste 1 m onder het maaiveld te bevinden om geen nadeligen gevolgen van opvriezi	Obj-01091	Tunneldeel 11	Kruising Alberti
TE-00213.1	Drukbelasting constructies op leidingen	Kunstwerken in de nabijheid van ondergrondse leidingen dienen zodanig ontworpen te worden dat er geen onaanvaardbare dr	Obj-01091	Tunneldeel 11	Kruising Alberti
TE-00490.1	Hoogte polderwand ten opzichte van maxima	De bovenzijde van de polderwand dient op een hoogte te liggen dat het Indien de top van de polderwand onder maaiveldnivea	Obj-01091	Tunneldeel 11	Kruising Alberti
TE-01564.3	Tunnelconstructie ruimtes	Een tunnelconstructie dient de van toepassing zijnde verkeerskoker-, vl BIM model: Een 3D BIM model waarin het minimale rui	Obj-01091	Tunneldeel 11	Kruising Alberti
TE-01603.1	Ruimte vluchtkoker	Een vluchtkoker in een tunnelconstructie dient conflictvrij ruimte te bieden BIM model: Een 3D BIM model waarin het minimale rui	Obj-01539	Tunneldeel 10	Splittingsconsti
TE-01605.1	Tunnelconstructie aansluiten ruimtes	Ruimtes in een tunnelconstructie dienen technisch en geometrisch aan BIM model: Een 3D BIM model waarin het minimale rui	Obj-01539	Tunneldeel 10	Splittingsconsti
TE-04057.1	Tunnelconstructie realiseren ruimtes vluchtk	Een tunnelconstructie dient de ruimtes in de vluchtkokers, dienstkokers Voor aantoning dient een ruimte BIM-model gemaakt tr	Obj-01539	Tunneldeel 10	Splittingsconsti
TE-04074.1	Slooppniveau maaiveld	Een te slopen constructie dient tot ten minste 2,0 m onder maaiveld geheel verwijderd te zijn. Hierbij inbegrepen funderingszol	Obj-01684	Leidingenkoker Bredastraat	
TE-04076.1	Slooppniveau waterbodem	Constructie(onderdelen) onder het Albertkanaal dienen tot ten minste TAW -2,75m verwijderd te zijn.	Obj-01684	Leidingenkoker Bredastraat	
TE-04077.1	Maximaal peil tunneldak Albertkanaal	De bovenzijde van de tunnelconstructies onder het Albertkanaal dienen maximaal het peil te hebben conform gekoppeld docur	Obj-01091	Tunneldeel 11	Kruising Alberti
TE-04096.1	Functie Afwatering Vluchtkoker	De Afwatering Vluchtkoker dient te voorkomen dat vloeistoffen blijven staan op diepe punten in de vluchtkoker.	Obj-01091	Tunneldeel 11	Kruising Alberti
TE-04141.3	PVBR in de verkeerskokers, viaducten en over	De profiel van beschikbare ruimte (PVBR) voor de verschillende deelget In de volgende ontwerp stappen, dient de PVBR voor de	Obj-01091	Tunneldeel 11	Kruising Alberti
TE-04281.1	Halve barriër, toepassing	Indien een halve barriër wordt toegepast dient deze over de volledige hoogte van het profiel en over de volledige geplaatste le	Obj-01091	Tunneldeel 11	Kruising Alberti
TE-04341.1	Tunnelconstructie PVBR langs de rijbaan	Een tunnelconstructie dient de ruimte tussen de weg en de afgewerkte BIM model: Een 3D BIM model waarin het minimale rui	Obj-01091	Tunneldeel 11	Kruising Alberti
E-03843.1	Tunneldelen Kanaalkruisingen Top-down	Tunneldelen 9, 10, 11, 12, 13 en 16 van de Vork dienen met een top-down bouwmethode gerealiseerd te worden.	Obj-01091	Tunneldeel 11	Kruising Alberti
E-03843.1	Tunneldelen Kanaalkruisingen Top-down	Tunneldelen 9, 10, 11, 12, 13 en 16 van de Vork dienen met een top-down bouwmethode gerealiseerd te worden.	Obj-01552	Tunneldeel 12	Busopstelplaat
TE-00058.1	Buisdiameter riolering verkeerskokers	De buisdiameter van de ingestorte riolering in de verkeerskokers dient afgestemd te zijn op de langshelling en de gevraagde tr	Obj-01091	Tunneldeel 11	Kruising Alberti
TE-00058.1	Buisdiameter riolering verkeerskokers	De buisdiameter van de ingestorte riolering in de verkeerskokers dient afgestemd te zijn op de langshelling en de gevraagde tr	Obj-01552	Tunneldeel 12	Busopstelplaat
TE-00060.1	Voorkomen bezwijken vloer verkeerskoker	Indien een afvoersysteem met rioolbuizen in verkeerskoker zonder ballastbeton wordt toegepast, dient er te worden aangetoo	Obj-01552	Tunneldeel 12	Busopstelplaat
TE-00060.1	Voorkomen bezwijken vloer verkeerskoker	Indien een afvoersysteem met rioolbuizen in verkeerskoker zonder ballastbeton wordt toegepast, dient er te worden aangetoo	Obj-01091	Tunneldeel 11	Kruising Alberti
TE-00190.1	Kunstwerk, minimale diepte fundering	Alle funderingsaanzetten dienen zich tenminste 1 m onder het maaiveld te bevinden om geen nadeligen gevolgen van opvriezi	Obj-01091	Tunneldeel 11	Kruising Alberti
TE-00190.1	Kunstwerk, minimale diepte fundering	Alle funderingsaanzetten dienen zich tenminste 1 m onder het maaiveld te bevinden om geen nadeligen gevolgen van opvriezi	Obj-01552	Tunneldeel 12	Busopstelplaat
TE-00213.1	Drukbelasting constructies op leidingen	Kunstwerken in de nabijheid van ondergrondse leidingen dienen zodanig ontworpen te worden dat er geen onaanvaardbare dr	Obj-01091	Tunneldeel 11	Kruising Alberti
TE-00213.1	Drukbelasting constructies op leidingen	Kunstwerken in de nabijheid van ondergrondse leidingen dienen zodanig ontworpen te worden dat er geen onaanvaardbare dr	Obj-01552	Tunneldeel 12	Busopstelplaat
TE-00490.1	Hoogte polderwand ten opzichte van maxima	De bovenzijde van de polderwand dient op een hoogte te liggen dat het Indien de top van de polderwand onder maaiveldnivea	Obj-01552	Tunneldeel 12	Busopstelplaat

**Figure B1.** Snippet from the export from Relatics.

The function `bim_labels` is created to label requirements as geometrical (1) and non-geometrical (0) and are added to a new column called 'Geometrical requirement' in the DataFrame. The columns 'Requirement text', 'Requirement title' and 'Verification method client' are concatenated to a new column called 'Tekst'. This combined text will be used as input for the machine learning model. After that, the dataset is divided into a training set (80%) and a test set (20%) to train and evaluate the model. A machine learning pipeline is created which first vectorizes the text data using TF-IDF (Term Frequency-Inverse Document Frequency) and then trains a logistic regression model to classify the data. The model is trained on the training set (`X_train`, `y_train`) to predict whether a requirement is geometrical or non-geometrical.

A separate dataset is created which maps geometrical requirements to four different classes. This data is used to train a second model for predicting the class of the requirements based on the requirement text. A second pipeline is created and trained using the same TF-IDF vectorizer and regression model but this time to predict the class for each requirement based on the `manual_data` set. For each requirement that is classified as geometrical, the script predicts which class it belongs to using the `class_model`.

```

import pandas as pd
import re
import nltk
from nltk.corpus import stopwords
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import make_pipeline
from sklearn.metrics import accuracy_score

nltk.download("stopwords")
stopwords = set(stopwords.words("dutch"))

df = pd.read_excel("Requirement database v2.xlsx")

def bim_labels(row):
    if row["Methode"] == "BIM model clash controle":
        return 1
    if any(re.search(r"BIM\S*", str(row[col]), re.IGNORECASE) for col in ["Verificatie voorschrift OG", "Verificatie toelichting"]):
        return 1
    return 0

df["Geometrical requirement"] = df.apply(bim_labels, axis=1)

df["Tekst"] = df["Eistekst"].astype(str) + " " + df["Eistitel"].astype(str) + " " + df["Verificatievoorschrift OG"].astype(str)

X_train, X_test, y_train, y_test = train_test_split(df["Tekst"], df["Geometrical requirement"], test_size=0.2, random_state=42)

model = make_pipeline(TfidfVectorizer(stop_words=stopwords), LogisticRegression(max_iter=500))
model.fit(X_train, y_train)

manual_data = pd.DataFrame({
    "EisID": [
        "TE-00190.1", "TE-00490.1", "TE-01260.1", "TE-01286.1",
        "TE-01564.3", "TE-01603.1", "TE-01605.1", "TE-04057.1", "TE-04341.1"
    ],
    "Eistekst": [
        "Alle funderingsaanzetten dienen zich tenminste 1 m onder het maaiveld te bevinden om geen nadeligen gevolgen van opvriezing te ondervinden.",
        "De bovenzijde van de polderwand dient op een hoogte te liggen dat het grondwater niet over de polderwand kan stromen tijdens en na de werken.",
        "Polderwanden dienen verticaal minimaal 2,0m in de Boomse kleilaag te worden geplaatst.",
        "De stortvoeg voor opstort na afkappen diepwand dient boven het grondwaterpeil te worden uitgevoerd.",
        "Een tunnelconstructie dient de van toepassing zijnde verkeerskoker-, vluchtkoker- en dienstkoker-ruimtes conform gekoppeld document te realiseren.",
        "Een vluchtkoker in een tunnelconstructie dient conflictvrij ruimte te bieden aan de van toepassing zijnde PVR, afbouwelementen, installaties en alle daarbij benodigde vrije ruimtes.",
        "Ruimtes in een tunnelconstructie dienen technisch en geometrisch aan te sluiten op het gedeelte van deze ruimtes in aangrenzende tunnelconstructies.",
        "Een tunnelconstructie dient de ruimtes in de vluchtkokers, dienstkokers en kabelkokers te realiseren conform gekoppelde documenten.",
        "Een tunnelconstructie dient de ruimte tussen de weg en de afgewerkte wand, vanaf de bovenkant van de barri r tot aan de bovenkant van het PVR volledig vrij te houden. Dit PVR dient voorts ter hoogte van de bovenkant van vluchtpictogrammen over de gehele tunnallengte minimaal 200mm en ter hoogte van de pictogrammen boven de hulppostkasten over de gehele tunnallengte minimaal 155mm breed te zijn."
    ]
})

```



```

    ],
    "Class": [
        "Class 2", "Class 1", "Class 1", "Class 1", "Class 3", "Class 4",
        "Class 3", "Class 3", "Class 2"
    ]
})

X_class = manual_data["Eistekst"]
y_class = manual_data["Class"]

class_model = make_pipeline(TfidfVectorizer(stop_words=stopwoorden_nl), LogisticRegression(max_iter=500))
class_model.fit(X_class, y_class)

df["Predicted Class"] = ""
geometrical_df = df[df["Geometrical requirement"] == 1]
df.loc[geometrical_df.index, "Predicted Class"] = class_model.predict(geometrical_df["Tekst"])

for _, row in manual_data.iterrows():
    df.loc[df["EisID"] == row["EisID"], "Predicted Class"] = row["Class"]

df["Geometrical requirement"] = df["Geometrical requirement"] | model.predict(df["Tekst"])

pd.set_option("display.max_rows", None)
pd.set_option("display.max_columns", None)
pd.set_option("display.width", 1000)
pd.set_option("display.max_colwidth", None)

df = df.loc[:, ~df.columns.str.contains('^Unnamed: 21')]
df = df.loc[:, ~df.columns.str.contains('^Unnamed: 22')]
df = df.loc[:, ~df.columns.str.contains('^Tekst')]

# print(df[["EisID", "BIM requirement", "Geometrical requirement", "Predicted Class", "Object ID"]])

output = df[["EisID", "BIM requirement", "Geometrical requirement", "Predicted Class", "Object ID"]]

pd.set_option("display.max_rows", None)
pd.set_option("display.max_columns", None)

display(output)

```

	EisID	BIM requirement	Geometrical requirement	Predicted Class	Object ID
0	E-00952.1	Nee	0		Obj-01539
1	E-03843.1	Nee	0		Obj-01091
2	TE-00058.1	Nee	1	Class 1	Obj-01539
3	TE-00060.1	Nee	0		Obj-01091
4	TE-00190.1	Ja	1	Class 2	Obj-01091
5	TE-00213.1	n.v.t.	0		Obj-01091
6	TE-00490.1	Ja	1	Class 1	Obj-01091
7	TE-00646.1	n.v.t.	0		Obj-01539
8	TE-00660.1	Nee	0		Obj-01539
9	TE-00687.1	Nee	0		Obj-01539
10	TE-00735.1	Nee	0		Obj-01091
11	TE-01041.1	Nee	0		Obj-01091
12	TE-01084.2	Nee	0		Obj-01539
13	TE-01085.1	Nee	0		Obj-01539
14	TE-01101.1	Nee	0		Obj-01091
15	TE-01260.1	Ja	1	Class 1	Obj-01539
16	TE-01283.1	Ja	1	Class 1	Obj-01539

17	TE-01286.1	Ja	1	Class 1	Obj-01539
18	TE-01287.1	Nee	0		Obj-01091
19	TE-01364.1	Nee	0		Obj-01091
20	TE-01485.1	Nee	0		Obj-01091
21	TE-01490.1	Nee	0		Obj-01091
22	TE-01491.1	Nee	0		Obj-01091
23	TE-01564.3	Ja	1	Class 3	Obj-01091
24	TE-01603.1	Ja	1	Class 4	Obj-01539
25	TE-01605.1	Ja	1	Class 3	Obj-01539
26	TE-04057.1	Ja	1	Class 3	Obj-01539
27	TE-04074.1	Nee	0		Obj-01684
28	TE-04076.1	Nee	0		Obj-01684
29	TE-04077.1	Nee	0		Obj-01091
30	TE-04096.1	Nee	0		Obj-01091
31	TE-04141.3	Nee	0		Obj-01091
32	TE-04281.1	Nee	0		Obj-01091
33	TE-04341.1	Ja	1	Class 2	Obj-01091
34	E-03843.1	Nee	0		Obj-01091
35	E-03843.1	Nee	0		Obj-01552
36	TE-00058.1	Nee	1	Class 1	Obj-01091
37	TE-00058.1	Nee	1	Class 1	Obj-01552
38	TE-00060.1	Nee	0		Obj-01552
39	TE-00060.1	Nee	0		Obj-01091
40	TE-00190.1	Ja	1	Class 2	Obj-01091
41	TE-00190.1	Ja	1	Class 2	Obj-01552
42	TE-00213.1	Nee	0		Obj-01091
43	TE-00213.1	Nee	0		Obj-01552
44	TE-00490.1	Ja	1	Class 1	Obj-01552
45	TE-00490.1	Ja	1	Class 1	Obj-01091
46	TE-00646.1	Nee	0		Obj-01552
47	TE-00646.1	Nee	0		Obj-01091
48	TE-00660.1	Nee	0		Obj-01091
49	TE-00660.1	Nee	0		Obj-01552
50	TE-00687.1	Nee	0		

...

The output is shortened.

```

from sklearn.metrics import confusion_matrix, classification_report, ConfusionMatrixDisplay
import matplotlib.pyplot as plt

valid_df = df[df["BIM requirement"].isin(["Ja", "Nee"])].copy()
valid_df["True Label"] = valid_df["BIM requirement"].map({"Ja": 1, "Nee": 0})

y_true = valid_df["True Label"]
y_pred = valid_df["Geometrical requirement"]

cm = confusion_matrix(y_true, y_pred)

print("Confusion matrix:")
print(cm)

print("\nClassification report:")
print(classification_report(y_true, y_pred, target_names=["Non-geometrical (0)", "Geometrical (1)"]))

disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=["Non-geometrical",
"Geometrical"])

```

```

fig, ax = plt.subplots()
disp.plot(cmap=plt.cm.Blues, ax=ax, include_values=False)
plt.title("Confusion matrix")

labels = [["TN", "FP"], ["FN", "TP"]]
for i in range(2):
    for j in range(2):
        ax.text(j, i, f"{labels[i][j]}\n{cm[i, j]}",
                ha='center', va='center', fontsize=12, fontweight='bold', color='grey'
        )

ax.set_xlabel("Geometrical label", fontsize=10)
ax.set_ylabel("BIM label", fontsize=10)
ax.set_yticklabels(["Nee", "Ja"], fontsize=10)

plt.show()

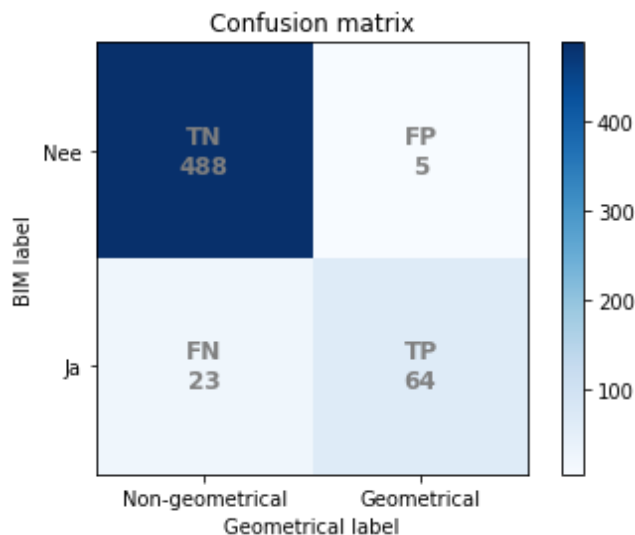
```

Confusion matrix:

```
[[488  5]
 [ 23 64]]
```

Classification report:

	precision	recall	f1-score	support
Non-geometrical (0)	0.95	0.99	0.97	493
Geometrical (1)	0.93	0.74	0.82	87
accuracy			0.95	580
macro avg	0.94	0.86	0.90	580
weighted avg	0.95	0.95	0.95	580



```
relevant_df = df[df["BIM requirement"].isin(["Ja", "Nee"])]
```

```

def check_result(row):
    if row["BIM requirement"] == "Nee" and row["Geometrical requirement"] == 0:
        return "Correct"
    elif row["BIM requirement"] == "Ja" and row["Geometrical requirement"] == 1:
        return "Correct"
    else:
        return "Wrong"

```

```
relevant_df["Result check"] = relevant_df.apply(check_result, axis=1)
```

```

correct = (relevant_df["Result check"] == "Correct").sum()
total = len(relevant_df)
percentage = correct / total * 100

print(f"Correctness: {correct} van {total} ({percentage:.2f}%)")

print(relevant_df["Result check"].value_counts())

```

```

Correctness: 552 van 580 (95.17%)
Result check
Correct      552
Wrong         28
Name: count, dtype: int64

```

```

filtered_df = df[(df["Geometrical requirement"] == 1) | ((df["BIM requirement"] == "Ja") & (df["Geometrical requirement"] == 0))]

```

```

display(filtered_df[["EisID", "BIM requirement", "Geometrical requirement", "Predicted Class", "Object ID"]])

```

	EisID	BIM requirement	Geometrical requirement	Predicted Class	Object ID
2	TE-00058.1	Nee	1	Class 1	Obj-01539
4	TE-00190.1	Ja	1	Class 2	Obj-01091
6	TE-00490.1	Ja	1	Class 1	Obj-01091
15	TE-01260.1	Ja	1	Class 1	Obj-01539
16	TE-01283.1	Ja	1	Class 1	Obj-01539
17	TE-01286.1	Ja	1	Class 1	Obj-01539
23	TE-01564.3	Ja	1	Class 3	Obj-01091
24	TE-01603.1	Ja	1	Class 4	Obj-01539
25	TE-01605.1	Ja	1	Class 3	Obj-01539
26	TE-04057.1	Ja	1	Class 3	Obj-01539
33	TE-04341.1	Ja	1	Class 2	Obj-01091
36	TE-00058.1	Nee	1	Class 1	Obj-01091
37	TE-00058.1	Nee	1	Class 1	Obj-01552
40	TE-00190.1	Ja	1	Class 2	Obj-01091
41	TE-00190.1	Ja	1	Class 2	Obj-01552
44	TE-00490.1	Ja	1	Class 1	Obj-01552
45	TE-00490.1	Ja	1	Class 1	Obj-01091
61	TE-01260.1	Ja	1	Class 1	Obj-01552
62	TE-01260.1	Ja	1	Class 1	Obj-01091
64	TE-01283.1	Ja	1	Class 1	Obj-01552
65	TE-01283.1	Ja	1	Class 1	Obj-01091
66	TE-01286.1	Ja	1	Class 1	Obj-01091
67	TE-01286.1	Ja	1	Class 1	Obj-01552
75	TE-01564.3	Ja	1	Class 3	Obj-01552
76	TE-01564.3	Ja	1	Class 3	Obj-01091
77	TE-01603.1	Ja	1	Class 4	Obj-01091
78	TE-01603.1	Ja	1	Class 4	Obj-01552
79	TE-01605.1	Ja	1	Class 3	Obj-01091
80	TE-01605.1	Ja	1	Class 3	Obj-01552
81	TE-04057.1	Ja	1	Class 3	Obj-01552
82	TE-04057.1	Ja	1	Class 3	Obj-01091
92	TE-04341.1	Ja	1	Class 2	Obj-01091
93	TE-04341.1	Ja	1	Class 2	Obj-01552
108	TE-00058.1	Nee	1	Class 1	Obj-01516
110	TE-00190.1	Ja	1	Class 2	Obj-01516
113	TE-00490.1	Ja	1	Class 1	Obj-01516
149	TE-01260.1	Ja	1	Class 1	Obj-01516
150	TE-01283.1	Ja	1	Class 1	Obj-01516
151	TE-01286.1	Ja	1	Class 1	Obj-01516

154	TE-01564.3	Ja	1	Class 3	Obj-01516
155	TE-01603.1	Ja	1	Class 4	Obj-01516
156	TE-01605.1	Ja	1	Class 3	Obj-01516
196	TE-04057.1	Ja	1	Class 3	Obj-01516
200	TE-04341.1	Ja	1	Class 2	Obj-01516
274	TE-01261.1	Ja	0		Obj-06766
275	TE-01261.1	Ja	0		Obj-06766
276	TE-01261.1	Ja	0		Obj-06766
282	TE-01281.1	Ja	0		Obj-06766
283	TE-01281.1	Ja	0		Obj-06766
284	TE-01281.1	Ja	0		Obj-06766
349	TE-01283.1	Ja	0		Obj-01099
352	TE-01286.1	Ja	0	Class 1	Obj-01099
361	TE-00490.1	Ja	0	Class 1	Obj-01099
362	TE-00490.1	Ja	0	Class 1	Obj-01099
363	TE-00490.1	Ja	0	Class 1	Obj-01099
364	TE-00490.1	Ja	0	Class 1	Obj-01099
365	TE-00490.1	Ja	0	Class 1	Obj-01099
366	TE-00190.1	Ja	0	Class 2	Obj-01099
368	TE-04141.1	Ja	0		Obj-01099
369	TE-01603.1	Ja	1	Class 4	Obj-01099
371	TE-01605.1	Ja	1	Class 3	Obj-01099
372	TE-04341.1	Ja	1	Class 2	Obj-01099
373	TE-04057.1	Ja	1	Class 3	Obj-01099
374	TE-04057.1	Ja	1	Class 3	Obj-01099
375	TE-04057.1	Ja	1	Class 3	Obj-01099
376	TE-04057.1	Ja	1	Class 3	Obj-01099
377	TE-04057.1	Ja	1	Class 3	Obj-01099
378	TE-04057.1	Ja	1	Class 3	Obj-01099
379	TE-04057.1	Ja	1	Class 3	Obj-01099
380	TE-04057.1	Ja	1	Class 3	Obj-01099
381	TE-01564.1	Ja	1	Class 3	Obj-01099
382	TE-01564.1	Ja	1	Class 3	Obj-01099
383	TE-01564.1	Ja	1	Class 3	Obj-01099
387	TE-01260.1	Ja	0	Class 1	Obj-01099
388	E-00631.1	Ja	0		Obj-01097
393	E-03843.1	Ja	0		Obj-01097
415	TE-00058.1	Nee	1	Class 1	Obj-01097
418	TE-00190.1	Ja	1	Class 2	Obj-01097
421	TE-00490.1	Ja	1	Class 1	Obj-01097
452	TE-01260.1	Ja	1	Class 1	Obj-01097
453	TE-01283.1	Ja	1	Class 1	Obj-01097
454	TE-01286.1	Ja	1	Class 1	Obj-01097
457	TE-01564.3	Ja	1	Class 3	Obj-01097
458	TE-01603.1	Ja	1	Class 4	Obj-01097
459	TE-01605.1	Ja	1	Class 3	Obj-01097
497	TE-04057.1	Ja	1	Class 3	Obj-01097
499	TE-04141.3	Ja	0		Obj-01097
501	TE-04341.1	Ja	1	Class 2	Obj-01097
562	TE-01261.1	Ja	0		Obj-06768
563	TE-01261.1	Ja	0		Obj-06768
569	TE-01281.1	Ja	0		Obj-06768
570	TE-01281.1	Ja	0		Obj-06768

```
filtered_df = df[df["Geometrical requirement"] == 1]
```

```
eistekst_table = filtered_df.groupby("EisID").agg({
    "Eistekst": "first",
    "Object ID": lambda x: ", ".join(x.dropna().astype(str)),
    "Predicted Class": "first"
})
```

```
}).reset_index()
display(eistekst_table)
```

EisID	Eisteks	t	Object ID	Predicted Class
0	TE-00058.1			
	De buisdiameter van de ingestortte riolering in de verkeerskokers dient afgestemd te zijn op de langshelling en de gevraagde transportcapaciteit, maar is minimaal 200 mm.			
	Obj-01539, Obj-01091, Obj-01552, Obj-01516, Obj-01097			Class 1
1	TE-00190.1			
	Alle funderingsaanzetten dienen zich tenminste 1 m onder het maaiveld te bevinden om geen nadeligen gevolgen van opvriezing te ondervinden			
	Obj-01091, Obj-01091, Obj-01552, Obj-01516, Obj-01097			Class 2
2	TE-00490.1			
	De bovenzijde van de polderwand dient op een hoogte te liggen dat het grondwater niet over de polderwand kan stromen tijdens en na de werken			
	Obj-01091, Obj-01552, Obj-01091, Obj-01516, Obj-01097			Class 1
3	TE-01260.1			
	Polderwanden dienen verticaal minimaal 2,0m in de Boomse kleilaag te worden geplaatst			
	Obj-01539, Obj-01552, Obj-01091, Obj-01516, Obj-01097			Class 1
4	TE-01283.1			
	Compartimenteringswanden, voor het compartimenteren van polders, dienen minimaal tot aan het hoogste drainagepeil niveau van de twee aansluitende poldercompartimenten te reiken.			
	Obj-01539, Obj-01552, Obj-01091, Obj-01516, Obj-01097			Class 1
5	TE-01286.1			
	De stortvoeg voor opstort na afkappen diepwand dient boven het grondwaterpeil te worden uitgevoerd			
	Obj-01539, Obj-01091, Obj-01552, Obj-01516, Obj-01097			Class 1
6	TE-01564.1			
	Een tunnelconstructie dient de van toepassing zijnde verkeerskoker-, vluchtkoker- en dienstkokerruimtes conform gekoppeld document te realiseren			
	Obj-01099, Obj-01099, Obj-01099			Class 3
7	TE-01564.3			
	Een tunnelconstructie dient de van toepassing zijnde verkeerskoker-, vluchtkoker- en dienstkokerruimtes conform gekoppeld document te realiseren			
	Obj-01091, Obj-01552, Obj-01091, Obj-01516, Obj-01097			Class 3
8	TE-01603.1			
	Een vluchtkoker in een tunnelconstructie dient conflictvrij ruimte te bieden aan de van toepassing zijnde PVR, afbouwlementen, installaties en alle daarbij benodigde vrije ruimtes. Het minimale PVR voor vluchtkokers in een tunnel bedraagt: - OKA tunnel: 2,0m breed x 2,1m hoog - Scheldetunnel: 2,0m breed x 2,1m hoog - Schijnpoorttunnel midden: 2,1m breed x 2,1m hoog - Schijnpoorttunnel zijanten: 1,8m breed x 2,1m hoog - Kanaalzonetunnel boven: 1,7m breed x 2,1m hoog - Kanaalzonetunnel onder: 1,7m breed x 2,1m hoog - Luchtbaltunnel: 2,0m breed x 2,1m hoog			
	Obj-01539, Obj-01091, Obj-01552, Obj-01516, Obj-01099, Obj-01097			Class 4
9	TE-01605.1			
	Ruimtes in een tunnelconstructie dienen technisch en geometrisch aan te sluiten op het gedeelte van deze ruimtes in aangrenzende tunnelconstructies			
	Obj-01539, Obj-01091, Obj-01552, Obj-01516, Obj-01099, Obj-01097			Class 3
10	TE-04057.1			
	Een tunnelconstructie dient de ruimtes in de vluchtkokers, dienstkokers en kabelkokers te realiseren conform gekoppelde documenten. Obj-01539, Obj-01552, Obj-01091, Obj-01516, Obj-01099, Obj-01099, Obj-01099, Obj-01099, Obj-01099, Obj-01099, Obj-01099, Obj-01099, Obj-01099, Obj-01097			
				Class 3
11	TE-04341.1			

Een tunnelconstructie dient de ruimte tussen de weg en de afgewerkte wand, vanaf de bovenkant van de barrier tot aan de bovenkant van het PVR volledig vrij te houden. Dit PVR dient voorts ter hoogte van de bovenkant van vluchtpictogrammen over de gehele tunnellingte minimaal 200mm en ter hoogte van de pictogrammen boven de hulppostkasten over de gehele tunnellingte minimaal 155mm breed te zijn.

Obj-01091, Obj-01091, Obj-01552, Obj-01516, Obj-01099, Obj-01097

Class 2



## Appendix C: Verification of pipe diameter (Class 1)

This appendix discusses the Python script that was used to verify a requirement in Class 1 (see **Table C1**). The script uses the IfcOpenShell module, which can extract information from the IFC file that is loaded at the beginning of the script. It loops through all elements in the model and groups them by the first part of their name. Then it searches for elements that match a specific target filter related to pipelines. Once the relevant elements are filtered, the script extracts specific property and quantity values from each element, such as the sectional area and element name. These values are stored in a DataFrame using the Pandas module. Based on the sectional area, the script calculates the diameter of each pipe using the formula for a circular cross-section. The calculated diameter is then compared to the requirement of a minimum diameter of 200 mm. For each element, the script adds a verification result indicating whether the requirement is met. The results of this verification process are shown in a **Table C2**. The script and the results presented in this appendix relate to Obj-01539. The remaining object codes and their corresponding BIM models have been verified in the same manner.

**Table C1.** Requirement text used for verification of Class 1.

Requirement text
The pipe diameter of the embedded sewer system in the traffic ducts must be adjusted according to the longitudinal slope and the required transport capacity, but it is at least 200 mm.

```
import ifcopenshell
import pandas as pd
from IPython.display import display
import numpy as np
import re
import time

start_time = time.time()

ifc_path = r"C:\Users\oveb\Downloads\IFC tunneldelen\OWRB-01539-ROC-DMO-W66-000001.ifc"
model = ifcopenshell.open(ifc_path)

category_map = {}

for element in model.by_type("IfcProduct"):
    if element.Name:
        base_name = element.Name.split(':')[0]
        category_map.setdefault(base_name, []).append(element)

target_filter = "[XXX-BUI] Multilayer pipelines for water_ROCO_(GMA2)"
filtered_elements = []

for category in category_map.values():
    for element in category:
        if element.Name and target_filter in element.Name:
            filtered_elements.append(element)

data = []

for element in filtered_elements:
    element_data = {
        "Element Name": element.Name or 'Unnamed',
        "GlobalId": element.GlobalId
    }

    for rel in element.IsDefinedBy:
```

```

        if rel.is_a("IfcRelDefinesByProperties"):
            prop_def = rel.RelatingPropertyDefinition
            if prop_def.is_a("IfcPropertySet"):
                for prop in prop_def.HasProperties:
                    if prop.Name in ["ROCO_CAR_Sectional_Area", "OWV_SE_fysiek_naam",
"Workset"]]:
                        value = getattr(prop, "NominalValue", None)
                        element_data[prop.Name] = value.wrappedValue if value else 'No
ne'

                    elif prop_def.is_a("IfcElementQuantity"):
                        for quantity in prop_def.Quantities:
                            if quantity.Name == "ROCO_CAR_Sectional_Area":
                                val = getattr(quantity, "LengthValue", None) or \
                                    getattr(quantity, "AreaValue", None) or \
                                    getattr(quantity, "VolumeValue", None) or \
                                    getattr(quantity, "CountValue", None) or \
                                    getattr(quantity, "WeightValue", None)
                                element_data[quantity.Name] = val

            data.append(element_data)

df = pd.DataFrame(data)

df['ROCO_CAR_Sectional_Area'] = pd.to_numeric(df['ROCO_CAR_Sectional_Area'], errors='c
oerce')

df['Diameter'] = 2 * np.sqrt(df['ROCO_CAR_Sectional_Area'] / np.pi)

df['Verification'] = df['Diameter'].apply(
    lambda d: f"d = {d:.2f} ≥ 0.20 ✓" if d >= 0.200 else f"d = {d:.2f} ≥ 0.20 ✗"
)

pd.set_option('display.max_rows', None)
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)
pd.set_option('display.max_colwidth', None)

display(df)

total_elements = len(df)
passed_elements = (df['Diameter'] >= 0.200).sum()
match = re.search(r'OWRB-(\d{5})', ifc_path)
obj_number = f"Obj-{match.group(1)}" if match else "Obj-XXXXX"

end_time = time.time()
total_time = end_time - start_time
print(f"Script executed in {total_time:.2f} seconds.")

summary = f"{passed_elements} out of {total_elements} requirements with requirementID
TE-00058.1 meet the requirement of a minimum pipe diameter of 200 mm for {obj_number}.
"
print(summary)

```

**Table C2.** DataFrame of verification results of requirement TE-00058.1 for each element in the model.

[illegible]

	Element Name	GlobalId	ROCO_CAR_Sectional_Area	OWV_SF_fysiek_naam	Workset	Diameter	Verification
67	[XXX-BUI]_Multilayer pipelines for water_ROCO_(GMA2):DN500:11553860	29Pxnphm91ABFMqQgqpoR2	0.157080	Wegriolingsstelsel TKZB-MO VRK	310. Waterhuishouding algemeen	0.447214	d = 0.45 ≥ 0.20
68	[XXX-BUI]_Multilayer pipelines for water_ROCO_(GMA2):DN500:11553864	29Pxnphm91ABFMqQgqpoRE	0.157080	Wegriolingsstelsel TKZB-MO VRK	310. Waterhuishouding algemeen	0.447214	d = 0.45 ≥ 0.20
69	[XXX-BUI]_Multilayer pipelines for water_ROCO_(GMA2):DN500:11553868	29Pxnphm91ABFMqQgqpoRA	0.157080	Wegriolingsstelsel TKZB-MO VRK	310. Waterhuishouding algemeen	0.447214	d = 0.45 ≥ 0.20
70	[XXX-BUI]_Multilayer pipelines for water_ROCO_(GMA2):DN500:11553872	29Pxnphm91ABFMqQgqpoRM	0.157080	Wegriolingsstelsel TKZB-MO VRK	310. Waterhuishouding algemeen	0.447214	d = 0.45 ≥ 0.20
71	[XXX-BUI]_Multilayer pipelines for water_ROCO_(GMA2):DN500:11553876	29Pxnphm91ABFMqQgqpoRI	0.157080	Wegriolingsstelsel TKZB-MO VRK	310. Waterhuishouding algemeen	0.447214	d = 0.45 ≥ 0.20
72	[XXX-BUI]_Multilayer pipelines for water_ROCO_(GMA2):DN500:11554205	29Pxnphm91ABFMqQgqpoSR	0.157080	Wegriolingsstelsel TKZO-MO VRK	310. Waterhuishouding algemeen	0.447214	d = 0.45 ≥ 0.20
73	[XXX-BUI]_Multilayer pipelines for water_ROCO_(GMA2):DN500:11554209	29Pxnphm91ABFMqQgqpoSd	0.157080	Wegriolingsstelsel TKZO-MO VRK	310. Waterhuishouding algemeen	0.447214	d = 0.45 ≥ 0.20

Script executed in 43.20 seconds.

74 out of 74 requirements with requirementID TE-00058.1 meet the requirement of a minimum pipe diameter of 200 mm for Obj-01539.

## Appendix D: Verification of diaphragm walls (Class 1)

This appendix discusses the Python script that was used to verify a requirement in Class 1 (see **Table D1**). The appendix is divided into two parts, with the first part (Appendix D1) focusing on understanding the requirement and filtering the BIM models and Appendix D2 looks at the verification.

**Table D1.** Requirement text used for verification of Class 1 (second example).

Class 1	Requirement text
	The top of the diaphragm wall must be at a height that prevents groundwater from flowing in during and after the works.

### Appendix D1: Filtering of the models

Prior to conducting the verification, it is essential to identify which elements belong to the diaphragm walls in the tunnel structure. The selection of these elements was based on their Workset within the BIM model. The Python script used to filter the IFC models used the IfcOpenShell module. Besides, the groundwater level, during and after the works, must be determined since it is not specifically specified within the requirement text. Based on project documentation, the groundwater level during the construction works is determined to be +3,00 m T.A.W. The groundwater level after the construction works is set at +4,75 m T.A.W. This represents an exceptionally high groundwater level that corresponds to an extreme event expected to occur once during the 100 year design life of the tunnel construction. **Figure D1a** shows the situation with a ground water level of +3,00 m T.A.W during construction and **Figure D1b** displays the situation after the construction works.

```
import ifcopenshell

ifc_path = r"C:\Users\oveb\Downloads\IFC tunneldelen\OWRB-01091-ROC-DMO-W66-000001.ifc"

model = ifcopenshell.open(ifc_path)

worksets_to_keep = "251. Diepwanden, CB en Baretten"

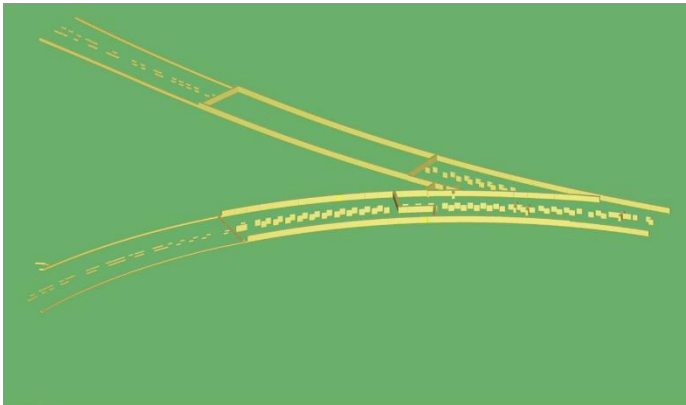
elements_to_keep = []

for rel_defines in model.by_type("IfcRelDefinesByProperties"):
    element = rel_defines.RelatedObjects[0]
    prop_set = rel_defines.RelatingPropertyDefinition
    if hasattr(prop_set, "HasProperties"):
        for prop in prop_set.HasProperties:
            if prop.Name == "Workset":
                workset_value = prop.NominalValue.wrappedValue
                if workset_value in worksets_to_keep:
                    elements_to_keep.append(element)
                    break

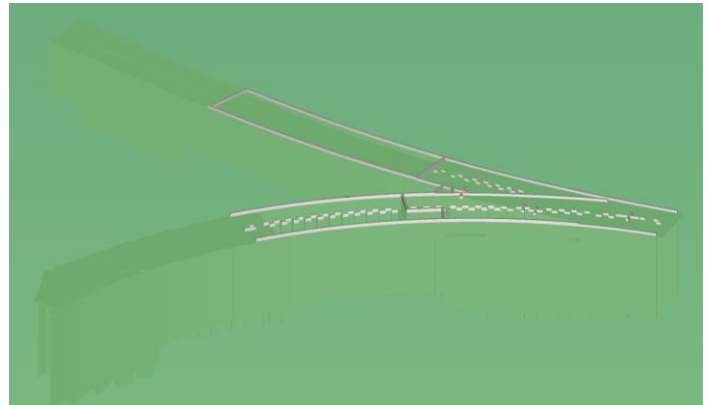
for element in model.by_type("IfcElement"):
    if element not in elements_to_keep:
        model.remove(element)

output_path = r"C:\Users\oveb\Downloads\FilteredIFC_Workset01091.ifc"
model.write(output_path)
print(f"Filtered IFC saved to:\n{output_path}")
```

Filtered IFC saved to:  
C:\Users\oveb\Downloads\FilteredIFC\_Workset01091.ifc



(a)



(b)

**Figure D1.** (a) Overview of tunnel elements related to the verification of the requirement in relation to the ground level during the works and (b) an overview of the tunnel elements in relation to the ground level after the works.

## Appendix D2: Verification in Navisworks

To verify the requirement, the diaphragm wall mentioned in the requirement text are filtered out in the IFC model. **Figure D1a** shows the situation displaying the diaphragm walls and a groundwater level of +3,00 m T.A.W. during the construction works. A clearance check using a plane representing the groundwater level shows that the diaphragm walls are high enough to prevent the groundwater from flowing in the excavation pit during the construction works. The verification must also be conducted for a groundwater level of +4,75 m T.A.W., after the construction works are completed. This situation is depicted in **Figure D1b** which shows that at some parts of the top of the diaphragm walls will be located underneath the groundwater level. Nevertheless, it can be concluded that after the construction phase, the tunnel will be equipped with a roof structure which ensures that the groundwater cannot enter the tunnel. This shows that the requirement, to prevent groundwater from flowing into the tunnel construction, is met.

## Appendix E: Verification of element depth (Class 2)

This appendix discusses the Python script that was used to verify a requirement in Class 2 (see **Table E1**). The appendix is divided into two parts, with the first part (Appendix E1) focusing on understanding the requirement and filtering the BIM models and Appendix E2 looks at the clearance check for the verification.

**Table E1.** Requirement text used for verification of Class 2.

Requirement text
All foundation bases must be located at least 1 m below the ground surface to prevent any adverse effects from frost heave.

### Appendix E1: Filtering of the models

To perform the verification, a selection was made of elements related to *foundation bases* based on their Family Name and Workset (see **Table E1.1**). The resulting selection was used to filter the BIM models. Only the elements associated with foundation bases that are relevant for the verification of the requirement were retained. In addition, all temporary elements were filtered out of the model as they will be removed after construction. These temporary construction elements are not relevant for the verification of the requirement concerning frost heave. The elements were filtered based on the BIM phasing parameter (called `OWV_BIM_Fasering` in the model).

**Table E1.1.** Overview of worksets and family names used for filtering the BIM model based on 'foundation base' name.

Workset	Family name
250. General Civil Foundations	Floor
251. Diaphragm Walls, CB and Barrettes	[DWA-DWP]_Concrete diaphragm wall units_ROCO_(GMA2)
255. Anchors	[XXX-GNK]_GM_Anchor head_vert [XXX-GNK]_grouted ground anchor_ROCO_(GMA2)
257. Underwater Concrete Floor	Toposolid
301. Floors	Floor
304. Columns	[XXX-KLM]_Concrete Columns_Round_ROCO_(GMA1)

The Python script uses the `IfcOpenShell` module to read the IFC models. The script makes it possible to filter the BIM model by selecting only specific building elements based on their properties and then write a new version of the model. A predefined list of Family Names is created that represent the specific categories of elements that are relevant for the verification. The script loops through all objects to find elements with a Family property that matches the predefined list. These elements are added to a list of elements to keep. It then filters out any temporary elements by checking the `OWV_BIM_Fasering` property. If this property equals `TT`, the element is removed from the list to ensure only permanent elements are retained. Finally, the script deletes all elements not in the list and writes the filtered model to a new IFC file. **Figure E1** provides a visual representation of the script's functionality where **Figure E1a** shows the unfiltered model and **Figure E1b** displays the model after filtering by the script.

```
import ifcopenshell

ifc_path = r"C:\Users\oveb\Downloads\IFC tunneldelen\OWRB-01516-ROC-DM0-W66-000001.ifc"

model = ifcopenshell.open(ifc_path)

family_names = [
    "Toposolid",
    "Floor",
```



```

    "[XXX-GNK]_GM_Anchor head_vert",
    "[XXX-GNK]_grouted ground anchor_ROCO_(GMA2)",
    "[XXX-KLM]_Concrete Columns_Round_ROCO_(GMA1)",
    "[DWA-DWP]_Concrete diaphragm wall units_ROCO_(GMA2)"
]

elements_to_keep = []

for rel_defines in model.by_type("IfcRelDefinesByProperties"):
    element = rel_defines.RelatedObjects[0]
    prop_set = rel_defines.RelatingPropertyDefinition

    if prop_set.Name == "Other":
        for prop in prop_set.HasProperties:
            if prop.Name == "Family":
                family_value = prop.NominalValue.wrappedValue
                if family_value in family_names:
                    elements_to_keep.append(element)
                    break

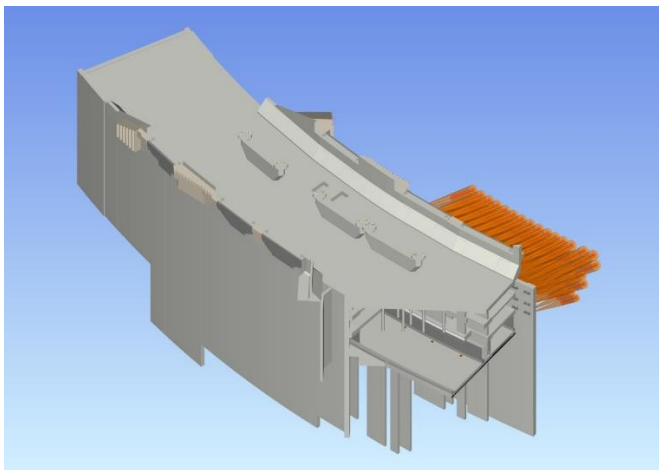
    if prop_set.Name == "Identity Data":
        for prop in prop_set.HasProperties:
            if prop.Name == "OWV_BIM_Fasering":
                fasering_value = prop.NominalValue.wrappedValue
                if fasering_value == "TT" and element in elements_to_keep:
                    elements_to_keep.remove(element)

for element in model.by_type("IfcElement"):
    if element not in elements_to_keep:
        model.remove(element)

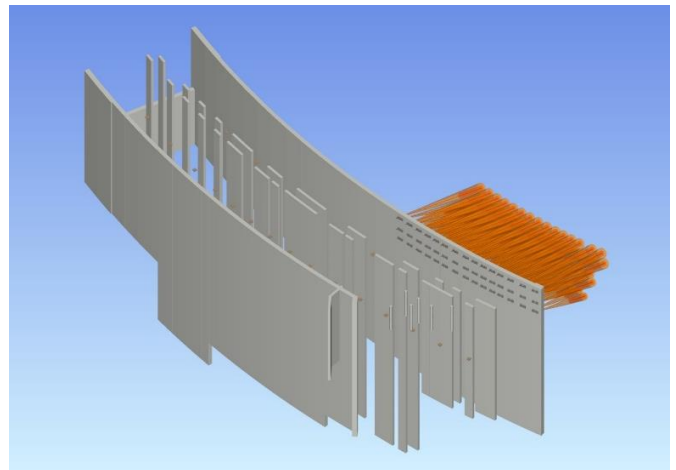
output_path = r"C:\Users\oveb\Downloads\01552 Filtered.ifc"
model.write(output_path)
print(f"Filtered IFC saved to:\n{output_path}")

Filtered IFC saved to:
C:\Users\oveb\Downloads\01516 Filtered.ifc

```



(a)

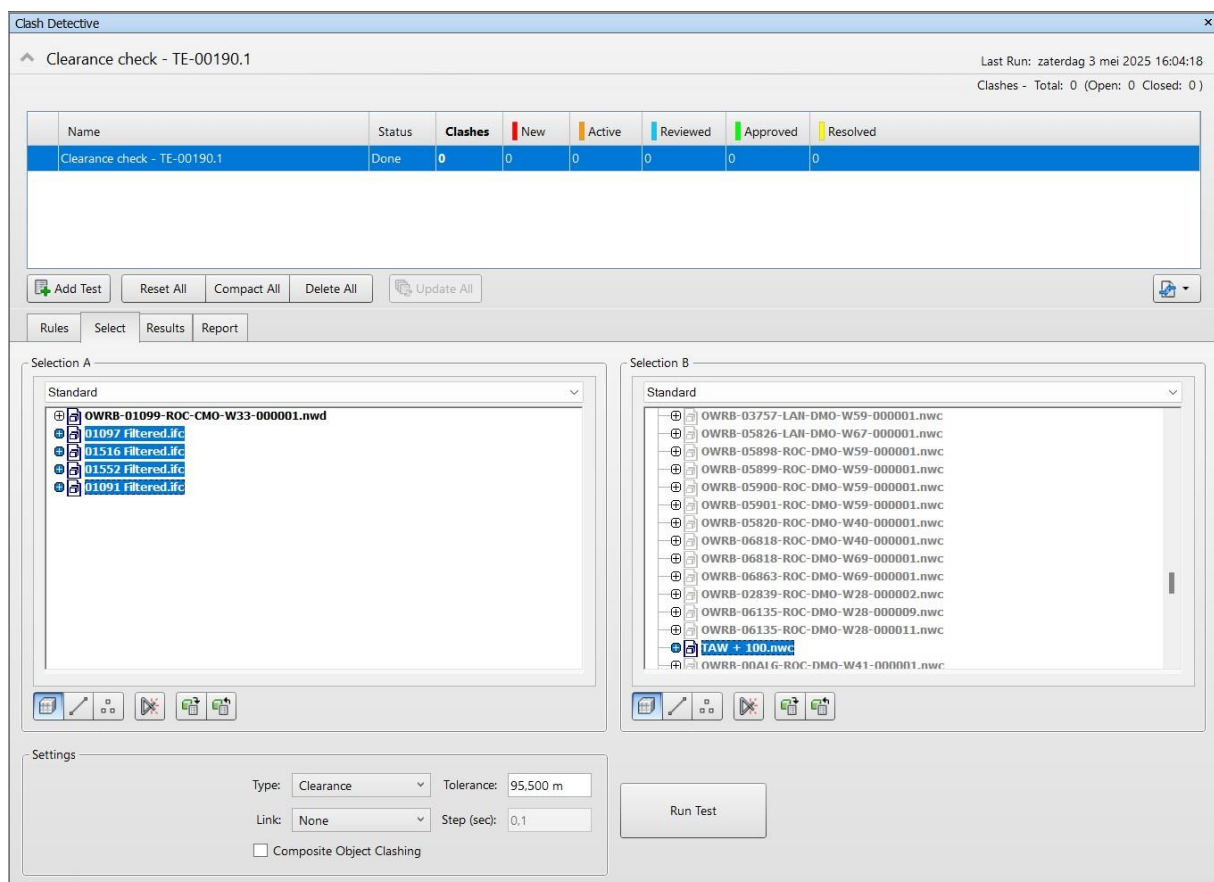


(b)

**Figure E1.** (a) BIM model with all the elements in place and (b) filtered model.

## Appendix E2: Verification in Navisworks

The verification was performed in Navisworks, where a clearance check was done between the models related to the requirement and a fictional layer. The layer was placed at +100 m T.A.W. while the ground level was previously determined at +5,5 m T.A.W. As indicated in the requirement text, all foundation base elements must be at least 1 meter below the ground level (see **Table E1**). This corresponds to a height of +4,5 m T.A.W. To verify compliance with the requirement, a clearance check was conducted with a tolerance of 95,5 meters (see **Figure E2**). No clashes were found in this clearance check with building elements in the models and the layer at the fictional level of +4,5 m T.A.W. (see **Figure E3**). In addition, a check was also carried out to determine from which level the first foundation base elements are present in the models. This turned out to be at a clearance check with a tolerance of 96 meters, which corresponds to 1,5 meters below ground level (i.e., + 4 m T.A.W.). This is within the margin of at least one meter below ground level to prevent frost heave. At this level, four clashes were observed which are shown in **Figures E4-E7**.



**Figure E2.** Clearance check (selection tab) between BIM models and fictional layer with a tolerance of 95,5 meters.



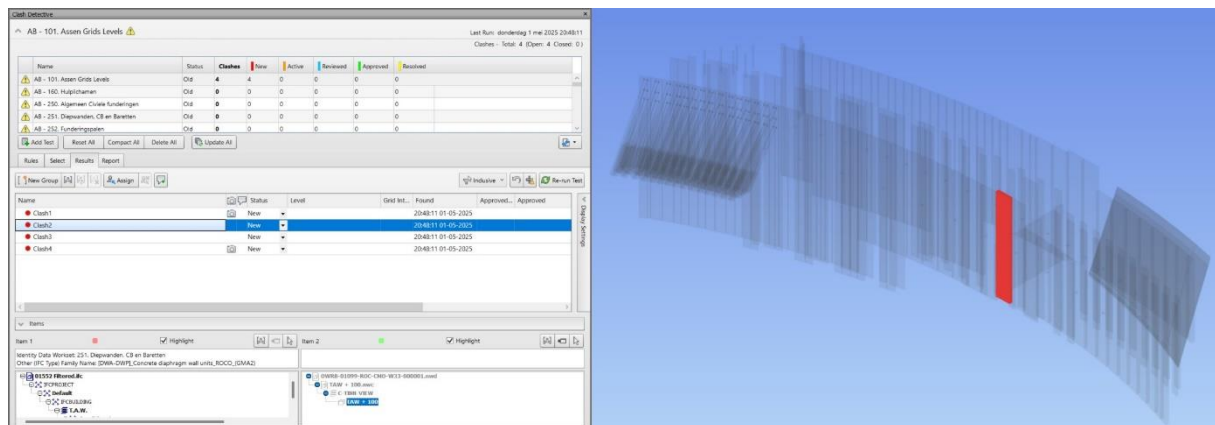


Figure E5. Visualization of clash 2 between the BIM model and fictional layer.

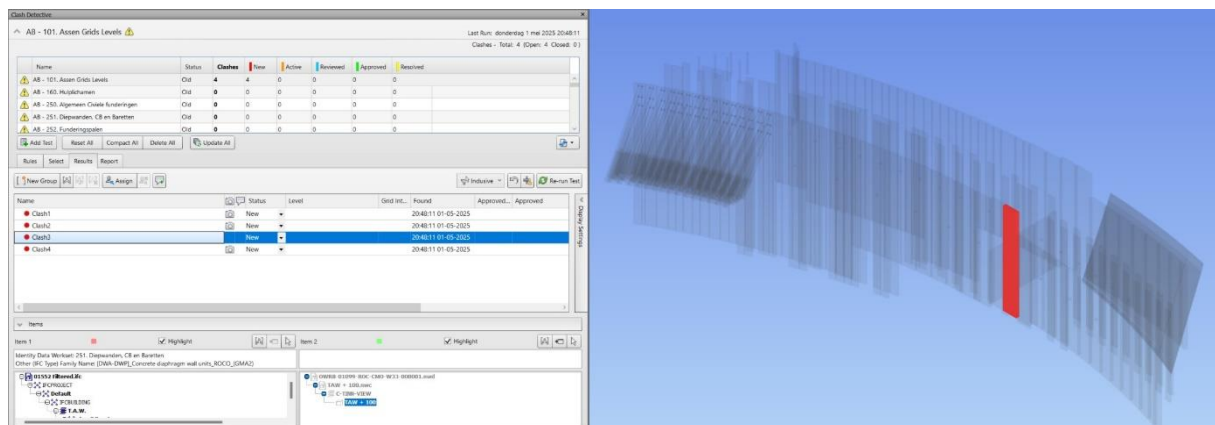


Figure E6. Visualization of clash 3 between the BIM model and fictional layer.

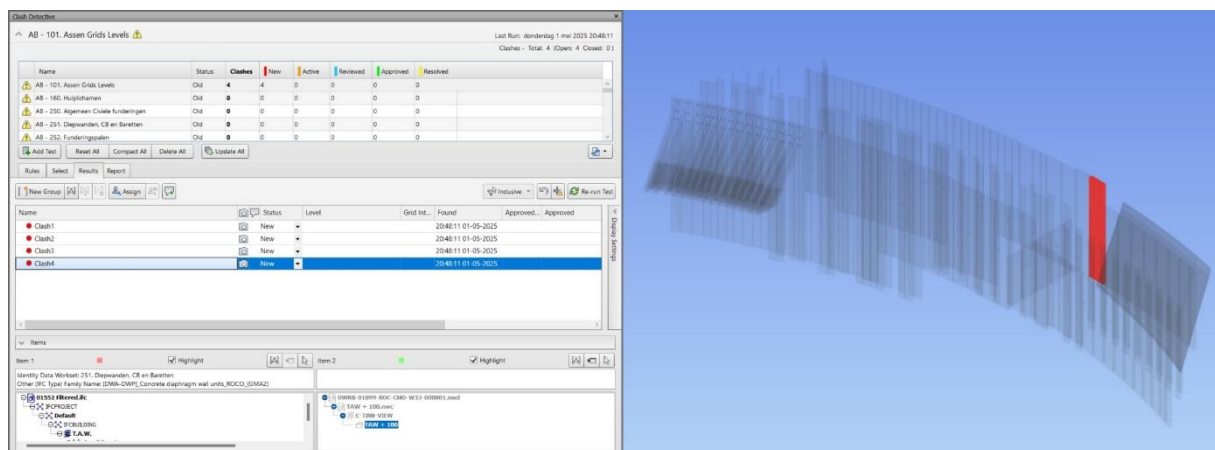


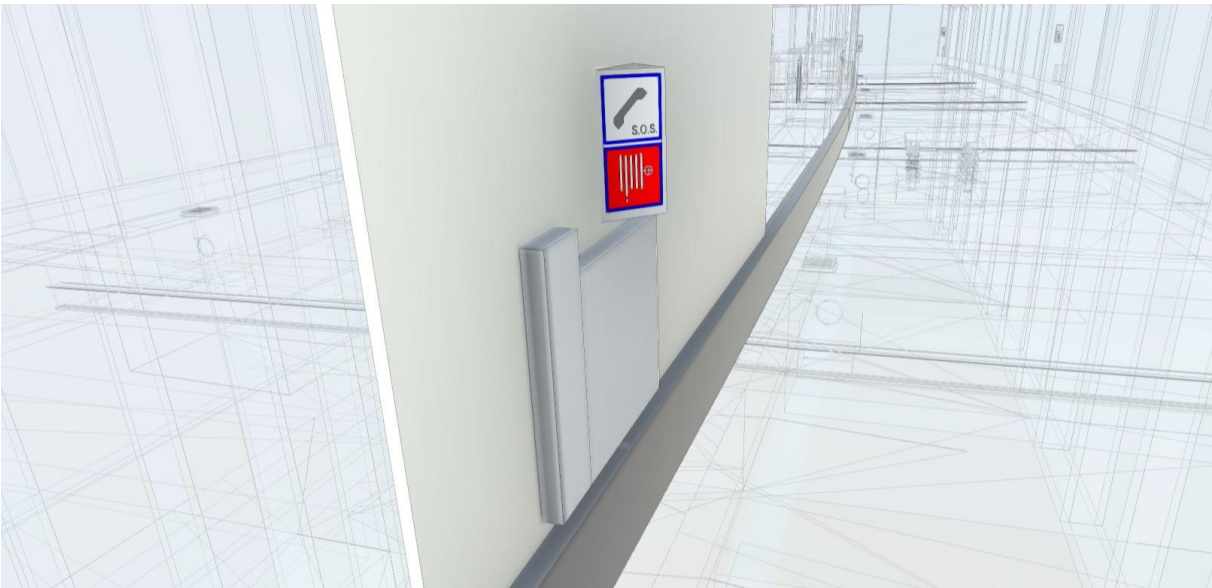
Figure E7. Visualization of clash 4 between the BIM model and fictional layer.

# Appendix F: Verification of tunnel equipment (Class 2)

This appendix looks at the verification of the tunnel equipment related to the roadway. The requirement used in this appendix for the verification is shown in **Table F1**. The requirement text distinguishes between a width of 200 mm at the top of the escape pictograms from the top of the barrier and a width of 155 mm above the auxiliary station boxes along the entire length of the tunnel construction.

Class 2	<b>Table F1.</b> Second example of an Class 2 requirement from the case study.	
	<table><tr><th>Requirement text</th></tr><tr><td>A tunnel structure should keep the space between the road and the finished wall, from the top of the barrier to the top of the PVR completely clear. This PVR should also be at least 200 mm wide at the top of escape pictograms along the entire length of the tunnel and at least 155 mm wide at the top of pictograms above the auxiliary station boxes along the entire length of the tunnel.</td></tr></table>	Requirement text
Requirement text		
A tunnel structure should keep the space between the road and the finished wall, from the top of the barrier to the top of the PVR completely clear. This PVR should also be at least 200 mm wide at the top of escape pictograms along the entire length of the tunnel and at least 155 mm wide at the top of pictograms above the auxiliary station boxes along the entire length of the tunnel.		

For this a clearance profile was used to perform a clash detection within the BIM model. The clash detection was conducted between the clearance profiles in the tunnel structure and the escape pictograms and auxiliary station boxes along the entire length of the tunnel (see **Figure F1**). The requirement must ensure that there is sufficient space between the signalling elements and the roadway used by traffic.



**Figure F1.** Impression of escape pictograms and auxiliary station boxes in the tunnel elements.

A clash detection was done by importing a XML file clash import into Navisworks (see **Figure F2**). The clash detection shows that almost all the clashes occurred at the lower part of the clearance profile. The clash detection is carried out by using the workset that contained the escape pictograms combined with the clearance profiles along the tunnel length. The clash detection identified multiple clashes which leads to non-compliance of the requirement.

```

<exchange
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation=
    "http://download.autodesk.com/us/navisworks/schemas/nw-exchange-12.0.xsd" units="m" filename="" filepath="">
  <batchtest name="Clash test tunnel signing" internal_name="Clash test tunnel signing" units="m">
    <clashtests/>
    <selectionsets>
      <viewfolder name="IFC Collections" guid="e84bca5f-b085-560c-8411-ed6704960964">
        <viewfolder name="Containers" guid="6e9b6173-b057-54a8-8f8d-78724b62805f">
          <viewfolder name="BuildingStorey" guid="720def6f-731a-50d4-907f-48108cee4902"/></viewfolder>
          <viewfolder name="Layers" guid="40c33990-1f92-546c-b7d5-3304f20a3e11"/></viewfolder>
        <viewfolder name="IFC Collections (OWRB-01097-ROC-DMO-W66-000001.ifc)" guid="251a18c9-9f95-5f02-8604-9056b80297b3">
          <viewfolder name="Containers" guid="5cbcbf88-8e01-5bea-addb-081eb930f03a">
            <viewfolder name="BuildingStorey" guid="5946b754-400e-50ec-9afe-13736e54eb2d"/></viewfolder>
            <viewfolder name="Layers" guid="6ba0758a-215a-5cf0-8675-aa839d32c570"/></viewfolder>
          <viewfolder name="IFC Collections (OWRB-01552-ROC-DMO-W66-000001.ifc)" guid="64cbc69b-2eb7-5228-994e-da30f0a1c3af">
            <viewfolder name="Containers" guid="d12e463a-1266-5fc9-9361-4270bc3d8ea9">
              <viewfolder name="BuildingStorey" guid="45ae04ad-d71b-5122-bae2-c1fac3677a4a"/></viewfolder>
              <viewfolder name="Layers" guid="0db7d61b-1249-5fd6-9fa4-d401a1fa8702"/></viewfolder>
            <viewfolder name="IFC Collections (OWRB-01539-ROC-DMO-W66-000001.ifc)" guid="e7994536-c836-5f12-b005-8e067a3861e0">
              <viewfolder name="Containers" guid="f2866027-824a-50dc-9156-8e01dcb10c5c">
                <viewfolder name="BuildingStorey" guid="e5230930-e4ff-572b-9968-3d448057f376"/></viewfolder>
                <viewfolder name="Layers" guid="528d0286-afd7-5227-87b0-ea467a976510"/></viewfolder>
              <viewfolder name="IFC Collections (OWRB-01091-ROC-DMO-W66-000001.ifc)" guid="aa043160-a5cb-5ab1-9ac4-d47da1762647">
                <viewfolder name="Containers" guid="0a033389-aefb-53f2-b3ae-db039dbee4f6">
                  <viewfolder name="BuildingStorey" guid="ce6ed071-efa4-5f04-bd6f-3788addf015a"/></viewfolder>
                  <viewfolder name="Layers" guid="f83efa0a-39b5-582b-94d4-03441f64a1bb"/></viewfolder>
                </viewfolder>
              </viewfolder>
            </viewfolder>
          </viewfolder>
        </viewfolder>
      </selectionsets>
    </batchtest>
  </exchange>

```

**Figure F2.** Clash detection import XML file between tunnel parts and signalling elements in Navisworks.



## Appendix G: Verification of tunnel compartments (Class 3)

This appendix explains the Python script that was used to verify a requirement in Class 3 (see **Table G1**). The appendix is divided into two parts, where the first part (Appendix G1) focuses on understanding the requirement and filtering of the BIM models and Appendix G2 looks at the verification.

**Table G1.** Requirement text used for verification of Class 3.

Requirement text
A tunnel construction must realize the applicable traffic, escape- and service tube spaces in accordance with the linked document.

### Appendix G1: Filtering of the models

Prior to conducting the verification, it is essential to identify which elements belong to the traffic, escape and service tube spaces. The selection of these elements was based on their Family property within the BIM model. A filtering process was applied to exclude diaphragm wall elements, as these are not relevant to the requirement being verified. In addition, all temporary elements were removed from the model, since they will not be present in the final construction and are therefore not applicable to the verification of the tunnel tube spaces. This filtering was based on the BIM phasing parameter (OWV\_BIM\_Fasering), which distinguishes between temporary and permanent components.

The Python script uses the `IfcOpenShell` module to process the IFC models. The script removes specific elements from the BIM model based on defined property criteria, thereby creating a filtered model that is suitable for the verification process. The filtering logic targets two specific properties, namely the family name property and the BIM phasing parameter. A predefined list of family names is used to identify categories of elements that must be excluded from the model. Additionally, elements with the `OWV_BIM_Fasering` property set to `TT` are excluded to ensure that only permanent elements remain. The script iterates through the property definitions of all model elements to identify those matching the exclusion criteria and removes them from the model. The filtered model only contains the relevant elements for the intended requirement verification.

**Figure G1** provides a visual overview of the process, where **Figure G1a** displays the unfiltered model and **Figure G2b** shows the filtered model with the specified elements removed.

```
import ifcopenshell

ifc_path = r"C:\Users\oveb\Downloads\IFC_tunneldelen\OWRB-01552-ROC-DMO-W66-000001.ifc"

model = ifcopenshell.open(ifc_path)

family_names_to_remove = [
    "[DWA-DWP]_Concrete diaphragm wall units_ROCO_(GMA2)",
    "[DWA-DWP-TOL]_Diaphragm wall tolerance zone_(GMA2)"
]

elements_to_keep = set(model.by_type("IfcElement"))

for rel_defines in model.by_type("IfcRelDefinesByProperties"):
    element = rel_defines.RelatedObjects[0]
    prop_set = rel_defines.RelatingPropertyDefinition

    if not hasattr(prop_set, "HasProperties"):
        continue
    for prop in prop_set.HasProperties:
        if prop.Name == "Family":
```



```

family_value = prop.NominalValue.wrappedValue
if family_value in family_names_to_remove:
    elements_to_keep.discard(element)
    break

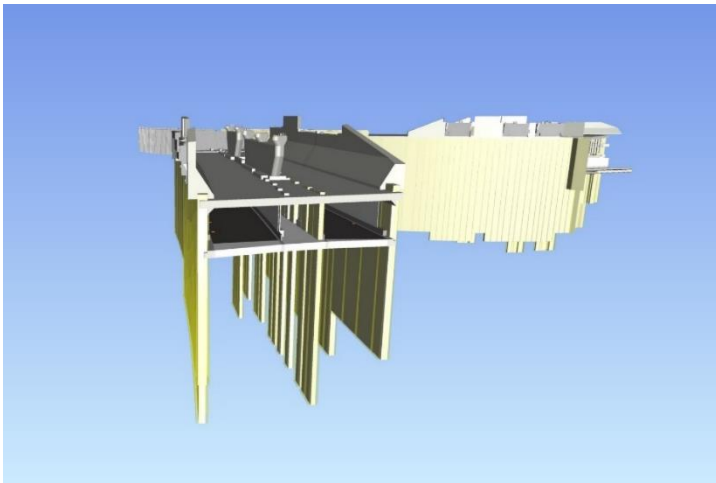
if prop.Name == "OwV_BIM_Fasering":
    fasering_value = prop.NominalValue.wrappedValue
    if fasering_value == "TT":
        elements_to_keep.discard(element)
        break

for element in model.by_type("IfcElement"):
    if element not in elements_to_keep:
        model.remove(element)

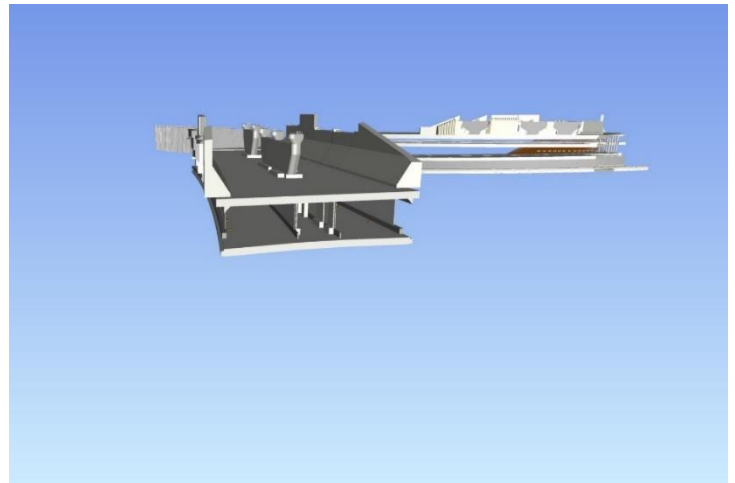
output_path = r"C:\Users\oveb\Downloads\01552 Filtered Tube.ifc"
model.write(output_path)
print(f"Filtered IFC saved to:\n{output_path}")

Filtered IFC saved to:
C:\Users\oveb\Downloads\01552 Filtered Tube.ifc

```



(a)



(b)

**Figure G1.** (a) BIM model with all the elements in place and (b) filtered model focusing on tunnel elements.

In addition to the fact that the applicable tunnel tube spaces are not clearly defined in the requirement text, the requirement also refers to supplementary documents. To fully understand the context of the requirement, it is important to take the information from these documents into account. These documents include technical drawings of the tunnel alignment and the corresponding coding. **Figure G2** presents an excerpt from the documents referenced in the requirement text. These documents indicate that the tunnel spaces must remain free of obstructions within the design and the model.

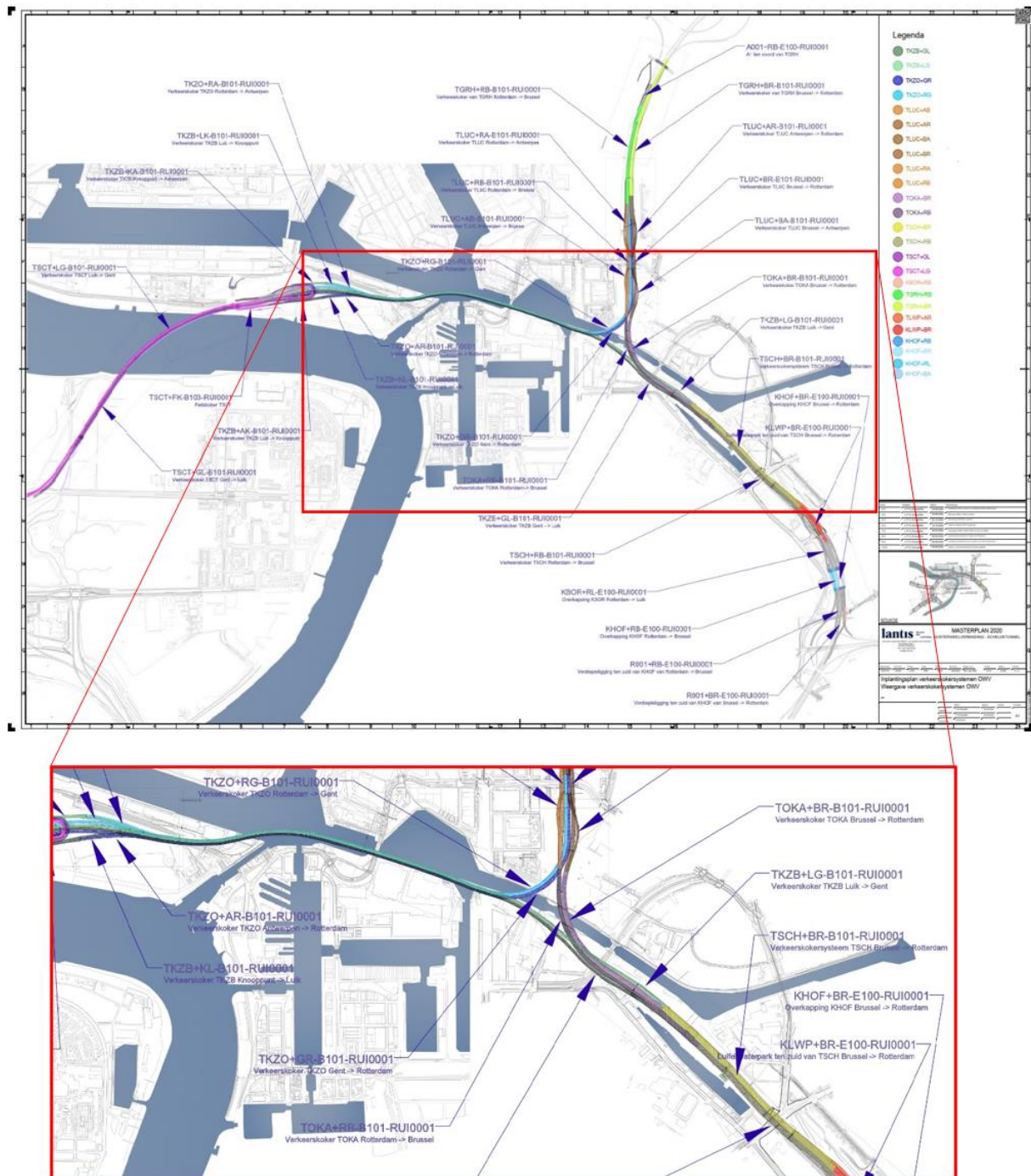


Figure G2. Impression from technical drawing referenced in the requirement.

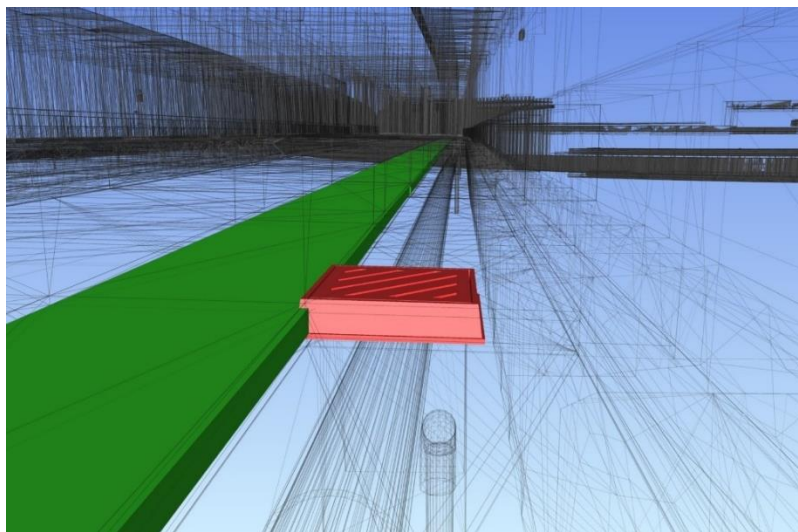
## Appendix G2: Verification in Navisworks

The verification was performed in Navisworks, where a clearance check was done between models related to the requirement and a clearance profile. The clearance profile is used to assess whether the traffic, escape- and service spaces within the tunnel elements are free from obstructions. A clearance check was done where 366 clashes were identified between the clearance profile and the model with a tolerance of 0 millimeters. These clashes were grouped by workset and revealed that over 70% of the total number of clashes occur in the Workset 310. General water management (see **Table G2**).

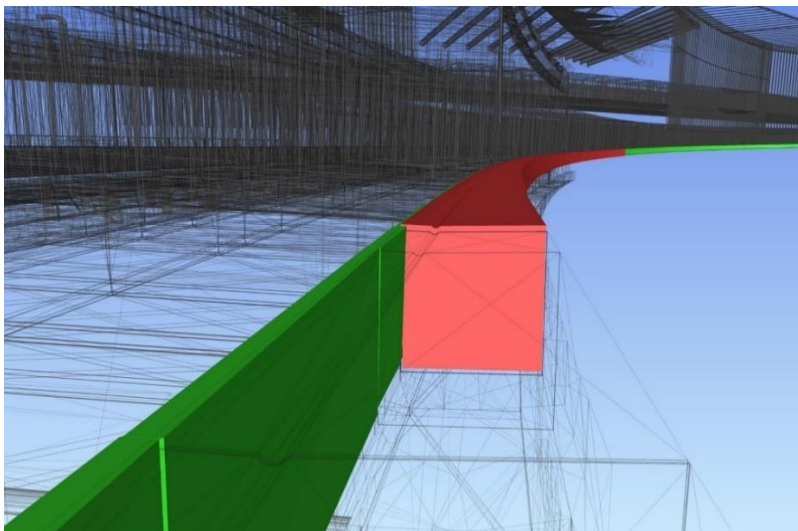
**Table G2.** Clashes broken down to workset.

Workset	Amount of clashes
301. Vloeren	33 (9%)
310. Waterhuishouding algemeen	270 (74%)
400. Algemeen Civiele Bovenbouw	8 (2%)
501. Voorzetwanden	16 (4%)
502. Vulwanden	8 (2%)
900. Algemeen Context & Equipment	31 (8%)

By increasing the tolerance to 0.05 meters, the total number of clashes decreased to 18. This reduction shows the sensitivity of clash detection outcomes to the selected tolerance level. **Figures G3-G8** show a clash for each workset from the clearance check outcomes. The clashes between the elements and the clearance profile are highlighted in green (clearance profile) and red (building model element).

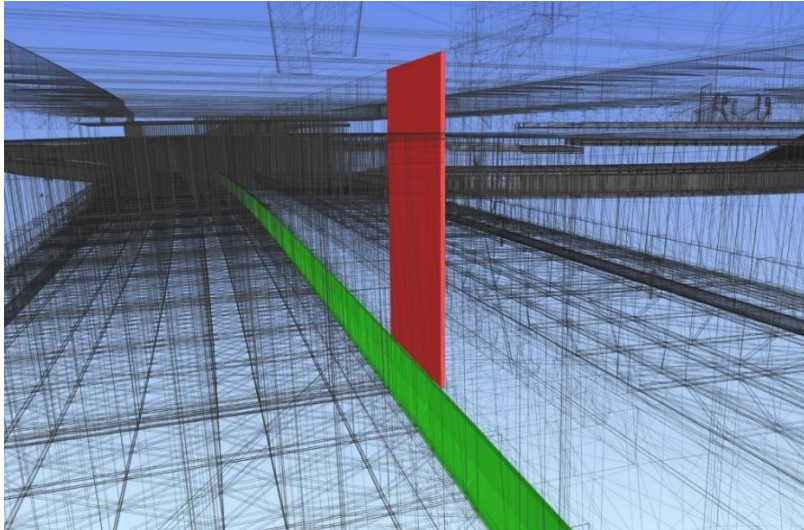


**Figure G3.** Clash example from Workset 310. Watermanagement general.

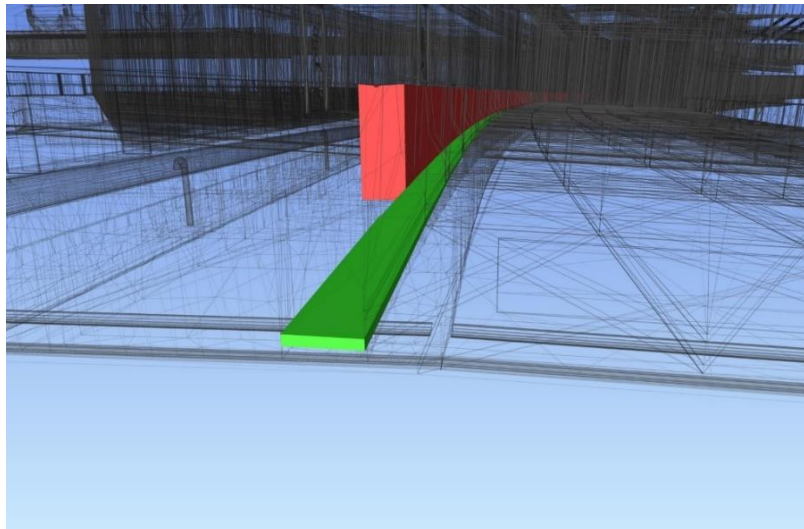


**Figure G4.** Clash example from Workset 501. Pre-walls.

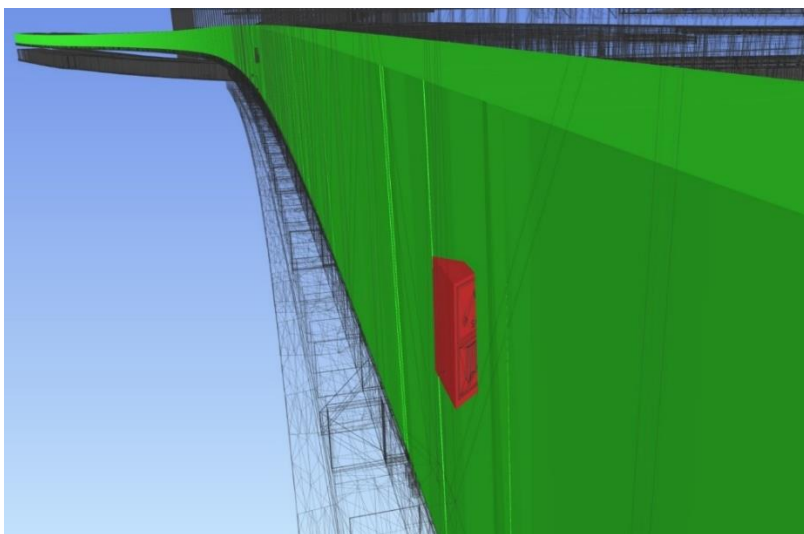




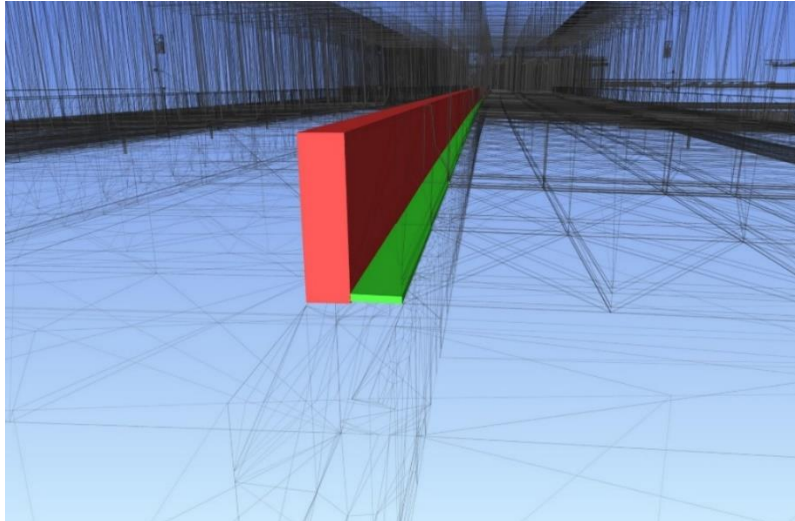
**Figure G5.** Clash example from Workset 502. Filling walls.



**Figure G6.** Clash example from Workset 400. General Civil Superstructure.



**Figure G7.** Clash example from Workset 900. General Context & Equipment.



**Figure G8.** Clash example from Workset 301. Floors.

## Appendix H: Verification of escape ducts (Class 4)

This appendix explains the Python script that was used to verify a requirement in Class 4 (see **Table H1**). The appendix is divided into two parts, where the first part (Appendix H1) looks at understanding the requirement and the filtering of the BIM models and Appendix H2 focuses on the verification itself.

**Table H1.** Requirement text used for verification of Class 4.

Class 4	Requirement text
	An escape duct in a tunnel construction must provide conflict-free space for the applicable PVR, finishing elements, installations, and all necessary free spaces.

### Appendix H1: Filtering of the models

To perform the verification, a Python script is used that systematically processes the IFC models to identify and retain only the elements relevant to the escape ducts within the tunnel constructions. Using the Python module `IfcOpenShell`, the script first loads the IFC file and begins iterating over all elements associated with property definitions. By looping through the model, it evaluates each elements attributes against predefined filtering criteria. The filtering is done based on the property `OWV_SE_fysiek_naam`, which contains the physical names of the elements in the model. This ensured that only elements corresponding to specific terms related to the escape ducts in the requirement text, such as ‘escape door’ and ‘escape corridor floor’, were retained. Also, an additional filtering was done using the `OWV_BIM_Fasering` property, which led to the removal of temporary construction elements in the models. These temporary construction elements are not directly related for the requirement verification. The filtered model only contains the relevant elements for the intended requirement verification. **Figure H1** provides a visual overview of the model, where **Figure H1a** displays the unfiltered model and **Figure H1b** shows the filtered model with the specified elements removed.

```
import ifcopenshell

ifc_path = r"C:\Users\oveb\Downloads\IFC tunneldelen\OWRB-01539-ROC-DM0-W66-000001.ifc"
model = ifcopenshell.open(ifc_path)

keywords = {
    "OWV_SE_fysiek_naam": ["Vloer vluchtgang", "Wandopvulling", "Vloer dienstgang", "Wandafwerking", "Constructiewand"],
    "OWV_SE_logisch_naam": ["Diepwand", "Vluchtdeur", "Uitvulling", "Bodemplaat tunnel", "Dak tunnel"]
}

excluded_family_name = "[DWA-DWP-TOL]_Diaphragm wall tolerance zone_(GMA2)"

element_to_propsets = {}

elements_to_keep = set()

for rel_defines in model.by_type("IfcRelDefinesByProperties"):
    prop_set = rel_defines.RelatingPropertyDefinition

    if not prop_set.is_a("IfcPropertySet"):
        continue

    for element in rel_defines.RelatedObjects:
        element_to_propsets.setdefault(element, []).append(prop_set)
```

```

if prop_set.Name == "Identity Data":
    for prop in prop_set.HasProperties:
        if prop.Name in keywords:
            try:
                value = prop.NominalValue.wrappedValue
            except:
                continue
            for keyword in keywords[prop.Name]:
                if keyword in value:
                    for element in rel_defines.RelatedObjects:
                        elements_to_keep.add(element)
                    break

for rel_defines in model.by_type("IfcRelDefinesByProperties"):
    prop_set = rel_defines.RelatingPropertyDefinition

    if not prop_set.is_a("IfcPropertySet"):
        continue

    if prop_set.Name == "Other":
        for prop in prop_set.HasProperties:
            if prop.Name == "Family":
                try:
                    family_value = prop.NominalValue.wrappedValue
                except:
                    continue
                if excluded_family_name in family_value:
                    for element in rel_defines.RelatedObjects:
                        elements_to_keep.discard(element)

    for rel_defines in model.by_type("IfcRelDefinesByProperties"):
        prop_set = rel_defines.RelatingPropertyDefinition

        if not prop_set.is_a("IfcPropertySet"):
            continue

        for prop in prop_set.HasProperties:
            if prop.Name == "OWV_BIM_Fasering":
                try:
                    fasering_value = prop.NominalValue.wrappedValue
                except:
                    continue
                if fasering_value == "TT":
                    for element in rel_defines.RelatedObjects:
                        elements_to_keep.discard(element)

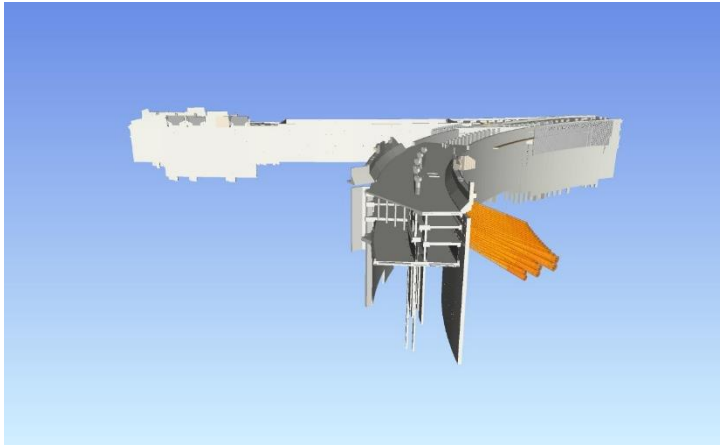
    for element in model.by_type("IfcElement"):
        if element not in elements_to_keep:
            model.remove(element)

output_path = r"C:\Users\oveb\Downloads\Escape tunnel 01539.ifc"
model.write(output_path)
print(f"Filtered IFC saved to:\n{output_path}")

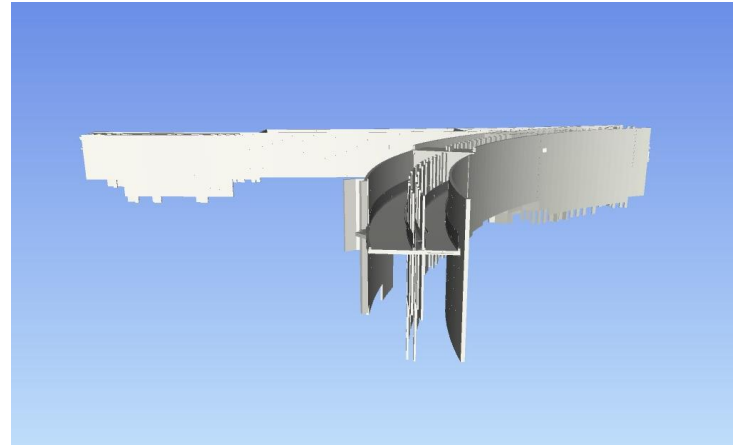
Filtered IFC saved to:
C:\Users\oveb\Downloads\Escape tunnel 01539.ifc

```





(a)

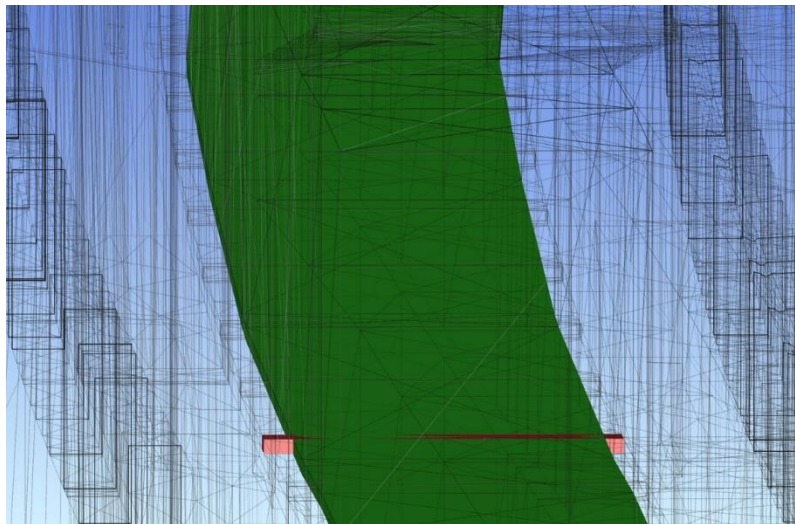


(b)

**Figure H1.** (a) BIM model with all the elements in place and (b) filtered model focusing on tunnel escape ducts.

## Appendix H2: Verification in Navisworks

The verification was done in Navisworks, where a clearance check was performed between the elements related to the requirement and a clearance profile. The clearance profile is used to assess whether the escape ducts within the tunnel elements provide conflict-free spaces. By doing the clash detection in Navisworks a total of 340 clashes between the model elements and the clearance profile were identified. For this clearance check a clearance tolerance of 0.001 meter was used. The majority of the detected clashes can be attributed to tunnel section 10, where 330 out of the total 340 clashes occur. An example of such a clash is shown in **Figure H2**. The remaining clashes occur across the other tunnel sections.



**Figure H2.** Clash between clearance profile and floor element in the BIM model.

## Appendix I: Efficiency metrics

This appendix elaborates on how the efficiency metrics were derived. The structure follows the same categorisation as presented in Section 3.5, distinguishing between classification, rule interpretation, preparation, execution and reporting of the verification process.

### Efficiency of the classification

To evaluate the efficiency requirement classification process, a sample of 10 requirements was used. These were manually categorised into geometrical and non-geometrical requirements which took 136 seconds in total. To estimate the total time for the complete dataset of 614 requirements, the following was done:

$$\begin{aligned}\text{Time} &= \frac{136}{10} \cdot 614 = 8350.40 \text{ seconds} \\ &= \frac{8350,4}{60} = 139.17 \text{ minutes}\end{aligned}$$

In contrast, the automated classification approach processed the full dataset in only 42 seconds. This results in a time reduction of:

$$\text{Time reduction} = \left(1 - \frac{42}{8350.40}\right) \cdot 100 = 99.5\%$$

### Efficiency of the rule interpretation

The efficiency of the rule interpretation using RASE cannot be directly compared to the traditional method, as RASE was not used previously and relies on manual input. Its repeatability is determined to be moderate while the structured algorithm ensures consistency, manual application introduces some variation. However, the quality of results is likely higher due to the improved clarity and structure RASE provides.

### Efficiency of the preparation

While precise time savings were not recorded, the semi-automated process for preparing building model data led to increased efficiency. Using IfcOpenShell scripts to automate element filtering minimizes manual work, indicating potential time reductions. This method also boosts repeatability by ensuring consistent filtering, avoiding the inconsistencies common in manual processes. Such consistency is especially valuable for Class 3 and 4 requirements that rely on semantic data. Additionally, quality improves as automation lowers the risk of errors, resulting in more reliable outcomes.

### Efficiency of the verification process

In terms of time savings, the Class 1 requirement was used as a quantifiable example. The verification of the requirement using manual checking took approximately 120 minutes, whereas the automated approach completed the same process in 80 seconds.

$$\text{Time reduction} = \left(1 - \frac{80}{120 \cdot 60}\right) \cdot 100 = 98.9\%$$

To assess the accuracy of the automated verification approach compared to the manual approach, ten measurements were analysed. These measurements were made in Navisworks<sup>1</sup>. **Table I1** shows the values and deviations for both approaches.

**Table I1.** Values of manual and automated measurements and their deviations.

Measurement	Value [m]		Deviation [m]	
	Manual	Automated	Manual	Automated
1	0.452	0.450	0.002	0.000
2	0.450	0.450	0.000	0.000
3	0.453	0.450	0.003	0.000
4	0.449	0.450	0.001	0.000
5	0.448	0.450	0.002	0.000
6	0.451	0.450	0.001	0.000
7	0.451	0.450	0.001	0.000
8	0.450	0.450	0.000	0.000
9	0.451	0.450	0.001	0.000
10	0.450	0.450	0.000	0.000

To quantify the average error, the mean absolute deviation (MAD) was calculated using the following formula:

$$\frac{1}{n} \sum_{i=1}^n |x_i - \bar{x}|$$

For the manual approach (man), this resulted in an MAD of:

$$MAD_{\text{man}} = \frac{0.002+0.000+0.003+0.001+0.002+0.001+0.001+0.000+0.001+0.000}{10} = \frac{0.0011}{10} = 0.0011 \text{ m}$$

For the automated approach (auto), the following MAD was calculated.

$$MAD_{\text{auto}} = \frac{0.000+0.000+0.000+0.000+0.000+0.000+0.000+0.000+0.000+0.000}{10} = \frac{0.000}{10} = 0.0000 \text{ m}$$

The relative deviation was calculated using the MAD for both approaches leading to the following results:

$$\text{Deviation}_{\text{man}} = \left( \frac{0.0011}{0.450} \right) \cdot 100 = 0.24\%$$

$$\text{Deviation}_{\text{auto}} = \left( \frac{0.0000}{0.450} \right) \cdot 100 = 0.00\%$$

## Efficiency of reporting the results

The reporting step cannot be directly compared, so no specific efficiency metric is available in terms of time savings. However, both manual and semi-automated methods are expected to yield similar results when repeated, as they follow consistent reporting procedures. While quality can't be measured quantitatively, both approaches likely produce comparable outputs. The semi-automated method does offer an advantage in traceability due to better integration with the CDE, avoiding reliance on external tools.

<sup>1</sup> When measurements done by a professional, they may vary from those measured by the author.

## Appendix J: openBIM approach

This appendix presents a verification example for a Class 1 requirement (see **Table J1**), previously introduced in Section 3.2.4.1, conducted using the openBIM approach. Central to this approach is the application of buildingSMART standards, aligning with several guidelines identified in Chapter 4, specifically guidelines S1, O1, O2 and B2. Although the use of the openBIM methodology deviates from the proposed workflow and therefore falls partially outside the primary scope of this research, it demonstrates the significant potential of bSI standards for enabling automated requirement verification.

**Table J1.** Requirement text used for verification of Class 1.

Requirement text
The pipe diameter of the embedded sewer system in the traffic ducts must be at least 200 mm.

By using the openBIM workflow, the requirement is expressed using the buildingSMART Information Delivery Specification standard. This standard directly addresses the challenges previously identified in this research, namely ambiguity in requirement texts and the lack of explicit linkage between requirements and building elements in the BIM model. The IDS is an XML-based schema that allows requirements to be defined in a machine-readable format. The IDS used for the example is presented in **Figure J1**.

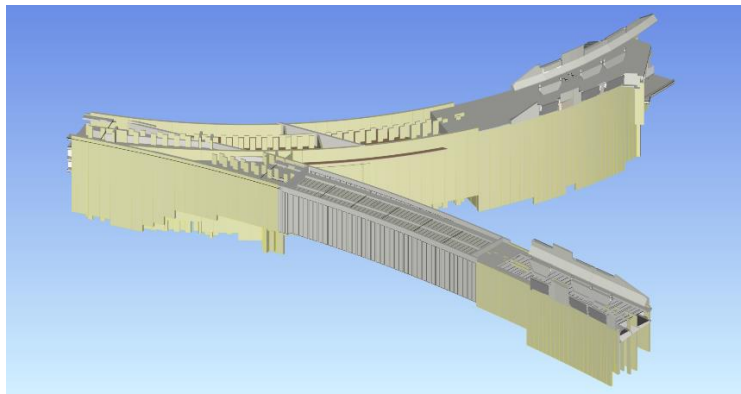
```
<ids:ids xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://standards.buildingsmart.org/IDS http://standards.buildingsmart.org/IDS/1.0/ids.xsd"
xmlns:ids="http://standards.buildingsmart.org/IDS">
  <ids:info>
    <ids:title>New ids file</ids:title>
  </ids:info>
  <ids:specifications>
    <ids:specification ifcVersion="IFC2X3" name="New Specification">
      <ids:applicability minOccurs="1" maxOccurs="unbounded">
        <ids:entity>
          <ids:name>
            <ids:simpleValue>IFCBUILDINGELEMENTPROXY</ids:simpleValue>
          </ids:name>
        </ids:entity>
      </ids:applicability>
      <ids:requirements>
        <ids:property dataType="IFCLENGTHMEASURE" cardinality="optional">
          <ids:propertySet>
            <ids:simpleValue>Dimensions</ids:simpleValue>
          </ids:propertySet>
          <ids:baseName>
            <ids:simpleValue>ROCO_CLE_Diameter_Inner</ids:simpleValue>
          </ids:baseName>
          <ids:value>
            <xs:restriction base="xs:double">
              <xs:minInclusive value="0.2" />
            </xs:restriction>
          </ids:value>
        </ids:property>
        <ids:property dataType="IFCTEXT" cardinality="required">
          <ids:propertySet>
            <ids:simpleValue>Identity Data</ids:simpleValue>
          </ids:propertySet>
          <ids:baseName>
            <ids:simpleValue>OWV_SE_logisch_naam</ids:simpleValue>
          </ids:baseName>
          <ids:value>
            <ids:simpleValue>Riool kunstwerk</ids:simpleValue>
          </ids:value>
        </ids:property>
      </ids:requirements>
    </ids:specification>
  </ids:specifications>
</ids:ids>
```

**Figure J1.** IDS schema for requirement definition using XML-format.

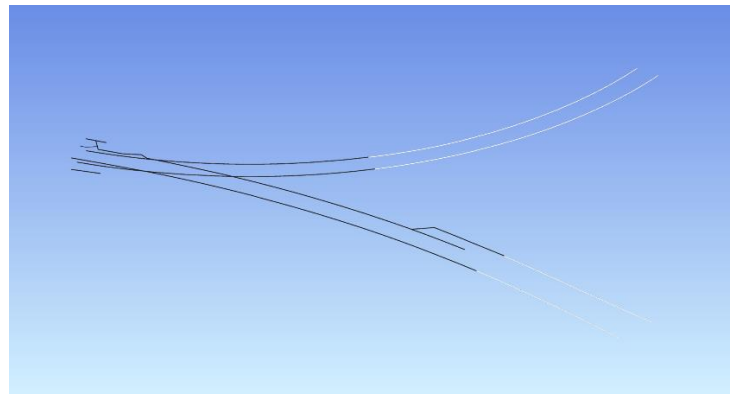
In this case, the IDS does specify that the IFC model MUST contain entities that have IFC class IFCBUILDINGELEMENTPROXY that MEET the following requirements:

- MAY HAVE property ROCO\_CLE\_Diameter\_Inner of Pset Dimensions (IFCLENGTHMEASURE)  $\geq 0.2$
- MUST HAVE property OWV\_SE\_logisch\_naam of Pset Identity Data (IFCTEXT) = Riool kunstwerk

This method of specifying requirements using the bSI IDS standard eliminates ambiguity from requirement descriptions and establishes a direct connection between the textual requirement and the corresponding model elements. As a result, the process of automated requirement verification becomes significantly less complex. Once the IDS is defined, it can be verified against the IFC file to determine whether the model complies with the specified requirements. **Figure J1a** shows the tunnel construction and in **Figure J1b** all the pipelines that need to be verified against the requirement are visible. The verification using the IDS showed that all the assessed pipelines comply with a minimum pipe diameter of 200 millimeters. These outcomes can be communicated to stakeholders.



(a)



(b)

**Figure J1.** (a) Overview of tunnel elements related to the verification of the requirement and (b) the isolated pipelines that need to be verified based on the requirement text.

