

Univerza v Ljubljani
Fakulteta *za gradbeništvo*
in geodezijo



Vicente MEDIANO CORONA

**AUTOMATED CODE COMPLIANCE CHECKS WITH
BIM**

**AVTOMATIZIRANO PREVERJANJE SKLADNOSTI
S PREDPISI V BIM**



European Master in
Building Information Modelling

Master thesis No.:

Supervisor:
Assist. Prof. Tomo Cerovšek, Ph.D.

Ljubljana, 2023



ERRATA

Page	Line	Error	Correction
-------------	-------------	--------------	-------------------

»This page is intentionally blank«

BIBLIOGRAFSKO – DOKUMENTACIJSKA STRAN IN IZVLEČEK

- UDK:** 004.946:005.591.6(043.3)
- Avtor:** Vicente Mediano Corona
- Mentor:** Assist. Prof. Tomo Cerovšek, Ph.D.
- Somentor:** Rok Jereb
- Naslov:** Avtomatizirano preverjanje skladnosti s predpisi v BIM
- Tip dokumenta:** Magistrsko delo
- Obseg in oprema:** 134 str., 8 pregl., 104 sl., 2 pril.
- Ključne besede:** BIM (Building Information Modelling), preverjanje skladnosti, preverjanje pravil, avtomatizacija, vizualno programiranje.

Izveček:

Preverjanje skladnosti s predpisi je ključnega pomena za zagotavljanje kakovosti, točnosti in ustreznosti gradbenih projektov. Ustaljeni, ročni, načini preverjanja skladnosti s predpisi so dragi in časovno potratni ter pogosto zahtevajo ponovno delo.

Pričujoča študija podaja znanstveno metodologijo za interdisciplinarni pristop k presoji, izvedbi in implementaciji avtomatizacije preverjanja skladnosti s predpisi z na osnovi BIM. Metodologija vključuje kvalitativne in kvantitativne metode ter aplikativen pristop. Kvalitativne metode vključujejo uporabo razlage pravil, klasifikacijske in kodirne sisteme ter prevajanje pravil z uporabo metodologije RASE (angl. Requirements, Applicability, Selection, and Exception) in jezikovne modele umetne inteligence. Kvantitativne metode se osredotočajo na standardizacijo in temeljijo na vizualnim programiranju in avtomatizaciji grafov za preverjanje skladnosti s predpisi.

Namen razvitega okvira je avtomatizirati preverjanje skladnosti s predpisi in izdelati praktično rešitev za bolj učinkovito in konsistentno načrtovanje. V sklepnem delu naloge dmonstriramo uporabo metodologije, na osnovi katere prikažemo njeno učinkovitost pri racionalizaciji avtomatizacije postopka preverjanja skladnosti.

Ugotovitve te študije prispevajo k napredku avtomatizacije preverjanja skladnosti na osnovi BIM, saj metodologija nudi praktično rešitev za strokovnjake. Z rezultati te raziskava želimo prispevati tudi k povečanju kakovosti in učinkovitosti načrtovanja stavb, ki temelji na postopni avtomatizaciji preverjanja skladnosti na prilagodljiv in sledljiv način.

»This page is intentionally blank«

BIBLIOGRAPHIC– DOCUMENTALISTIC INFORMATION AND ABSTRACT

- UDC:** 004.946:005.591.6(043.3)
- Author:** Vicente Mediano Corona
- Supervisor:** Assist. Prof. Tomo Cerovšek, Ph.D.
- Co-supervisor:** Rok Jereb
- Title:** Automated Code Compliance Checks with BIM
- Document type:** Master Thesis
- Scope and tools:** 134 p., 104 fig., 8 tab., 2 ann.
- Keywords:** BIM (Building Information Modelling), Compliance Checks, Rule-Checking, Automation, Visual Programming.

Abstract:

Compliance checks are crucial for ensuring the quality, accuracy, and regulatory compliance of building design projects. However, the industry's current manual workflows for code compliance checks often result in time and money loss, re-work, and frustration. This study presents a mixed-approach research methodology to automate BIM compliance checks and address these challenges.

The methodology incorporates qualitative and quantitative methods and an applied research approach. The qualitative methods involve rule interpretation using classification and encoding systems and rule translation using the RASE (Requirements, Applicability, Selection, and Exception) Methodology and AI-language models. In contrast, the quantitative methods focus on model standardisation, visual programming, and Automated Code Compliance Check graphs.

The developed framework aims to automate compliance checks for general regulations and provide a practical solution for efficient and regulatory-compliant building design processes. The methodology is applied to a real-world case scenario, demonstrating its effectiveness in streamlining the compliance checking process.

The findings of this study contribute to the advancement of BIM-based compliance checking methodologies and offer practical insights for industry professionals. Ultimately, this research seeks to improve the overall quality and effectiveness of building design projects by automating compliance checks in a seamless and adaptable manner.

»This page is intentionally blank«

ACKNOWLEDGEMENTS

This work represents the culmination of immense dedication, effort, time, and care. However, its realisation was only possible with everyone who stood alongside me during this journey.

First, I am grateful to my supervisor, Professor Tomo Cerovšek, for his patience and guidance throughout this dissertation. I am also thankful to Rok Jereb, my co-supervisor, and Blaz Budja for allowing me to work as a BIM Specialist in Jereb and Budja Arhitekti this semester. Thanks to my co-workers, Maja, Jernej, Petra, Nina, and Adrijana, for fostering a friendly and collaborative environment in the office. Their kindness, sense of humour, and professionalism made every day a joy to be part of.

My family have been a constant source of inspiration, as they were on my mind this whole time. I must express my profound appreciation to my mother, whose unconditional support and wisdom have guided me throughout my life, particularly these past years. I am genuinely thankful to my father for always being present and ready to lend a helping hand when needed. I must also thank my sisters, Fran and Cata, who have looked up to me and taken pride in my accomplishments. I dedicate this work to them.

To my dear BIM A+ classmates and friends, who have been more than peers, they became my substitute family during our master's program. Since September, we bonded closely, and their love, laughter, affection, and company will always have a special place in my heart. I genuinely hope each achieves a bright and prosperous future, fulfilling all their aspirations and dreams. I also wish we could meet again in some part of the world.

Finally, to my friends and the rest of my family in Chile and other parts of the world, who constantly checked on me, wished me the best, gave me support, and advised me on life. Thanks for making my past year in Europe less distant from home and for having me present in your thoughts.

Reflecting on my decision to study abroad nearly two years ago, I am overwhelmed with gratitude for the care and affection I have received from all those mentioned above and many others. I consider myself incredibly fortunate to have crossed paths with such remarkable people. Their support has been a driving force behind my achievements, and I will forever treasure them.

»This page is intentionally blank«

TABLE OF CONTENTS

ERRATA.....	I
BIBLIOGRAFSKO – DOKUMENTACIJSKA STRAN IN IZVLEČEK.....	IV
BIBLIOGRAPHIC– DOCUMENTALISTIC INFORMATION AND ABSTRACT	VI
ACKNOWLEDGEMENTS.....	VIII
TABLE OF CONTENTS	X
INDEX OF FIGURES.....	XII
INDEX OF TABLES.....	XV
GLOSSARY OF ACRONYMS	XVI
1. Introduction.....	1
1.1 Background and Context of the Problem	1
1.2 Motivation	2
1.3 Goals and Objectives.....	3
1.4 Research Questions	4
1.5 Partner Company.....	4
1.6 Thesis Structure	5
2. Theoretical Review	7
2.1 Code Compliance Checking in the AEC Industry	7
2.2 State of the Art of Computer-Assisted Code Compliance Checks	8
2.3 Automated Code Compliance Checking	16
2.4 Visual Programming Language.....	26
3. Methodology	28
3.1 Research Design and Approach	28
3.2 Framework Definition Methodology.....	30
3.3 Applied Methodology	44
4. Automated Compliance Check Framework	45
4.1 Use Cases	45
4.2 Model Preparation	49
4.3 Rule Codification	57

4.4	Rule Classification	61
4.5	Rule Interpretation	67
4.6	Rule Translation	75
4.7	Rule Execution	78
4.8	Checking Results Reporting	81
4.9	Continuous Refinement of the Framework.....	84
5.	Application to Case Study.....	86
5.1	Slovenian Regulation for Residential Building (SRRB).....	86
5.2	Case Study Selection: Podbreznik B7-B9	88
5.3	Model Preparation	89
5.4	Case Study: Article 12, Rule N°1	94
5.5	Case Study: Article 14, Rule N°2	99
5.6	Case Study: Article 20	105
6.	Results and Discussion	110
6.1	Case Study Analysis.....	110
6.2	Findings from the Automated Compliance Checking implementation.....	112
6.3	Analysis of the effectiveness and efficiency of the ACCC process	113
6.4	Challenges and limitations of the process	114
7.	Conclusions	116
7.1	Summary of the study and main findings	116
7.2	Contributions and implications for the industry.....	117
7.3	Limitations of the study and recommendations for future research	117
7.4	Final Thoughts.....	119
8.	References	121
9.	Appendices	¡Error! Marcador no definido.

INDEX OF FIGURES

Figure 1: Typical Manual Compliance Check Workflow in BIM.....	1
Figure 2: Automated Compliance Check Workflow in BIM	2
Figure 3: Jereb in Budja Arhitekti Logo.....	4
Figure 4: Typical Building Lifecycle in AECO Projects.....	7
Figure 5: Rule Semantics and Spatial Elements.	8
Figure 6: Main issues with Manual Code Compliance Checks	8
Figure 7: Timeline of International Research into CCC [14].....	9
Figure 8: Decision Tables examples in SASE [16].	10
Figure 9: CORENET platform [18].....	10
Figure 10: e-PlanCheck Interface [18].....	11
Figure 11: Fornax Interface [19].....	11
Figure 12: Solibri Model Checker Interface.	12
Figure 13: ResCheck-Web interface [20].	13
Figure 14: Timeline of major ACC systems [1].	14
Figure 15: Automation Challenges for Code Compliance Check.....	16
Figure 16: Common structure of an ACCC process, inspired by [4]	17
Figure 17: Taxonomy and Ontology	19
Figure 18: Rule Taxonomy Example	20
Figure 19: Rule Ontology Example using RDF.....	22
Figure 20: Pseudo-coded Regulation Example	23
Figure 21: Artificial Intelligence in ACCC.....	24
Figure 22: Advantages of VPL	26
Figure 23: Methodology Phases	28
Figure 24: Methodology Workflow.....	29
Figure 25: Framework Functions and Methods Steps	31
Figure 26: Software and tools used.....	32
Figure 27: Rule Codification Step's IDEF0.....	32
Figure 28: Rule Classification Step's IDEF0.....	33
Figure 29: Rule Codification & Classification Concept.....	33
Figure 30: Rule Interpretation Concept	35
Figure 31: Rule Interpretation Step's IDEF0.....	36
Figure 32: Model Preparation Concept	37
Figure 33: Model Preparation Step's IDEF0.....	37
Figure 34: Rule Translation Concept.....	38
Figure 35: Rule Translation Step's IDEF0.....	39
Figure 36: Rule Execution Concept.....	40
Figure 37: Rule Execution Step's IDEF0	40
Figure 38: Report Checking Results Concept	41
Figure 39: Report Checking Results Step's IDEF0.....	42
Figure 40: ACCC Framework's Application IDEF0	42

Figure 41: ACCC Framework's Application IDEF0 decomposition	43
Figure 42: General Use Case	45
Figure 43: AECO Disciplines and Building's Life Cycle for Use Cases	46
Figure 44: Architecture & Design Phase use case.	47
Figure 45: Engineering & Design and Construction Phases use case.	48
Figure 46: Facility Management & Operation Phase use case.	48
Figure 47: Model Preparation Considerations	49
Figure 48: QA & QC Aspects	50
Figure 49: Consistency and Good Practices Aspects.....	51
Figure 50: Data Organisation and Formatting Aspects.....	52
Figure 51: Adherence to Industry Standards and Guidelines Aspects.....	53
Figure 52: Model Checker's Wizard for configuring QA/QC checks.....	55
Figure 53: Filtering Views with Project Browser Parameter.	55
Figure 54: Model Checker Report for pre-defined checks.	56
Figure 55: Predefined Check Set for Uniclass and its Check Report.....	57
Figure 56: Encoding Structure	61
Figure 57: Classification Table Structure.....	62
Figure 58: RASE Methodology.....	68
Figure 59: Main AI-Language Models in the Market.....	69
Figure 60: Rule Translation Considerations	75
Figure 61: Rule Translation Procedural Approach.....	77
Figure 62: Dynamo Player Interface.....	78
Figure 63: Rule Execution Considerations	78
Figure 64: Input Node Definition	79
Figure 65: Input Node Renaming	79
Figure 66: Instructions for Rule Execution.....	80
Figure 67: Type of Compliance Reports.....	81
Figure 68: Compliance Report Formats	82
Figure 69: Overridden Window within a Model Plan View	84
Figure 70: Case Study presentation: Podbreznik B7-B9	88
Figure 71: Model Preparation's Check Set Configuration for Rooms.....	89
Figure 72: Model Preparation's Parameter Filtering for Rooms.....	89
Figure 73: Model Checker Report Run.....	90
Figure 74: Missing Metadata in Rooms.....	90
Figure 75: Apartment Rooms Coding System Format.....	91
Figure 76: Room Naming Graph - Example for Main Bedroom	92
Figure 77: Room Naming Complete Graph	92
Figure 78: Room Numbering Graph	93
Figure 79: Room Apartment Numbering Graph.....	93
Figure 80: Room Apartment Coding Graph.....	93
Figure 81: Model Checker Final Report Run.	94
Figure 82: Article 12, Rule N°1 Dynamo Graph – Complete.....	96

Figure 83: Article 12, Rule N°1 Dynamo Graph – Section 1	97
Figure 84: Article 12, Rule N°1 Dynamo Graph – Section 2	97
Figure 85: Article 12, Rule N°1 Dynamo Graph – Section 3	97
Figure 86: Article 12, Rule N°1 Dynamo Player Settings	98
Figure 87: Article 12, Rule N°1 - Compliance Report's Dynamo Graph	98
Figure 88: Article 12, Rule N°1 - Compliance Report	99
Figure 89: Article 14, Rule N°2 Dynamo Graph – Complete	101
Figure 90: Article 14, Rule N°2 Dynamo Graph – Section 1	102
Figure 91: Article 14, Rule N°2 Dynamo Graph - Section 2	102
Figure 92: Article 14, Rule N°2 Dynamo Graph - Section 3	102
Figure 93: Article 14, Rule N°2 Dynamo Graph - Section 4	103
Figure 94: Article 14, Rule N°2 - Dynamo Player Settings	103
Figure 95: Article 14, Rule N°2 - Compliance Report's Dynamo Graph	104
Figure 96: Article 14, Rule N°2 - Compliance Report	104
Figure 97: Article 20 Dynamo Graph – Complete	107
Figure 98: Article 20 Dynamo Graph – Section 1	107
Figure 99: Article 20 Dynamo Graph – Section 2	107
Figure 100: Article 20 Dynamo Graph – Section 3	108
Figure 101: Article 20 - Dynamo Player Settings	108
Figure 102: Article 20 - Compliance Report's Dynamo Graph	108
Figure 103: Article 20 - Compliance Report #1	109
Figure 104: Article 20 - Compliance Report #2	109

INDEX OF TABLES

Table 1: State of the Art of Computer-Assisted Code Compliance Checks	15
Table 2: Rule Encoding Examples	61
Table 3: Rule Classification Template	67
Table 4: Urban Compliance Tables as a Revit Annotation Families Example	83
Table 5: Slovenian Residential Building Code Articles	87
Table 6: SRRB Article 12, Rule N°1 Classification Table	95
Table 7: SRRB Article 14, Rule N°2 Classification Table	100
Table 8: SRRB Article 20 Classification Table.....	105

GLOSSARY OF ACRONYMS

ACCC: Automated Code Compliance Checks.

ACC: Automated Compliance Checks.

AEC: Architecture, Engineering, and Construction.

AECO: Architecture, Engineering, Construction, and Operation.

AI: Artificial Intelligence.

AISC: American Institute of Steel Construction.

BERA: Building Environment Rule and Analysis.

BIM: Building Information Modelling.

BEP: BIM Execution Plan.

CCC: Code Compliance Check.

CORENET: Construction and Real Estate Network

EDM: Express Data Manager.

FM: Facility Management

IDEF0: Integration Definition for Function Modelling.

IDS: Information Delivery Specification.

IDM: Information Delivery Manual.

ISO: International Organisation for Standardisation

IFC: Industry Foundation Classes.

LOD: Level of Development.

LOIN: Level of Information Need.

MCCC: Manual Code Compliance Check.

MDA: Model-Driven Architecture.

ML: Machine Learning.

MVD: Model View Definitions.

NLP: Natural Language Processing.

QA: Quality Assurance.

QC: Quality Control.

RASE: Requirements, Applicability, Selection, and Exception.

RDF: Resource Description Framework.

SASE: Standards, Analysis, Synthesis, and Expression.

SRRB: Slovenian Regulations for Residential Buildings

SICAD: Standards Interface for Computer-Aided Design.

UML: Unified Modelling Language.

VPL: Visual Programming Language.

»This page is intentionally blank«

1. INTRODUCTION

1.1 Background and Context of the Problem

In Architecture, Engineering, and Construction (AEC) projects, ensuring compliance with regulations and bylaws is a critical aspect of the building design process. Compliance checks are vital in maintaining quality, safety, and legal adherence for construction projects. However, the industry's current state of compliance checks is an intricate process (Figure 1). It often relies heavily on manual processes, leading to inefficiencies, errors, and delays in project delivery.

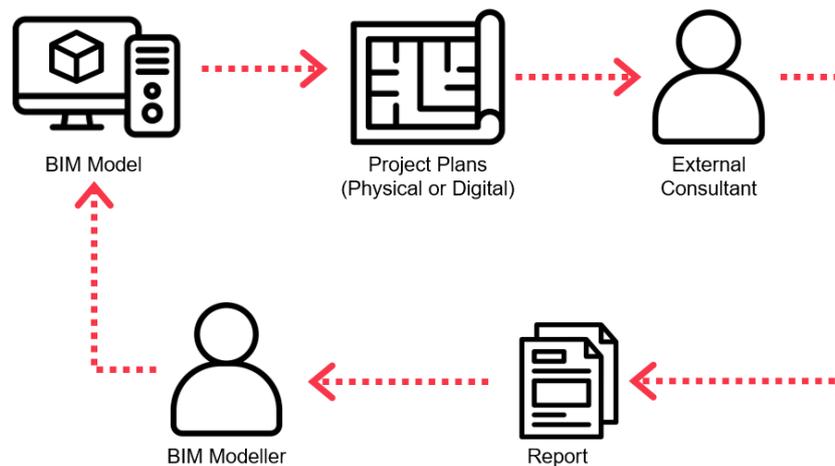


Figure 1: Typical Manual Compliance Check Workflow in BIM

Nevertheless, Building Information Modelling (BIM) has emerged in the past decades as a transformative technology in the construction industry, offering significant advantages in collaboration, visualisation, and data management. However, integrating compliance checks into the BIM workflow has proven to be an incomplete task.

The existing workflows for compliance checks often involve time-consuming manual inspections and rule interpretation, which utilise substantial resources and present risks of oversight and non-compliance. These challenges are particularly evident during the design phases, where intricate regulations and complex design models must be evaluated simultaneously.

This study aims to explore and develop an automated process for rule checking and the design of checking tools in the context of building design. The aim is to create a streamlined and efficient workflow (or Framework) to generate automated code compliance check tools that can be customised and applied to different regulatory

environments. By doing so, the study seeks to address the challenges associated with manual compliance checks, such as time-consuming processes, potential errors, and the need for repetitive manual interventions.

This work's purpose is also double-sided: first, to enhance the overall efficiency of the building design process by automating rule checking, thus reducing the time and effort required for compliance assessments. Second, to ensure regulatory compliance by using automated checks that are accurate, consistent, and adaptable to various regulations and bylaws.

By using Visual Programming Language (VPL) and avoiding "black box" tools, this work aims to create an automated and practical solution that enables architects, designers, and other stakeholders to focus more on the creative aspects of design while ensuring compliance with relevant regulations.

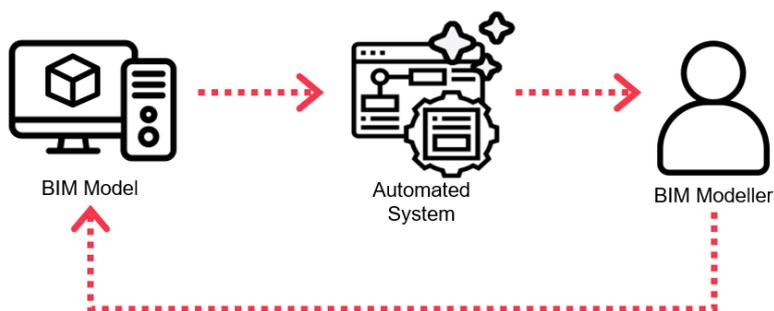


Figure 2: Automated Compliance Check Workflow in BIM

Furthermore, by integrating adjustable parameters into the automated process, the study aims to provide flexibility in accommodating different regulatory environments and adapting to specific project requirements. This allows the automated system to be customised and modified according to the needs of other building projects.

1.2 Motivation

The motivation behind this study originated from previous experiences and observations as an architect and professional in the field of building design, particularly in the context of Building Information Modelling (BIM) and architectural project development. In this field, the existing state of code compliance workflows often presents challenges and limitations hindering smooth project progression.

Manual compliance checks present significant challenges. The iterative nature of these checks, especially when dealing with complex models or regulations, generates

inefficiencies and frustration in the design process. Such features contradict BIM principles, which aim to enhance collaboration, accuracy, and efficiency in the construction industry.

One of the primary incentives for this research is to address the substantial time loss and rework associated with manual compliance checks. This approach is significantly time-consuming, prone to human error, and increases the risk of overlooking requirements or misinterpreting complex regulations, potentially leading to non-compliant designs. These errors raise legal and safety concerns and result in additional costs and delays in project delivery.

Another motivation comes from the belief that building design needs a more fluid and adaptable approach to compliance checks. Current workflows often rely on static rule sets that fail to capture various regulations and bylaws' dynamic nature and complexities.

A final incentive originates from the desire to align compliance checks with the principles of BIM and incorporate the potential of technology and automation in this part of the design process. By automating compliance checks, the study aims to free up designers' time and mental resources, enabling them to focus on creative aspects and innovative solutions while ensuring adherence to regulations, which is not happening in the current situation.

1.3 Goals and Objectives

This research aims to develop an efficient and adaptable framework for automated compliance checks that can be incorporated to aid in the BIM design process. The specific objectives include:

- Identifying the critical challenges associated with manual compliance checks and exploring their impact on the building design process.
- Ensuring the quality of BIM models for an efficient automation process.
- Creating a general workflow for automated code compliance checks, adaptable to various regulatory environments and eventual modifications.
- Analysing different bylaws and regulations to classify them and their characteristics.
- Developing visual programming-based tools and graphs for automated code compliance checks.

1.4 Research Questions

This study aims to address the following research questions:

1. What are the challenges associated with manual compliance checks, and how do they impact the building design process?
2. How can compliance checks be effectively integrated into the BIM building design process to ensure efficiency and regulatory compliance?
3. Which tools and methods can be helpful in the transformation of written-text rules into machine-readable coding?
4. What are the critical attributes and characteristics of various categories of regulations that must be captured and translated into automated compliance checks?
5. How can rules' attributes be structured and standardised to enable accurate interpretation by automated tools?
6. How can visual programming languages be employed to develop practical and user-friendly tools for automated code compliance checks?
7. What role do transparency and customizability play in ensuring professionals can trust and effectively use these tools?
8. How can Automated Compliance Checks be adaptable and flexible for the AECO industry in the BIM context?

1.5 Partner Company

This dissertation was developed in collaboration with Jereb in Budja arhitekti. The company is an Architecture Studio based in Ljubljana and was founded by Rok Jereb and Blaž Budja in 2007, both architects from the Faculty of Architecture, University of Ljubljana.



Figure 3: Jereb in Budja Arhitekti Logo

They have developed several projects throughout the country, including architecture and urban planning. They have been awarded numerous architectural prizes and awards and have won several competitions in the Slovenian context.

Their focus on Architecture Design is based on a holistic approach while embracing concepts like integration into the local environment, the spatial context, care for materials and details, and attention to the user. Their projects usually focus on providing high-quality architecture while incorporating socially responsible solutions.

In the past years, they decided to embrace the challenge of incorporating BIM technologies in their workflows, which is an ongoing process. Currently, the office is working with a hybrid mix of Level 1 and Level 2 BIM implementation, having projects in 2D, 3D, and some BIM Models. The company's leader's interest in a seamless BIM workflow is high. This is why, before the end of the current year (2023), they intend to have a complete Level 2 Implementation.

During the whole process of this work's development, I was hired as a BIM Specialist. This was a way of being familiarised with their workflows and projects, which helped implement the automated compliance-check workflow on the provided project and meet their needs.

1.6 Thesis Structure

Chapter 1: Introduction. The opening chapter provides an overview of the research problem, the reasons for conducting the study, the objectives, the research questions, and the involvement of a partner company will be provided.

Chapter 2: Theoretical Review. This chapter examines the existing practices of code compliance checking in the AEC industry. It explores the current state of automated code compliance solutions by studying and analysing academic and scientific literature.

Chapter 3: Methodology. Outlines the research design and approach to address the automated code compliance check problem, including the framework definition methodology and applied research methods. The chapter establishes the robust foundation upon which the subsequent research develops.

Chapter 4: Automated Code Compliance Check Framework. Throughout this pivotal chapter, the critical components of the framework are outlined and designed, including rule analysis and interpretation techniques, model standardisation methods, and visual programming workflows.

Chapter 5: Application to Case Studies. Chapter 5 focuses on practically demonstrating the automated code compliance framework by applying it to selected use cases provided by the Partner Company. The chapter provides insights into the Case Study selection process and then proves the application and effectiveness of the framework.

Chapter 6: Result and Discussion. This chapter presents the research findings. The results are analysed and interpreted, offering a complete understanding of the automated code compliance check's performance and limitations.

Chapter 7: Conclusion and Future Work: The thesis's final substantive chapter summarises the research's main findings and contributions. It highlights the study's practical implications for the AEC industry, emphasising the potential benefits of automated code compliance checking. Additionally, it acknowledges the limitations of the research and provides recommendations for future work and improvements.

2. THEORETICAL REVIEW

2.1 Code Compliance Checking in the AEC Industry

Code Compliance Checking (CCC) is crucial for the completion of Architecture, Engineering, Construction, and Operation (AECO) projects and throughout their lifecycle [1], [2]. Nevertheless, nowadays, there is a significant breach between legislation and BIM models, as regulations have existed since the beginning of civilisation [3]. The digital revolution has appeared only in the past decades [4] and continues to thrive.

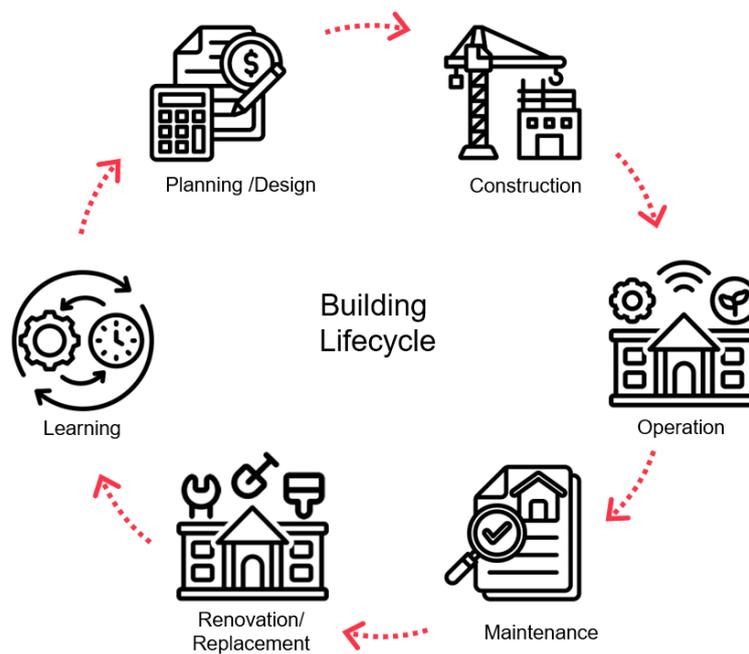


Figure 4: Typical Building Lifecycle in AECO Projects.

This problem is mainly associated with the relationship between written-word concepts and spatial elements and their representations (geometrical, technical, figurative, conceptual, etc.). This implies that rules may have varying semantic interpretations and sometimes present challenges regarding congruence with building elements and their constraints. Consequently, this issue has separated projects' Design Processes and Code Compliance Checks processes.

The most common workflow for CCC is to analyse project compliance manually and directly on their 2D representations, like floorplans, sections, and elevations [4], and usually through the evaluation of an external consultant or professional [5] (as shown in Figure 1).



Figure 5: Rule Semantics and Spatial Elements.

This manual workflow has been widely proven tedious, time-consuming, and prone to errors. It can generate enormous losses of time and money in the Design, Construction, and Operation Phases if there are any non-compliances [4], [6]–[9]. Moreover, the issue could be accentuated in the late phases of development, especially if there is any rework to be done.

Solihin and Eastman (2015) even state that visual inspection is unreliable for assuring BIM model compliance and quality [10]. This happens because the complexity and detail level of the latter is increasing with time, and analogical methods cannot cope with it. Paradoxically, this fact is also a barrier to achieving automation because of the efforts in time and money needed for software development [11].



Figure 6: Main issues with Manual Code Compliance Checks

Another relevant issue with Compliance Check is that currently, there are no standards regulating the processes and workflows [12]. This has led to countless approximations for checking projects and no consistency regarding their procedures.

2.2 State of the Art of Computer-Assisted Code Compliance Checks

Even though the Manual process of Code Checking in the AEC industry is still nowadays the primary methodology for Compliance Checks, its digitalisation is critical for a more efficient and accurate process. Efforts for computerising it have existed as early as the late 60s -with elementary implementation levels- and had subsequently evolved into

more complex solutions (Figure 7), making its complete automation one of today's primary goals.

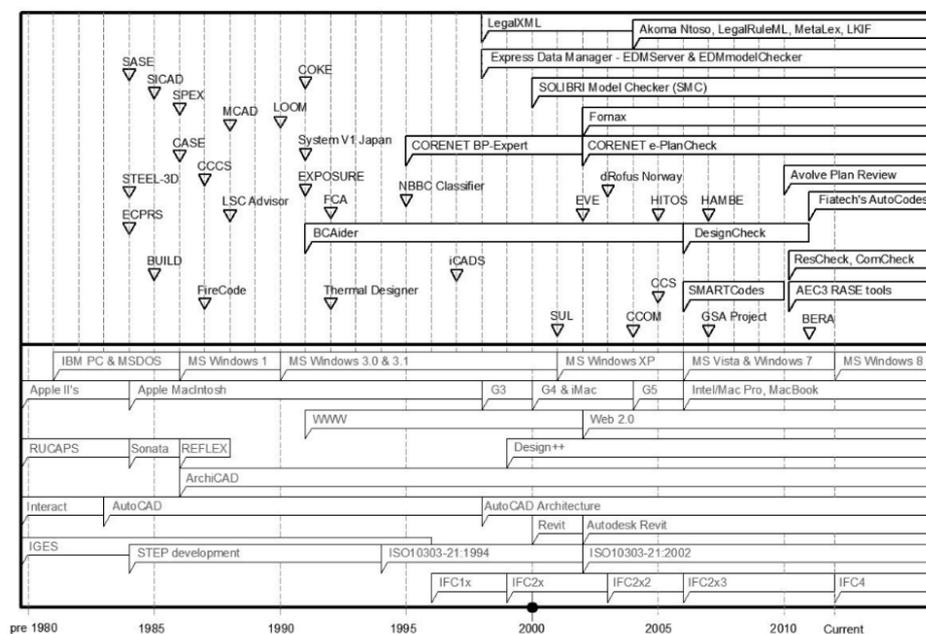


Figure 7: Timeline of International Research into CCC [14]

In 1969, the American Institute of Steel Construction (AISC) made the first approach towards Computer-Assisted Compliance Checks with AISC Specifications by proposing a set of interconnected decision tables for requirements in the steel building process. The idea was to use these representations (decision trees) to design and construct steel buildings to be machine-readable [13]. Even though this was not yet fully computer-assisted, it was used as a guideline for introducing computerisation into the compliance check workflows [14].

Various years later, in 1983, the Standards Interface for Computer-Aided Design (SICAD) was introduced by the US National Bureau of Standards [15], providing structural engineers with a user-friendly Standard to create and analyse structural models and to perform automated compliance checks with building codes and regulations.

The SICAD prototype demonstrated how designed components could be checked for compliance with design standards using application program databases and included three standards processors (organisational network, information network, and mapping processor), as well as a one-way mapping functionality that facilitated the user's ability to navigate, evaluate, and extract the required information from standards. Moreover, additional inputs from the user could address any missing or incomplete data [13], [14].

In 1987, the same organisation published SASE (Standards Analysis, Synthesis, and Expression), a significant early system for managing and maintaining standards. SASE provided a systematic structure for families of related codes across many US jurisdictions and tools for creating and checking decision tables, information networks, classification systems, and the organisation of provisions of those standards. It also enabled standards writers to add additional features according to specific standards and develop and manage alternative or successive versions of a standard [13], [14], [16].

		1	2	E
1	Use group L2 or L3 (2)	T	F	
2	Special order of building official (3) AND existing building (4) AND height not greater than 5 stories (6) AND height not greater than 65 feet (7) AND more adequate exitway impossible (9)		T	
1	Requirement = satisfied	X	X	
2	Requirement = violated			X

		1	2
1	Design live load \geq 100 psf (16)	T	F
1	Requirement = satisfied	X	
2	Requirement = violated		X

Figure 8: Decision Tables examples in SASE [16].

Later, in 1995, Singapore’s Ministry of National Development created, through the Building and Construction Authority [17], the Construction and Real Estate Network (CORENET) as a platform (Figure 9) for information exchange, integration, and collaboration between projects’ stakeholders in the AEC industry [18]. Afterwards, in September of 2000, the e-PlanCheck Project was launched within the same platform as an approach of using AI in code-checking for digital models using IFC (Figure 10), which was a very ambitious attempt for its time [3].

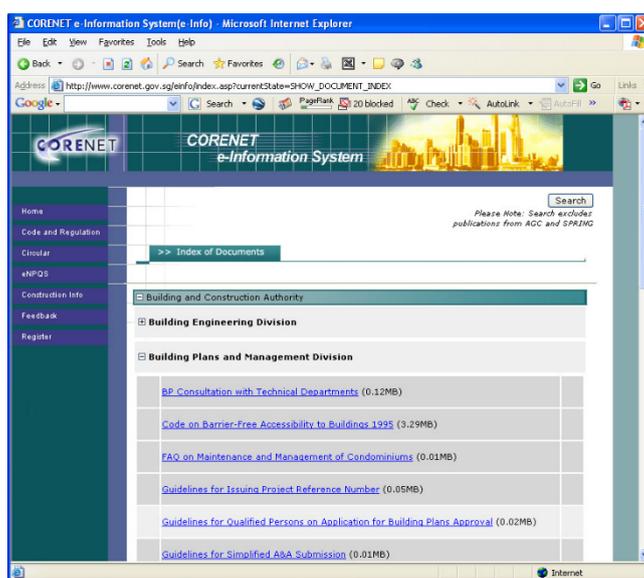


Figure 9: CORENET platform [18]

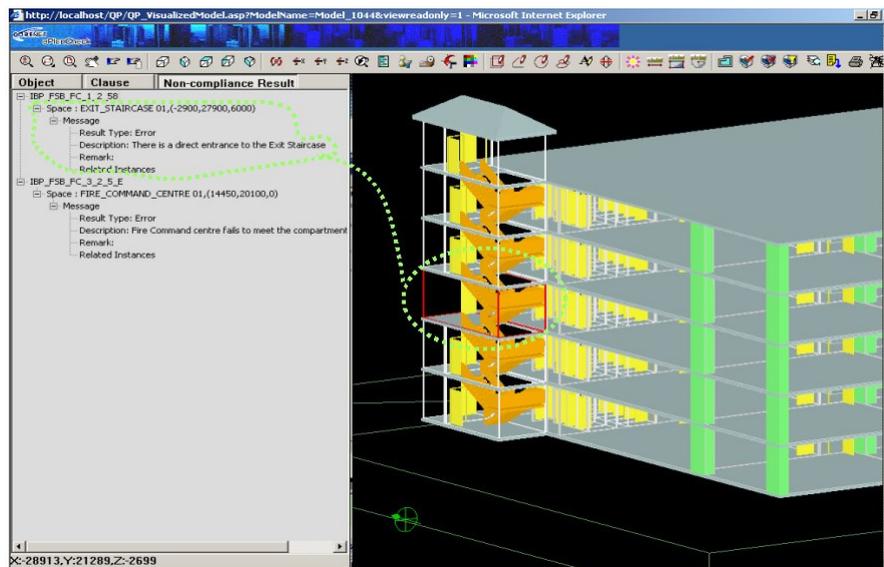


Figure 10: e-PlanCheck Interface [18]

Finally, in 2002, an independent solution called FORNAX was provided for the platform, developed by a private institution. It was an object-based solution with higher semantic levels in the context of code compliance Checks that used C++ as its programming language [4], [6], [18], [19].

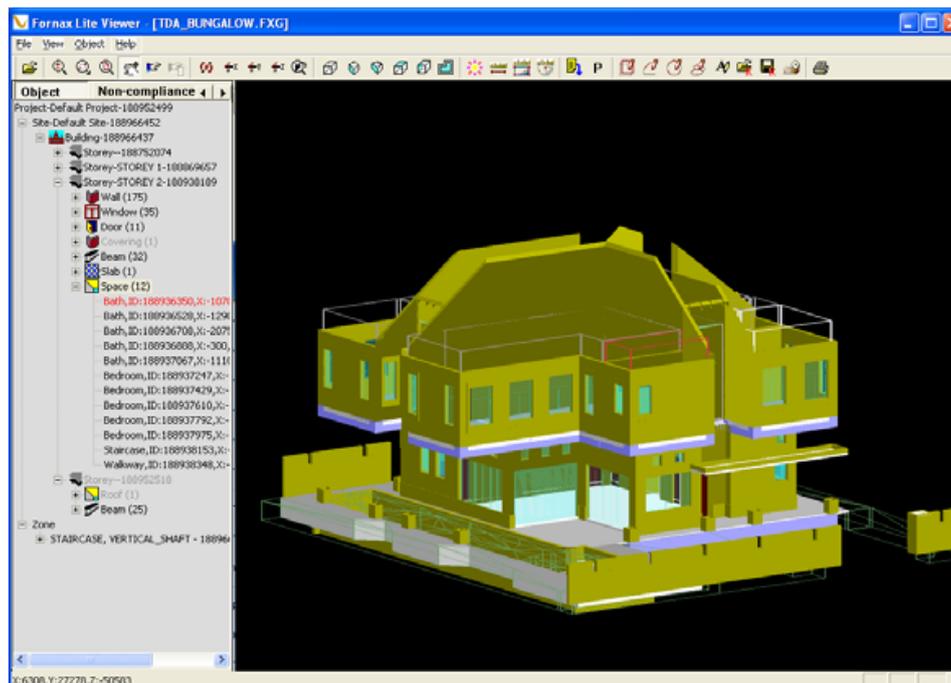


Figure 11: Fornax Interface [19]

Parallely, the Express Data Manager (EDM) Suite was developed by Jotne EPM Technologies in Norway in 1998 [14]. This software integration platform facilitated the interoperability of IFC models [6]. Moreover, although it was initially designed as a

collaboration tool, EDM now includes various applications, such as EDMmodelChecker and EDM Model Server [14]. EDM supports open development using EXPRESS and EXPRESS-X, object-oriented query languages that offer flexibility but require broad expertise to use effectively [8].

Solibri Model Checker was developed in Finland in 2000 as an IFC-compatible Java-based program to check models automatically. Nowadays, it has built-in capabilities for quality and integrity, geometry, issues, spatial programs, and path checking within models, among others. Even though the checking rules can be parametrised through an easy-to-use interface (Figure 12), the solution is limited in flexibility as it does not publicly provide a customisable programming interface [4], [6]. Solibri is considered the commercially most (if not only) successful commercially available automated compliance check tool [3].

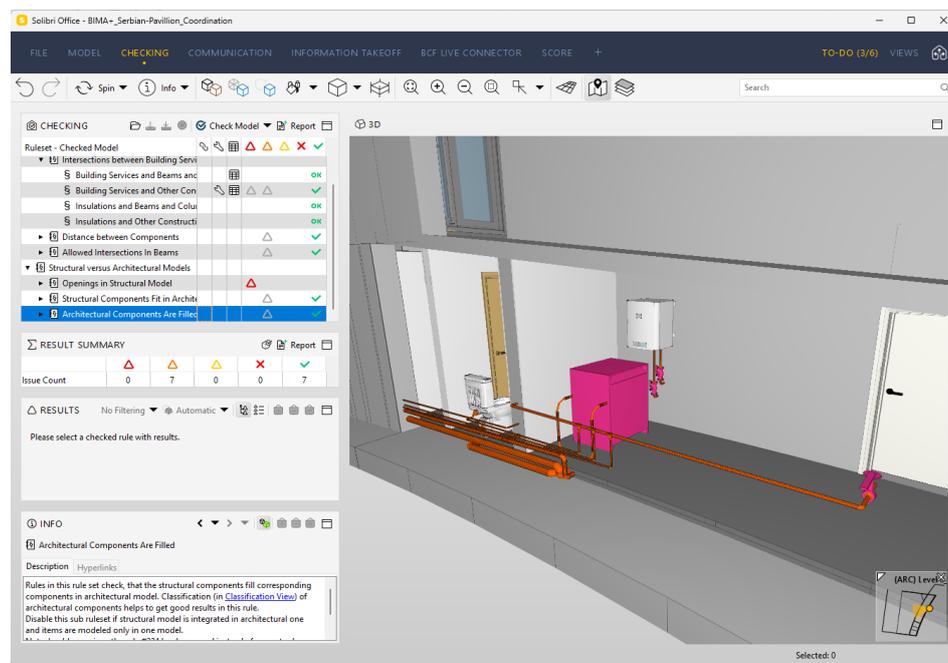


Figure 12: Solibri Model Checker Interface.

Using EDM as its core and base compliance-checking engine, another software named DesignCheck was created in Australia by the Cooperative Research Centre (CRC) for Construction Innovation in 2006. It consists of thorough design information mapping onto building codes. It allows compliance checks in different design stages using the IFC model, offering flexibility in the automation of code checking for architects, designers, and certifiers [6]. The system provides an interactive reporting interface but does not contain an integrated 3D graphic display [4]. As EDMmodelChecker, it uses EXPRESS language, so it requires trained software specialists for its management [7].

Unfortunately, DesignCheck has not been implemented for commercial use, and there are no indicators of any future development plans [14].

In 2009, the US Department of Energy developed ResCheck and ComCheck to verify building designs against energy standards, such as IECC and ASHRAE Standards 90.1, in Residential and Commercial projects, respectively [20]. These compliance checks solutions had predefined criteria based on those standards (that were hard-coded into them), where any amendments to the standards had to be managed by the responsible governmental department [14].

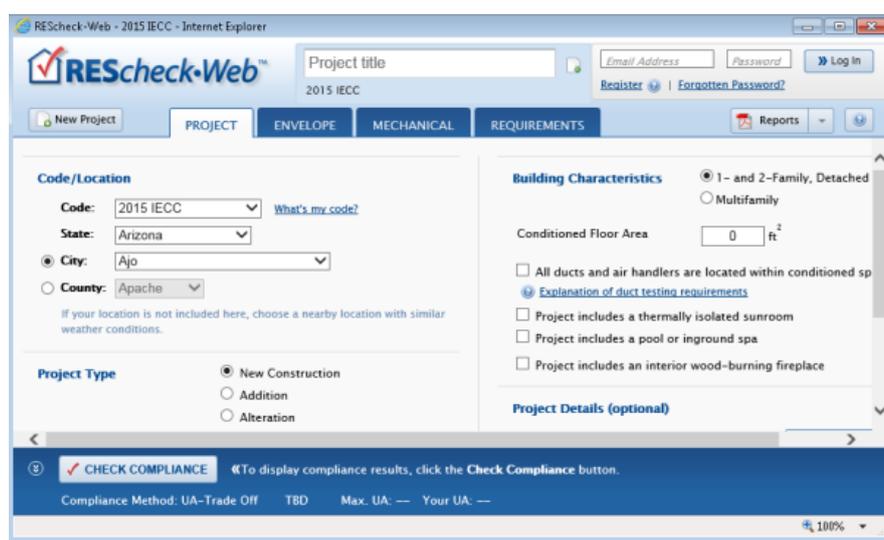


Figure 13: ResCheck-Web interface [20].

The Building Environment Rule and Analysis (BERA) was developed in 2011 by Lee as a domain-specific programming language designed for Code Compliance Checking in building models. It gives users greater flexibility than black-box methods and allows for encoding complex rules [5]. Nevertheless, as its application mainly focuses on evaluating a building's circulation and special programs, it lacks a logical base, limiting its versatility for more intricate structures [5], [7]. Some authors have labelled it as promising, but it does not adequately address the gap between requirements and available data of building models [3], [21]. Regardless of all this, BERA is defined to serve as a relevant foundation for future advancements in the area [7].

Afterwards, in 2015, a joint effort project from various UK institutions [22] called RegBIM was developed [11]. The RegBIM project combined the IFC data model with the expertise and knowledge of regulation and the BIM domain [9] to automate compliance checks. This methodology included various components such as marking regulatory documents using RASE (Requirement, Applies, Select, Exception), mapping regulations to IFC data

models, performing compliance checks using a rule engine or semantic model, and presenting results through a user-friendly interface. This allowed experts and professionals to maintain regulatory systems without coding or software knowledge [11].

Other solutions for ACCC Systems include Statsbygg, International Code Council (ICC) & SMARTcodes, General Services Administration (GSA) & Design Assessment Tool (DAT), Korean Research Studies, LicA, ACCBEP, GTPPM, Natural Language Processing (NLP), SMARTreview APR, UpCodes AI, ACABIM, and other Artificial Intelligence (AI) approaches [1], [2], among others.

Notably, since the '80s, the attempts to develop automated (or semi-automated) solutions have exponentially increased (Figure 14). This showcases the fast and constant evolution of digital and information technologies. Table 1 illustrates this by compiling the previously mentioned efforts and their characteristics.

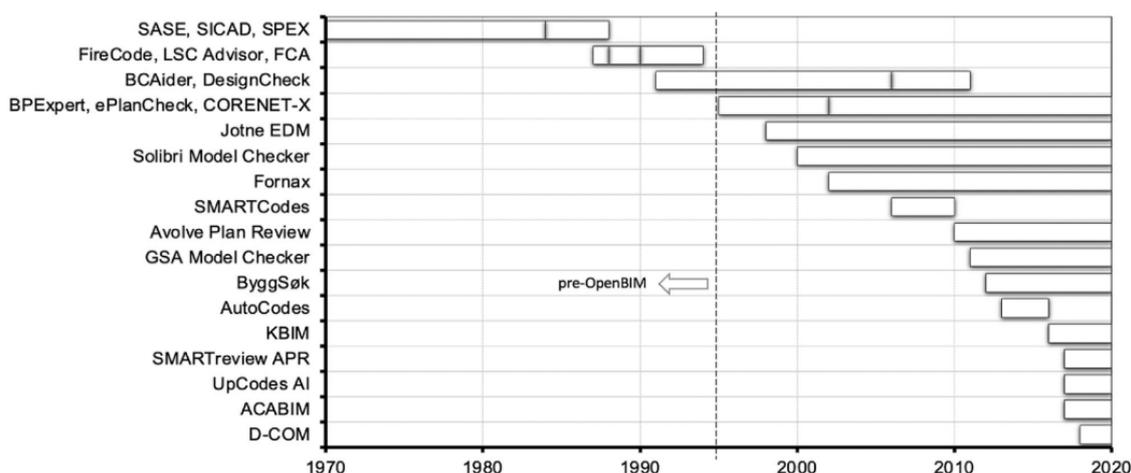


Figure 14: Timeline of major ACC systems [1].

Also, in Table 1, a significant change in the interface of these tools can be observed. Logically, the initial approaches were more analogical and text or code-based, while the newer solutions provide more advanced interfaces with 3D graphical representations and collaborative features. This showcases how technological advancements allow more capable solutions and grant the tool usability.

Tool Name	Year	Country	Company/ Organization	Level of Automation	Type of Interface	Additional Characteristic
AISC Specification	1969	USA	American Institute of Steel Construction	Manual (Machine-readable)	Decision Tables	White-box.
SICAD	1984	USA	National Bureau of Standards	Semi-automated	Text-based	Methodology for CAD standards integration and efficiency.
SASE	1987	USA	National Bureau of Standards	Semi-automated	Text-based	Methodology for Standards Representation
CORENET & e-PlanCheck	1995-2000	Singapore	Ministry of National Development	Automated	Graphical user interface	Hard-coded. IFC-compatible
Express Data Manager	1998	Norway	Jotne EPN Technologies	Semi-automated	Graphical Interface based on EXPRESS	White-box. Coding expertise is necessary.
Solibri	2000	Finland	Solibri, Inc.	Automated and Semi-automated	Graphical user interface	Hard-coded. IFC-compatible. Customisable checks
FORNAX	2002	Singapore	novaCITYNETS Pte. Ltd.	Semi-Automated	Graphical user interface	Hard-coded. IFC-compatible
DesignCheck	2006	Australia	Cooperative Research Centre (CRC) for Construction Innovation	Automated	Code-based	White-box solution. IFC is compatible. Coding expertise is necessary.
ResCheck and ComCheck	2009	USA	U.S. Department of Energy	Semi-automated	Graphical user interface	Hard-coded
BERA	2011	USA	College of Architecture, Georgia Institute of Technology.	-	Programming Language	-
RegBIM	2015	UK	Technology Strategy Board	Semi-automated	Graphical user interface	Hard-coded

Table 1: State of the Art of Computer-Assisted Code Compliance Checks

2.3 Automated Code Compliance Checking

Automated Code Compliance Check (ACCC) is defined by Eastman (2009) as the assessment of a project by looking at how things are organised, how they interact with each other, or their specific characteristics through the application of rules, conditions, or constraints and receiving an output regarding its result [4].

Despite being identified before the arrival of BIM in the industry, ACCC comes into play to improve efficiency in manual and computer-assisted processes. Solihin and Eastman (2015) sustain that the most relevant goal of ACCC is to create complete automated systems ultimately; that way, professionals can focus on more relevant processes such as security, sustainability, and environmental performance [10]. Additionally, Amor et al. (2021) also developed the idea that the automation of quantitative and tedious compliance checks may be helpful for professionals to concentrate their efforts on more qualitative tasks, such as performance-based design checks [1]. Furthermore, Ding (2006) states that ACCC is crucial for identifying non-compliance as early as possible, which can avoid costly modifications in the late stages [6].

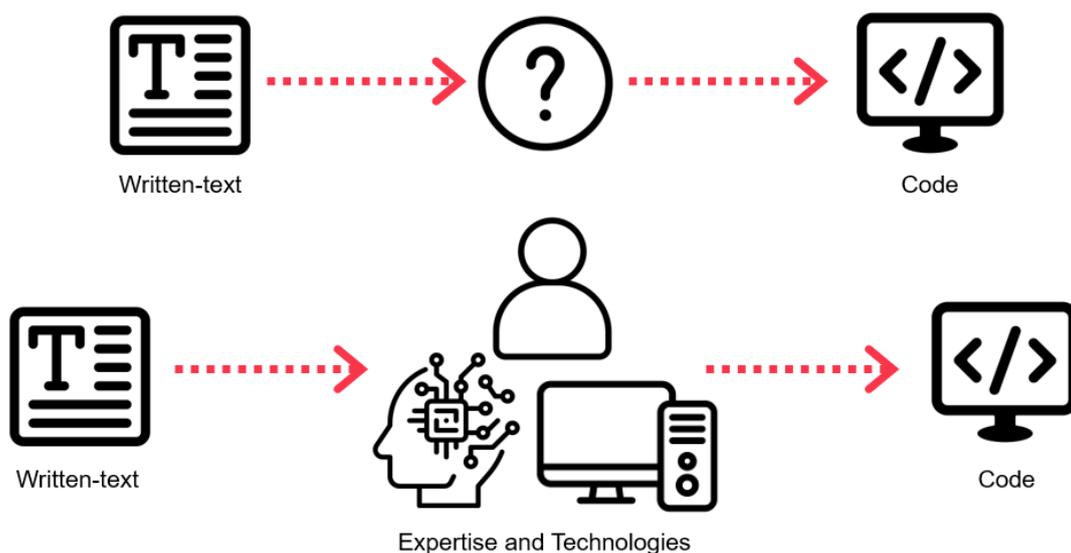


Figure 15: Automation Challenges for Code Compliance Check

Nevertheless, there are various challenges to address in achieving this goal, mainly because of the syntactic differences between Written-text Rules and Machine-Readable Code necessary for Automation (Figure 15). The human mind can easily understand and associate the text with building elements, but machines must be for that task. Additionally, another considerable challenge relates to the complexity that fully

automated processes mean, especially in the context of BIM models that, only by themselves, are already large, highly complex and intricate sources of information.

Solihin (2016) also mentions that by fully automating this code-checking procedure, those processes' quality, innovation, and efficiency will improve [3]. In addition, this enhanced characteristic could also be extrapolated to other design and construction processes.

Other authors sustain that achieving fully automated CCC is unlikely; instead, efforts should be put into developing guiding tools (Beach, 2020). This is why Hybrid Validation is an optimal solution for approaching the Automation of Code Compliance Checking, where human oversight by experts and qualified stakeholders could complement the process if needed.

Solihin and Eastman (2015) emphasise the importance of systematising rules to ensure effective and efficient ACCC. Their work focuses explicitly on classifying different rules for creating a future framework that links the process obstacles and technological requirements needed for rule automation.

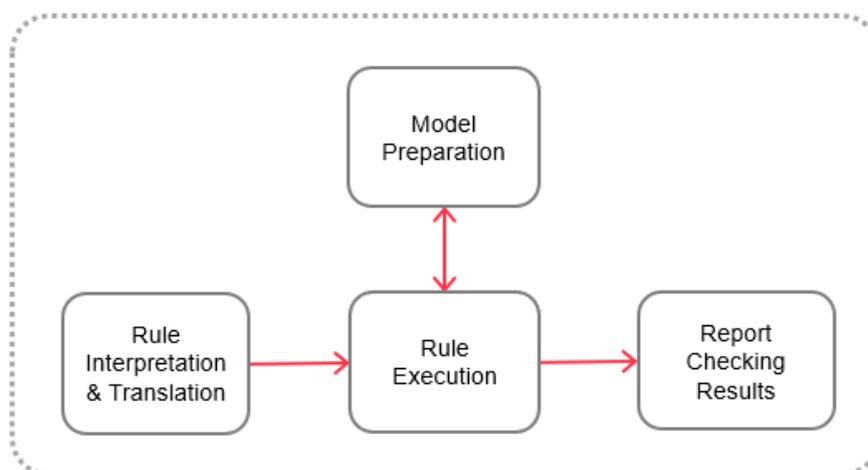


Figure 16: Common structure of an ACCC process, inspired by [4]

Regarding the Automation of the process itself, Eastman (2009) describes four stages for carrying it out: Building Model Preparation, Rule Interpretation and translation, Rule Execution, and Report Checking results [4], as seen in Figure 16. This Theoretical Review will treat Interpretation and Translation as distinct processes, each with different complexities. While Rule Interpretation will be boarded as the understanding and analysis of rules, Rule Translation involves converting them into machine-readable concepts and developing the actual rule-checking tool. Despite their differences, these two processes are closely intertwined.

2.3.1 Model Preparation

The standardisation of the information is a crucial requirement within any BIM Model, as it involves consistency, collaboration, quality, and efficiency. It helps reduce errors and improves design and construction processes [3]. Additionally, it is a vital previous step to achieving the machine-readability desired for ACCC. Nevertheless, it can also mean an enormous amount of work as it can englobe a vast number of topics, from the quality of the model's metadata to how its elements were modelled [1].

Model preparation usually considers standards and recommendations on good practices in BIM models for machines to analyse them consistently and make them machine-interpretable [4], [23]. This involves ensuring the BIM models are appropriately formatted and structured before performing automated compliance checks. It may also include ensuring the information's quality, sufficiency, correctness, and consistency by removing duplicate or erroneous data, validating data integrity, and standardising parameters and properties[1]. The quality of the model preparation stage is crucial to the accuracy of the compliance checks.

For Amor and Dimyadi (2021), this previous step for automated compliance checking is necessary to ensure the accessibility of data [1]. According to the author, it is a rather challenging task, especially in the context of OpenBIM, where IFC data sets can be too large and complex to cope with. Model Preparation is also directly related to specifying the information needed, deriving data for a specific regulation check, and extracting groups of data from a model for more efficient checking [4], [24] Information Delivery Manual (IDM) and Model View Definitions (MVD) come into play for minimising complications that unused information may bring [1].

Additionally, Information Requirements are essential in this stage, as they define the Model and its elements in detail. Various authors have referred to the importance of defining or identifying the Level of Development within a model in the context of the Code Compliance Check. It is consensual that the Level of Development (and, more precisely, the Level of Information Need) must be enough for specific rules to be checked and calculated [10], [25].

It is crucial to remark that throughout the phases of a Project, the LOD of elements progresses at different rates [10] and that some bylaws require different Levels of Development on the model's elements to be checked [26]. The authors highlight this by indicating that for some checks in the early-stage design process, the LOD could be 100 or 200, while in the design development phases, it could be 300 or 350.

This might suggest the need to define separate Code Compliance Checking (CCC) tools for specific phases and categorise them accordingly in relation to the specificity of the model. Therefore, there is a need for a structured and well-defined approach to LOD in AEC projects to ensure compliance with legal requirements while also meeting project requirements.

2.3.2 Rule Interpretation

This stage needs a thorough understanding of the specific rules, standards, and regulations. This is crucial for translating them from natural written language into a machine-readable format [4], [10], [24], which can be challenging due to the complexity and variability of regulations and standards. Nevertheless, it should be considered a significant one-time effort towards automation (as CORENET did).

The process involves interpreting the meaning and semantics of rules, including their intent, base, hidden assumptions, presumed general knowledge of the subjects, and relations or dependencies with other rules. Usually, expert knowledge is required to perform this interpretation, which is crucial in clarifying and checking concepts or principles. It may also involve defining parameters or thresholds for the rules to ensure that the check is sensitive to specific requirements or circumstances.

Furthermore, authors have had different approaches towards this step. Most of them tend to categorise rules as a way of understanding them, and, as there are numerous codes and rules to consider [9], there are also many ways to classify them. The main trends go towards the Taxonomy and Ontology of regulations (Figure 17), even though other approaches have been made, like using pseudo-code [4], [8], [27] or, more lately, Artificial Intelligence [1]. None of these are exclusionary and could be used in a combined way.

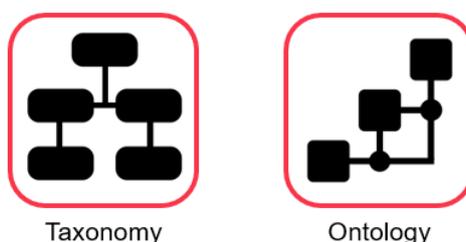


Figure 17: Taxonomy and Ontology

Both taxonomies and ontologies could approach Rule Interpretation according to regulations' context, origin, application stage, scale, element type (if it is a plot, building,

construction component, etc.), complexity (how many parameters they involve, if there is a need to create new, if they need additional elements to be checked, etc.), geometrical or conceptual properties, difficulty in translating into digital or automated processes, specificity, among others. Thus, a detailed analysis of the rules is vital to define the correct approach for machine readability in each specific case.

Additionally, Amor and Dimyadi (2021) describe two general rule types. On the one hand, prescriptive regulations provide specific rules and guidelines to follow and tend to be more straightforward to apply. On the other hand, performance-based codes focus on achieving desired outcomes without setting specific methods and are usually more flexible in checking [1].

2.3.2.1 Taxonomy of Rules

Taxonomy is a hierarchical system of classification for entities, and in the context of Rule Interpretation, it allows for categorising different codes and regulations. Prescriptive regulations can be classified within the taxonomy framework based on their specific criteria and standards for various building elements and systems. While performance-based codes can be categorised within the taxonomy according to the performance objectives they aim to achieve, their complexity, or the criteria they define.

A simple Taxonomy may Classify Rules regarding their main function or Type, as seen in Figure 18.

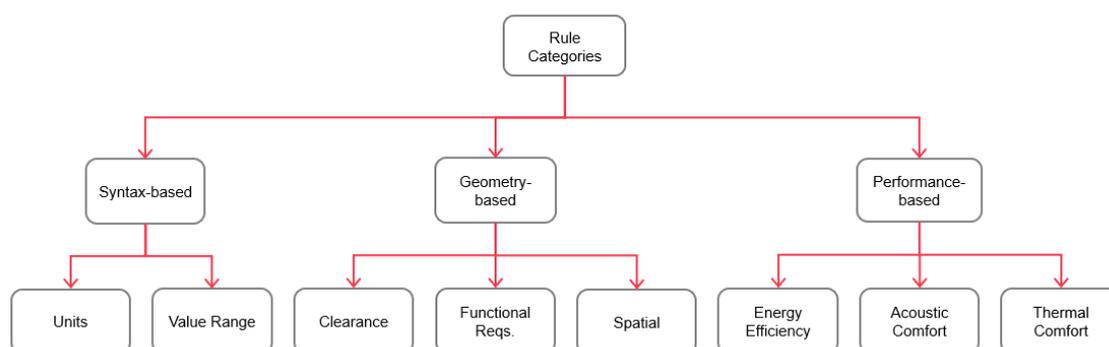


Figure 18: Rule Taxonomy Example

Solihin & Eastman (2015) define four general classes for rules according to the complexity needed for their translation and the amount, accessibility, and correctness of data: **Class 1**, with laws that require a single or small number of explicit data, and the extraction is direct (it is also closely related to the improvement of the model's quality and predictability, as in "Model Preparation"); **Class 2**, describing rules that require simple derived attribute values, and analysing relationships between entities; **Class 3**

having rules that require extended data structure, like a Solids Modeler or Graphs; and **Class 4** with regulations that require proof of solution on how a model demonstrate compliance and possibly correcting it if not [10]. Additionally, Solihin (2016) states that for more complex regulations, there is a need for more complex level semantics as well [3].

Moreover, for the authors, the classifications of rules do not suggest a correlation in their importance. Instead, they all serve a valuable purpose for automated rule checking, but each type has a different scope and aim. Additionally, they suggest that identifying patterns within and between rules is key for a more efficient process.

Additionally, they propose seven categories for rules in the AEC industry [10]:

1. Well-formedness checks: Ensure building models meet syntactic standards and required conditions.
2. Building regulatory code checking: Verify compliance with established building codes and regulations.
3. Specific client requirements: Address unique client design requirements (e.g., hospitals, courthouses).
4. Constructability and contractor requirements: Rules for temporary elements and conditions during construction.
5. Safety and corrective actions: Automate identification of hazards for workers and maintenance staff.
6. Warranty approvals: Check post-construction models for issues affecting warranties and maintenance costs.
7. BIM data completeness for facilities management handover: Ensure BIM data is complete for handover to facilities management, considering downstream requirements.

2.3.2.2 Ontology of Rules

Ontology-based interpretation of rules is a more complex process, consisting of organising and categorising the corresponding rules and relationships between or within them. This method may be more straightforward regarding embracing complex rules translation, but it also requires significant understanding, expertise, and effort [10]. Prescriptive codes can be classified and organised within an ontology by representing the connections and relationships among different requirements and elements. Similarly, performance-based codes can be represented by categorising the rule's performance goals, measurements, and the relations between different criteria.

Graphically, Ontologies may be represented in various ways, too. For instance, one representation possibility can be done by using RDF (Reference Description Framework). Here, resources described as unique identifiers are related to others in triples [28], [29]. In the example in Figure 19, various relations between two elements (Rooms and Windows) can be illustrated using this model.

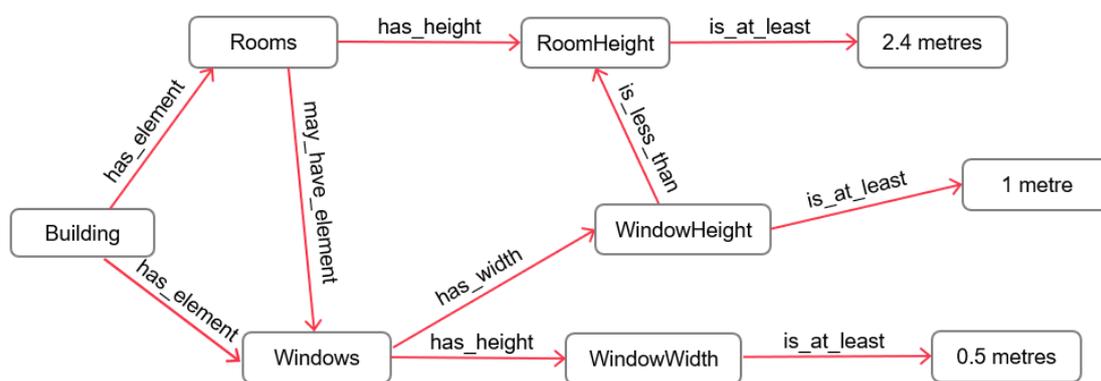


Figure 19: Rule Ontology Example using RDF.

A significant issue with the ontology approach is to accurately recognise the general concept behind each specific rule to understand its general aim and have a smoother development of code-checking. According to Solihin and Eastman (2015), this may be achieved by fragmenting and questioning each concept and definition of the bylaw and asking further questions regarding their implications and ranges of applicability [10]. This logic-based approach allows the user to comprehend the contents and meaning of each regulation entirely.

One approach is described by Eastman (2009) by first identifying the context where a specific rule applies and then its properties. This way, the relationships between entities and their parameters can be made when analysing bylaws [4]. The author also specifies the importance of Omniclass and the International Framework for Dictionaries (IFD) as a base for Rule Interpretation.

Another approximation, made by Hjelseth (2011), consists of using the RASE mark-up Methodology as an ontology-based way of interpreting rules. The idea behind it is to semantically break down rules by classifying them according to their content within four operators: Requirements, Applicability, Selection, and Exception [30]. Later, it is easier to translate them into code using these simplified statements

2.3.2.3 Pseudo-Coding

Pseudocode is a technique for creating and presenting algorithms like programming code without following specific syntax. It uses human-readable, informal language to portray algorithms at a high level [31]. Using natural language, simple math, and programming elements like conditionals and loops gives a broad depiction of problem-solving steps. Programmers use it to discuss and refine algorithms before coding, concentrating on logic and flow while avoiding language-related limitations.

As in coding, it generally serves as a bridge between the problem-solving and implantation phases; in the context of ACCC, it may serve a similar purpose. This means that pseudo-coding could connect the phase of Rule Interpretation with Rule Translation, where domain-specific knowledge, or rules, can be transformed into functions and rules that can be later easily transformed into proper coding [4].

```
Regulation: Maximum Building Height

»Buildings within the city are subject to a maximum height
restriction of 12 units.«

CONSTANT MAX_HEIGHT = 12

PROCEDURE CheckBuildingHeight(height):
  IF height <= MAX_HEIGHT:
    PRINT "Building meets height regulation."
  ELSE:
    PRINT "Building exceeds height regulation. Maximum height:",
MAX_HEIGHT, "Actual height:", height

BUILDING building1
building1.height = 8
CheckBuildingHeight(building1.height)

BUILDING building2
building2.height = 15
CheckBuildingHeight(building2.height)
```

Figure 20: Pseudo-coded Regulation Example

2.3.2.4 Artificial Intelligence

Additionally, Amor et al. (2021) discuss the multiple benefits of AI models in Automated Compliance Checks. The authors state that exploring Natural Language Processing

(NLP) and Machine Learning (ML) techniques can extract normative knowledge from written codes and standards. Furthermore, they claim that AI-based ontologies enable efficient mapping of objects and attributes and rule applications for compliance-checking purposes [1]. This implies that AI may fill the gap between legislation and computers by replacing the domain expertise in interpreting bylaws and regulations and transforming them into machine-readable formats with no additional effort (Figure 21).

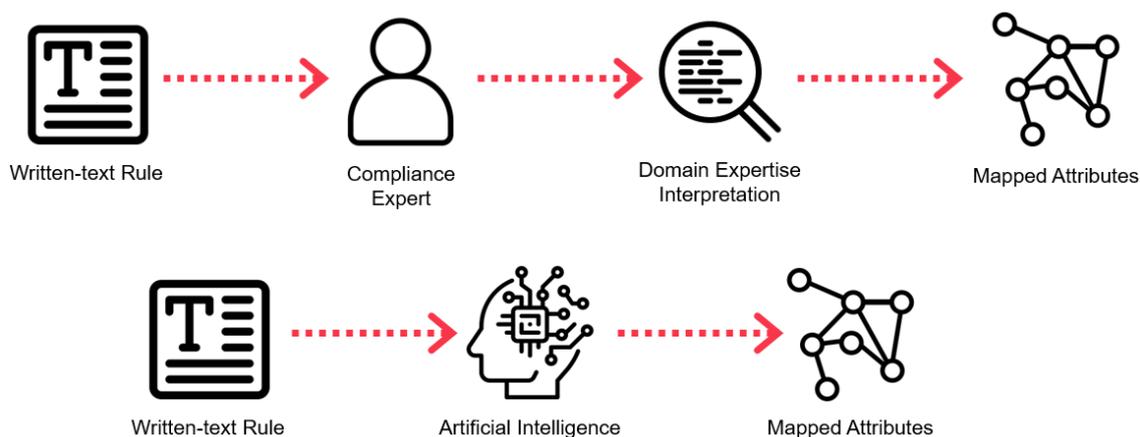


Figure 21: Artificial Intelligence in ACCC

2.3.3 Rule Translation

As Rule Translation happens after the Interpretation and Classification of rules, this step may also have different requirements and levels of complexity. The design of automated algorithms can be highly intricate depending on the nature of the bylaw being checked. There are various approaches regarding Translation, as it highly depends on how the Interpretation process is done.

Eastman (2009) defines two ways to implement ACCC. One refers to using computer language-encoded rules, which require expertise and may not be easily spread. The other approach uses parametric tables, which are defined to be easier to understand and execute but a more limited method [4].

Solihin (2016) developed a SQL-based Domain Specific Language (DSL) for Code Compliance Checks, named BIM Rule Language (BMRL), that is integrated with the BIM Model's data and elements. The author used Conceptual Graphs to achieve a better breakdown, illustration, and comprehension of the rules and their relationships. This reduction into basic concepts helps to remove ambiguity and facilitates the approach to the programming of the automated tool [3].

Additionally, there is one last fundamental issue to address. It involves balancing the need for accurate data and efficient processing. On the one hand, relying on user input can ensure the accuracy of the data, but it can also be time-consuming and lead to errors. On the other hand, depending on computer logic can be faster and more efficient, but it requires accurate models and assumptions and may not be as reliable as user input [10]. This is a critical situation regarding Rule Translation, as the outcome of an Automated solution depends highly on the human-based work behind it.

2.3.4 Rule Execution

Automated code compliance checks in BIM models involve executing rules using software tools to identify non-compliant elements. This process requires defining regulatory and design regulations run within the BIM model. Then, the tool or software evaluates the model against the rules and generates a report detailing any violations. This step can be time-consuming and computationally intensive, especially for large or complex models. The software may also perform automated corrections or suggest changes to bring the BIM model into compliance.

Eastman (2009) specifies that if the rules have been interpreted correctly and consistently with machine-readable functions compatible with building model elements, the Rule Execution process is reasonably uncomplicated [4].

2.3.5 Report Checking Results

Report Checking Results involves reviewing the results of the automated compliance check to identify any issues that need to be addressed. This may include reviewing detailed reports that identify the specific non-compliant elements and provide guidance on managing them. It could also contain graphical information for a more precise or thorough understanding of an issue.

Once compliance is assured, the report could be represented officially or formally, which could be done as a table or type of graphics [4]. This may include referencing the original textual rule or describing how specific regulations are not complied with.

The report may also track compliance over time and identify areas where the model can be improved. Back in 2009, Eastman predicted the applicability of error reporting in online platforms and even within the same authoring tool [4], as nowadays software like Revit and Navisworks do through BIM 360, or Solibri Model Checker can achieve through online servers like BIMcollab or others.

It is essential to mention that even though Code Checking could be fully automated, the legal need for a human expert to review them inevitably makes this process a Hybrid one.

Moreover, Amor and Dimyadi (2021) state that due to the complexity that ACCC represent and the highly different approaches that hence exist towards its implementation, it is only appropriate to assume that currently, there is not a complete understanding of what specific characteristic are needed to achieve full automation [1].

Nevertheless, Beach (2020) identifies a series of issues regarding the current state of Automated Code Compliance checks [4], including the model quality, the data extraction is too complex, the need for expertise in a programming language that is not applicable in every context, the lack of international standards for rules, complexity in the interpretation of rules (semantics v/s pragmatics), lack on flexibility (black-box solutions), lack of query systems with the models, and that IFC models are highly interoperable and standardised, but rigid and need to be exported.

2.4 Visual Programming Language

The rigidity of existing Compliance Check tools is exposed by their inability to adapt to regulation changes, different contexts, or error detection. This is described as a “Black box” process [32], representing a weakness regarding the digital world's constant changes and fluctuating nature.

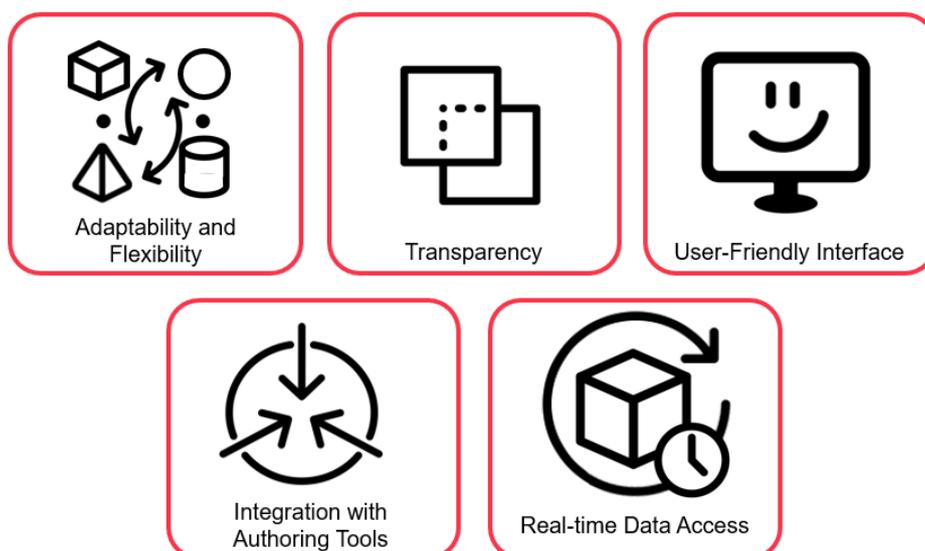


Figure 22: Advantages of VPL

Additionally, various studies, articles, prototypes, and practical applications have been made in the industry regarding Automated Code Compliance Checks. Still, they are always focused explicitly on specific projects, contexts, or local regulatory systems [2], [25], [26], [33]–[35]. This generates isolation in their explorations, making integrating them into other circumstances impossible. Therefore, there is an underlying need for compliance check tools to be adaptable and flexible enough to be used in various contexts or countries.

Also, VPL is an excellent solution for one of the main problems ACCC has had in the identified past experiences: the necessity for expert knowledge in writing rules, which is inconvenient for stakeholders [4], [12]. VPLs have the outstanding advantage of being user-friendly by using a graphical interface in a way that project developers can handle and learn without much training or previous knowledge of programming and coding.

Preidel and Borrman (2015) identify Visual Programming Languages (VPL) as the most favourable solution for bridging the gap between the complexity of incorporating automation in rule-checking and the transparency and flexibility that the process needs. Additionally, the authors claim that a mandatory requirement for automated compliance checks to work is the communication between the user and the machine, which VPL highly provides [5].

Furthermore, Visual Program Language has the immense advantage of live updates and access to the projects' geometry and metadata. There is a need for integrating BIM data with the query system that external solutions do not possess [3]. This provides a faster and more efficient workflow, as the information within a project can be directly managed and interpreted without the need for exporting processes or external programming tools or software that connects the BIM data with the regulations.

Finally, VPL tools generally have the benefit of not needing additional software licences, as the most commercially available solutions are already integrated into the authoring tools or are instead plugins for them (e.g., Dynamo for Revit and Grasshopper for ArchiCAD, respectively).

3. METHODOLOGY

3.1 Research Design and Approach

This dissertation employs a systematic research design and approach to thoroughly address automating compliance checks in Building Information Modelling (BIM). Its methodology includes several key processes that build upon each other to provide a holistic approach to addressing the problem.

The adopted methodology will be mixed methods research consisting of an initial Investigation Phase (Problem Definition, Problem Analysis, Literature Survey, and Data Collection), a Frameworking Phase (Framework Definition and Framework Development), and an Application Phase (Case Study Selection, Solution Application, and Problem Resolution).

The first Phase, already carried out in the previous Chapters, mainly consisted of researching to set a theoretical background towards the issue and defining the main guidelines. The subsequent Phases will include the design of the Framework for tackling the problem (Frameworking Phase) alongside an Applied Research approach for the Case Study application (Application Phase).

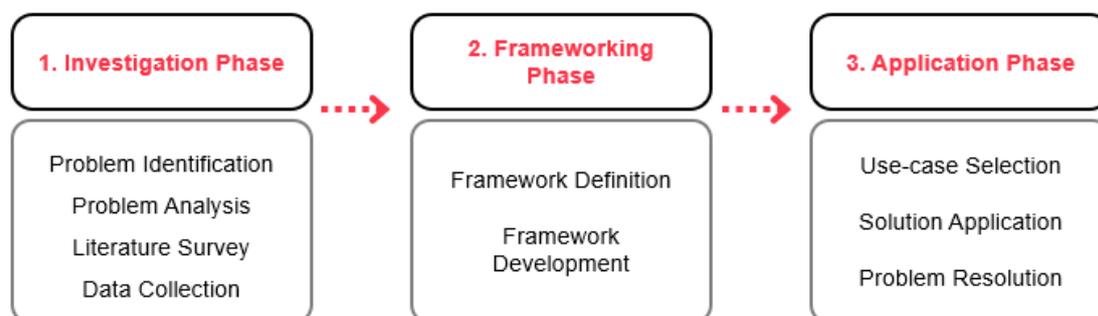


Figure 23: Methodology Phases

In this research phase, included in Chapter 1, the Problem Identification process was first carried out. This involved identifying and articulating the challenges and issues associated with ACCC and setting the foundation for further analysis and research. This exploration was based on industry practices, project requirements, and compliance regulations to clearly understand the main points and limitations of the existing process of compliance checks.

In Chapter 2 -Through a Literature Survey- existing research, best practices, frameworks, and approaches in ACCC and compliance checks were examined. This

review provided valuable insights into the industry's current state, identified gaps in knowledge, and set some bases for the Problem Analysis. The process was transversal throughout other stages, as it was constantly revisited for deepening knowledge, clarifying concepts, and including findings as an iterative process. This step was vital for defining Data Collection and Case Study Selection processes.

Additionally, based on the Problem Identification, Chapters 1 and 2 included the Problem Analysis. In Chapter 1, some initial explorations were done, whereas in Chapter 2, it was deepened into the interactions between design complexities, regulatory requirements, and the efficiency and accuracy of compliance assessments. This phase examined the existing manual compliance check processes' underlying causes, implications, and limitations.

The next step, Data Collection, consisted of gathering relevant and accurate information from the Literature Survey necessary to develop the automated compliance check solutions. Regulatory requirements, industry standards, academic research, design guidelines, and other pertinent data sources were found and analysed for use in this work's application section.

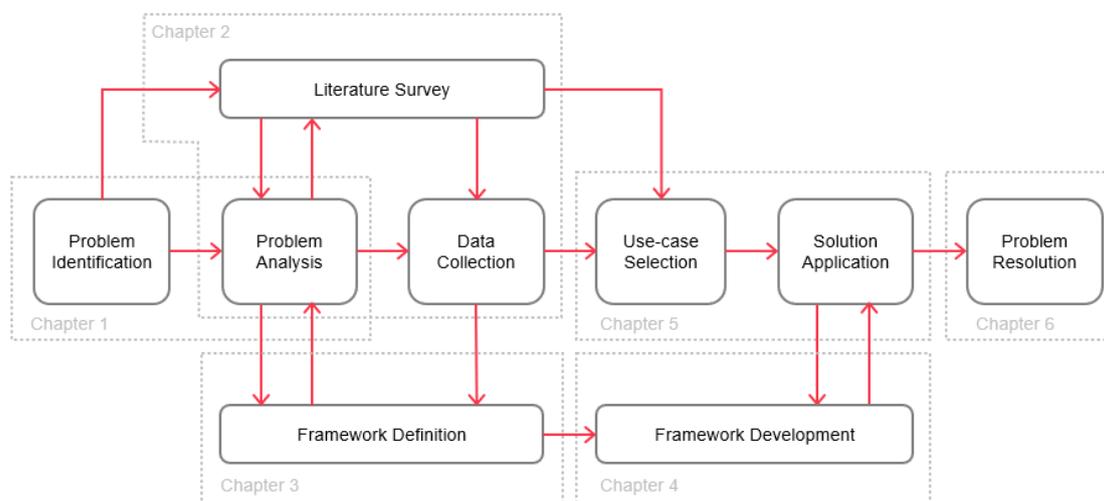


Figure 24: Methodology Workflow

After the initial Phase, the Frameworking Phase will be conducted. By extracting the insights from the Problem Analysis and Data Collection, the Framework Definition will be outlined thoroughly in the current Chapter. This framework will provide a conceptual structure for addressing the challenges of ACCC and guide the subsequent methodology development. It will consist of a mixed research approach, including quantitative and qualitative methods.

Later, the methods outlined in the Framework Definition will be executed in the Framework Development. This step involves transforming the conceptual framework into practical and functional components that can be applied to the selected use cases. Here, processes, tools, algorithms, and methodologies will be designed and created to automate the code compliance check process. This will enable the integration of qualitative and quantitative methods.

The Framework Development process will also include rigorous testing, refinement, and validation of the developed tools and methods. This will ensure their reliability, accuracy, and compatibility with the selected use cases. Iterative testing cycles, feedback incorporation, and improvement will be conducted to enhance the effectiveness and efficiency of the compliance check tools.

Then, by carefully selecting representative use cases, the study aims to capture different scenarios, regulatory contexts, and design challenges. The Case-Study Selection will be based on relevant criteria, such as project complexity, regulatory requirements, or specific industry sectors. This enables a diverse range of use cases to be examined, providing a robust evaluation of the proposed methodology.

After this, the Solution Application will be carried out. Here, the developed Framework will be implemented and executed in the selected use cases for more than one rule and project. The tools will be designed specifically for each case using the input extracted from the Framework. This step aims to demonstrate the effectiveness and feasibility of the automated compliance check solution.

Lastly, the methodology will include a Problem Resolution step, where the results obtained from the Solution Application will be carefully analysed, and any identified issues or challenges will be addressed. This process aims to refine and improve the automated compliance check solution and discuss possible feedback or improvements.

3.2 Framework Definition Methodology

This section aims to define a general framework that can broadly englobe the automation of as many bylaws as possible, independent of their context, origin, and characteristics. It will incorporate qualitative (Rule Interpretation, Analysis, and Classification) and quantitative (Model Standardization and Automated Graphs Development) methods.

The Framework will be structured around the main concepts of the five processes described in the Theoretical Review: Model Preparation, Rule Interpretation, Rule Translation, Rule Execution, and Report Checking Results. The idea will be to design a

workflow for each to be executed in any possible context and during all phases of the Design process. Additional steps or sub-processes may be added for both quantitative and qualitative processes. Later, the applied research will consummate and demonstrate the practical use of the workflow.

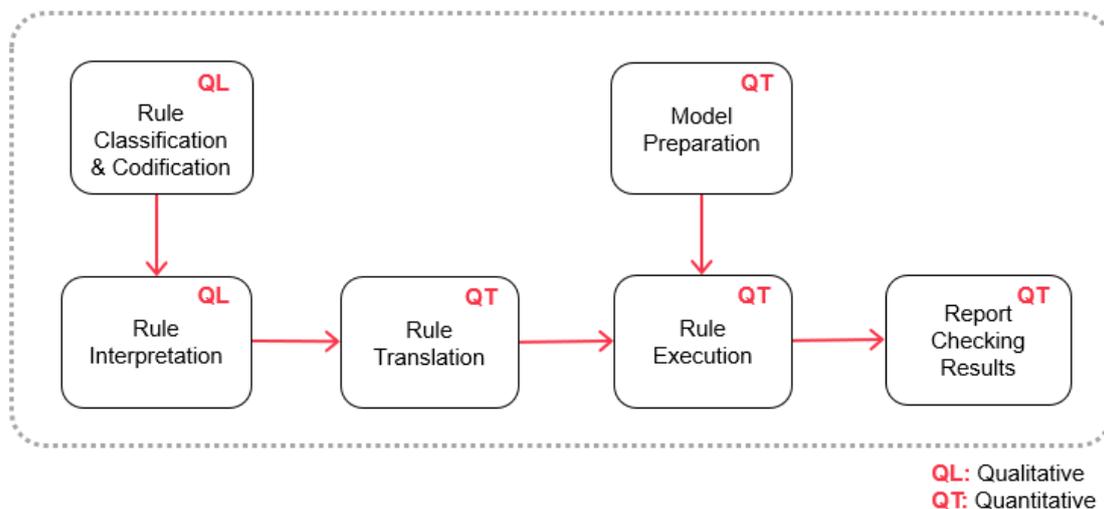


Figure 25: Framework Functions and Methods Steps

Different rule-checking tools will be developed to represent the methodology's versatility. This approach will allow showcasing multiple rule types as an example of the applied methods in action and representing different stages of the design process to cover a broad range of complexity levels.

Moreover, as mentioned in the Theoretical Review, Visual Programming Languages offer a favourable trade-off between transparency (being a white-box solution) and a user-friendly interface (that professionals could adapt, modify, or twitch when needed). This is why the ACCC tools will be developed using this language.

Additionally, this framework will be designed as an iterative process, as reviewing previous steps is crucial for identifying errors or malfunctions, especially in the developed automated codes (or VPL graphs). Also, constant improvement is a critical aspect of programming. The output and experiences gained in later steps of one rule's automation checking development could be valuable feedback for initial or even other regulations.

Due to the collaborative nature of this work between the BIM A+ master's organisation and Jereb in Budja Arhitekti, the implementation of the framework will be aligned with the latter's workflows and approaches regarding the use of BIM technologies. This is why

the development and application of the study will be made using Revit 2021.1 as the authoring tool and Dynamo 2.6.1 as the automation software.



Figure 26: Software and tools used

Finally, depending on each type of rule, a different Report will be designed in the VPL graph to illustrate the result of every kind of code, depending on their characteristics.

3.2.1 Qualitative Methods

3.2.1.1 Rule Codification

To understand how rules work, they will be converted into a simple coding system that depicts these characteristics in a single-string format, structured from general to specific. This classification and coding system will enable users to face the following steps (Classification, Interpretation, Translation, Execution, and Reporting Check Results) quickly and to know what to expect, what could happen, or what could go wrong with each of them. Additionally, it will provide a universal way of representing codes for comparison, consistency, and simplification purposes.

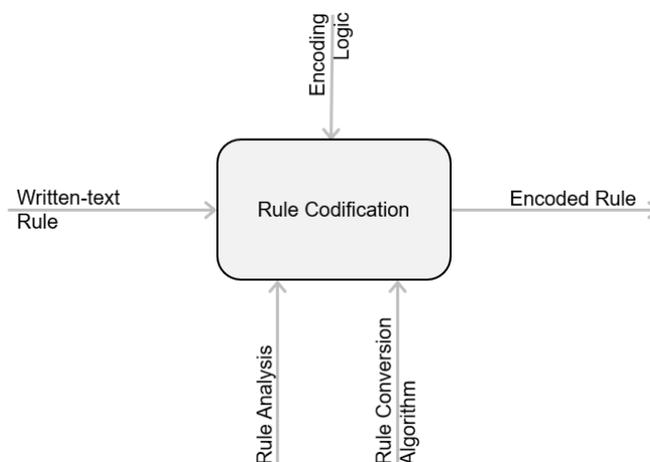


Figure 27: Rule Codification Step's IDEF0

3.2.1.2 Rule Classification

Then, to have a general and structured idea of how rules should be processed, it is adequate to analyse them further to generate a classification system for rules. This previous step is crucial for later Interpreting and Translating them correctly. Here, rules will be described and analysed using a single table template that displays its main characteristics, such as origin, context, type, geometric rules, spatial requirements, specifications, etc. This categorisation will help us understand the rules' nature and how they can be related to the BIM model. Also, it will help identify common patterns and similarities among regulations.

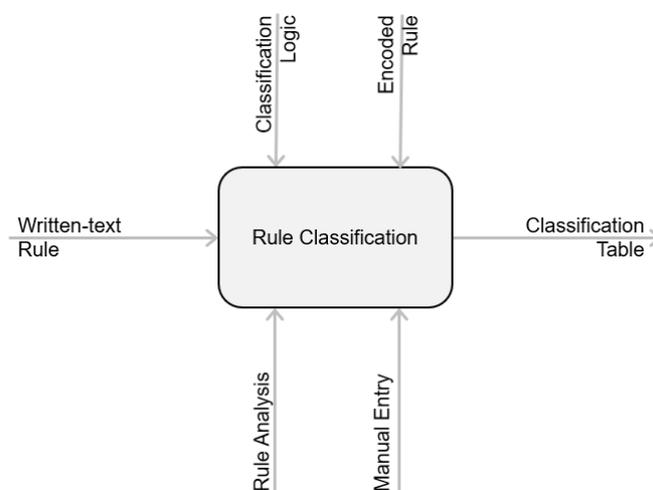


Figure 28: Rule Classification Step's IDEF0

Both the classification tables and the encoding system of rules will be handy in the subsequent steps for better and consistent understanding. Their purpose will be to assist in the later processes when human oversight is needed to edit and review the automated workflows. It is relevant to mention that these tools will only describe and organise the characteristics of the codes and not their semantic content.

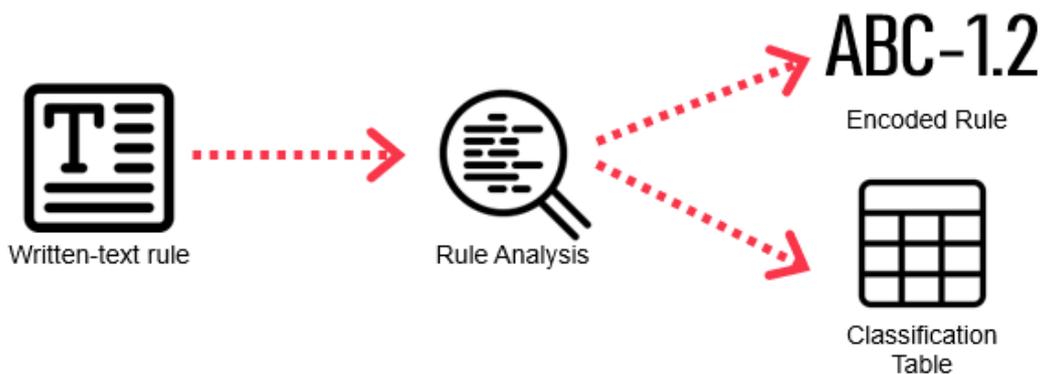


Figure 29: Rule Codification & Classification Concept

3.2.1.3 Rule Interpretation

As previously discussed, the most crucial issue regarding the Interpretation of rules is understanding them on both a semantic and practical level. This is where a combination of different methodologies studied in the Theoretical Review will be used to obtain a fluid workflow for this step: RASE Methodology, Pseudo-coding, and AI language models.

For the actual Interpretation of Rules and transformation into a machine-readable format, using RASE Methodology (which provides an ontology-based approach to rule classification and semantic breakdown) will serve as the general structure for rule interpretation and analysis. This methodology allows for a more detailed understanding of rules by classifying them according to their content within four operators: Requirements, Applicability, Selection, and Exception.

- **Requirements:** This operator focuses on the precise requirements stated in the rules, including specific conditions, constraints, or actions that must be fulfilled. By identifying and categorising the requirements within a rule, a clearer understanding of its intent and objectives can become apparent.
- **Applicability:** The applicability operator considers the context and scope of the rule, determining where and when the rule is intended to be applied. This classification helps assess the rule's relevance and applicability to different projects, design scenarios, phases, and parts of a building/construction/structure, among others.
- **Selection:** The selection operator deals with the choices or options provided within a rule. It involves identifying alternative paths or variations the rule allows and understanding the decision-making process. Potential alternatives or variations can be explored by classifying the Selection aspects of regulations.
- **Exception:** The exception operator is used to identify conditions or circumstances under which the rule may not apply. It involves identifying scenarios where the rule can be exempted or modified based on specific conditions or constraints. By classifying the exceptions, potential deviations from the rule can be recognised and managed appropriately.

As Beach (2020) did, the RASE mark-up methodology will also be colour-coded for each operator. Thus allowing the user to understand the process better and structure ideas

more clearly and graphically. The Requirement will be blue, Application green, yellow for Selection, and red for Exception.

To facilitate the interpretation process into the next step, pseudo coding will be employed. As Pseudo coding involves using a simplified programming-like language to describe the logic and steps involved in interpreting the rules, it will be easier for later translation into a machine-readable format. Breaking down the rules into smaller, more manageable components makes analysing and understanding their requirements easier.

Finally, to achieve this in a more automated and simplified way, Artificial Intelligence language models (such as Bard, Notion, ChatGPT, etc.) will assist in the interpretation process. These AI models can understand and analyse human-written texts and re-write them using given structures or guidelines.

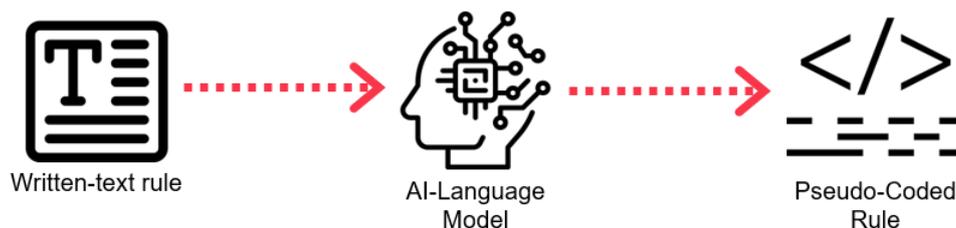


Figure 30: Rule Interpretation Concept

They are valuable tools for interpreting complex rules and providing explanations or clarifications when needed. To achieve this, a defined input or prompt will be designed for every interpretation process, defining both structure and desired output format (e.g., “Transform the following rule into a Pseudo Code, using RASE methodology as its structure.”). Explaining the RASE methodology and colour-coding for each entry will be necessary for consistent results.

Combining the RASE Methodology, pseudo-coding, and AI language models makes the qualitative analysis and classification of rules more efficient, accurate, and streamlined. The RASE Methodology provides a structured framework for understanding the semantic aspects of regulations, while pseudo-coding allows for a logical breakdown of rules into smaller components. AI language models can then assist in interpreting and generating human-readable explanations for these rules, enhancing the overall understanding and analysis.

Adopting this approach will contribute to a more comprehensive and systematic interpretation and classification of rules, enabling a deeper analysis of their

requirements, applicability, selection options, and exceptions. This detailed understanding will be used as a guideline in the subsequent steps in the methodology, such as rule translation and execution, leading to more effective and accurate automated compliance checks in BIM.

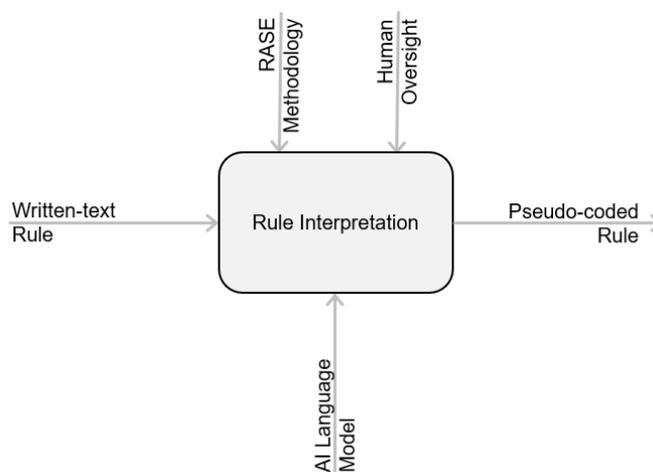


Figure 31: Rule Interpretation Step's IDEF0

While bylaws become more complex, using this combination of methods might not be enough to generate a clear output for the Translation step. This is expected, especially with performance-based and more complex prescriptive regulations. Thus, a constant human assessment throughout the process will be needed to secure accuracy and preciseness.

3.2.2 Quantitative Methods

3.2.2.1 Model Preparation

To automate compliance checks in BIM, it is crucial to establish a standardised model representation that captures the relevant information needed for rule checking. This includes developing a consistent and structured approach to modelling that aligns with the requirements of the rule set. Additionally, these corrections may include highlighting errors or areas that need improvement in the BIM Model.

Data quality assurance and control checks will ensure the BIM Model's quality. This involves verifying the accuracy, completeness, and adherence to specified criteria of the model's data. The BIM Model will also be organised, structured, and formatted according to established conventions or guidelines, ensuring consistency and ease of understanding.

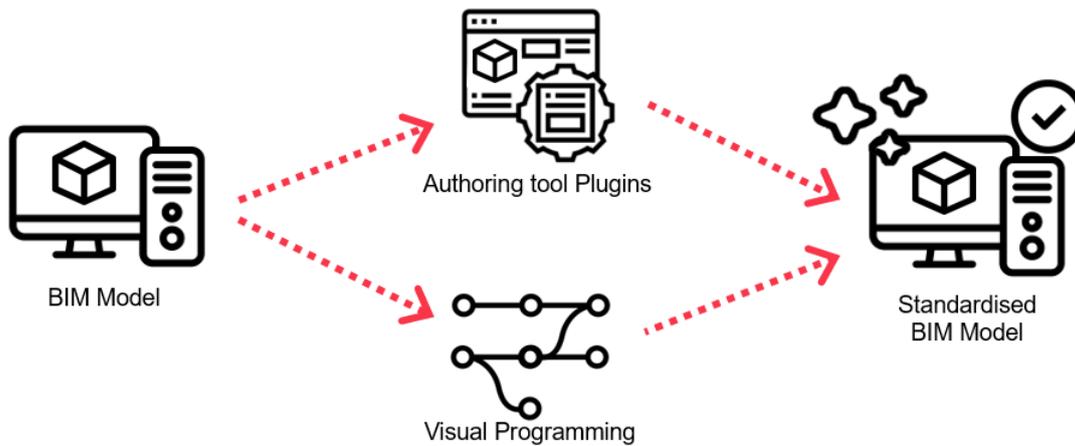


Figure 32: Model Preparation Concept

Additionally, this process should follow specific guidelines and specifications. These guidelines, such as the Level of Information Need, will dictate the required level of detail and accuracy for different elements within the BIM Model. Adhering to industry standards and guidelines is essential to ensure compliance and uniformity.

According to each case, several tools and techniques might be used for this. This includes specialised plugins, add-ons on the authoring device, or even VPL tools developed for general effects or specific requirements.

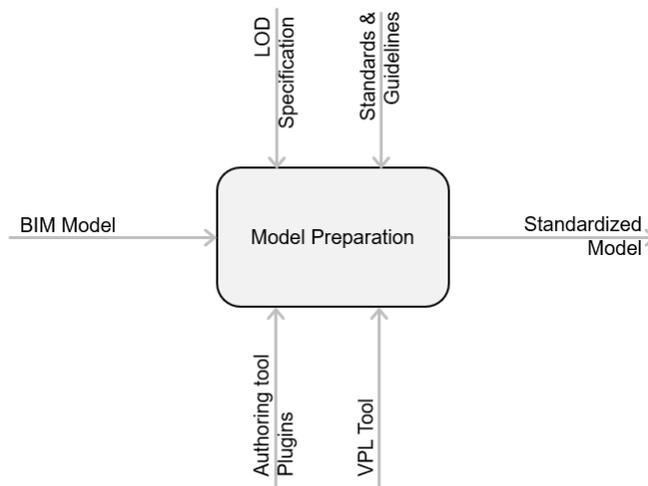


Figure 33: Model Preparation Step's IDEF0

It is crucial to mention that different rules might require various tasks within the model's preparation, including, for instance, the creation of new parameters, specific naming conventions, or auxiliary elements modelling, among others.

3.2.2.2 Rule Translation

This step involves transforming the interpreted rules into machine-readable instructions or algorithms. This process requires logical reasoning, algorithm design, and syntax translation to correctly capture and convert the pseudo code's semantics, conditions and actions from the previous step into executable commands or scripts. This ensures that the translated rules faithfully represent the intended behaviour.

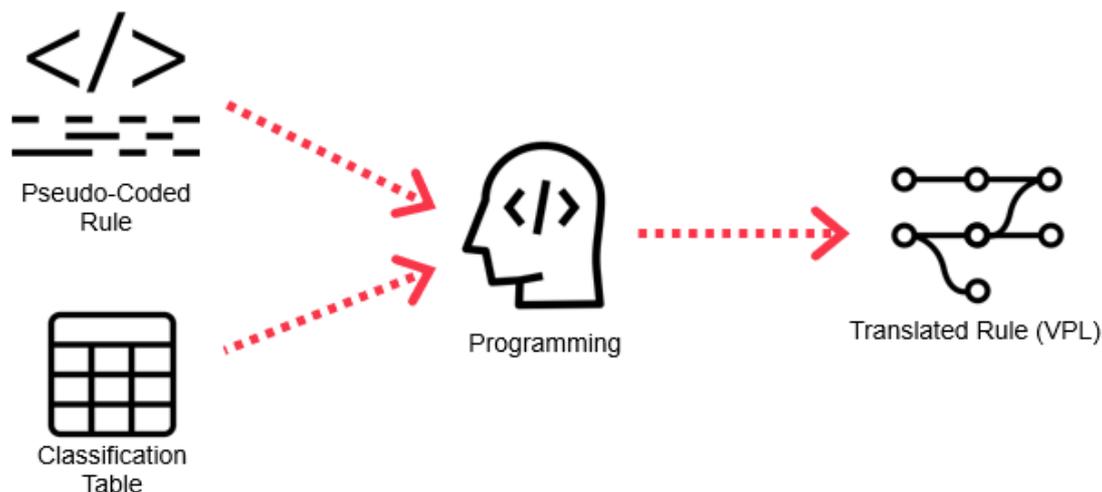


Figure 34: Rule Translation Concept

As reviewed in Section 2, Visual Programming Languages provide a powerful way to integrate rule-checking functionalities within the BIM workflow. In the case of Dynamo, there is a direct relationship between the model's elements and the interface, which highly benefits the process of designing Automated Compliance Checks (ACCC). For example, it allows for querying, editing, and modifying model elements based on the rule requirements. Also, its visual interface enables the user to interact with the rules in a user-friendly manner, facilitating customisation and adaptability when needed.

Specifically, the workflow will transform the small operations extracted from the pseudo coding process into functions -or nodes- in the VPL environment. The nodes -and their connections- will represent the rules' syntax, thus relating them to the Model's elements.

Additionally, to ensure consistency and clarity in this process, the Classification Tables and the Encoded rules will serve as guidelines throughout this conversion process. These guidelines provide a framework for structuring and expressing the rules in the visual programming representation, enabling easier comprehension and utilisation of the translated rules.

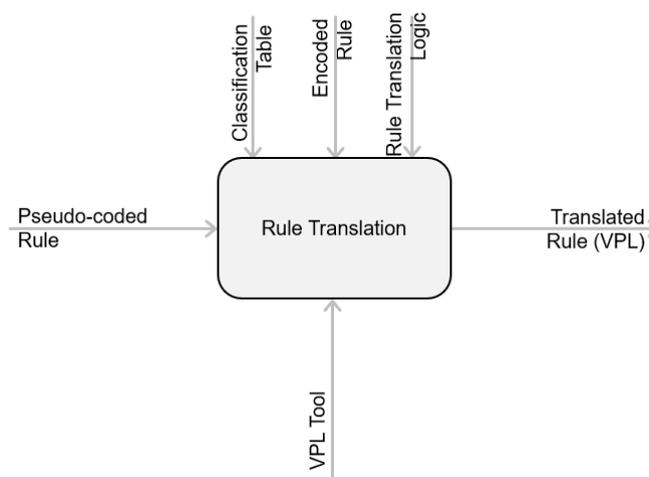


Figure 35: Rule Translation Step's IDEF0

Moreover, as the VPL allows synchronising the display of model elements with the rule requirements, the codes will have real-time feedback. This way, immediate improvement may be made within the same process. This may also allow us to check non-compliance or potential issues during the Translation Process.

Finally, custom node packages, such as Clockworks, Archilab, or Spring Nodes, will be most likely used, as they provide advanced or complex nodes that will allow the creation of simplified Dynamo graphs.

3.2.2.3 Rule Execution

During Rule Execution, the previously defined and translated rules are run. The process begins by identifying the conditions or triggers and determining when and where a rule should be executed. These conditions can be based on specific phases or milestones, data inputs, or predefined criteria.

Once the conditions for rule execution are met, the rules are activated and applied to the relevant data or system. This typically involves evaluating the requirements specified in the regulations and determining whether they are satisfied.

The actions specified in the rules are executed if the conditions are met. These actions can include calculations, data transformations, updates to the system, or any other established operations that drive the desired behaviour or outcome.

Throughout the Rule Execution process, it is essential to ensure that the execution is performed accurately and efficiently. This may involve implementing optimisation

techniques, handling rule conflicts, and monitoring the execution for errors or inconsistencies.

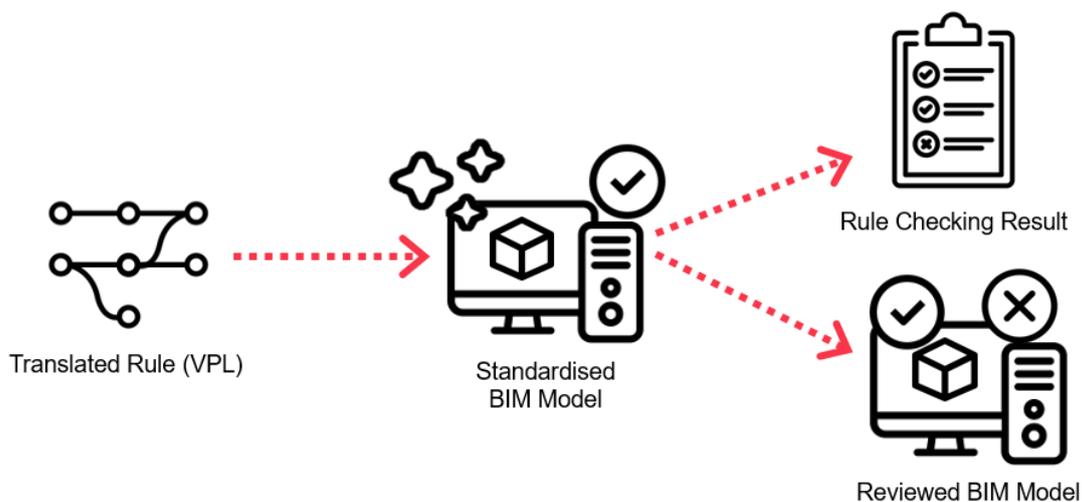


Figure 36: Rule Execution Concept

The Rule Execution process may be iterative, with multiple rules executed sequentially or simultaneously based on the specific scenario or system requirements. The process continues until all applicable checks have been completed and the desired outcomes have been achieved.

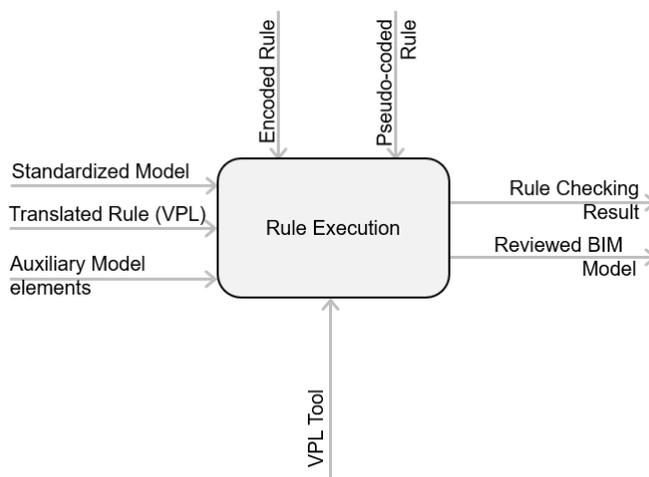


Figure 37: Rule Execution Step's IDEF0

The output of the Rule Execution process is the impact or result of the applied rules on the system or data. This can include updated data values, system state changes, or any other observable effects resulting from the execution of the rules.

3.2.2.4 Report Checking Results

The generation of compliance check reports involves reviewing, verifying, and representing the outcomes of the checks. It includes data visualisation, statistical analysis, and report generation. The idea is to generate visual or graphic representations regarding the compliance (or non-compliance) of the rules directly related to the BIM Model and its elements. This programming task must be directly integrated into the VPL graph.

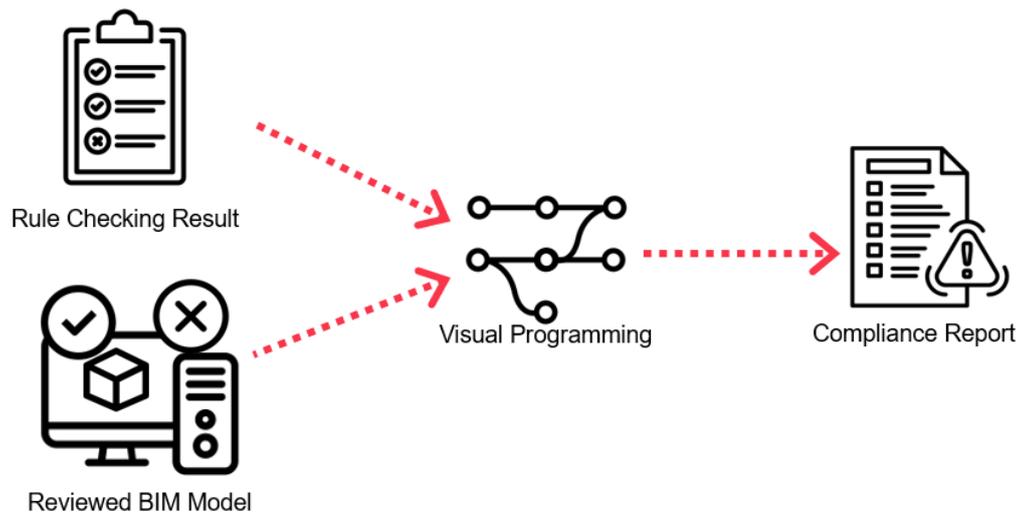


Figure 38: Report Checking Results Concept

The results of the compliance checks must be summarised and presented in a thorough and structured format. Different types of reports may be created depending on the rule being checked. Each of them may also have individual formats, and some may include compliance tables or written-text reports. These formats or templates are designed and thought through for each rule, or at least a specific rule, to facilitate clear communication and analysis.

In specific cases, these reports are mandatory and or specified within the same rule. In other cases, the results may be included in a table representing several rules being complied with for legal purposes (e.g., Urban and Building Rules Compliance Tables inserted on a Sheet for obtaining Building Permits). Other reports may not require a specific template and may be generated solely for internal purposes. For example, a simple message or alert indicating non-compliance can be a report for reviewing the Model or Design.

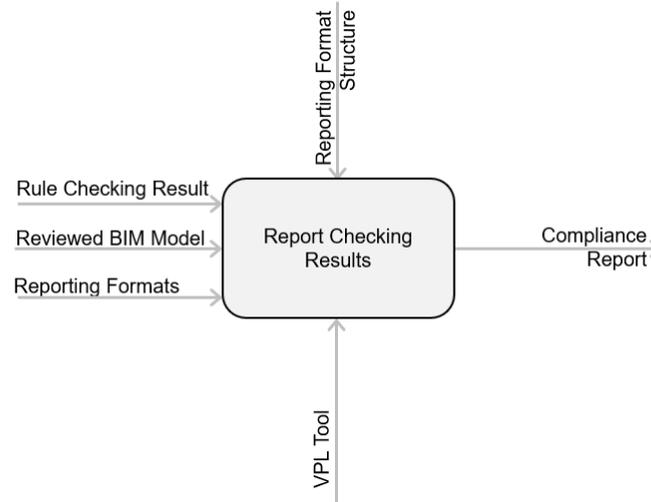


Figure 39: Report Checking Results Step's IDEF0

Using Reports for Rule Checking is a crucial outcome for the whole Automation Process. They need to be structured and standardised to ensure consistency and efficiency. Additionally, they may be used for comparing results across different stages of the building's lifecycle or even among projects.

Finally, the whole Framework's Application is shown in Figure 40, and a decomposition of its function is shown in Figure 41 (a higher resolution version of Figure 41 may be found in Annex A).

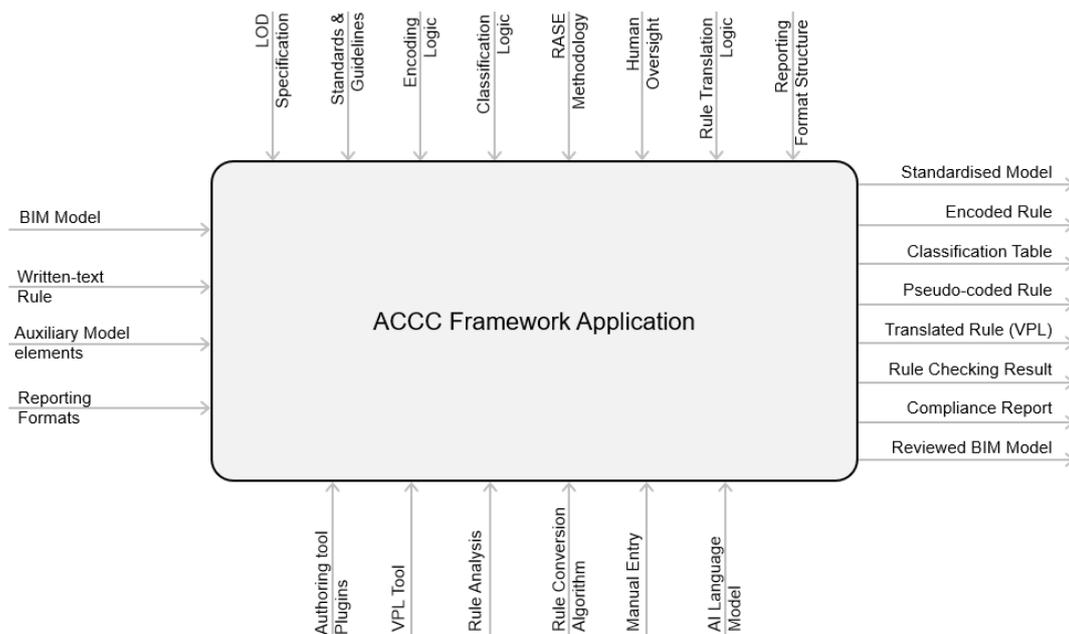


Figure 40: ACCC Framework's Application IDEF0

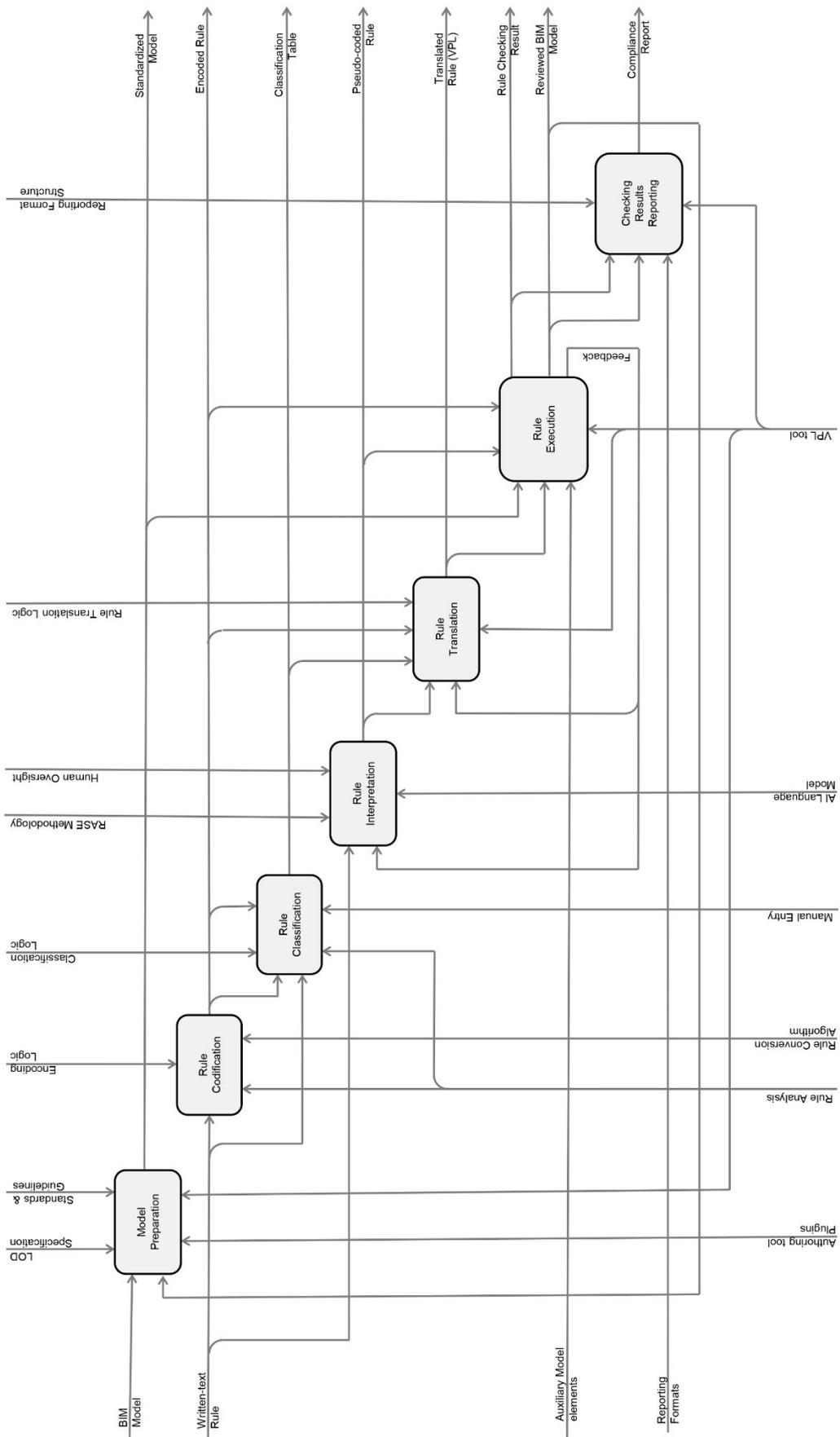


Figure 41: ACCC Framework's Application IDEF0 decomposition

3.3 Applied Methodology

The applied research approach involves using the methodology in a case study provided by Jereb in Budja Arhitekti. This will demonstrate the effectiveness and applicability of the framework in a real-world scenario.

The case study should represent a suitable example that aligns with the research objectives and the company's expertise and interests. Factors such as the project's complexity, the presence of diverse rule requirements, and the potential for showcasing the capabilities of the methodology will be considered.

The Application of the Solution will include the following key aspects:

- Implementation of the methodology: The developed methodology will be applied to the selected case study, following the established workflow for each process and adapting it to the specific requirements and constraints of the project.
- Validation and evaluation: An assessment of the performance and effectiveness of the methodology in the applied context will be done. This will evaluate the accuracy and reliability of the automated compliance checks using the developed tools and techniques. Then, feedback will be collected from the execution regarding the methodology's usability, efficiency, and practicality.
- Iterative refinement: Iterations and methodology refinement will be done as necessary based on the findings and feedback from the applied research. Any identified limitations, challenges, or areas for improvement will be addressed to enhance the framework's overall effectiveness and usability.

Applying the methodology to a specific use case will provide valuable insights and practical experience that will contribute to the further development and validation of the framework. It will identify potential issues or areas where the methodology can be fine-tuned to meet better professionals' needs in automating compliance checks within the BIM environment.

The applied research approach ensures the methodology is theoretically accurate, practically viable, and relevant in real-world architectural projects. By collaborating with Jereb in Budja Arhitekti, the research may benefit from their expertise and contribute to improving compliance-checking practices in the architectural industry.

4. AUTOMATED COMPLIANCE CHECK FRAMEWORK

The primary objective of this chapter is to create a well-structured framework that integrates qualitative and quantitative methods based on the definitions made in the Methodology chapter. This framework will serve as the blueprint for designing and developing the tools, algorithms, and methodologies required for ACCC.

4.1 Use Cases

Based on Unified Modelling Language (UML) and Model-Driven Architecture (MDA), the Use Cases approach will provide a structured and intuitive method for defining actor interactions and the ACCC Framework on determined contexts.

4.1.1 General Approach

Defining a general Use Case establishes a foundational representation of the Framework's functionalities during compliance verification. This overview illustrates how different components interact seamlessly to perform compliance checks. In this dissertation, Use Cases demonstrate the practical engagement of various actors and components with the framework, as seen in Figure 42.

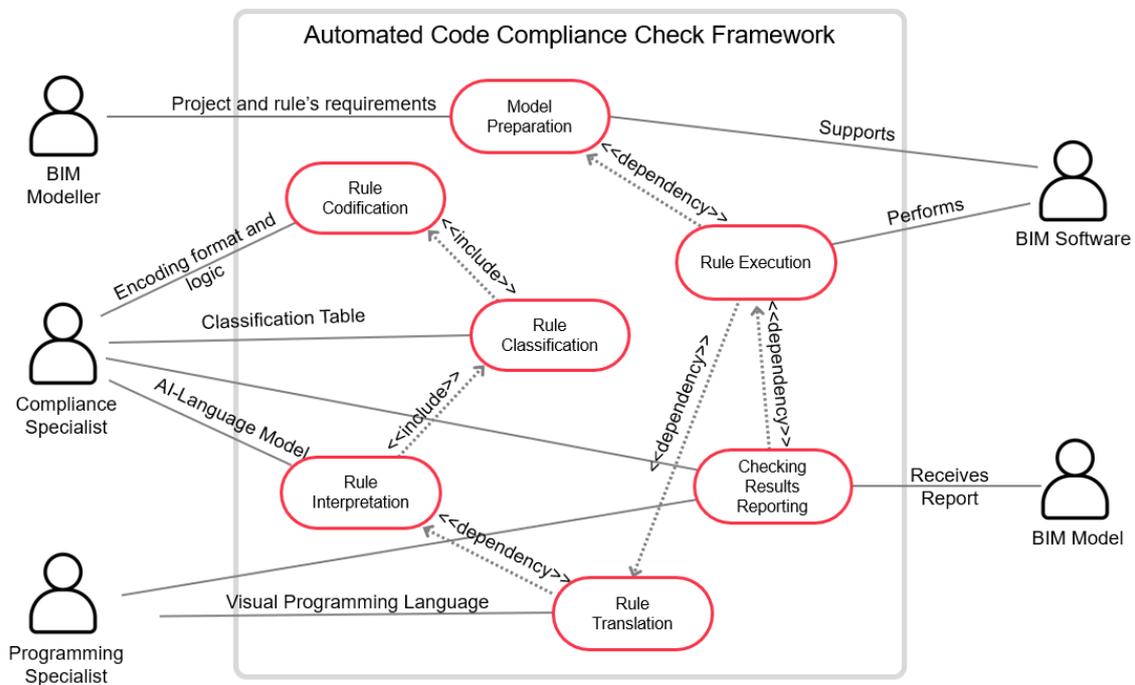


Figure 42: General Use Case

The General Use Case Diagram (Figure 42) visually captures the relationships between the Framework's seven core steps or functionalities. It showcases how actors like Professionals, Software and Models engage with each functionality to ensure a thorough compliance verification process. It illustrates the flow of information and actions between different stages, underscoring the systematic nature of the Framework's operations.

4.1.2 BIM Approach

As previously mentioned, the primary objective of this dissertation is to provide a flexible and adaptable Framework applicable to various scenarios. This is achieved by defining specific use cases within different contexts of BIM project stages, effectively showcasing its adaptability within disciplinary (AECO) and temporal (building lifecycle) dimensions.

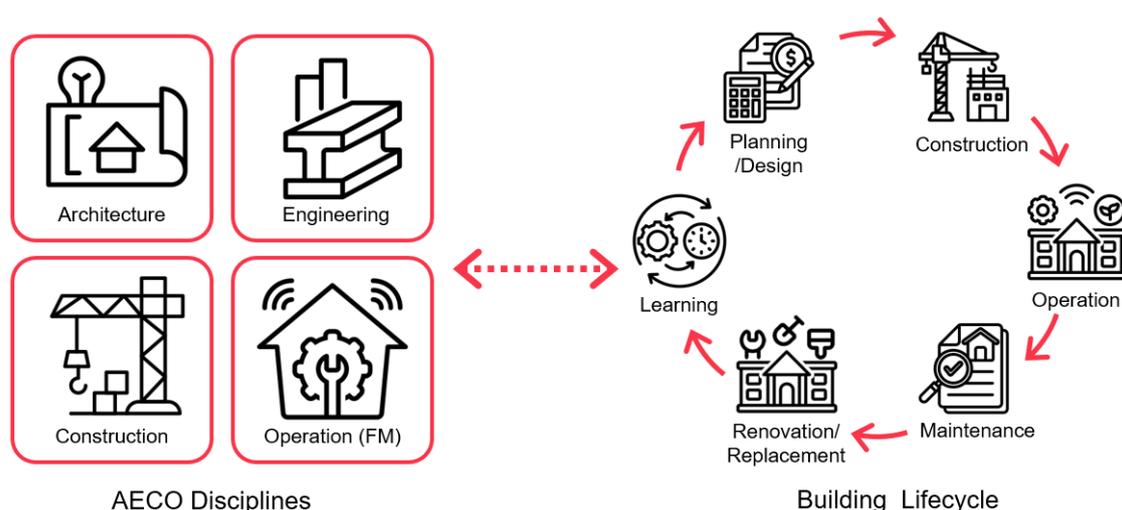


Figure 43: AECO Disciplines and Building's Life Cycle for Use Cases

Integrating a BIM approach into this framework aims to align its functionalities with specific compliance requirements on each Building's Lifecycle stage and according to the role of the AECO Discipline. For instance, Architects might focus on design compliance, Structural Engineers on integrity compliance and MEPs functionality compliance, Construction on implementation and material compliance, and Facility Managers on operation and maintenance compliance. These may happen in one or more stages of the Building's Lifecycle.

Additionally, BIM Uses may be incorporated into this analysis for a clearer scenario of how the Framework could be applied to several stages of a Project.

4.1.3 Examples

As mentioned above, the proposed Framework's adaptability across distinct stages of the building lifecycle and within diverse professional domains is one of its most crucial aspects. Its effectiveness in automating compliance checks can be exhibited by showcasing how it can be integrated into concrete examples.

Generally, Architecture and Engineering Disciplines perform compliance checks on the Early Stages of the Building's Life Cycle. As construction starts to be developed, the Construction professionals ensure the accurate translation of designs into physical reality, adhering to approved plans and complying with material specifications and safety requirements. During the Operations Phase, Facility Management Disciplines handle maintenance, accessibility, and occupant well-being.

4.1.3.1 Architecture - Design Phase

During the Design Phase, an architect may employ the Framework to verify compliance with Concept, Draft, Basic, and Detail Designs. The Framework may streamline the architect's design process, ensuring compliance with relevant regulations and standards. Here, several aspects like urban, spatial, constructive, program-related, or accessibility compliances may be checked upon the BIM Model.

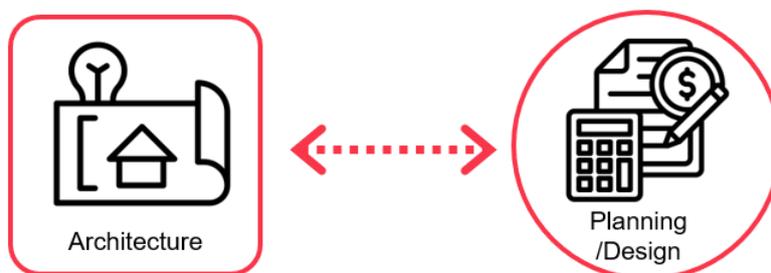


Figure 44: Architecture & Design Phase use case.

4.1.3.2 Structural Engineering–Design and Construction Phases

In this scenario, Structural Engineers may use the Framework to perform compliance checks during the Design Phases, too. It may be applied to assess the proposed structural design, considering factors such as load-bearing capacity, material specifications, safety codes, and stability requirements. Additionally, as the project progresses to the Construction Phase, Structural Engineers may continue to use the Framework to monitor the construction process, ensuring that executed elements match the approved designs and assess construction methodologies.

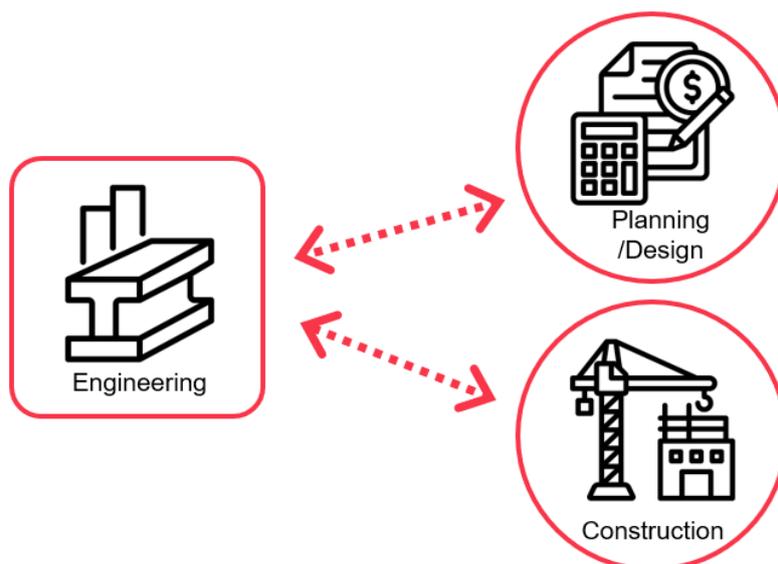


Figure 45: Engineering & Design and Construction Phases use case.

4.1.3.3 Facility Management - Operation Phase

Facility managers may employ the Framework to perform compliance assessments during the Operation Phase of a Building's Lifecycle. Operation professionals may evaluate compliance checks regarding accessibility regulations and preservation requirements and ensure building systems and components adhere to maintenance protocols. They could have immense importance in the context of As-Built Models and Digital Twins.

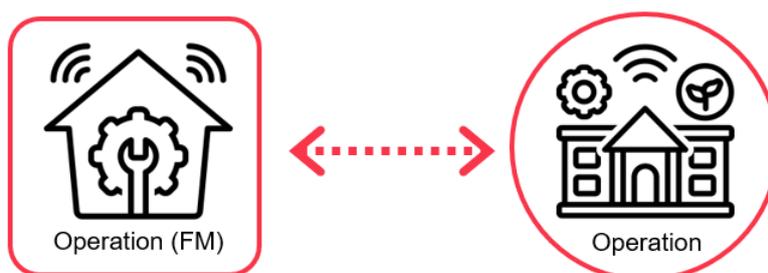


Figure 46: Facility Management & Operation Phase use case.

Employing the Use Case Approach within the Framework allows a general understanding of practical scenario applications to be easily represented. Additionally, by doing so, the potential of the Framework to solve compliance issues across several Disciplines and Stages is proof of its flexibility and adaptability.

4.2 Model Preparation

As mentioned in previous chapters, the importance of Models' Standardisation is immense. This statement does not apply only to the purpose of Automated Code Compliance Checks but to the BIM process in general. Nevertheless, as the present work is focused on ACCC, efforts will be directed at establishing a standardised model representation that captures the relevant information needed for rule checking.

4.2.1 Main Considerations

By incorporating quality assurance and control techniques, emphasising consistency and good practices, adhering to industry standards and guidelines, and integrating with plugins, software tools, and VPL tools, the Model Preparation phase establishes a structured approach to automate compliance checks in BIM models. This framework ensures the BIM model is accurately represented, compliant, and ready for rule-based analysis and verification.

This process encompasses several critical activities aimed at achieving this goal.

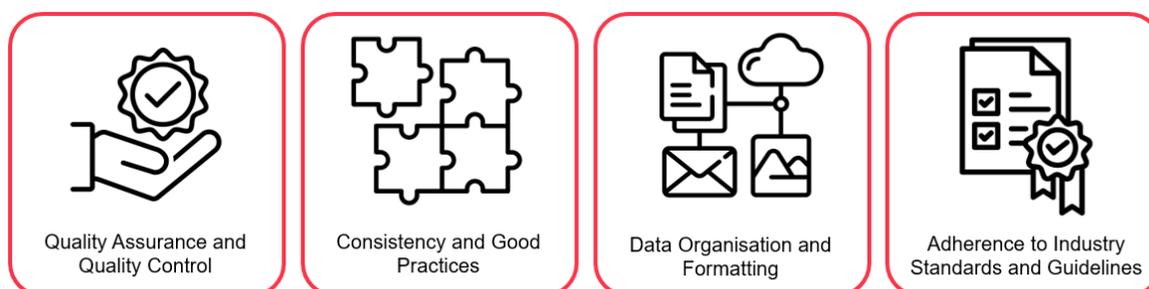


Figure 47: Model Preparation Considerations

4.2.1.1 Quality Assurance and Control

Quality assurance and control techniques should be employed to verify the accuracy, completeness, and adherence to the BIM model's data criteria. This involves implementing automated checks to identify errors, inconsistencies, or missing information within the model. Tools, plugins, and add-ons provide data validation, error detection, and cleansing features, enhancing the BIM model's overall quality. Some of the main aspects related to QA/QC are:

- Error identification: This involves automatically detecting and flagging any inaccuracies or mistakes within the data. By identifying errors early in the process, potential issues that might impact compliance checks can be addressed and corrected promptly.

- **Inconsistencies identification**: Consists of detecting discrepancies or conflicts within the BIM model. These inconsistencies can occur in naming conventions, parameter values, or element classifications. Correcting inconsistencies helps maintain uniformity and accuracy throughout the model.
- **Missing information identification**: Aims to identify any data gaps or essential information absent in the BIM model. Missing data can hinder accurate compliance checks and analysis. Recognising this aspect enables modellers to fill in the missing information to ensure the model is comprehensive and ready for compliance checks.



Figure 48: QA & QC Aspects

4.2.1.2 Consistency and Good Practices

This concept involves establishing and adhering to standardised naming conventions, parameter definitions, and modelling practices. Consistency ensures uniformity across the model and enhances its comprehensibility. Software plugins can assist in enforcing naming conventions and modelling standards by automatically checking and modifying element names, parameter values, and other attributes to ensure consistency. Aspects to be checked may include:

- **Standardized Naming Conventions**: Involves establishing and adhering to a consistent and predefined set of rules for naming elements within the BIM model. These conventions ensure that elements are named systematically and organised, making identifying and understanding them easy.
- **Parameter Definitions**: This entails defining a clear and consistent set of parameters representing specific attributes of BIM elements. These parameters should be well-defined, providing accurate and relevant information for each

element. By consistently defining these parameters, the BIM model becomes more informative and suitable for compliance checks and analysis

- Modelling practices: Refers to the standardised approaches and techniques employed when creating and modifying BIM model elements. Consistent modelling practices enhance the model's structure and improve its accuracy and integrity. Enforcing good modelling practices results in a more coherent and reliable BIM model.

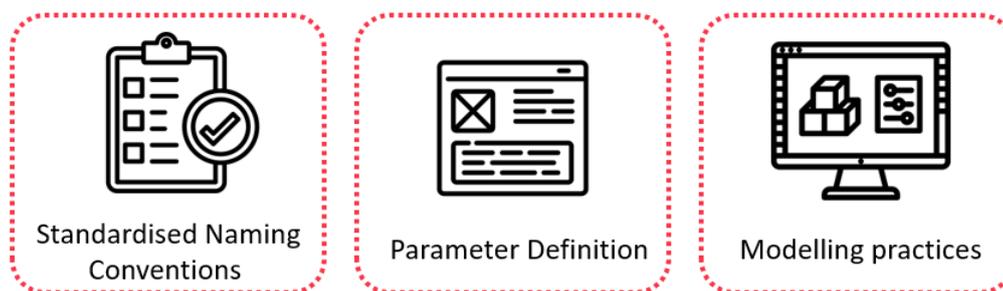


Figure 49: Consistency and Good Practices Aspects

4.2.1.3 Data Organisation and Formatting

Organising and formatting the BIM model according to established conventions enhances its clarity and ease of understanding. This facilitates efficient navigation, analysis, and compliance checks. Following adequate considerations will lead users to guide users in consistently organising and formatting the model. Some important aspects to consider are:

- Model Organization and Structure: Refers to the systematic arrangement and grouping of elements within the model to facilitate efficient management and navigation. It involves logically grouping related elements and establishing a clear hierarchy representing the building's physical and functional relationships.
- Project Browser Management: Consists of organising views, sheets, and other project-related items in the project browser to improve accessibility and ease of use. It involves creating a well-structured and intuitive hierarchy to access specific project components quickly.
- Data Categorization: Involves using predefined categories and defining custom categories to represent different building components accurately. It also includes

using datum (grids, levels, reference planes, etc.) to position and classify elements based on their type and properties.

- Workset Management (Revit-Specific): Focuses on dividing the BIM model into smaller subsets known as worksets to enable efficient collaboration among team members. It includes defining clear workset names, ownership, permissions, and coordination protocols.
- Model Filtering and Visibility: Refers to controlling the display of elements in various views to enhance model clarity and reduce visual confusion. It includes using view filters and templates to apply consistent visibility settings and graphic overrides.
- Assemblies, Families, and Groups: Involve combining multiple elements into one unit to simplify complex components and improve model performance. They allow for efficient management of repeated elements and facilitate collaborative design.
- File Management and Collaboration: Covers practices related to managing BIM files and ensuring effective collaboration among team members. It includes adopting centralised model management, backup strategies, version control, and collaborative platforms for seamless data sharing.

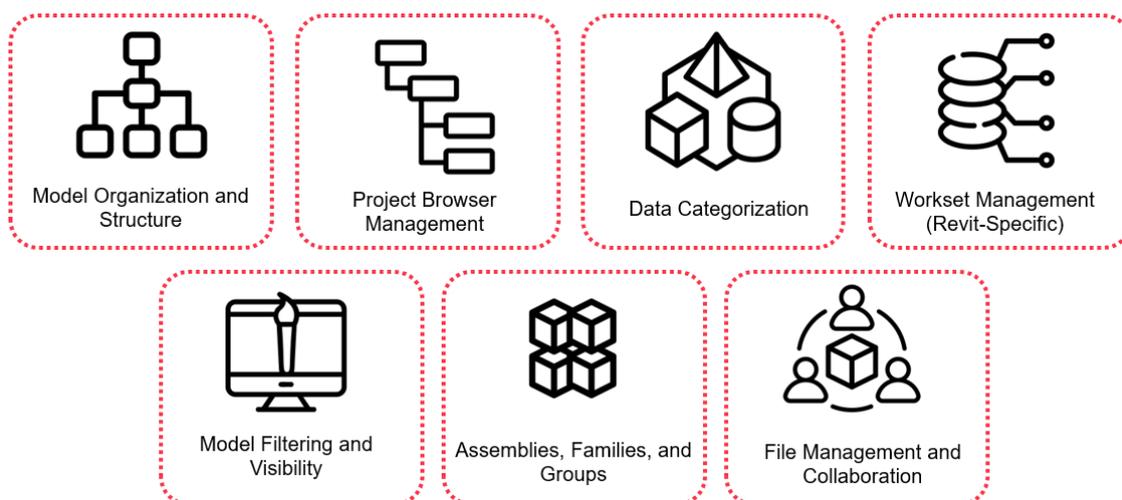


Figure 50: Data Organisation and Formatting Aspects

4.2.1.4 Adherence to Industry Standards and Guidelines

Adherence to Industry Standards and Guidelines refers to conforming to established norms, protocols, and specifications in BIM implementation and data exchange. Following these standards ensures that the BIM model complies with industry best practices, promotes interoperability, and facilitates seamless collaboration among various stakeholders. Adhering to recognised guidelines makes the BIM model more reliable, consistent, and ready for automated compliance checks and rule-based analysis. For example, some aspects to evaluate might include:

- Level of Information Need (LOIN) Compliance: Involves determining the required level of detail for different elements within the BIM model based on project and stakeholder requirements. The LOIN specifies the extent to which elements need to be modelled and documented, ensuring the model meets the specific needs of various project phases.
- Element Classification and Property Assignment: Entails categorising BIM elements based on standard classification systems or industry-specific standards. Property assignment involves assigning relevant attributes and properties to each element for consistent data representation and analysis.
- Data Exchange Formats and Interoperability: Refers to standardised file formats for sharing BIM data between different software platforms. Interoperability involves ensuring that data can be seamlessly transferred and understood by various BIM tools and applications.
- Compliance with BIM Execution Plans (BEP) and Project Standards: BIM Execution Plans (BEP) outline the project-specific approach to BIM implementation and collaboration. Adherence to the BEP ensures that all stakeholders follow the agreed-upon protocols and workflows throughout the project.

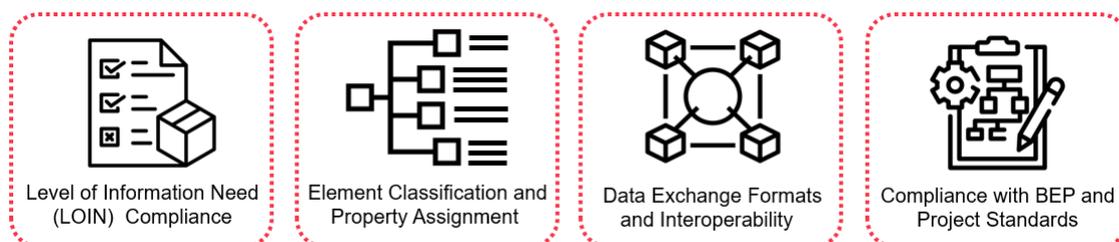


Figure 51: Adherence to Industry Standards and Guidelines Aspects

4.2.2 Integration of VPL Tools

Collaboration with Virtual Programming Language (VPL) tools can enhance the Model Preparation phase by addressing complex modelling requirements and enabling advanced design effects. VPL tools offer specialised functionalities and mechanics that can be utilised to improve the accuracy and representation of the BIM model.

For example, collaborating with a VPL tool designed explicitly for generative modelling can be beneficial if the compliance rules involve intricate geometries or parametric design requirements. Such tools enable the creation of complex geometries and parametric relationships, ensuring that

Additionally, as VPLs allow the development of precise and customised graphs, it is most likely that while running some of them, mistakes (geometric or meta-data related) that were not known will be put in evidence, thus creating opportunities to improve Model aspects that were not foreseen or intended to be corrected.

For instance, some of the most common nodes that can aid with the previous aspects or more complex standardisation processes in Dynamo nodes may Include:

- *String.Contains*
- *Object.IsNull*
- *Element.BoundingBox*
- *Geometry.DoesIntersect*
- *(Control Flow) If*
- *Element.GetParameterValueByName*
- *Element.SetParameterByName*
- *Element.OverrideColorInView*
- *(Math Operators) ==, >, <, etc.*
- *Filter.ByBoolMask*

4.2.3 Authoring Tool Plug-in Example: Autodesk Model Checker

Revit's Model Checker Plugin will be showcased in the present section. Nevertheless, third-party Software, such as Solibri or BIMcollab ZOOM, are also efficient and easy-to-use tools. Revit's Model Checker can be crucial in achieving various aspects of BIM Model Preparation, including Quality Assurance and Quality Control (QA/QC), Consistency and Good Practices, Data Organization and Formatting, and Adherence to Industry Standards and Guidelines.

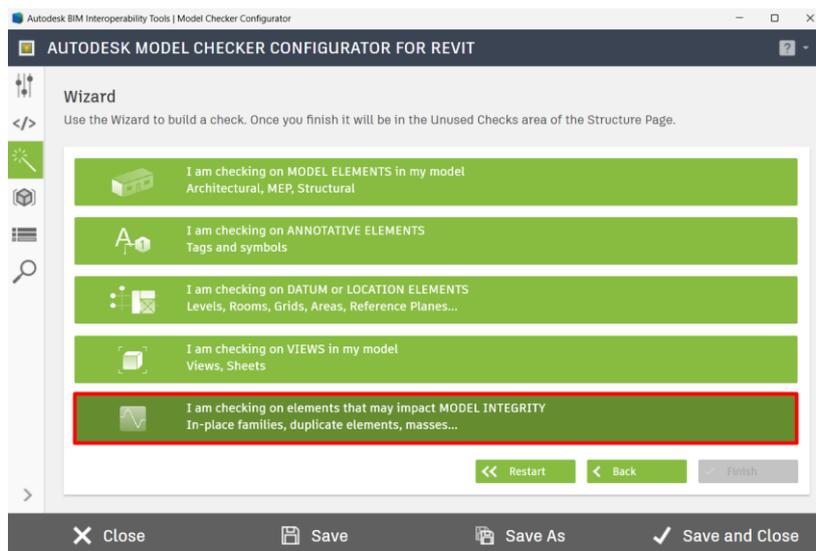


Figure 52: Model Checker’s Wizard for configuring QA/QC checks.

Firstly, for QA/QC, Model Checker ensures error identification by running predefined rule checks that flag inconsistencies, inaccuracies, and potential clashes within the BIM model. It helps maintain data quality by detecting missing information and highlighting elements that require improvement. By automating these checks, Model Checker reduces manual review efforts and ensures a more accurate and compliant BIM model.

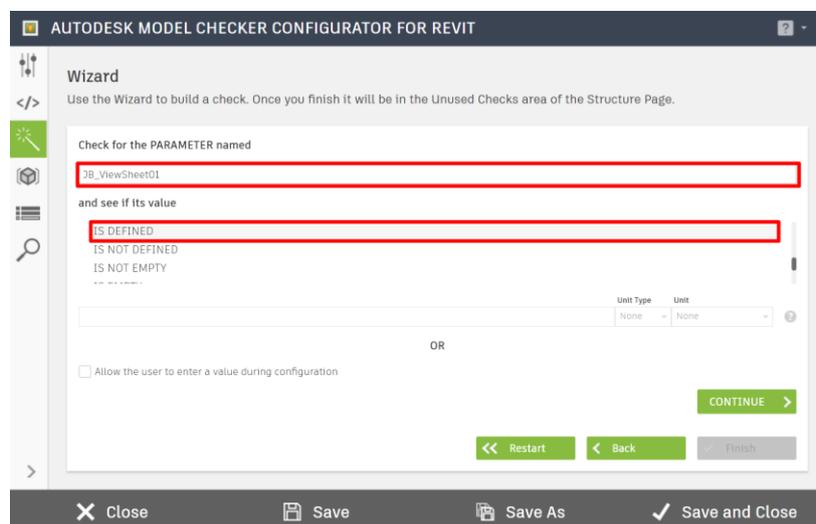


Figure 53: Filtering Views with Project Browser Parameter.

Additionally, concerning Consistency and Good Practices, the Model Checker may enforce standardised naming conventions and parameter definitions throughout the model. It validates that elements are correctly classified and grouped, promoting a consistent modelling approach. By verifying adherence to modelling practices, Model

Checker aids in creating a well-structured and organised BIM model that facilitates effective collaboration and data understanding.

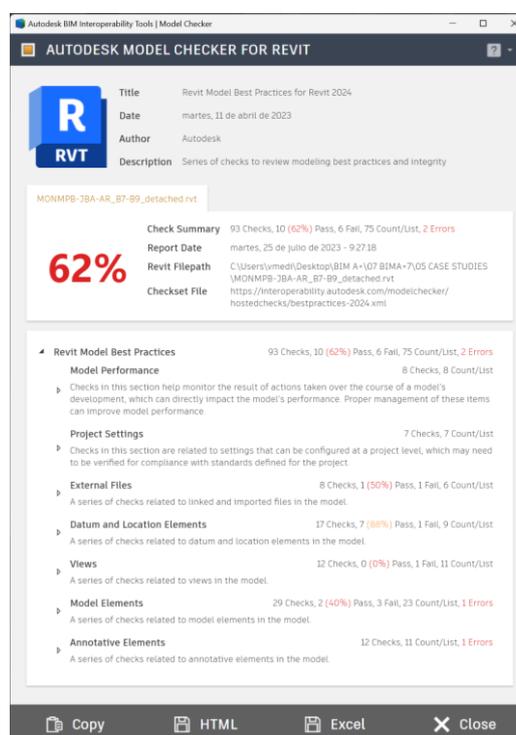


Figure 54: Model Checker Report for pre-defined checks.

Regarding Data Organization and Formatting, the Model Checker verifies that elements are correctly categorised and assigned to appropriate data sources, as seen in Figure 53. It ensures the project browser is well-organized, correctly labelled views and data exchange formats comply with industry standards. By validating data organisation, Model Checker improves the overall coherence of the BIM model and simplifies information retrieval.

Furthermore, Model Checker contributes significantly to Adherence to Industry Standards and Guidelines. It validates that the BIM model complies with LOIN requirements, element classification systems, and custom project standards. Model Checker enhances interoperability with other software platforms by ensuring data exchange compatibility and promoting collaborative workflows. It also verifies that the BIM Execution Plan (BEP) guidelines are followed, minimising discrepancies in project execution and improving consistency across all project phases.

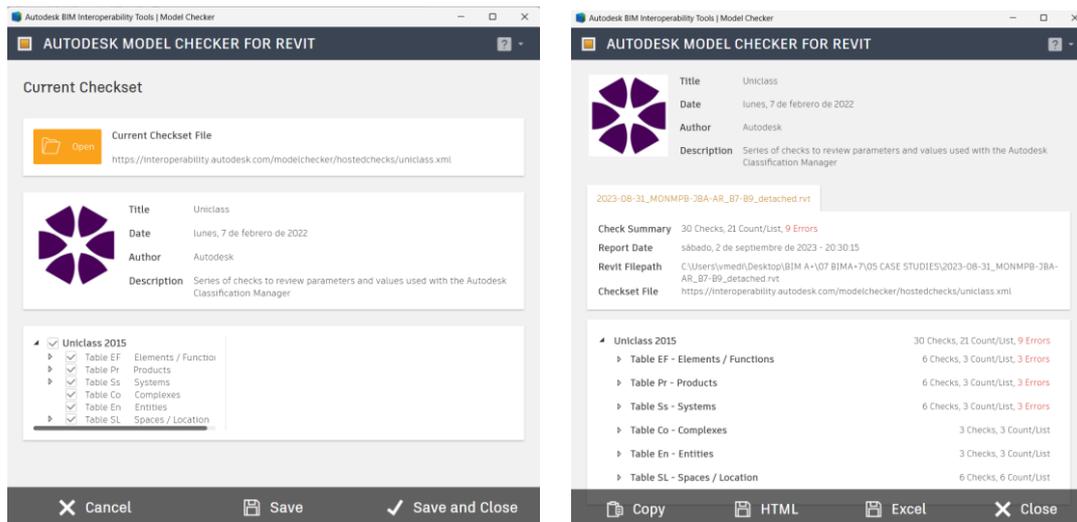


Figure 55: Predefined Check Set for Uniclass and its Check Report.

It is crucial to mention that while Model Checker software is powerful in identifying violations of various requirements, as mentioned above, it is just a checking tool. To reach a Standardized Model, additional measures must be undertaken, such as manually addressing compliance or running other automated tools for doing so.

4.3 Rule Codification

Rule Codification involves their transformation into a structured coding system, enabling more efficient development of subsequent steps in a more organised way of displaying, storing, and labelling. The design considerations, characteristics, and structure will be developed to provide a universal representation for comparison, consistency, and simplification. Several vital aspects will be considered for designing the Coding Structure to ensure its effectiveness and usability.

4.3.1 Formatting

First, the coded rule shall be easily readable and interpretable by humans and machines. A single-string format, structured from general to specific, will be used. Depending on each case, the Categories will consist of 2 to 4 characters and be separated with hyphens (-). Underscores (_) shall not be used. The general rule shall be to use alphanumeric characters and capital letters to improve clarity and readability, though exceptions will be allowed if explicitly needed. Most fields will have abbreviations of their concepts, and no spaces, special characters, or other punctuation marks shall be used unless explicitly required by the rule (e.g., Rule numbering as in article or legal documents)

4.3.2 Universality and Consistency

This system shall be applicable across different projects, domains, and jurisdictions. This guarantees consistent representation and comparison of rule compliances, regardless of a specific context. Considerations such as using English as a universal language and ISO-standardized country codes will be crucial.

4.3.3 Rule Characteristics

Based on the previously described considerations, the following coding structure is proposed for the rule encoding process:

1. Domain Code: Represents the domain or discipline to which the compliance rule belongs. It will consist of 2 or 3 characters in capital letters. This field shall be mandatory. Some examples of domain codes include:
 - **AR**: Architecture
 - **STR**: Structure
 - **URB**: Urbanism
 - **PLU**: Plumbing
 - **MEC**: Mechanical
 - **EL**: Electrical
2. Category Code: Specifies further categorisation of the rule within its domain. From two up to four characters in capital letters shall be used. This shall be a mandatory field. Examples of categories may include:
 - **FS**: Fire Safety
 - **SAF**: Safety
 - **ACC**: Accessibility
 - **ELEC**: Electrical Systems
 - **VENT**: Ventilation
 - **SPA**: Spaces
 - **AC**: Air Conditioning
 - **HV**: Heat and Ventilation
 - **ENE**: Energy
 - **SUS**: Sustainability
 - **CON**: Construction
 - **LGT**: Lighting Systems
 - **MOD**: Model-related
 - **LOA**: Load Bearing
3. Rule Identification: A unique numeric or alphanumeric identifier for each rule within a specific category or domain. This identifier helps in referencing and organising rules. They may have different formats according to the specific documents, standards, or guidelines from which the rule shall be extracted. Including the Document Name for more precise identification and representing

the document's organisation or company is recommended. This field shall be mandatory. For example:

- **CNYBC05**: City of New York Building Code, Rule 05
- **4.3.2.3**: Article 4.3.2.3
- **NBSCH1a**: NBS BIM Object Standard, Chapter 1a
- **IBCL23**: International Building Code, Law N° 23

4. Country Code: Indicates the country-specific regulations or standards to which the compliance rule adheres, according to the ISO 3166 country codes. "Alpha-3" codes shall be used. Examples of country codes include:

- **AUS**: Australia
- **BRA**: Brazil
- **CHL**: Chile
- **CHN**: China
- **COL**: Colombia
- **DEU**: Germany
- **EGY**: Egypt
- **ESP**: Spain
- **FRA**: France
- **GBR**: United Kingdom
- **INT**: International
- **JAP**: Japan
- **RUS**: Russia
- **SGP**: Singapore
- **SVN**: Slovenia
- **USA**: United States

5. Document Type Code: Refers to the kind of document in which the compliance rule may be found. It provides information about the document's specific type or format containing the rule. Also, the name or a code representing the document's official name could be used. Examples of this category may be:

- **STN**: Standard
- **REG**: Regulation
- **GUI**: Guideline
- **COD**: Building Code
- **REC**: Recommendation

6. Year Code: Represents the year the compliance rule was introduced or last updated. Four-digit numbers will be used. Additionally, a distinction for the month may be added, if necessary, preceded by a dot. For instance, the Year Code could be represented as the following:

- **2006**: Year 2006
- **1995.04**: April 1995

7. Type of Rule Code: A code indicating the type or nature of the rule. This helps classify rules based on their purpose or function. A three-character code will be used to represent each of them. Examples of rule types would be:
 - **GEN**: General
 - **DIM**: Dimensional
 - **MAT**: Material spec
 - **ARE**: Area
 - **VAL**: Validation
 - **PER**: Permit
 - **GEO**: Geometric
 - **REV**: Revision

8. Compliance Framework Code: This concept indicates the compliance type of the rule or how the rule works. It is related to the rule's specific needs or requirements. This Category may be helpful for the Interpretation and Translation Phases. An up-to-four-character abbreviation of the type shall be used. Some of them may include:
 - **PRCD**: Procedural rule
 - **PRSC**: Prescriptive rule
 - **PERF**: Performance-based rule

9. Other Categories Code: Any other category with additional information for classifying the rule may be added to the coding structure. It is a flexible category to accommodate various aspects, such as versioning, specific notes, or any requirements not covered by the existing categories. They shall follow the same formatting logic described previously

4.3.4 Structure

Combining the codes of each category will generate a unique identifier for each compliance rule. It is vital to mention that the only mandatory fields shall be Domain, Category, and Rule Identification. The other categories shall be optional, as some rule category's information may be unavailable, redundant or unnecessary if used.

The most crucial aspect is that the criterion among a group of rules should be consistent when applying this step. This way, an eventual rule-checking tools library will be easier to navigate.

According to international or local standards, the Company's Code may be included as a prefix on the tool's file name to comply with existing guidelines or protocols.

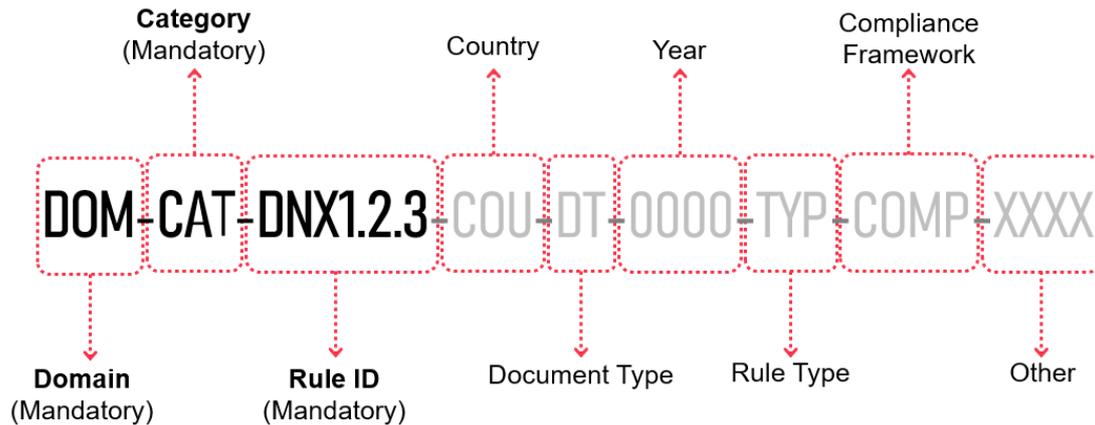


Figure 56: Encoding Structure

Finally, the coding structure of some rules may be represented as follows:

Rule	Code Structure	Formatted Code
Section 5.1 of UK's NBS BIM Object Standard	Domain-Category - Identification-Type	AR-MOD-NBS5.1-QUA
Article 2.6.1 of the Chilean General Urbanism and Construction Code	Domain-Category-Identification-Type-Compliance	URB-CON-OGUC.2.6.1-LEG-PRSC
Sub-chapter 2.2, Singapore Virtual Design and Construction Guide	Domain-Category-Identification	MOD-GUI-VDC-2.2

Table 2: Rule Encoding Examples

4.4 Rule Classification

Classifying rules provides a structured approach to categorising and organising rules, facilitating efficient interpretation and analysis. This section focuses on designing a classification table or sheet template for analysing and describing rules, selecting relevant characteristics -included in the Rule Codification step-hierarchy, and other important considerations. This sheet will serve as a central tool for organising and analysing the characteristics of individual rules, providing a systematic approach to categorising and understanding the rules based on their characteristics.

4.4.1 Classification Structure

The classification sheet/table will provide a tabular format that enables an organised categorisation and understanding of the rules concerning the BIM model. The design of the classification table will ensure clarity, ease of use, and effective data organisation.

Its design will include a general hierarchy and structure to aid in organising and categorising the rules' characteristics. Specifically, it will be divided into three sections: Identification Characteristics, Content Characteristics, and Translation Characteristics. Each section captures specific aspects of the rules, allowing for a thorough analysis and interpretation:

- Identification Characteristics: This section focuses on the identification and basic information of the rules. It provides essential context and references for each rule.
- Content Characteristics: This section delves into the content-related attributes of the rules. It captures the nature and framework of the rules.
- Translation Characteristics: This section focuses on the aspects related to implementing and assessing rules. It includes characteristics related to compliance check methods and specific elements within the BIM software.

Each characteristic will correspond to a blank field filled in using text in an alphanumerical format, depending on the origin of the information.

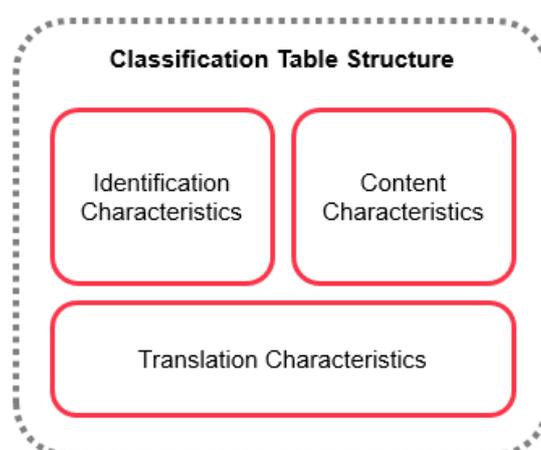


Figure 57: Classification Table Structure

4.4.2 Characteristics

The classification table captures various characteristics of each rule. The following characteristics are commonly used in the classification of rules:

4.4.2.1 Identification Characteristics:

- **Domain:** The primary domain or field to which the rule belongs. It categorises the rule within a specific discipline, such as architectural, structural, MEP (mechanical, electrical, plumbing), energy, or other relevant domains. This characteristic helps provide context for the rule and assists in organising rules based on their respective domains.
- **Category:** The specific subdomain or category within the domain to which the rule pertains. It further categorises the rule based on its relevance to a particular area, such as architectural design, structural calculations, MEP systems, or other categories. This characteristic helps in organising rules with similar characteristics and purposes. Categories may include Fire Safety, Fire Prevention, Accessibility, Electrical Systems, Ventilation, Impermeabilization, Energy, Sustainability, Construction, Lighting Systems, or Insulation.
- **Country:** The country or region to which the rule applies. It specifies the geographical scope or jurisdiction of the rule. This characteristic is essential in understanding the rule's applicability to regional or national regulations and standards. In some cases, this field shall indicate its international condition for International Standards or Guidelines.
- **Document Type:** The type of document where the rule is specified. This characteristic provides information about the source and format of the rule. It can be a building code, industry standard, project-specific requirement, recommendations, client requirements, or any other relevant document type.
- **Document Name:** The document's name or title containing the rule. It helps identify the specific document from which the rule is derived.
- **Document Year:** The year the document containing the rule was published. It provides temporal information about the rule and helps track changes and

updates over time. This characteristic may also display the month or specific date and document versioning.

- **Rule ID or Number:** This characteristic provides the unique identifier or number for each rule or where they could be contained. It will be a reference point for easy identification and retrieval of specific rules during the compliance check process.

4.4.2.2 Content Characteristics:

- **Rule Description:** This characteristic captures the written rule found in its original format. It includes the exact wording and language of the rule as it appears in the source document, without any modifications or interpretations. The rule description provides a textual representation of the rule's requirements, specifications, or constraints as stated by the regulatory authority, industry standard, or governing document
- **Type of Rule:** The categorisation of the rule based on its purpose or nature. This characteristic helps in understanding the focus and scope of the rule. Suggested examples of rule types may include:
 - General Rule: A rule that applies broadly and covers multiple aspects or disciplines within the domain. It provides general guidelines or requirements for compliance.
 - Dimensional Rule: A rule that focuses on the dimensional aspects of a BIM model or its elements, specifying tolerances, dimensions, or spatial requirements.
 - Material Specification Rule: A rule specifies the types of materials, their properties, or their usage within the BIM model.
 - Safety Rule: A rule that addresses safety requirements or regulations to ensure the safety of occupants or construction personnel.
 - Permit Rule: A rule that pertains to obtaining necessary permits or approvals for construction or design activities.

- Validation Rule: A rule that checks the accuracy, consistency, or validity of data or information within the BIM model.
- Legal Rule: A rule derived from legal regulations, codes, or statutes, ensuring compliance with the applicable laws and regulations.
- Geometric Rule: A rule that focuses on geometric or spatial relationships, constraints, or configurations within the BIM model.
- Quality Rule: A rule that defines quality standards or requirements for the BIM model, ensuring high-quality deliverables.

It is important to note that these examples are not exhaustive, and the specific types of rules may vary depending on the domain and context of the compliance check. Additionally, some rules may consist of a combination of these categories, which should be indicated if required.

- **Compliance Framework**: The framework or approach used to achieve compliance with the rule. This characteristic categorises the rule based on the approach or methodology it follows. Some examples of compliance frameworks could be:
 - Procedural Compliance: A framework that outlines step-by-step procedures or processes to ensure compliance with the rule. It may provide specific instructions, workflows, or checklists for compliance assessment.
 - Prescriptive Compliance: A framework that provides explicit requirements or specifications that must be met to achieve compliance. It may include specific measurements, limits, or thresholds that must be adhered to.
 - Performance-Based Compliance: A framework focusing on achieving desired performance outcomes rather than prescribing specific methods or requirements. It emphasises meeting performance criteria or targets specified in the rule.

- Risk-Based Compliance: A framework that assesses and manages risks associated with non-compliance. It may prioritise compliance efforts based on non-compliance's potential risks or impact.
- Reporting and Documentation Compliance: A framework that emphasises documentation, record-keeping, and reporting of compliance-related activities. It ensures proper documentation of compliance measures and outcomes.
- Continuous Improvement Compliance: A framework that emphasises continuous monitoring, evaluation, and improvement of compliance practices. It promotes ongoing review and enhancement of compliance measures

As in the Type of Rule Category, other compliance frameworks could be described and identified, and more than one of these aspects could be present in one rule.

4.4.2.3 Translation Characteristics:

- **Compliance Check Methods**: The method or approach used to conduct the compliance check for the rule. It describes the specific methods, algorithms, calculations, or techniques used to evaluate compliance with the rule.
- **Model Elements**: The specific elements or components within the Authoring Tool or software that the rule pertains to. It helps identify the relevant elements to consider when assessing compliance.
- **VPL Nodes**: It specifies relevant VPL nodes or scripts that might be used to implement or check compliance with the rule. This approximation will be helpful for the Rule Translation step, as it provides a guideline for designing a VPL graph
- **Observations**: Any observations needed or suggested for the translation process may be written here. It may also include anticipated outcomes or requirements that should be met according to the rule or the expected compliance conditions or results for the rule, providing a target for assessing compliance.

4.4.3 Template

In this section, the design of the sheet template for the rule classification will be displayed. The template serves as a visual representation of how the classification table is structured and organised. It provides a clear, intuitive layout for capturing and documenting the relevant rule characteristics.

The Rule Classification Table Template will be the following:

DOM-CAT-DNX1.2.3-COU-DT-0000-TYP-COMP-XXXX			
Identification Characteristics		Content Characteristics	
Domain		Description	
Category			
Country			
Document Type			
Document Name		Type of Rule	Compliance Framework
Document Year			
Rule ID or Number			
Translation Characteristics			
Compliance Check Methods	Model Elements	VPL Nodes	Observations

Table 3: Rule Classification Template

4.5 Rule Interpretation

Rule Interpretation involves the previous step of transforming written-text rules for later translating them into machine-readable formats. To accomplish this, a combination of tools working in combination are employed. This section focuses on the design of the prompt for an effective pseudo-code obtention using AI-language tools, which facilitate the interpretation process and generate optimal outputs for translation.

4.5.1 Prompt Design

A well-designed prompt is essential to ensure effective communication with AI language models and obtain accurate and detailed interpretations. The prompt serves as a guide for the AI model, specifying the desired structure and output format. In this research, the prompt is structured to align with the RASE methodology, which provides a classification framework based on four operators: Requirements, Applicability, Selection, and Exception, as seen in Figure 58.



Figure 58: RASE Methodology

The prompt instructs the AI model to interpret a rule and generate an optimal pseudo-code representation suitable for Revit Dynamo, a widely-used BIM tool; by explicitly mentioning the use of RASE methodology and pseudo-code representation for Revit Dynamo, the prompt sets clear expectations for the interpretation process.

The designed prompt will be as follows:

Using the RASE methodology as the structure, interpret the following rule and generate an optimal pseudo-code representation for Revit Dynamo. The RASE methodology involves classifying rules into four operators: The Requirement (in blue), Application (in green), Selection (in yellow), and Exception (in red). Generate a colour-coded written-text rule, and for the pseudo-coding, ensure that the interpretation output is detailed, accurate, and well-structured.

Rule: [Paste the written-text rule here]

This prompt effectively guides the AI language model, providing clear and specific instructions for the interpretation task, the desired structure, and the expected output format.

4.5.2 AI-Language Tools

In this step, a selection of AI language models will be tested to assist in interpreting and providing a comparative analysis of results. The chosen models will be the most

renowned text-based AI-language models nowadays: Phind AI, Open AI's ChatGPT, Google Bard, and Notion AI.



Figure 59: Main AI-Language Models in the Market

These AI language models have demonstrated proficiency in understanding and analysing human-written text, making them valuable resources for interpreting complex rules and generating explanations or clarifications. Some of them, like Phind, are also exclusively efficient in the programming field.

Their performance and effectiveness will be evaluated by comparing the results of different AI language tools. This analysis will contribute to understanding the strengths and limitations of each tool in the context of automated compliance checks for BIM.

It is important to note that while AI language models enhance the efficiency and accuracy of the interpretation process, human assessment remains critical throughout the workflow to ensure the reliability and precision of the automated compliance check system.

As an example, the defined prompt was input in the AI-language models using the following arbitrary rule:

"The perimeter railings or walls of the roof of the top floor of the building may not have a height greater than 1.5 m from the finished floor level, must have at least 80% transparency, and not be climbable."

The responses for each of the engines were as follows:

4.5.2.1 Phind AI

To generate a pseudo-code representation for the given rule in Revit Dynamo, we first need to understand the rule and its components. The rule can be broken down into the following RASE components:

Requirement: The perimeter railings or walls of the roof of the top floor of the building.

Application: May not have a height greater than 1.5 m from the finished floor level and must have at least 80% transparency.

Selection: Not applicable in this rule.

Exception: Not be climbable.

Now, let's translate this into pseudo-code:

```
// Requirement (Blue)
FOR each building
  IF floor is top floor
    FOR each object on the roof
      IF object is a perimeter railing or wall
        // Application (Green)
        IF object height from finished floor level > 1.5m
          ALERT "Object height exceeds 1.5m limit"
        END IF
        IF object transparency < 80%
          ALERT "Object transparency is less than 80%"
        END IF
        // Exception (Red)
        IF object is climbable
          ALERT "Object should not be climbable"
        END IF
      END IF
    END FOR
  END FOR
END IF
END FOR
```

4.5.2.2 Open AI's ChatGPT

Using the RASE methodology, we can interpret the given rule as follows:

1. Requirements:

- *The perimeter railings or walls of the roof of the top floor of the building may not have a height greater than 1.5 m from the finished floor level.*
- *The railings or walls must have at least 80% transparency.*
- *The railings or walls must not be climbable.*

2. Applicability:

Apply the following conditions to the perimeter railings or walls of the roof of the top floor of the building:

- *Limit the height to 1.5 m from the finished floor level.*
- *Ensure at least 80% transparency.*
- *Make the railings or walls non-climbable.*

3. Selection:

- *Perimeter railings or walls of the roof*
- *Top floor of the building*

4. Exception:

- *No specific exceptions are mentioned in the rule.*

Based on this interpretation, here is an optimal pseudo-code representation in Revit Dynamo:

```
# Define parameters
max_height = 1.5 # Maximum height allowed
min_transparency = 0.8 # Minimum transparency required

# Get all perimeter railings or walls of the roof of the top floor
roof_perimeter_elements = GetRoofPerimeterElements()

# Iterate over each element
for element in roof_perimeter_elements:
    height = GetElementHeight(element)
    transparency = GetElementTransparency(element)
    climbable = IsElementClimbable(element)

# Check conditions
if height > max_height:
    RaiseError("Height of element exceeds maximum allowed.")

if transparency < min_transparency:
    RaiseError("Transparency of element is below the minimum
requirement.")

if climbable:
    RaiseError("Element is climbable, which is not allowed.")
```

Note: The pseudo-code provided is a simplified representation and assumes the availability of appropriate functions and methods in the Revit Dynamo environment for selecting elements, getting and setting their properties (such as height and transparency), and determining climbability. The actual implementation may vary depending on the specific tools and functions available in the Revit Dynamo platform.

4.5.2.3 Google Bard

RASE Methodology *Interpretation*

1. **Requirement:** *The perimeter railings or walls of the roof of the top floor of the building may not have a height greater than 1.5 m from the finished floor level. The perimeter railings or walls must have at least 80% transparency. The perimeter railings or walls must not be climbable.*
2. **Application:** *The rule applies to the perimeter railings or walls of the roof of the top floor of the building.*
3. **Selection:** *The rule does not apply to any other railings or walls in the building.*
4. **Exception:** *There are no exceptions to the rule.*

Pseudo-code Representation for Revit Dynamo:

```
def perimeter_railings_rule(top_floor_railings):  
    for railing in top_floor_railings:  
        if railing.height > 1.5:  
            raise Exception("Railing height must be less than  
or equal to 1.5 m")  
        if railing.transparency < 0.8:  
            raise Exception("Railing transparency must be at  
least 80%")  
        if railing.is_climbable:  
            raise Exception("Railing must not be climbable")  
  
def main():  
    top_floor_railings = get_top_floor_railings()  
    perimeter_railings_rule(top_floor_railings)  
  
if __name__ == "__main__":  
    main()
```

This pseudo-code first gets the list of railings on the top floor of the building. Then, it calls the `perimeter_railings_rule()` function, which iterates through the list of railings and checks if they meet the rule'. If any railings do not meet the requirements, the `raise Exception()` statement is executed, which will cause the script to stop.

4.5.2.4 Notion AI's

Rule

::: Requirement ❤️

The perimeter railings or walls of the roof of the top floor of the building may not have a height greater than 1.5 m from the finished floor level, must have at least 80% transparency, and must not be climbable.

Pseudo-Code

```
# Select all perimeter railings or walls of the roof of the
top floor of the building

selected_railings_walls =
AllElementsOfCategory("Railing/Wall").Where(lambda x:
x.IsTopRoof() and x.IsPerimeter())

# Check the height of the selected railings or walls and if
it is greater than 1.5 m from the finished floor level, set
it to 1.5 m

for element in selected_railings_walls:
    if element.Height > 1.5:
        element.Height = 1.5

# Check the transparency of the selected railings or walls
and if it is less than 80%, set it to 80%

for element in selected_railings_walls:
    if element.Transparency < 80:
        element.Transparency = 80

# Ensure that the selected railings or walls are not
climbable

for element in selected_railings_walls:
    if element.IsClimbable():
        element.MakeNotClimbable()
```

When comparing these four different tools, it is evident that ChatGPT's response was more detailed and well-structured, both for the written-text RASE methodology and pseudo-coded rule. This is why the Application Phase will be recommended as the tool for Rule Interpretation.

Nevertheless, none of the responses show erroneous or misleading information, meaning none can be used indistinctly. It will depend on the preferences of the user or stakeholder using the framework.

It is essential to mention that the results may vary within the same AI tool, even when the same prompt is inserted. If the response is unsatisfactory, another request can be asked for or even iterate on specific modifications.

4.6 Rule Translation

The Rule Translation process is a critical bridge between abstract rule descriptions and their machine-readable execution. It consists of the transformation of Rule Interpretation outputs into machine-executable instructions. This sub-chapter outlines key considerations and procedural details for the successful execution of this step.

4.6.1 Main Considerations

Before diving into the practical implementation, certain considerations must be acknowledged. This section highlights essential factors that are decisive in achieving accuracy and reliability in Rule Translation. These considerations include Semantic Precision, Adaptability, and Real-time Feedback.

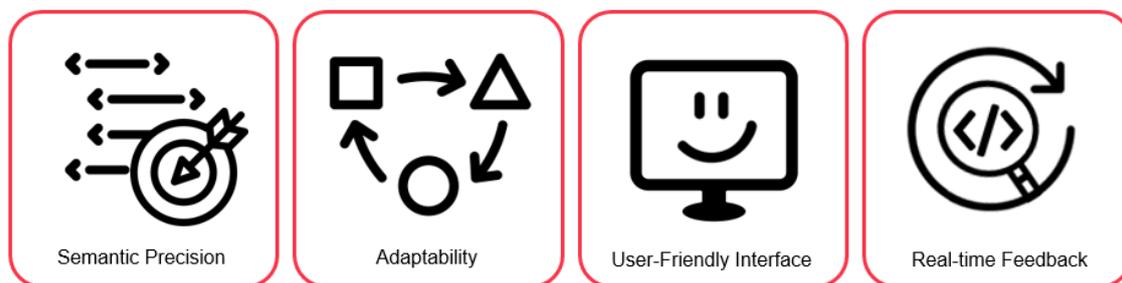


Figure 60: Rule Translation Considerations

- **Semantic Precision:** Translating rules from a human-readable form to a machine-executable format requires meticulous attention to detail to ensure that the rules' intended semantics, conditions, and actions are accurately preserved. Any deviation from the original intent could lead to erroneous compliance evaluations. It's essential to validate that the translated rules return the same outcomes as the

human-interpreted rules. This can be achieved through rigorous testing and validation to ensure they align.

- **Adaptability:** The Rule Translation process should be designed to accommodate diverse contexts, rule sources, and compliance requirements. Different projects might have varying elements, families, parameters, and metadata. Therefore, the translation approach should be adaptable and modular to handle different types of rules and projects effectively. Considering this adaptability, the framework becomes more versatile and can be applied to various BIM projects with varying compliance needs.
- **User-Friendly Interface:** If a visual programming language, such as Dynamo, is utilised for Rule Translation, the interface should be user-friendly. Not all users will have a deep technical background, and ensuring the interface is intuitive and easily navigable is important. A well-designed graph should allow users to interact with nodes and connections without requiring extensive programming knowledge. This empowers non-technical users, like architects and designers, to actively participate in the rule translation process. Actions like grouping nodes according to their combined functions and colour-coding help to achieve this.
- **Real-time Feedback:** Real-time feedback mechanisms within the translation environment enhance the efficiency and accuracy of the Rule Translation process. This feature enables immediate identification and correction of errors or inconsistencies as the rules are translated. Visual indications or alerts notify users of potential issues or deviations from the intended behaviour. Addressing issues in real time makes the Rule Translation process more iterative and dynamic, leading to more accurate translations and reducing the need for subsequent adjustments.

4.6.2 Procedural Approach

The Rule Translation process includes a set of procedures. It is a process that requires meticulous attention to detail to ensure that the translated rules accurately encapsulate the intended behaviour. Outputs from Rule Encoding and Classification are recommended as constant support tools. The following approach is proposed for the effective implementation of Rule Translation:

- 1. Rule Extraction:** This step involves extracting rule attributes from the developed sources: Classification Table, Coded Rule, and Pseudo-code. These rule attributes embody the compliance requirements to be automated.
- 2. Decomposition into Operations:** Deconstructing the main actions into discrete, manageable operations. Each operation should encapsulate a specific action or condition contributing to the overall rule's logic.
- 3. Algorithmic Design:** Formulating algorithms that capture the logic of each isolated operation. This involves defining the sequence of actions, conditions, and decision-making processes necessary for evaluating compliance.
- 4. Syntax Translation:** Translate the algorithmic design into a syntax that aligns with the chosen programming language or visual programming environment (e.g., Dynamo). This is done using visual components known as nodes. This step ensures that the logical constructs are faithfully represented in a machine-readable format. These nodes visually depict the logical steps inherent in the rules and are connected.
- 5. Mapping to Model Elements:** Establishing a direct correlation between the model's constituent elements and the nodes within the visual programming environment. Nodes or groups of nodes should correspond to specific model elements and their attributes relevant to compliance evaluation. VPL languages often represent model elements in the form of Lists.
- 6. Testing and Debugging:** Rigorously testing the translated rules through diverse scenarios and input data to ensure accuracy and correctness. Identify and rectify any discrepancies between intended behaviour and translated execution.

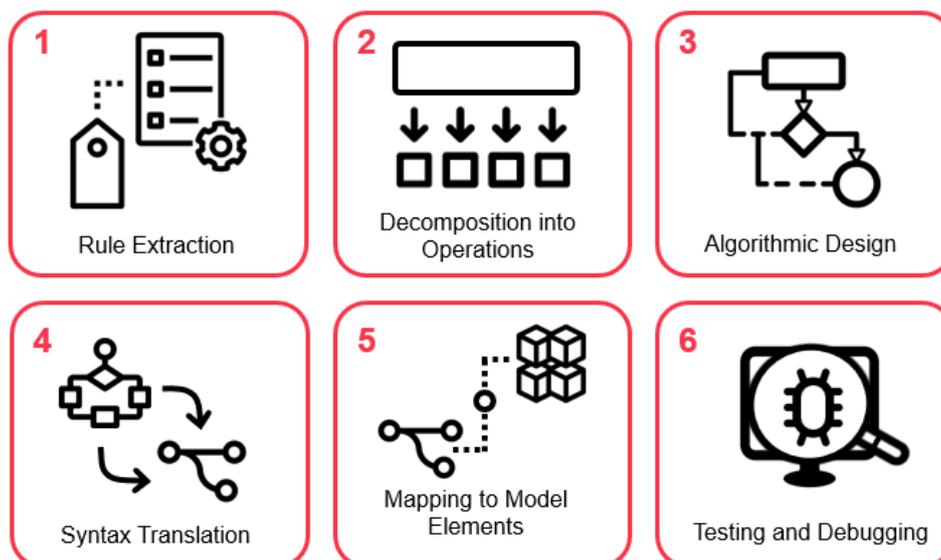


Figure 61: Rule Translation Procedural Approach

4.7 Rule Execution

In the Rule Execution phase of the Framework, ensuring the smooth operation of the code and usability by various stakeholders is vital. To achieve this, procedures and recommendations are established.

If using Dynamo as a main tool, integrating Dynamo Player or similar user-friendly interfaces provides a simplified way for the compliance checking graphs execution (Figure 62). Thus making it accessible to both technical and non-technical stakeholders.

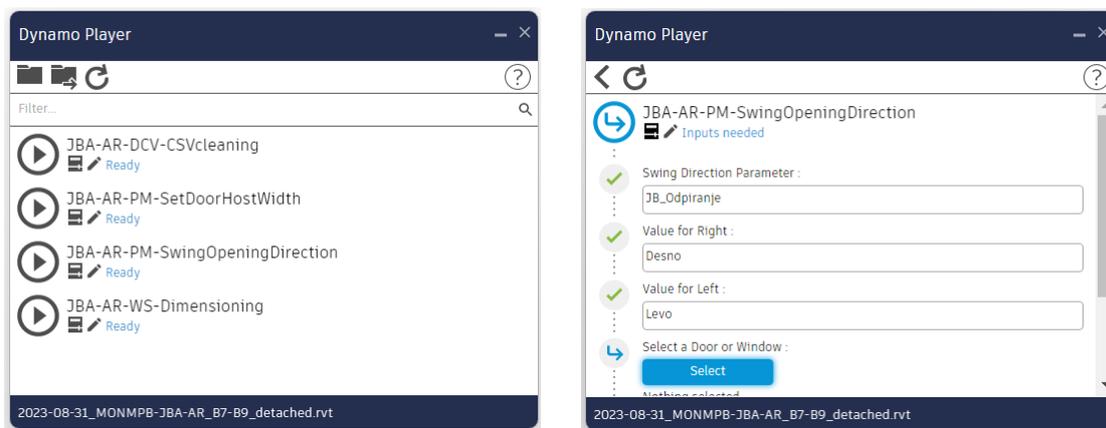


Figure 62: Dynamo Player Interface

Furthermore, considerations for a simpler Rule Execution will promote efficiency, functionality, applicability, and accessibility within VPL tools. These may include Input Nodes Definitions, Input Nodes Renaming, Writing Instructions, and Defining Protocols, among others.

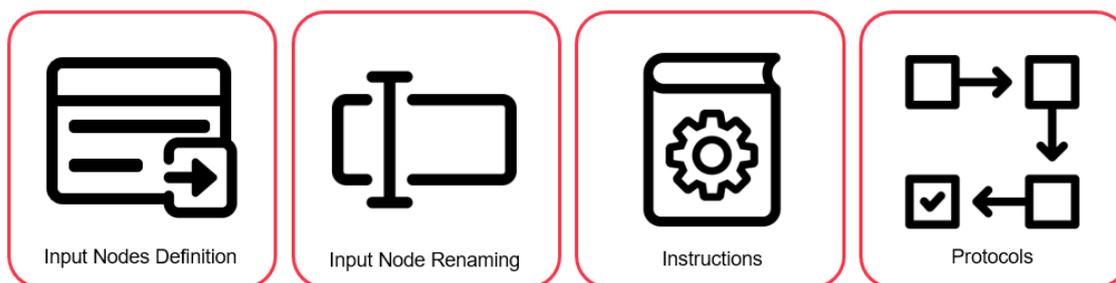


Figure 63: Rule Execution Considerations

4.7.1 Input Nodes Definition

Defining nodes as inputs when programming the VPL graph will further enhance usability. This step ensures that stakeholders can interact with the system intuitively,

facilitating rule execution. Additionally, it will be vital if the graph is eventually used for additional purposes like Generative Design.

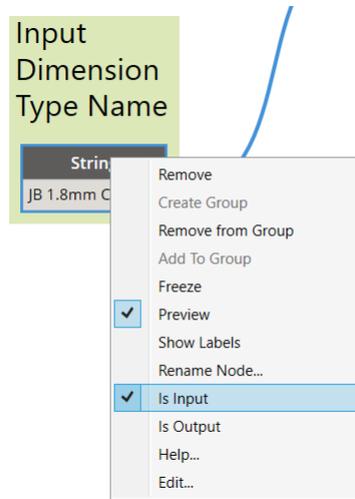


Figure 64: Input Node Definition

4.7.2 Renaming Input Nodes

As sometimes the Nodes' names are not as intuitive as desirable, simplifying user interactions by renaming input nodes with descriptive labels is helpful. This practice enhances user-friendliness and ensures stakeholders can easily identify and input relevant data. Additionally, Renamed Input Nodes are necessary for users to know how to fill them in with the required information in the Dynamo Player.

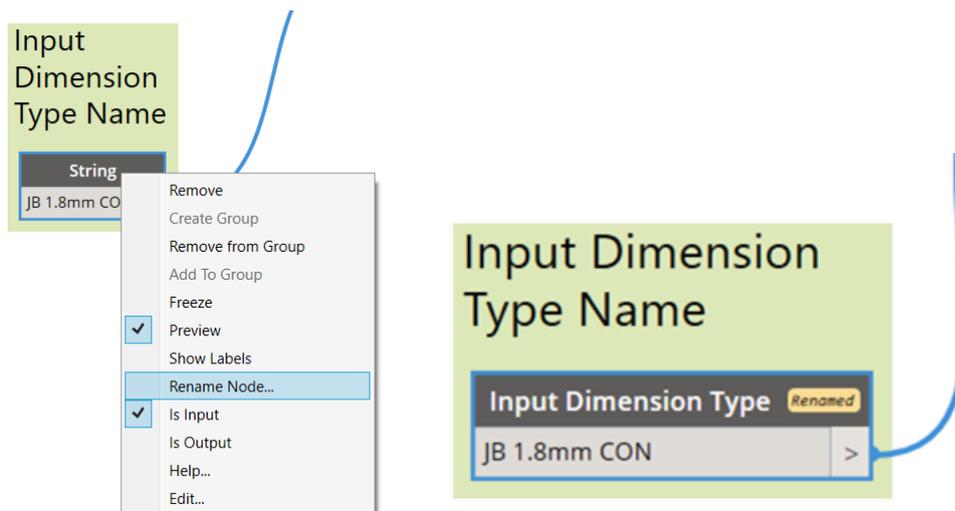


Figure 65: Input Node Renaming

4.7.3 Writing Instructions:

When guiding users through the Rule Execution process, providing clear and comprehensive instructions for each graph is decisive. These instructions will serve as a bridge between the system and its users, ensuring that stakeholders understand how to navigate and utilise the compliance check system effectively without the help of a programming professional. Instructions may be written in the graph or as an input node for Dynamo Player, as seen in Figure 66.

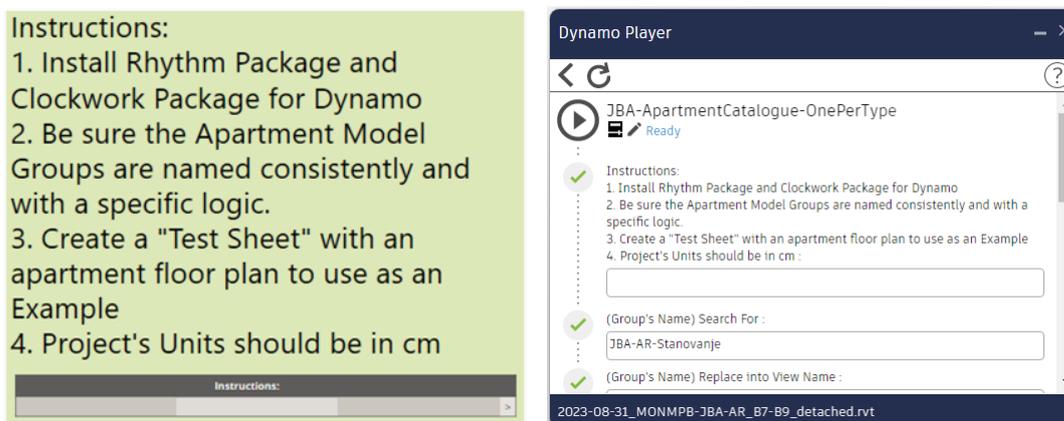


Figure 66: Instructions for Rule Execution

4.7.4 Protocols Definition

Establishing clear protocols is crucial to the execution process and promoting consistency and efficiency. Some protocols may include:

- **Condition Identification:** Clearly define the criteria for when a rule should be executed. This could be based on project phases, data inputs, or other specified conditions.
- **Rule Activation:** Document how to activate and apply rules once the conditions are met. Make use of user-friendly interfaces like Dynamo Player for easier rule activation.
- **Error Monitoring:** Implement error monitoring procedures to catch and address issues that may occur with incompatibilities between some rules and projects. This may lead to potential improvements in the VPL graphs.
- **Version Control:** Implement a version control system to track changes and maintain a record of modifications to each graph.

4.8 Checking Results Reporting

Checking Results Reporting involves presenting compliance assessment outcomes to facilitate understanding, decision-making, and communication among project stakeholders. This includes various types of reports that can be generated based on the nature of the compliance rules and the intended receiver.

4.8.1 Report Types and Formats

Compliance reports can broadly be categorised into two main types: Internal and External Reports (Figure 67). Internal Reports are designed to be utilised within the BIM model environment itself. They enhance the understanding of compliance status and assist in making necessary adjustments directly within the model. On the other hand, External Reports are intended for stakeholders outside the immediate BIM environment, such as project managers or regulatory authorities.



Figure 67: Type of Compliance Reports

In the scope of this dissertation, the main recommendation is to use Internal Report Types, as they provide a more streamlined workflow and do not rely on external tools or software. Additionally, generating External Reports within Visual Programming Language may require much greater effort. Nevertheless, depending on the situation or context (e.g., stakeholders don't have access to the BIM Model), External Reports may be needed regardless.

Various reporting formats are vital for understanding the diverse ways of representing compliances. These formats include written, tabular, visual, and cloud-based reports (Figure 68), offering distinct ways to present and communicate compliance information.

Moreover, combining various report formats to provide a broader representation of compliance is possible. For instance, an Internal visual report may be combined with a written report to accommodate both in-model adjustments and ease of understanding and identifying the checked elements.

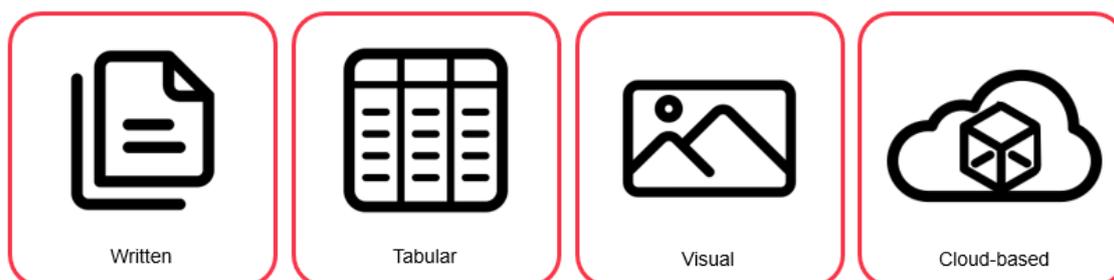


Figure 68: Compliance Report Formats

Additionally, different compliance rules may also require specific reporting formats. For example, urban regulations compliance checks may need the creation of specialised compliance tables, while structural integrity checks may require visual reports showcasing structural element highlights. Adapting the reporting format to the checked rule ensures the information is most meaningfully presented.

4.8.2 Written Reports

Written reports serve as a textual representation of compliance findings. They provide a structured narrative that helps stakeholders understand the compliance status and any necessary actions.

For internal use, written reports within the BIM model environment offer professionals insights into compliance, aiding in adjusting directly within the model. Additionally, it could be incorporated directly into the Model's Views to identify compliance with specific elements or situations. This may be done in simple text boxes or parametrised Annotation Families like Tags, providing context and insights to design teams or modellers.

Externally, they are useful to project managers, regulatory authorities, or other stakeholders, offering a clear, organised overview of compliance results for broader analysis and decision-making. External Software like Notepad or MS Word may be used for this purpose.

4.8.3 Tabular Reports

Tabular reports are all about structured data presentation, often in the form of tables. They offer a systematic way to summarise compliance information.

Internally, tabular reports are valuable for project teams working within the BIM model, providing them with structured compliance data for making informed decisions or

presentation purposes. Tabulated reports may use annotation families and parameters. For instance, compliance tables specific to urban regulations can be automatically generated, summarising compliance status for different elements within the model, as seen in Table 5.

NORMAS URBANISTICAS			
Superficie de Terreno	3.385,28 m ²		
Sup. Construida	13.866,03 m ²		
Sup. Util	10.099,04 m ²		
Sup. Util Sobre Terreno	9.309,21 m ²		
Ocupación de Suelo	Máxima	0,26	880,17 m ²
	Propuesta	0,26	855,33 m ²
	Área Libre	0,75	2.529,95 m ²
Constructibilidad	Máxima + 30 % Beneficio Excepc.	3,90	13.202,59 m ²
	Sup. Util Sobre Terreno	2,75	9.309,21 m ²
	Propuesta (Sup. Util + Excepc.)	2,94	9.969,23 m ²
Subterranidad (Área Excavada)	Máxima	0,70	2.369,70 m ²
	Propuesta	0,60	2.034,96 m ²
Antejardín	Máximo	5,00 m	
	Propuesto	5,00 m	
Distanciamiento	Mínimo	5,40 m (Art. 2.6.12 OGUIC)	
	Propuesto	5,74 m (Art. 2.6.12 OGUIC)	
Altura de Edificación	Máxima	28,00 m	
	Propuesta	27,40 m	
Adosamiento	Máximo	N/A	
	Propuesto	N/A	
Transparencia Cierros	Mínima	50% - 80% - 50%	
	Propuesta	100% - 98,50% - 89,12% (Cumple Art. 7 PRC)	
Altura Cierros	Máxima	1,30 m	
	Propuesta	1,75 m	
EXCEDENTE SUPERFICIE COMÚN			
Sup. Util Habitacional Sobre Terreno	9.309,21 m ²		
20% Máximo Superficie Común	1.861,84 m ²		
Superficie Común Sobre Terreno (Propuesta)	2.521,86 m ²		
Excedente Superficie Común Propuesta	660,02 m ²		
SUPERFICIES PROYECTADAS			
Superficie Edificada	Útil	Común	Total
Superficie Bajo Terreno	789,83 m ²	1.245,13 m ²	2.034,96 m ²
Superficie Sobre Terreno	9.309,21 m ²	2.521,86 m ²	11.831,07 m ²
Total	10.099,04 m ²	3.766,99 m ²	13.866,03 m ²

UNIDADES PROYECTADAS				
Unidades	Subterráneo	1º Piso	2º-10º Piso	Total
Departamentos	0	13	198	211
Estacionamientos	44	33	0	77
Biciclisteros	0	60	0	60
Bodegas	82	0	0	82
Total	126	106	198	430

CÁLCULO ESTACIONAMIENTOS		
Estacionamientos Vehiculares	Mínimo	Propuesto
Estacionamientos Vehiculares Exigidos (1 cada 3 viviendas)	71	71
+ Estacionamientos Vehiculares reemplazados por Bicicletas (máx 1/3 = 23 unidades)		4
+ Exigencia de estacionamientos de Bicicletas para descuento de estac. vehiculares		12
Sub-total Estacionamientos Vehiculares luego de reemplazo	71 - 4	67
Estacionamientos de visitas	1 (PcD)	10 (3 son PcD)
+ Est. habilitados para personas con discapacidad (no adicionales)	3	3
Estacionamientos dobles (hasta un 30% del total)	23 max.	8
Total Estacionamientos Vehiculares (Exigencia + visitas y PcD)	68	77
Estacionamientos de Bicicletas		
Estacionamientos de Bicicletas Exigidos (1 cada 2 Est. Propyectados)	39	48
+ Estac. de Bicicleta por reemplazo (máx 23 unidades)	12	12
Total Estacionamientos de Bicicletas	51	60

CÁLCULO ESCALERAS SUBTERRÁNEO - DESTINO: OTROS Art. 4.2.18. OGUIC			
Carga de Ocupación	Sup. / persona	Superficie Útil	Carga (personas)
Bodegas	40 m ² /persona	239,83 m ²	6
Total			
Cantidad y ancho mínimo: (Hasta 60 personas)	Exigido	1	1,10 m
	Proyectado	3	1,50

CÁLCULO ANCHO PASILLO SUBTERRÁNEO - DESTINO: OTROS Art. 4.2.18. OGUIC				
Nivel	Superficie Útil	Carga Ocupación	Mínimo (Art. 4.2.18)	Proyectado
Subterráneo -1	789,83 m ²	41	1,10 m	1,20 m

Table 4: Urban Compliance Tables as a Revit Annotation Families Example

When used externally, these reports serve project managers, regulatory authorities, and other stakeholders by presenting compliance results in a tabulated format suitable for analysis and decision-making. External tabulated reports typically utilise MS Excel to present compliance data in a spreadsheet format. This format is useful for in-depth analysis and further data manipulation.

4.8.4 Visual Reports

Visual reports focus on enhancing the visual representation of compliance findings. They may employ various visual indicators to highlight compliances and non-compliances, making them easily detectable.

Within the BIM Model, visual reports help teams identify compliant and non-compliant elements directly through colour coding (overrides) and highlighting elements, as seen in Figure 69. This may be done by creating dedicated views within the model that emphasise specific compliance issues.

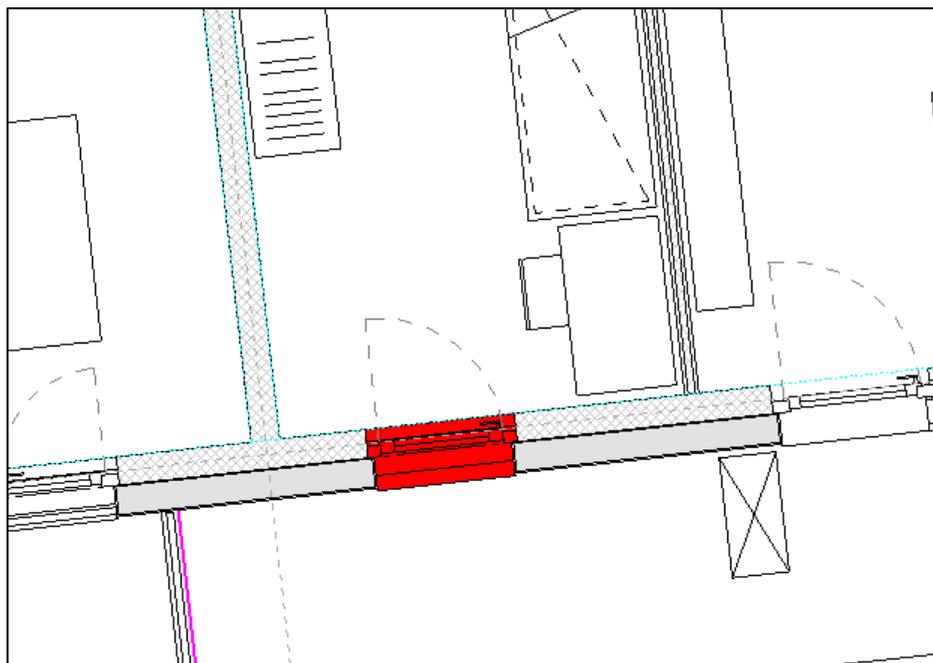


Figure 69: Overridden Window within a Model Plan View

Externally, these visual aids are less common but useful for stakeholders who require a more graphical representation of compliance results. Visual reports for external use may involve the capture of screenshots or images from the BIM model, providing a visual representation of compliance status that non-BIM professionals can easily understand.

4.8.5 Cloud-based Reports

Cloud-based reporting may be done through BIM collaboration platforms to report compliance checks directly in the cloud model. It's particularly useful for real-time tracking and issue resolution.

Internally, cloud-based reports enhance collaboration among project teams within the BIM environment. Externally, stakeholders benefit from real-time access to compliance data, facilitating communication and decision-making. This approach is especially valuable for collaboration with external parties.

4.9 Continuous Refinement of the Framework

While the developed Framework provides a structured approach for automating code compliance checks in BIM, it's important to remember that it's not set in stone. It should be viewed as a flexible guide that can evolve to stay aligned with changing industry standards and technologies.

The ACCF should be integrated into an organisation's protocols or project-specific documents to ensure its practical use. This could be part of a Building Execution Plan (BEP) or an internal organisational protocol. This way, it becomes an integral part of project management, emphasising its importance in consistently conducting compliance checks.

It's essential to recognise that, as with any complex system, the Framework may have limitations and potential errors due to data quality issues or complex rule checking. Continuous improvement requires collaboration among stakeholders to improve its accuracy and effectiveness.

5. APPLICATION TO CASE STUDY

This chapter will focus on the developed Framework's Application to a real-world case study. Because of the academic nature of the present work and the collaboration with Jereb in Budja Arhitekti in the Slovenian context, the Case Study will be developed within the local rules and bylaws. Specifically, the Slovenian "*Rules on minimum technical requirements for the construction of residential buildings and apartments*" document will be used as the primary source of Rules to be Automated.

The framework will be applied in a selection of rules from the Slovenian Regulation for Residential Buildings and into an existing project provided by Jereb in Budja Arhitekti. Nevertheless, by creating a general workflow regarding regulations classifications, the objective is to apply it to any context by customising it according to the regulatory environment's criteria or fundamental concepts. The present case study will serve as a model for this.

Once the framework is implemented, it will then be validated and evaluated. It is crucial to assess automated compliance checks' accuracy, efficiency, and reliability. This phase will involve testing and examination to determine the framework's effectiveness in identifying compliance deviations, potential errors, and areas for improvement

5.1 Slovenian Regulation for Residential Building (SRRB)

The Slovenian Regulation for Residential Buildings translated as "*Rules on minimum technical requirements for the construction of residential buildings and apartments*" (or "*Pravilnik o minimalnih tehničnih zahtevah za graditev stanovanjskih stavb in stanovanj*" in its original Slovene) is the current Code that every Residential Construction must adhere to in the country [36].

The document was first published in January of 2011, and have had several amendments and modifications since. It consists of 27 Articles (of which two have been deleted), each with different requirements for spatial and functional compliance.

The full Code translated into English can be found in Annex B, and a summary table with all the Articles will be displayed as follows:

Article (#)	General Purpose	N° of Rules
Article 1	Editing Area	2
Article 2	deleted	
Article 3	deleted	
Article 4	Entrance to multi-apartment buildings	3
Article 5	Common areas in multi-apartment buildings	3
Article 6	Safety of use	2
Article 7	Measuring points in multi-apartment buildings	2
Article 8	Elevators in multi-apartment buildings	4
Article 9	Size of garages and parking spaces	2
Article 10	Parts of apartments	4
Article 11	Installations and equipment in the apartment	1
Article 12	Room height	3
Article 13	Room width	1
Article 14	Apartment lighting	8
Article 15	Opening windows and doors	1
Article 16	Security of window openings	1
Article 17	Entrance to the apartment	1
Article 18	Transitions	1
Article 19	The width of the staircase in the apartment	1
Article 20	Beds	1
Article 21	Food preparation area	1
Article 22	Rooms for personal hygiene	5
Article 23	Surface treatment	1
Article 24	Minimum room equipment	10
Article 25	Equipment offsets	4
Article 26	Design requirements	3
Article 27	Supervision	1

Table 5: Slovenian Residential Building Code Articles

For this dissertation, three rules will be selected for application in the Case Study. The selection will be made to have a broad range of rule types to prove the Framework's effectiveness. Nevertheless, as requested by the Jereb team in Budja Architect, the selected rules will be Room-related. This will also simplify the Model Preparation Process.

The selected rules were the following, including Dimensional, Area, and Validation Rule Types, respectively:

1. Article 12: Room Height.

“(1) The clear height of residential premises is at least 2.50 meters, except for the vestibule and rooms from the second and third paragraphs of Article 10 of these regulations, where the clear height must not be lower than 2.20 meters.”

2. Article 14: Apartment lighting.

“(2) Direct lighting is achieved if the total area of treated masonry openings (in this case, only that part of the opening that is more than 0.50 meters above the finished floor) intended for lighting reaches at least 20 per cent of the net floor area of these parts of the apartment.”

3. Article 20: Beds.

“A maximum of two beds may be provided in a room or part of a room intended for living or sleeping. An exception is a bed for an infant, which is considered a temporary bed.”

5.2 Case Study Selection: Podbreznik B7-B9

For the case study, an on-going project in Jereb in Budja Arhitekti was selected to be checked. It consists of two buildings -B7 and B9- that together are the second stage of a residential complex in which the first stage was buildings B1, B3, and B5. As they share an underground garage, both buildings were modelled in the same file. Building B7 consists of 40 apartments, while B9 has 41

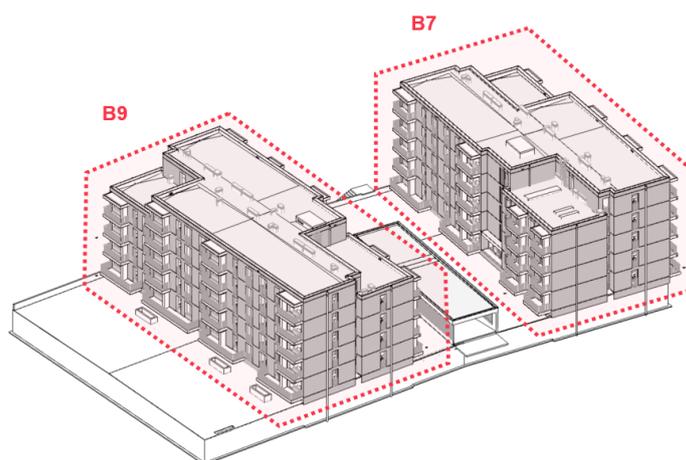


Figure 70: Case Study presentation: Podbreznik B7-B9

The project is currently being developed in its Detail Design Stage, so the checks performed will be the last before final delivery. This also means that not many incompliances are expected to be found.

5.3 Model Preparation

As mentioned earlier, the application will be focused on Regulation regarding Rooms for simplification purposes. This will make the Model Preparation step more streamlined and straightforward. First, a general Model Checker inspection will be done, and then VPL tool graphs will be used to try and solve potential issues.

5.3.1 Model Checker

The first identified need is to detect whether the modelled elements (Rooms) have adequate metadata for them to be detected by the automated tool. Custom Checking Sets were configured for this initial step using Model Checker Configurator's Wizard (Figure 71). Here, the wizard will help build the Check Sets to be run afterwards.

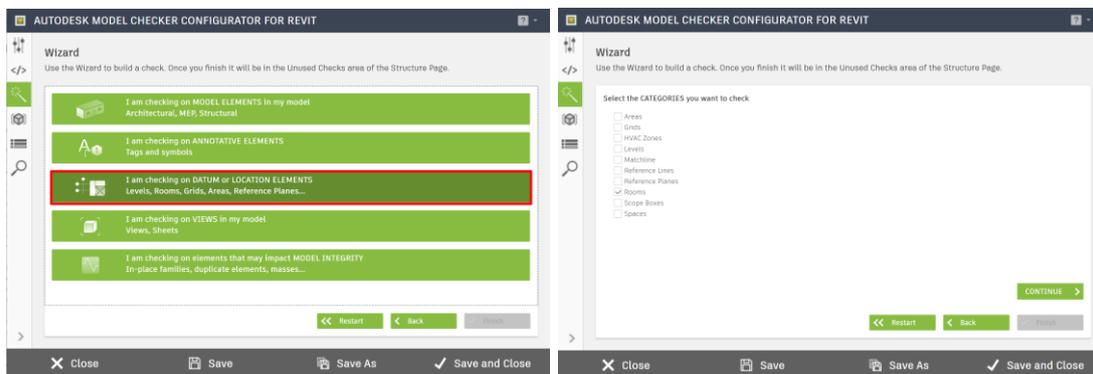


Figure 71: Model Preparation's Check Set Configuration for Rooms

Then, by adding filters in the Check Set, the required elements can be inspected based on parameters and conditional statements (Figure 72). Various checks can be added within the same Check Set in different Sections and Headings.

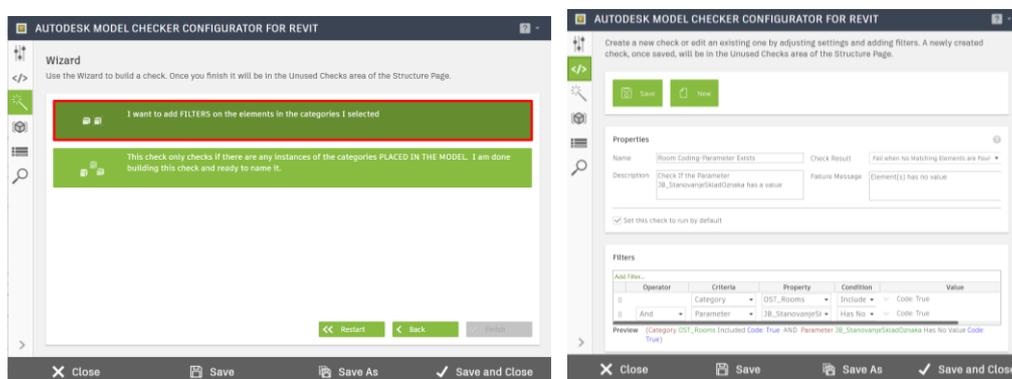


Figure 72: Model Preparation's Parameter Filtering for Rooms

Then, the Check was run, and a Report was emitted. It could be seen that the Rooms didn't have the Required parameters for the automated compliance check to be developed.

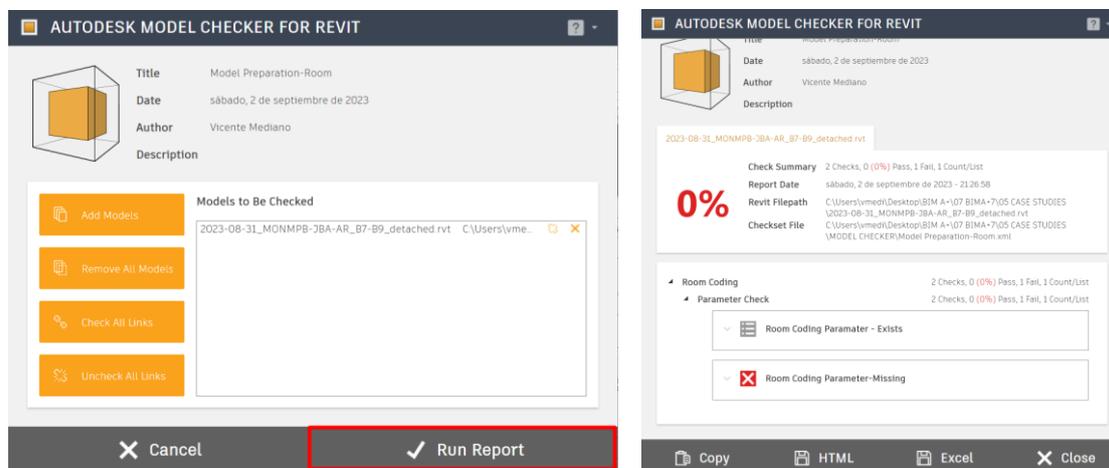


Figure 73: Model Checker Report Run

The issue was also detected by manually selecting them in the Model, where Rooms had no parameter metadata to identify them. This means that both the Automated check will not work, and Architectural and technical information will not be able to be managed (for scheduling or tagging them in Floor Plans and Details).

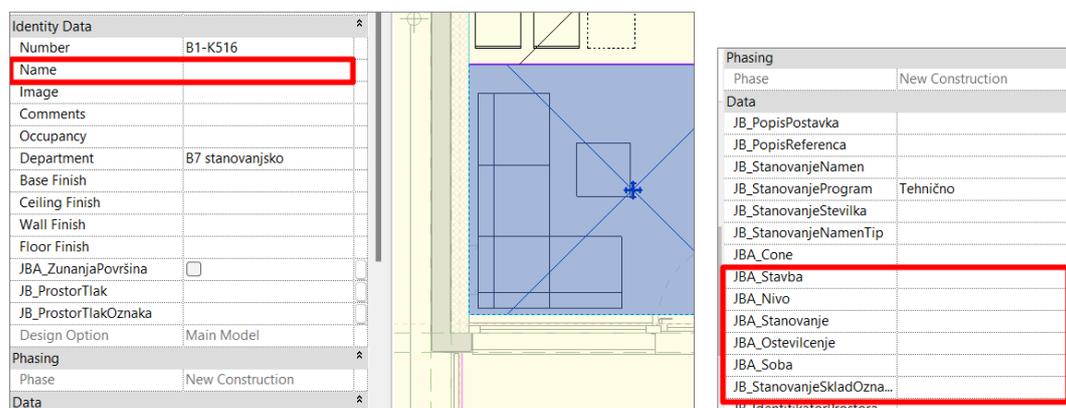


Figure 74: Missing Metadata in Rooms

As the Framework Definition mentions, Model Checker only identifies issues against filters or groups of conditional statements. Therefore, to address these issues, automated tools can also be developed. Even though, in some cases, where a small part of the information may be filled in, manual attempts could be made. Additionally, both could be combined to have a more efficient parameter population.

5.3.2 Dynamo

As an internal modelling agreement, both inside the office and within external stakeholders, a Coding System for identifying Apartment Rooms was implemented in Jereb in Budja Arhitekti. This code is a unique identifier per apartment unit that helps manage rooms easily within the Model.

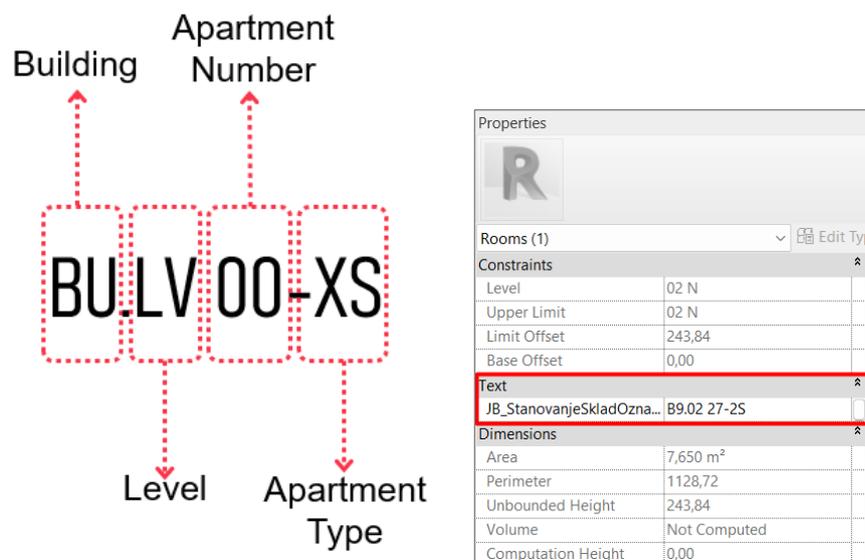


Figure 75: Apartment Rooms Coding System Format

The code displays relevant information within the apartment the room is in, like Building, Level, Apartment Number, and Apartment Type (Figure 75). Each of the categories has its parameter in which the information can be filled in (“*JB_Stanovanje*”, “*JB_Nivo*”, “*JB_Stanovanje*”, and “*JB_StanovanjeTip*”, respectively). Every room inside a single apartment should have this same code. Additionally, each room possesses an identification number correlative to the other rooms in the unit under a parameter named “*JB_Ostevilcenje*”.

Nevertheless, the first parameter that needed to be filled in was the “Name” parameter. This was done by developing a Dynamo graph that identified the different characteristic elements inside each Room and the Name abbreviation parameter. The “*JBA_Soba*” parameter was also filled in, as seen in the example for Main Bedroom in Figure 76.

The process was repeated for all the different types of Rooms, as seen in Figure 77.

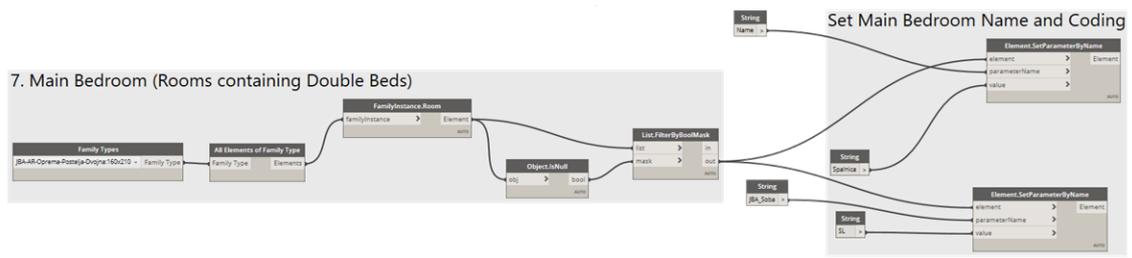


Figure 76: Room Naming Graph - Example for Main Bedroom

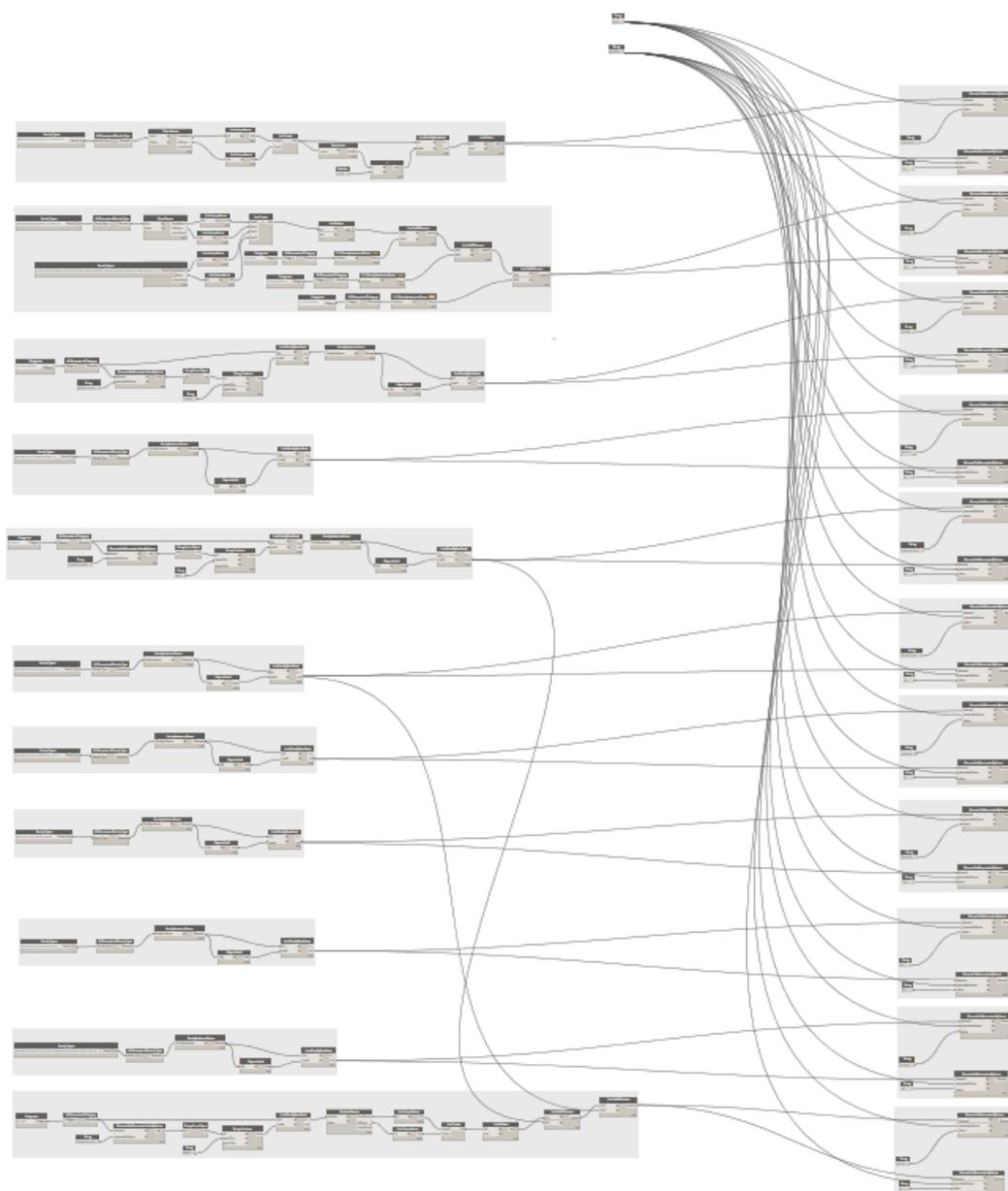


Figure 77: Room Naming Complete Graph

A similar logic was used for the different needed parameters, building several graphs, as seen in Figure 78 and Figure 79. For the rest of the more simple-to-populate parameters (like Level or Apartment Type), schedules were used.

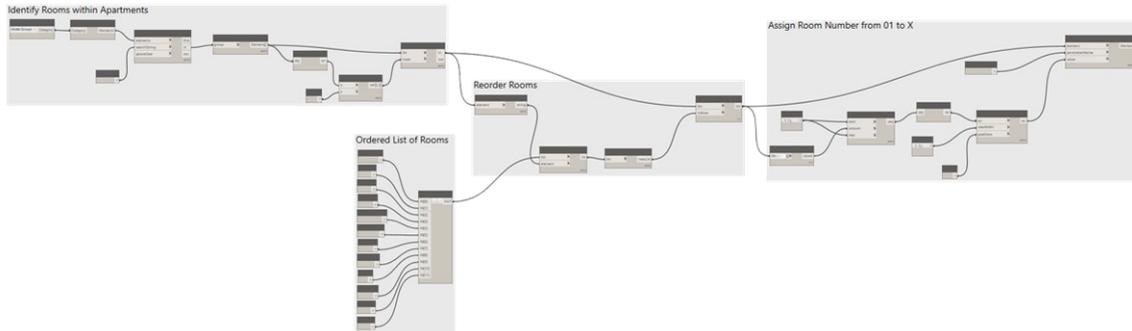


Figure 78: Room Numbering Graph

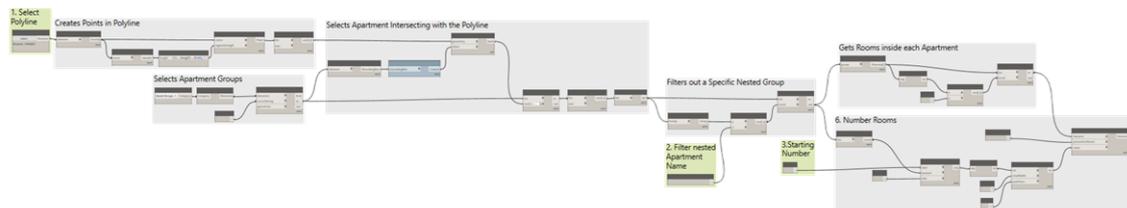


Figure 79: Room Apartment Numbering Graph

With all the stand-alone parameters, the Room Apartment Coding parameter was populated.

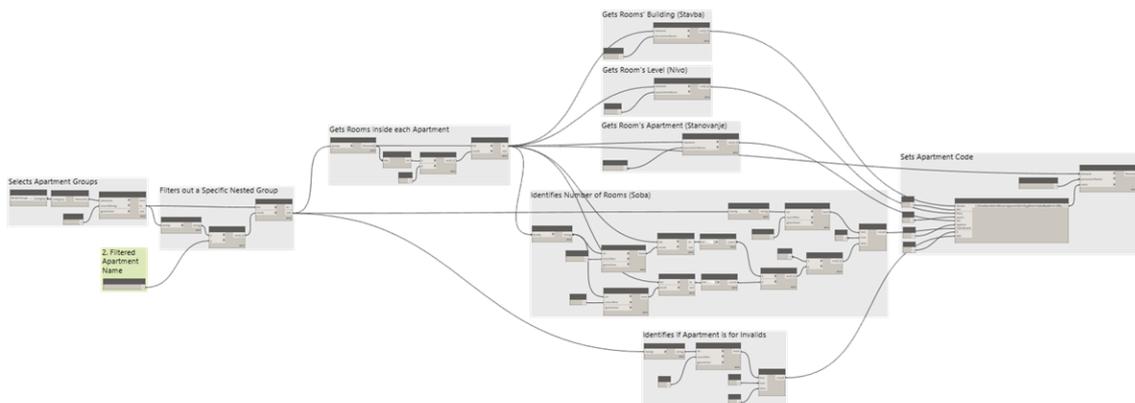


Figure 80: Room Apartment Coding Graph

Finally, the Model Checker was run again with the same Check Set to confirm that all the Rooms had consistent and precise information towards the development and execution of the Automated Compliance Checks.

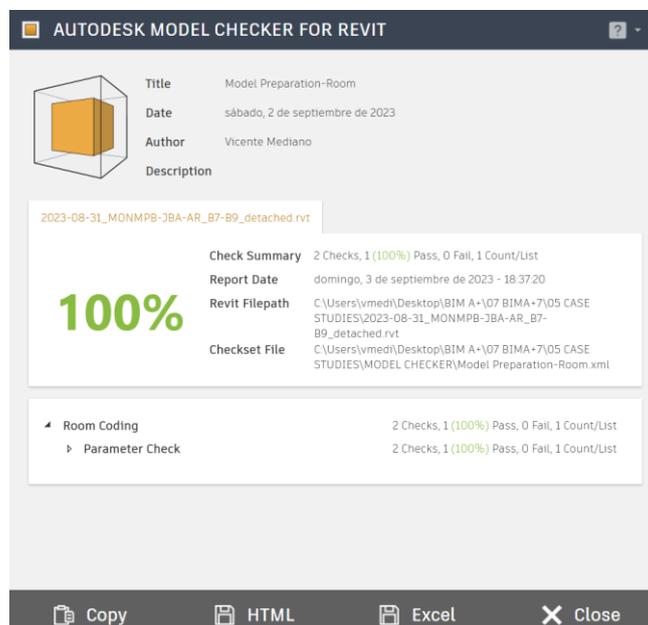


Figure 81: Model Checker Final Report Run.

As seen in Figure 81, the Selected rules could now be developed with all the desired rooms having adequate information.

5.4 Case Study: Article 12, Rule N°1

5.4.1 Rule Codification

Mandatory Fields:

- **Domain:** AR (Architecture)
- **Category:** SPA (Spaces)
- **Rule ID:** PrSt.12.1
 - Document Name: PrSt (“Pravilnik” ...”Stanovanj”).
 - Rule Number: 12.1 (Article 12, Rule N°1)

Optional Fields:

- **Type of Rule:** DIM (Dimensional)
- **Compliance Framework:** PRSC (Prescriptive)

Encoded Rule: AR-SPA-PrSt.12.1-DIM-PRSC

5.4.2 Rule Classification

AR-SPA-PrSt.12.1-DIM-PRSC			
Identification Characteristics		Content Characteristics	
Domain	Architecture	Description	
Category	Spaces	<u>Room Height:</u> "(1)The clear height of residential premises is at least 2.50 meters, except for the vestibule and rooms from the second and third paragraphs of Article 10 of these regulations, where the clear height must not be lower than 2.20 meters.	
Country	Slovenia		
Document Type	Building Code		
Document Name	Rules on minimum technical requirements for the construction of residential buildings and apartments*		
Document Year	2011	Type of Rule	Compliance Framework
Rule ID or Number	Article 12.1	Dimensional	Prescriptive
Translation Characteristics			
Compliance Check Methods	Model Elements	VPL Nodes	Observations
Filter Rooms by Names into two Groups. Compare height with the specified criteria.	Rooms	Element.Name; Room.Height; (Math Operator) >=	-

*Original Document Name: Pravilnik o minimalnih tehničnih zahtevah za graditev stanovanjskih stavb in stanovanj

Table 6: SRRB Article 12, Rule N°1 Classification Table

5.4.3 Rule Interpretation

Rule Interpretation in Phind using RASE Methodology:

Requirement: "The clear height of residential premises is at least 2.50 meters."

Application: "for the vestibule and rooms from the second and third paragraphs of Article 10 of these regulations."

Selection: "The clear height of the rooms."

Exception: "Storage and Bathrooms must not be lower than 2.20 meters."

Pseudo-code:

```
# Import Revit API
import RevitAPI

# Define the minimum heights
min_height = 2.50
exception_height = 2.20

# Get all residential premises
premises = RevitAPI.GetElementsByCategory("Residential Premises")

# Iterate through each premise
for premise in premises:
    # Get the clear height
    clear_height = premise.GetParameter("Clear Height").AsDouble()

    # Check if premise is a vestibule or a room from Article 10
    if premise.Name == "Vestibule" or premise.Name in ["Room2Article10",
"Room3Article10"]:
        # Check if clear height is less than the exception height
        if clear_height < exception_height:
            # Print an error message
            print(f"Error: {premise.Name} has a clear height less than
{exception_height} meters.")
        else:
            # Check if Clear height is less than the minimum height
            if clear_height < min_height:
                # Print an error message
                print(f"Error: {premise.Name} has a clear height less than
{min_height} meters.")
```

5.4.4 Rule Translation

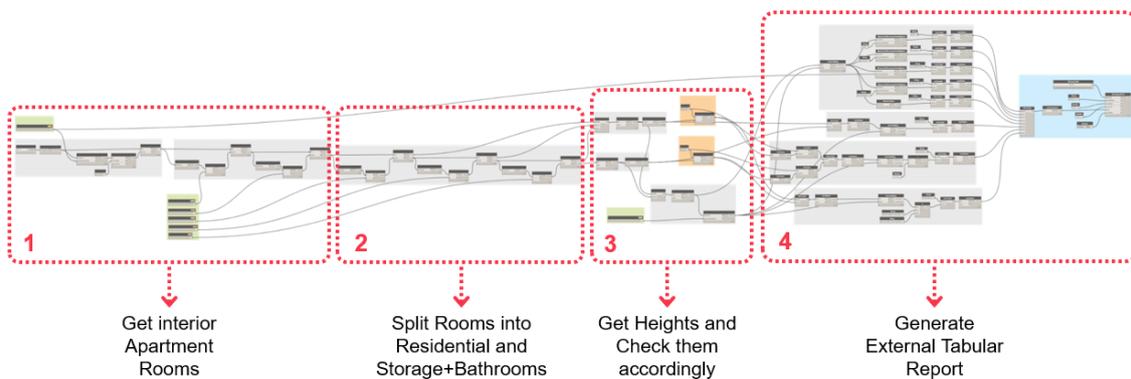


Figure 82: Article 12, Rule N°1 Dynamo Graph – Complete

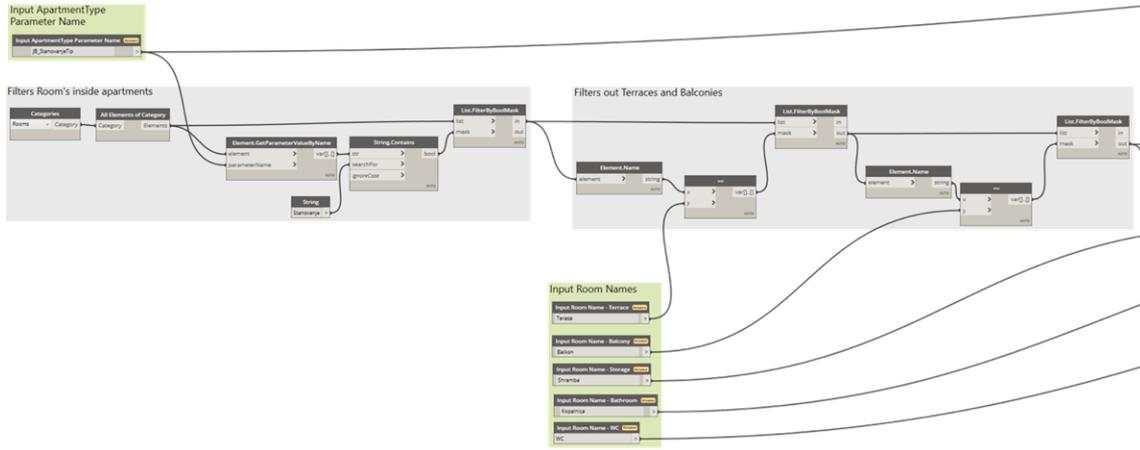


Figure 83: Article 12, Rule N°1 Dynamo Graph – Section 1

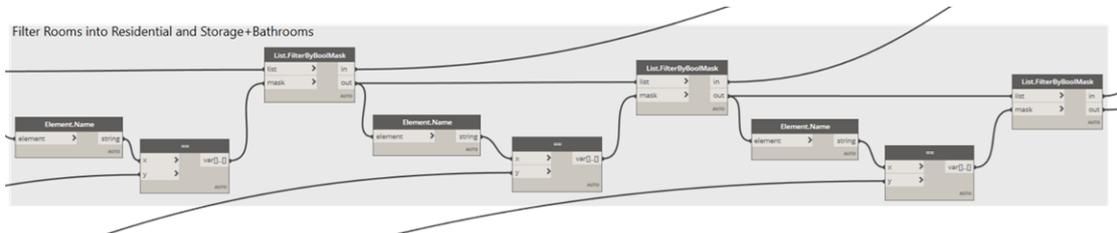


Figure 84: Article 12, Rule N°1 Dynamo Graph – Section 2

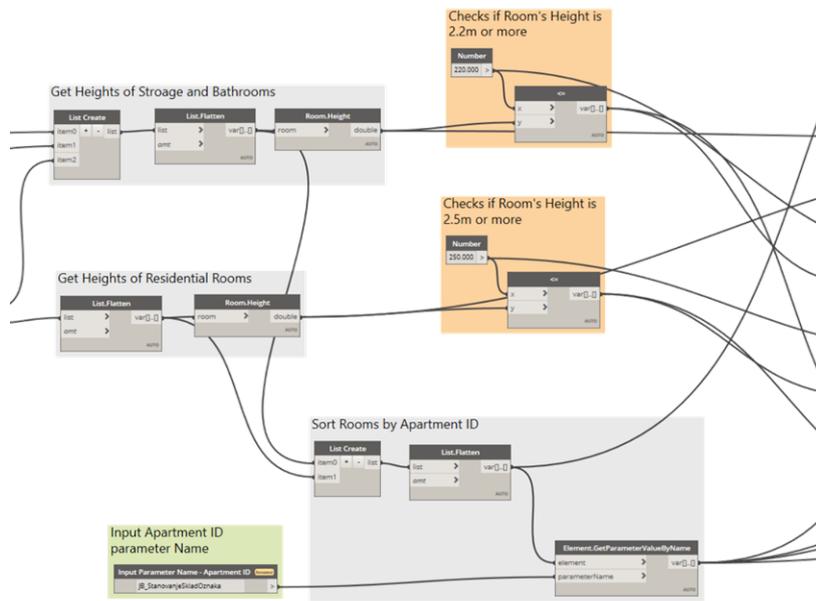


Figure 85: Article 12, Rule N°1 Dynamo Graph – Section 3

5.4.5 Rule Execution

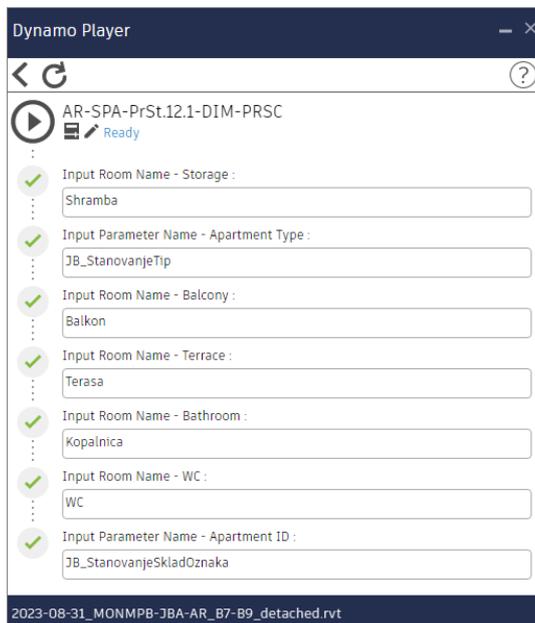


Figure 86: Article 12, Rule N°1 Dynamo Player Settings

5.4.6 Checking Results Reporting

Creation of External and Tabular Compliance Report: Excel Sheet listing all rooms, indicating their compliance according to their requirements, and including relevant metadata for identification.

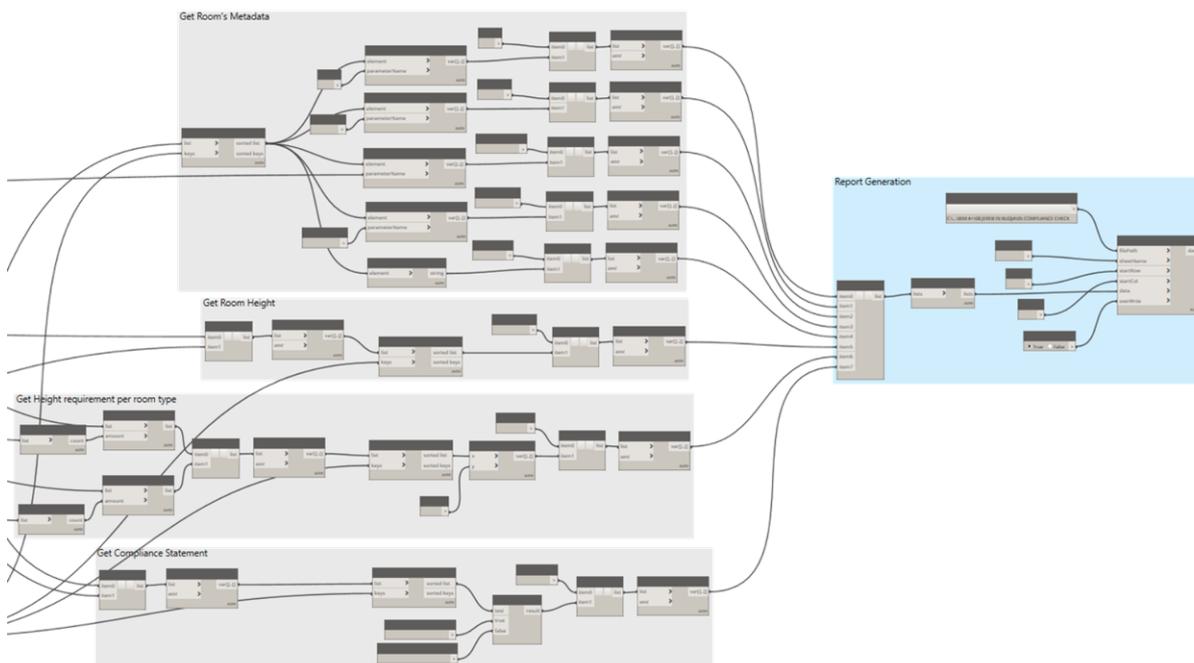


Figure 87: Article 12, Rule N°1 - Compliance Report's Dynamo Graph

LEVEL	BUILDING	APARTMENT TYPE	APARTMENT N°	ROOM NAME	ROOM HEIGHT	MIN HEIGHT	COMPLIANCE
04 N	B7	Stanovanje C Stranski	37	Hodnik	2,54	2,5	Room's Height is compliant
04 N	B7	Stanovanje C Stranski	37	Dnevni prostor	2,54	2,5	Room's Height is compliant
04 N	B7	Stanovanje C Stranski	37	Predprostor	2,54	2,5	Room's Height is compliant
04 N	B7	Stanovanje C Stranski	37	Kuhinja	2,54	2,5	Room's Height is compliant
04 N	B7	Stanovanje E+	38	Shramba	2,44	2,2	Room's Height is compliant
04 N	B7	Stanovanje E+	38	Kopalnica	2,44	2,2	Room's Height is compliant
04 N	B7	Stanovanje E+	38	WC	2,44	2,2	Room's Height is compliant
04 N	B7	Stanovanje E+	38	Otroška soba	2,44	2,5	Room's Height is not compliant
04 N	B7	Stanovanje E+	38	Otroška soba	2,44	2,5	Room's Height is not compliant
04 N	B7	Stanovanje E+	38	Kuhinja	2,44	2,5	Room's Height is not compliant
04 N	B7	Stanovanje E+	38	Spalnica	2,44	2,5	Room's Height is not compliant
04 N	B7	Stanovanje E+	38	Hodnik	2,44	2,5	Room's Height is not compliant
04 N	B7	Stanovanje E+	38	Predprostor	2,44	2,5	Room's Height is not compliant
04 N	B7	Stanovanje E+	38	Jedilnica	2,44	2,5	Room's Height is not compliant
04 N	B7	Stanovanje E+	38	Dnevni prostor	2,44	2,5	Room's Height is not compliant
04 N	B7	Stanovanje C+ stranski	39	Shramba	2,54	2,2	Room's Height is compliant
04 N	B7	Stanovanje C+ stranski	39	Kopalnica	2,54	2,2	Room's Height is compliant
04 N	B7	Stanovanje C+ stranski	39	WC	2,54	2,2	Room's Height is compliant
04 N	B7	Stanovanje C+ stranski	39	Dnevni prostor	2,54	2,5	Room's Height is compliant
04 N	B7	Stanovanje C+ stranski	39	Predprostor	2,54	2,5	Room's Height is compliant
04 N	B7	Stanovanje C+ stranski	39	Spalnica	2,54	2,5	Room's Height is compliant
04 N	B7	Stanovanje C+ stranski	39	Otroška soba	2,54	2,5	Room's Height is compliant

Figure 88: Article 12, Rule N°1 - Compliance Report

5.5 Case Study: Article 14, Rule N°2

5.5.1 Rule Codification

Mandatory Fields:

- **Domain:** AR (Architecture)
- **Category:** LGT (Lighting)
- **Rule ID:** PrSt.14.2
 - Document Name: PrSt ("Pravilnik" ... "Stanovanj").
 - Rule Number: 14.2 (Article 14, Rule N°2)

Optional Fields:

- **Type of Rule:** ARE (Area)
- **Compliance Framework:** PRSC (Prescriptive)

Encoded Rule: AR-LGT-PrSt.14.2-ARE-PRSC

5.5.2 Rule Classification

AR-LGT-PrSt.14.2-ARE-PRSC			
Identification Characteristics		Content Characteristics	
Domain	Architecture	Description	
Category	Lighting	Apartment Lighting: "(2) Direct lighting is achieved if the total area of treated masonry openings (in this case, only that part of the opening that is more than 0.50 meters above the finished floor) intended for lighting reaches at least 20 percent of the net floor area of these parts of the apartment."	
Country	Slovenia		
Document Type	Building Code		
Document Name	Rules on minimum technical requirements for the construction of residential buildings and apartments*		
		Type of Rule	Compliance Framework
Document Year	2011	Area	Prescriptive
Rule ID or Number	Article 14, Rule N°2		
Translation Characteristics			
Compliance Check Methods	Model Elements	VPL Nodes	Observations
Identify Interior rooms with windows. Get Room area and Window area. Subtract window area below 50cm. Compare areas and check 20%.	Windows, Rooms	Element.GetParameterValue, Window.Room, Room.Window, Room.Area	Not Rectangular openings (e.g., Glass doors with windows) will require additional steps.

*Original Document Name: Pravilnik o minimalnih tehničnih zahtevah za graditev stanovanjskih stavb in stanovanj

Table 7: SRRB Article 14, Rule N°2 Classification Table

5.5.3 Rule Interpretation

Rule Interpretation in Google Bard using RASE Methodology:

Requirement: Direct lighting is achieved if the total area of treated masonry openings

Application: Rooms intended for lighting reach at least 20 per cent of the net floor area of these parts of the apartment.

The application is the action or result required when the requirement is met. In this rule, the application is achieving direct lighting.

Selection: Only that part of the opening that is more than 0.50 meters above the finished floor

Exception: None.

Pseudo-code:

```

def is_direct_lighting_achieved(total_area_of_treated_masonry_openings,
net_floor_area):
    Returns True if direct lighting is achieved, False otherwise.

    Args:
        total_area_of_treated_masonry_openings: The total area of treated masonry
openings intended for lighting.
        net_floor_area: The net floor area of these parts of the apartment.

    Returns:
        True if direct lighting is achieved, False otherwise.

    # Get the area of the treated masonry openings that are more than 0.50 meters
above the finished floor.
    treated_masonry_openings_above_0_5_meters =
get_treated_masonry_openings_above_0_5_meters()
    treated_masonry_openings_area = treated_masonry_openings_above_0_5_meters.area

    # Get the net floor area.
    net_floor_area = get_net_floor_area()

    # Check if the total area of the treated masonry openings that are more than
0.50 meters above the finished floor is at least 20 percent of the net floor area.
    if treated_masonry_openings_area / net_floor_area >= 0.2:
        return True
    else:
        return False

```

5.5.4 Rule Translation

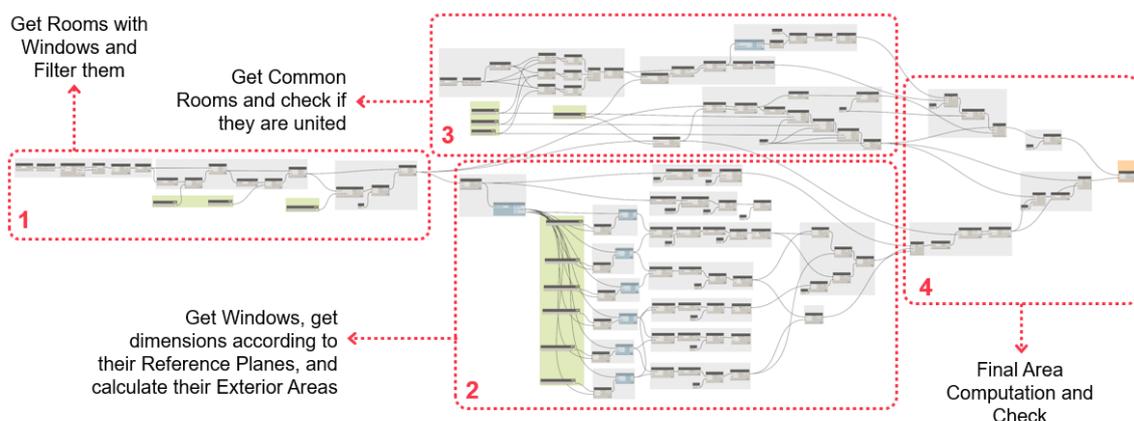


Figure 89: Article 14, Rule N°2 Dynamo Graph – Complete

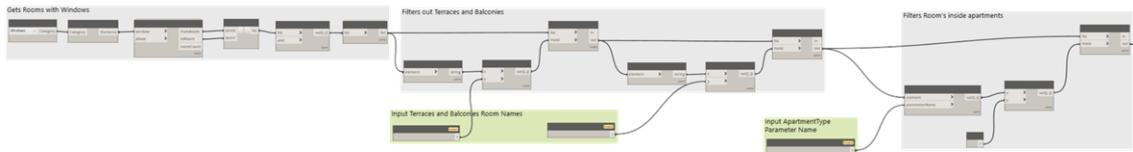


Figure 90: Article 14, Rule N°2 Dynamo Graph – Section 1

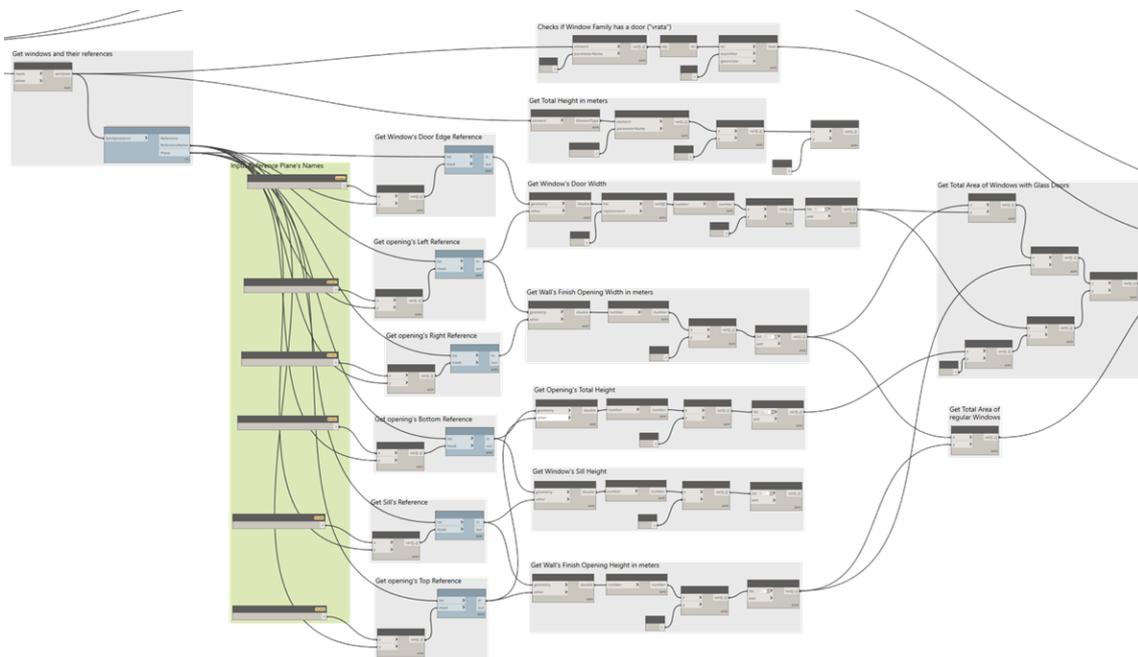


Figure 91: Article 14, Rule N°2 Dynamo Graph - Section 2

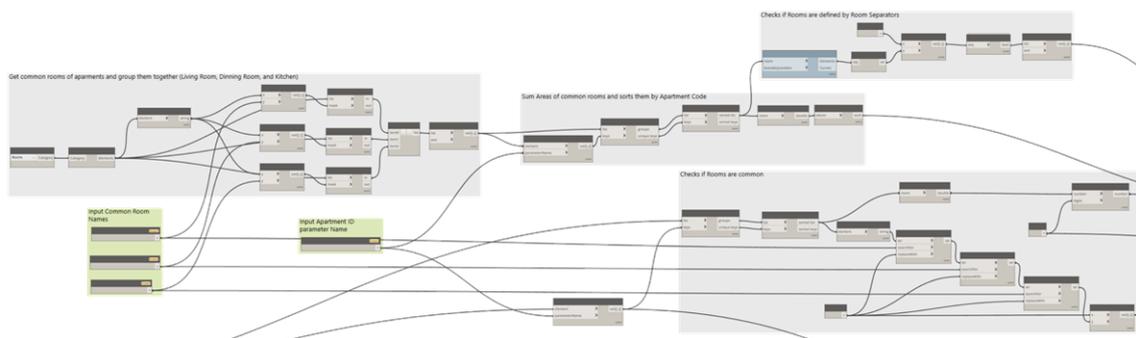


Figure 92: Article 14, Rule N°2 Dynamo Graph - Section 3

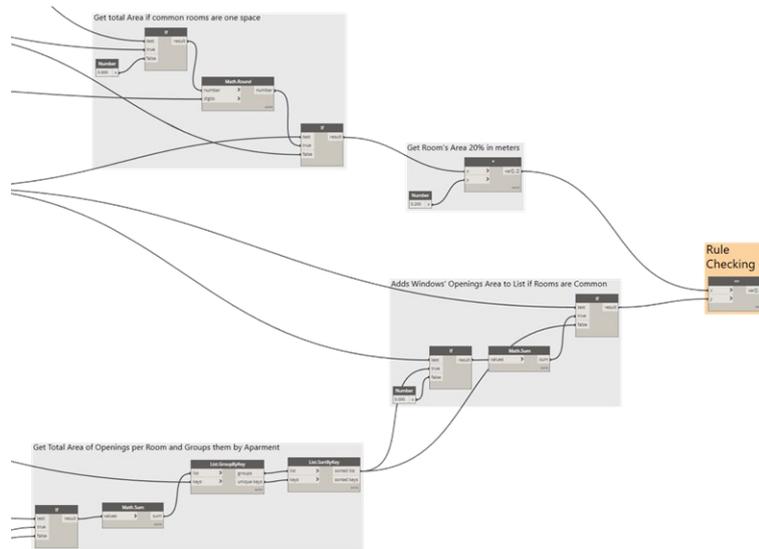


Figure 93: Article 14, Rule N°2 Dynamo Graph - Section 4

5.5.5 Rule Execution

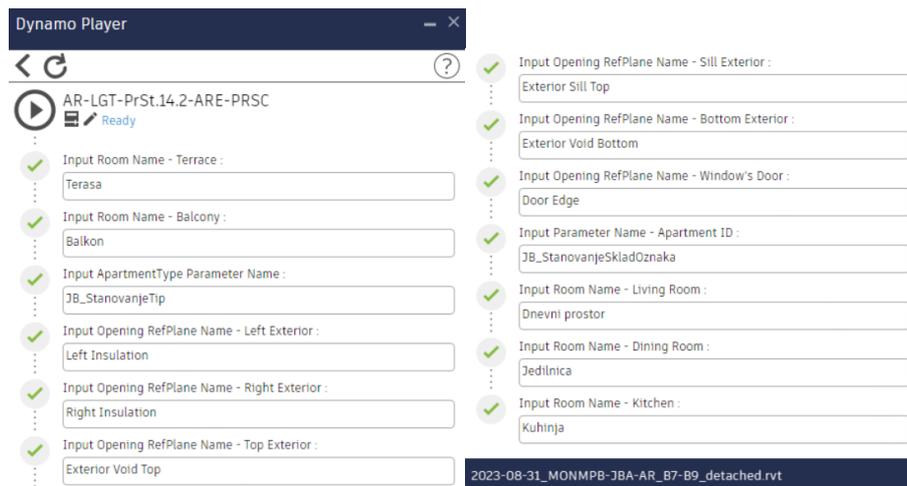


Figure 94: Article 14, Rule N°2 - Dynamo Player Settings

5.5.6 Checking Results Reporting

Creation of Internal and Visual Compliance Report: Plan Views with overridden Revision Clouds on each Window smaller than 20% of the room net area.

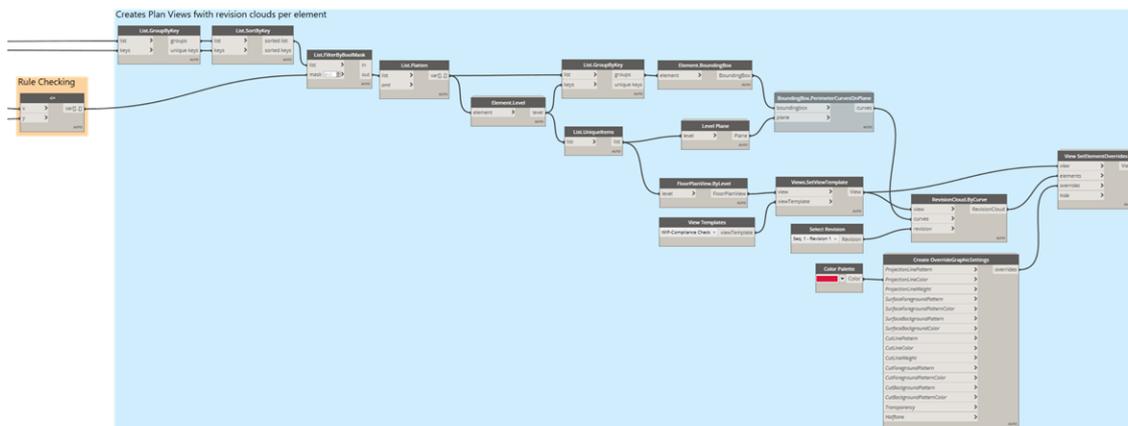


Figure 95: Article 14, Rule N°2 - Compliance Report's Dynamo Graph

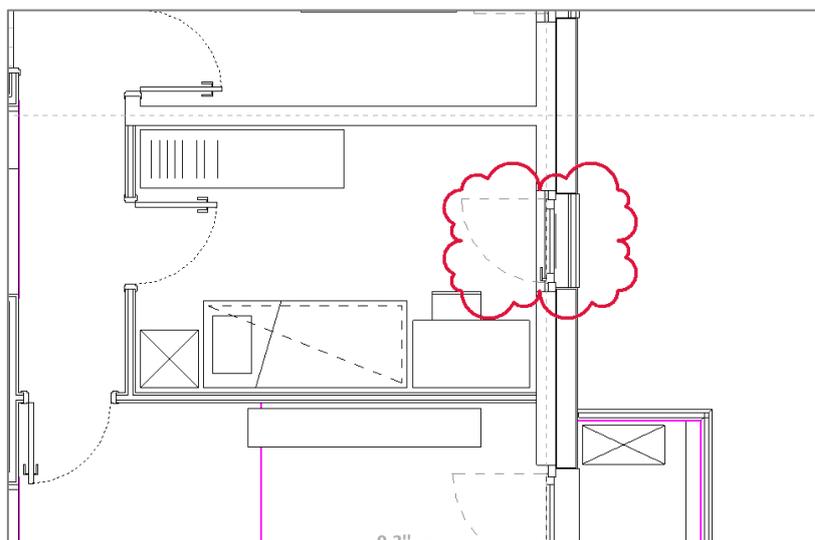


Figure 96: Article 14, Rule N°2 - Compliance Report

5.6 Case Study: Article 20

5.6.1 Rule Codification

Mandatory Fields:

- **Domain:** AR (Architecture)
- **Category:** SPA (Spaces)
- **Rule ID:** PrSt.20
 - Document Name: PrSt ("*Pravilnik*" ... "*Stanovanj*").
 - Rule Number: 20 (Article 20)

Optional Fields:

- **Type of Rule:** VAL (Validation)
- **Compliance Framework:** PRSC (Prescriptive)

Encoded Rule: AR-SPA-PrSt.20-VAL-PRSC

5.6.2 Rule Classification

AR-SPA-PrSt.20-VAL-PRSC			
Identification Characteristics		Content Characteristics	
Domain	Architecture	Description	
Category	Spaces	Beds: "They must be provided in the room or part of the room intended for living or sleeping maximum two beds. An exception is a bed for an infant, which is considered a temporary bed."	
Country	Slovenia		
Document Type	Building Code		
Document Name	Rules on minimum technical requirements for the construction of residential buildings and apartments*		
		Type of Rule	Compliance Framework
Document Year	2011	Validation	Prescriptive
Rule ID or Number	Article 20		
Translation Characteristics			
Compliance Check Methods	Model Elements	VPL Nodes	Observations
Filter rooms by their category or type. Count the number of beds in each room. Compare the bed count against the specified maximum (2 beds).	Rooms, Furniture.	Element.Location, Room.IsInsideRoom, Filter.ByBoolMask, Element.Count, (Math Operator) <	Two check are needed: If beds are in rooms intended for living, and if there are 2 or less beds on each.

*Original Document Name: Pravilnik o minimalnih tehničnih zahtevah za graditev stanovanjskih stavb in stanovanj

Table 8: SRRB Article 20 Classification Table

5.6.3 Rule Interpretation

Rule Interpretation of Chat GPT using RASE Methodology:

Requirement: Beds must be provided in a room's designated living or sleeping area. A maximum of two beds are allowed in this area.

Application: This rule applies to providing beds within a room or part of the room intended for living or sleeping. It places a limit of two beds in such areas.

Selection: When designing rooms or spaces for living or sleeping, ensure that you include beds in those areas. Ensure that the number of beds in these areas does not exceed two.

Exception: An exception is made for an infant's bed. An infant's bed is considered temporary and is not counted towards the maximum of two beds.

Pseudo-code:

```
function DesignRoomWithBeds(room):  
  
    // Requirement: Beds must be provided in the living or sleeping area  
    if room.intendedUse == "living" or room.intendedUse == "sleeping":  
        bedsCount = 0  
        infantBedPresent = false  
  
        // Application: Maximum of two beds  
        for each furnitureItem in room.furniture:  
            if furnitureItem.type == "bed":  
                if bedsCount < 2:  
                    bedsCount += 1  
                else:  
                    raise Exception("Maximum of two beds allowed.")  
  
            // Exception: Infant's bed  
            if furnitureItem.type == "infant_bed":  
                infantBedPresent = true  
  
        // Selection: Design the room with beds  
        if bedsCount > 0:  
            if infantBedPresent:  
                displayMessage("Design successful with beds for infants and up to  
two beds.")  
            else:  
                displayMessage("Design successful with up to two beds.")  
        else:  
            displayMessage("Design should include at least one bed.")  
    else:  
        displayMessage("Design does not require beds for this type of room.")
```

5.6.4 Rule Translation

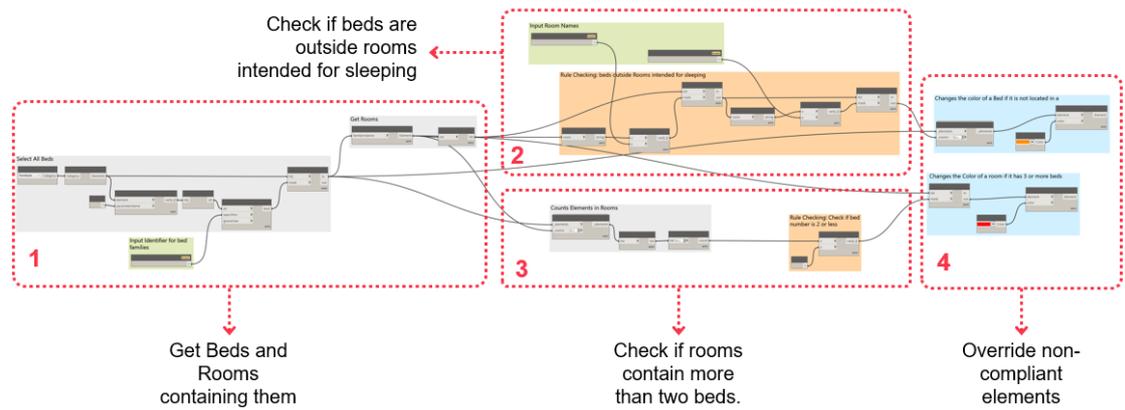


Figure 97: Article 20 Dynamo Graph – Complete

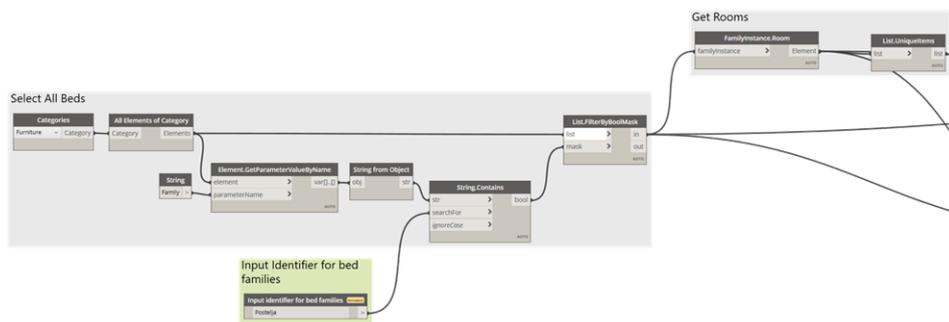


Figure 98: Article 20 Dynamo Graph – Section 1

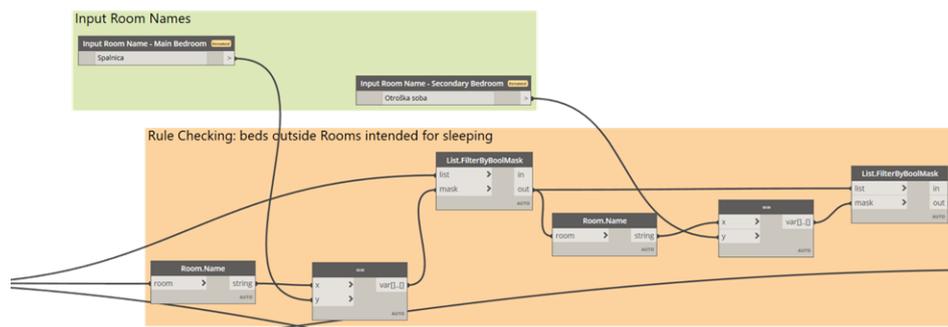


Figure 99: Article 20 Dynamo Graph – Section 2

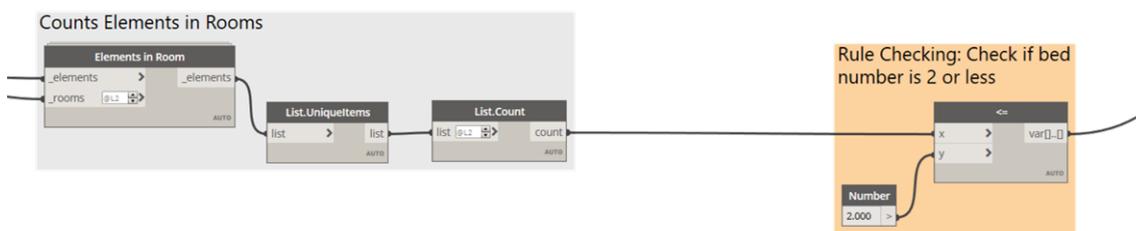


Figure 100: Article 20 Dynamo Graph – Section 3

5.6.5 Rule Execution



Figure 101: Article 20 - Dynamo Player Settings

5.6.6 Checking Results Reporting

Creation of Internal and Visual Compliance Report: Overridden Elements in View

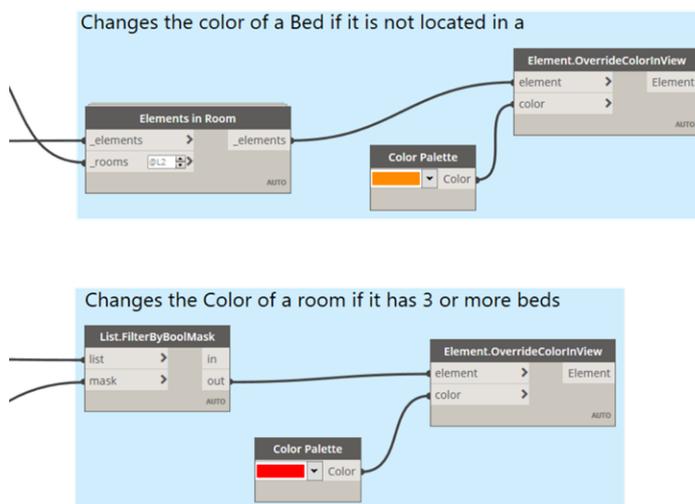


Figure 102: Article 20 - Compliance Report's Dynamo Graph

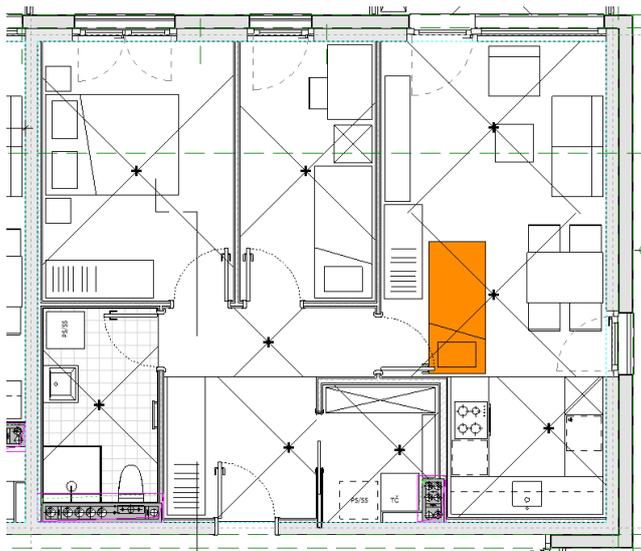


Figure 103: Article 20 - Compliance Report #1

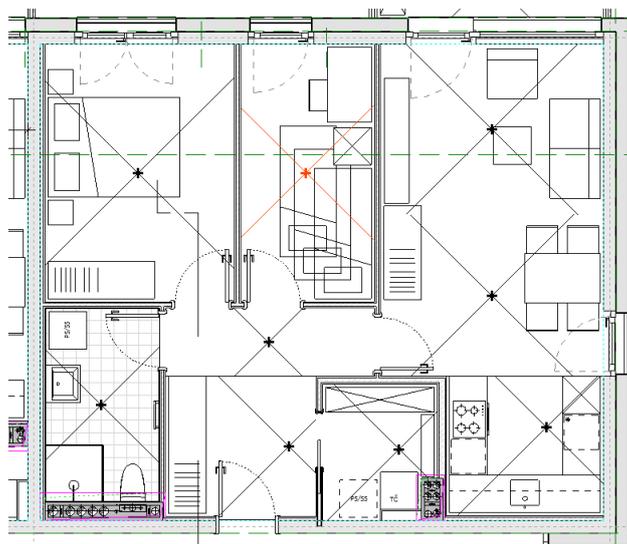


Figure 104: Article 20 - Compliance Report #2

6. RESULTS AND DISCUSSION

6.1 Case Study Analysis

6.1.1 Article 12.1

This was a simple compliance check, so no major complications were found in the graph's development. The Classification Table and Pseudo Code were not that useful. Nevertheless, it was discovered that some factors must be considered when working with datum elements, such as Rooms.

Generally, Rooms are modelled in Floor Plans, but their Height is usually not verified. This becomes a complication when building elements are not set to be "Room Bounding" or the "Areas and Volumes" computation options are not turned on. Thus, the importance of Modelling Standards and Model Preparation phase become more relevant.

This also highlights the importance of VPL and Automated Compliance Checks regarding Quality Assurance and Control. The benefit of having a direct integration and relationship of modelled elements and what is being shown in the VPL tool is that it is easy to detect and even address modelling and metadata issues immediately. Additionally, as Automated Compliance Check needs this type of consistency to be present, it further increases the quality and correctness of BIM Models.

6.1.2 Article 14.2

This regulation's compliance check was rather complicated, as seen in the Dynamo graph's complexity in Figure 89. Because of this, as expected, there were various challenges to address, and several findings must be considered. Using the Classification Table and the Pseudo Code was crucial for the tool's development and debugging.

One of the main issues regarding this checking tool was that windows are usually not modelled with the same criteria. This meant that the families had to be modified to extract geometrical information (Width and Height of the external Voids that cut the insulation).

As those Voids were locked into Reference Planes, for the VPL tool to read them thoroughly, they had to have a consistent naming convention, which was not the case. Each window family had to be checked, and their Reference Planes were renamed accordingly.

This problem is strictly related to Model Preparation. Nevertheless, as it is a fine granularity issue located at a deep level within the Project's elements and metadata

(Project → Families → Window → Reference Plane → Name), it becomes hard to detect with a generic check. This suggests the idea of generating specific Model Preparation checks per rule,

Also, it became evident that the lack of modelling standards undermines the efforts for automation. When Models are developed without common considerations or agreements, most likely, no consistency is reached, even within the same organisation. The checking tool will present errors in parameter identification, relationships between elements, geometric relations, and referencing, among many others.

Moreover, it is detected that different approximations for the same compliance checking tool may be developed depending on the types of families being checked, their geometry, and how they were modelled. It is unlikely to develop a universal tool that considers all the factors different models and projects may have on the first try. Thus, constant improvement of tools becomes crucial in Automated Compliance Checks.

Finally, it is observed that this kind of regulation may be ideal for experimenting with technologies like Generative Design. With inputs like the Window's Dimensions and the elements that determine the Room's Area, new design options may be created, such as generating new Window Families that comply with a specific room or altering the room's dimensions to achieve compliance.

6.1.3 Article 20

As in Article 12.1, no major complication was found in the development of this graph. Nevertheless, the anticipated identification of the two checking instances in the Classification Table was vital for a clearer idea of how to face the general graph structure.

Additionally, Custom Packages in Dynamo considerably simplified the efforts to detect the elements inside the Rooms, which was achieved with only one Node, called "Elements in Room" by Archilab.

It is noted that this kind of graph could be used not only for this specific regulation but also for detecting modelling errors or oversights, which can be especially frequent when many stakeholders manage and modify a BIM Model. As seen in Figure 103, an element—in this case, a bed—could have been moved or duplicated by mistake into another room highlighted in the Model—this helps achieve consistency, Quality Assurance, and Quality Control of Models.

Finally, regarding the Compliance Check Report, it is important to mention that a simple colour override in the view was more than enough to showcase non-compliance. Because of the purpose of the rule and the effortless methods needed to solve non-compliances, the Modellers would need to check the highlighted elements in a view and delete or move Furniture instances. This emphasises that the Reports' complexity should be adequate to the Rules' complexity and purpose.

6.2 Findings from the Automated Compliance Checking implementation.

The automated compliance check framework proved its practicality, efficiency, and adaptability. It streamlined the process, demonstrating enhanced speed and flexibility compared to manual checks. Real-time rule validation emerged as a notable advancement in the process.

The primary advantage of automated tools is evident in the upfront investment required for script development. While creating a robust script demands time and dedication, it offers a one-time effort that continuously improves efficiency and accuracy across projects. In contrast, manual checks are labour-intensive and repetitive, needing to be conducted for each regulation request and project modification.

Moreover, employing various methods such as natural language processing, rule extraction algorithms, and semantic analysis enabled a systematic approach to understanding and translating rules into machine-readable formats. This structured procedure dissected rules into core attributes, including Domain, Category, Type, and Compliance Framework, structured into groups like Identification, Content, and Translation Characteristics.

Additionally, proposing Visual Programming Language as a tool for designing white-box solutions promoted transparency and customizability. This builds trust and allows users to easily understand and modify a compliance-checking system. Furthermore, VPL simplifies complex programming logic, making compliance tools accessible to professionals without extensive coding expertise.

Finally, adaptability and flexibility were achieved by designing a general Framework incorporating adjustable parameters, rule templates, and dynamic algorithms. This capacity allows the framework to be applied in different contexts, regardless of Discipline, Building Lifecycle stage, project requirements, and evolving regulations.

6.3 Analysis of the effectiveness and efficiency of the ACCC process

The ACCC process demonstrated remarkable effectiveness in addressing the challenges inherent in manual compliance checks. By automating the assessment of design elements against regulatory standards, the process significantly reduces the risk of oversight and non-compliance. Through systematic rule interpretation and validation, the ACCC process ensured that designs adhere to regulations consistently and accurately. This improved effectiveness reduces legal liabilities, minimises costly revisions, and enhances overall project quality.

Moreover, the real-time validation capability introduced by the ACCC process enables immediate feedback during the design phase. This iterative loop promotes proactive design adjustments, preventing potential compliance issues from growing cumulatively throughout the project lifecycle. Thus, the ACCC process not only enhances regulatory adherence but also facilitates a more agile and responsive design methodology.

As mentioned in the previous sub-chapter, the ACCC process surpassed manual checks in efficiency through script development and streamlining tasks. Once a script is developed, it can be applied repeatedly to various projects, resulting in consistent and accurate compliance assessments. This eliminates the need for redundant manual checks and reduces the potential for human error. The cumulative time saved over multiple projects showcases the inherent efficiency of the ACCC process.

Furthermore, automation eliminates the delays associated with waiting for manual compliance checks to be conducted, allowing the design process to proceed without unnecessary interruptions. This speed-up workflow enhances project timelines and contributes to efficient project delivery.

Finally, the ACCC process found a harmonious balance between automation and human expertise, efficiently handling routine compliance checks while retaining the necessity of human intervention for intricate rule interpretation, complex design scenarios, and crucial decision-making. This cooperative approach ensures that the automated process remains transparent and dependable, augmenting the capabilities of professionals. Analysing the process's effectiveness and efficiency revealed its potential for scalability and growth, with its adaptable framework applicable across various regulatory contexts, project types, and building lifecycle stages. The modular structure of ACCC facilitates the seamless integration of new rules and regulations, guaranteeing relevance in an ever-evolving regulatory landscape.

6.4 Challenges and limitations of the process

While implementing the Automated Code Compliance Checking (ACCC) process showcased its effectiveness and efficiency in streamlining compliance assessments, it's important to acknowledge certain challenges and limitations.

One of the most prominent challenges comes from the complex transition between conceptualisation and practical implementation in programming. The gap between envisioning how a compliance rule should be translated into machine-readable code and the intricate steps required for an accurate and efficient translation became evident. Often, initial ideas had to undergo several iterations to align with the specific syntax, data structures, and algorithms demanded by the digital environment despite having developed the Classification Table and Pseudo-coded rule. This phenomenon emphasises the need for a deep understanding of both the regulatory domain and the programming landscape, effectively bridging the conceptual and digital fields.

Another critical aspect to consider is the applicability of the automated compliance checking tools in diverse projects. While the framework demonstrates its value within specific contexts, questions appear around its adaptability across a range of projects. The importance of standardisation in modelling and model preparation becomes apparent, as consistency in data structure and attribute definitions greatly impacts the accuracy and efficiency of the automated interpretation process.

Additionally, the choice of tools used in the process introduces its challenges. For instance, using Excel for Classification Tables raises questions about its added value compared to alternatives like annotation families in Revit and potential database integration. While the first may be beneficial for its familiarity, flexibility, and ease of use, it lacks automation and scalability.

Furthermore, Visual Programming Language (VPL), while promoting transparency and ease of use, has disadvantages, particularly in the context of complex rules or more advanced programming operations, like looping, which is not supported by Dynamo Nodes alone. The framework's robustness could sometimes be considered an "overkill" for simpler rules, while advanced users might find certain proposed methods unnecessary and prefer direct coding.

Debugging and troubleshooting also emerged as significant challenges during the implementation. Identifying and rectifying errors, especially in intricate rules, can be time-consuming and demanding. Additionally, the hybrid nature of the process, combining

human expertise with automation, raises questions about the reliability and consistency of results, particularly when interpreting more complex or intricate rules.

7. CONCLUSIONS

7.1 Summary of the study and main findings

This work addressed the challenges of compliance checking within the AECO industry by investigating the practicality, efficiency and adaptability of automated compliance checking processes.

The investigation started by analysing the background and context of the issue, highlighting the limitations associated with manual compliance checks and the potential for automation to improve the process. With the appearance of BIM as a transformative technology, integrating compliance checks into the BIM workflow emerged as a viable solution to address the challenges that traditional compliance assessment methods presented.

To address these challenges, a Framework was proposed and implemented. This Framework utilised various techniques and methodologies to translate textual regulations into machine-readable formats for Automated Compliance Checking. Initially, the Framework employed analysis and categorisation methods to improve the understanding of rules. It further incorporated an intermediate step to interpret and convert these rules into a code-familiar structure, facilitating the next translation step into executable code within a Visual Programming Language (VPL) environment, effectively serving as a white-box solution. Additionally, the Framework provided guidelines for standardising the BIM model before executing rule checks and addressed diverse reporting methods for compliance checks.

The effectiveness of this Framework was demonstrated through its successful application to diverse rules within a consistent context, giving insights that could carry significant implications for the industry. The main findings of this study include the usefulness and flexibility of the automated process. The process proved notably faster and more dynamic than traditional manual compliance checks, thus highlighting the potential for automation to revolutionise compliance assessments. Additionally, it acknowledges the importance of a thorough and holistic approach to integrating automation into complex industry processes.

Finally, by enabling efficient integration within the BIM framework, this study contributes to the ongoing discussion surrounding the future of compliance assessment, emphasising the transformative role of technology in reshaping industry practices.

7.2 Contributions and implications for the industry

This study has brought significant insights into developing a General Workflow for Automated Code Compliance Checks and showcasing how mixed-methods research can effectively tackle BIM-related challenges across the broader AECO industry. This approach emphasises not only technical automation but also the broader perspective needed to solve complex problems in innovative ways.

An essential contribution of this research is the importance of standardising processes within the BIM context. A foundation for consistent interpretation and application has been laid by carefully translating regulatory attributes into automated compliance checks. This commitment to consistency ensures that diverse stakeholders can work together more cohesively and efficiently, a crucial aspect of an industry with diverse compliance needs.

Furthermore, this study encourages the integration of Open BIM principles. Open BIM's flexible and transparent approach aligns seamlessly with the study's character, enabling better collaboration and interoperability across various software and professionals. Embracing Open BIM can seamlessly incorporate innovative solutions like the Framework for Automated Code Compliance Checks, fostering a culture of continuous improvement.

Interdisciplinary collaboration emerges as another key contribution. Bringing together expertise from various domains, including BIM, compliance, programming, and technology, has proven essential for driving progress. This combined approach ensures that solutions break down disciplinary barriers and tap into the collective knowledge of professionals from diverse fields.

In essence, this study's contributions extend beyond the framework itself. It models how thorough research, technical know-how, Open BIM principles, and interdisciplinary collaboration can combine to shape a future where technology enhances human capabilities. By embracing these implications, the AECO industry can confidently navigate the digital era and be better equipped to address challenges, innovate solutions, and improve project quality, efficiency, and sustainability on a global scale.

7.3 Limitations of the study and recommendations for future research

As with any research development, this study encountered limitations that provide paths for further investigation and improvement in automated compliance checking within the AEC industry. One significant limitation revolves around the scope and applicability of

the proposed automated compliance check framework. As this study focused on a specific set of tools, using Visual Programming Language (VPL) might present challenges in contexts with limited BIM model access, calling for exploring more versatile solutions that encompass the possibilities and compatibility issues that this may present within the BIM industry.

Applying the proposed framework within a specific authoring software environment highlights potential variations when implemented across different platforms. Further research should assess the framework's performance and adaptability within diverse authoring tools, considering each platform's distinct features and capabilities.

While effective within its context, the study's approach to tackling compliance issues may present limitations in providing universally applicable solutions. There's a need to consider alternative approaches that encompass Open BIM, Domain-Specific Languages, Natural Language Processing, or Open-Code tools. These alternatives could simplify the process for users and offer broader accessibility and effectiveness in complex scenarios.

Furthermore, considering the rapid evolution of Artificial Intelligence (AI) technologies, integrating AI into the automated compliance checking process holds immense potential, especially for tackling complex regulations or even developing Automated Checking Tools. However, this advancement also introduces challenges, including the possibility of error margins and the dependence on the ongoing development of AI capabilities. In the future, and with proper advancements, research may analyse the feasibility and benefits of integrating AI solutions into the compliance checking workflow while addressing potential risks that human intervention might easily detect.

Moreover, another approach that carries the issue on the other end of the process is creating standardised machine-readable versions of regulatory codes from their source. Nevertheless, this presents both promise and complexity. Developing standardised formats that bridge the gap between textual regulations and machine-interpretable formats could significantly enhance accuracy and efficiency in compliance checks (like what IFC did with interoperability, BCF with collaboration, or what IDS is currently achieving with Exchange Requirements). However, this effort would require huge collaborative efforts across national and international levels, alongside private and public organisations, to establish consistent standards that ensure reliable translation from regulatory language to digital code.

Additionally, incorporating the automated compliance checking workflow into Internal Execution Plans also emerges as a pertinent -and more achievable- chance for future exploration. Evaluating how the framework aligns with this context and the challenges and benefits that arise could offer insights into optimising compliance throughout project execution, giving a more consistent development of workflows.

Finally, this study recommends future researchers embrace a quantitative approach for comparing the efficiency gains of automated compliance checking versus traditional manual methods. The evaluation in this Dissertation primarily relied on qualitative observations, user feedback, and anecdotal evidence rather than numerical data. Quantitative analysis, including metrics such as time and money savings and error reduction, would provide a more concrete basis for evaluating the framework's advantages. By addressing these limitations and pursuing the recommended areas of future research, the field of automated compliance checking can continue to advance and reshape the landscape of the AEC industry.

7.4 Final Thoughts

The rapid growth of the Digital World is an inevitable force, constantly driven by technological advancements that shape various industries. In this context, BIM is evidence of digitalisation's profound influence on the AECO field. As society delves deeper into the digital age, BIM has evolved into a dynamic force that reestablishes how professionals conceive, design, construct, and manage buildings and infrastructure projects. And it will most likely continue to do so (as a considerable portion of the industry still has not adopted BIM).

The continuous evolution of BIM technologies, often at the pace of years, months or weeks, is intriguing and exciting. This is particularly evident in the ongoing development of groundbreaking technologies such as Artificial Intelligence, Generative Design, Machine Learning, Cloud Computing, the Internet of Things, and Digital Twins. These innovations have catalysed a transformation transcending BIM's original role as a mere 3D modelling methodology. Instead, it has evolved into a multi-disciplinary, integral, and holistic paradigm that redefines how AECO projects are conceived and executed. Given the ongoing trajectory of technological advancement, BIM's evolution is bound to continue reshaping the industry.

Additionally, in this everchanging era, the necessity for professionals is multifaceted. Mastery of BIM methodologies encompasses the practicalities of software, procedures, and standards and a profound adaptability to new technologies and ways of thinking.

Notably, while not all AECO professionals are familiar with programming languages, the challenges of compliance checks are universal and frequently encountered. This accentuates the importance of bridging the gap between conventional expertise and emerging technological knowledge.

With this in consideration, the evolution of the Digital World has propelled professionals ever closer to programming and technology. As the lines between disciplines blur and traditional boundaries dissolve, a deeper understanding of programming and technology becomes an asset, even for non-programmers. This convergence further accentuates the potential for transformative innovation and interdisciplinary collaboration.

In conclusion, this work is more than just an exploration of automated compliance checking; it reflects the unavoidable journey of technological progress, its profound impacts on the AECO industry, and the growing relation of traditional expertise with digital knowledge. As professionals navigate these transformative circumstances, the hope is that this dissertation serves as a guide, inspiring the streamlining of workflows, the embrace of technology, and the focus on the profound opportunities that lie ahead.

8. REFERENCES

- [1] R. Amor and J. Dimyadi, 'The promise of automated compliance checking', *Developments in the Built Environment*, vol. 5, Mar. 2021, doi: 10.1016/j.dibe.2020.100039.
- [2] M. Aydın, 'A Review of BIM-Based Automated Code Compliance Checking: A Meta-Analysis Research', 2021. [Online]. Available: www.intechopen.com
- [3] W. Solihin, 'A Simplified BIM Data Representation Using a Relational Database Schema for an Efficient Rule Checking System and its associated Rule Checking Language', 2016.
- [4] C. Eastman, J. min Lee, Y. suk Jeong, and J. kook Lee, 'Automatic rule-based checking of building designs', *Automation in Construction*, vol. 18, no. 8. pp. 1011–1033, Dec. 2009. doi: 10.1016/j.autcon.2009.07.002.
- [5] C. Preidel and A. Borrmann, 'Automated Code Compliance Checking Based on a Visual Language and Building Information Modeling', 2015, doi: 10.13140/RG.2.1.1542.2805.
- [6] L. Ding, R. Drogemuller, D. Marchant, R. Drogemuller, M. Rosenman, and J. Gero, 'Automating code checking for building designs-DesignCheck', 2006. [Online]. Available: <https://www.researchgate.net/publication/322299545>
- [7] C. Preidel and A. Borrmann, 'Towards code compliance checking on the basis of a visual programming language', Preidel & Borrmann, 2016. [Online]. Available: <http://www.itcon.org/2016/25>
- [8] H. Lee, S. Lee, S. Park, and J.-K. Lee, 'An Approach to Translate Korea Building Act into Computer-readable Form for Automated Design Assessment', 2015.
- [9] T. H. Beach, J. L. Hippolyte, and Y. Rezgui, 'Towards the adoption of automated regulatory compliance checking in the built environment', *Autom Constr*, vol. 118, Oct. 2020, doi: 10.1016/j.autcon.2020.103285.
- [10] W. Solihin and C. Eastman, 'Classification of rules for automated BIM rule checking development', *Autom Constr*, vol. 53, pp. 69–82, 2015, doi: 10.1016/j.autcon.2015.03.003.

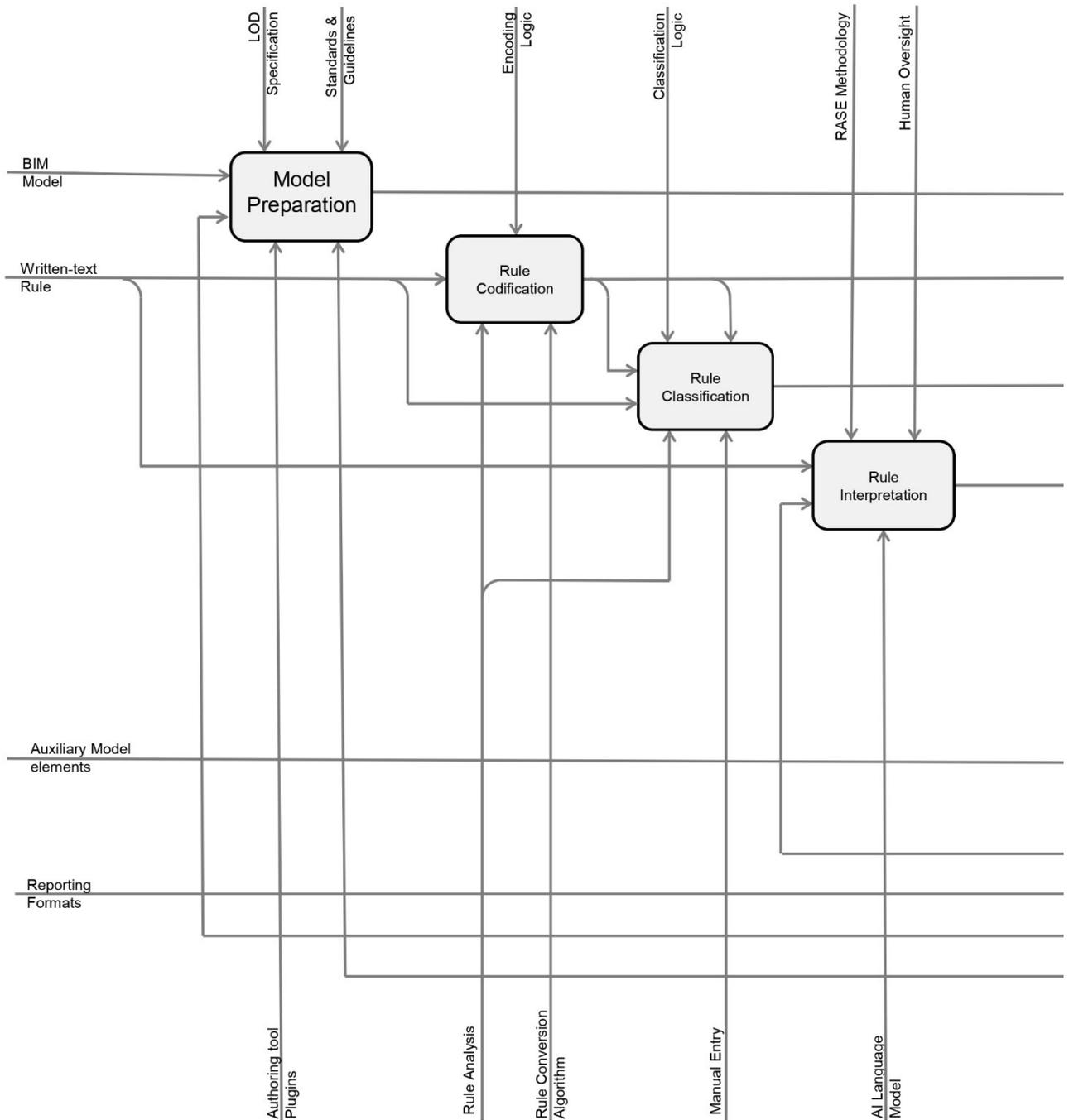
- [11] T. H. Beach, Y. Rezgui, H. Li, and T. Kasim, 'A rule-based semantic approach for automated regulatory compliance in the construction sector', *Expert Syst Appl*, vol. 42, no. 12, pp. 5219–5231, Jul. 2015, doi: 10.1016/j.eswa.2015.02.029.
- [12] T. H. Beach, J. L. Hippolyte, and Y. Rezgui, 'Towards the adoption of automated regulatory compliance checking in the built environment', *Autom Constr*, vol. 118, Oct. 2020, doi: 10.1016/j.autcon.2020.103285.
- [13] J. Dimyadi and R. Amor, 'Automated Building Code Compliance Checking-Where is it at?', 2013, doi: 10.13140/2.1.4920.4161.
- [14] S. J. Fenves, J. H. Garrett, H. Kiliccote, K. H. Law, and K. A. Reed2, 'Computer representations of design standards and building codes', 1995.
- [15] Stahl and Fred I., 'The Standards Interface for Computer-Aided Design (SICAD)', 1983.
- [16] S. J. Fenves, R. N. Wright, F. I. Stahl, and K. A. Reed, 'Introduction to SASE: Standards, Analysis, Synthesis, and Expression', Gaithersburg, MD, 1987. doi: 10.6028/NBS.IR.87-3513.
- [17] Ministry of National Development, 'CORENET Oficial Web Page - About Us'. Khemlani (accessed May 05, 2023).
- [18] L. Khemlani, 'CORENET e-PlanCheck: Singapore's Automated Code Checking System', Oct. 26, 2005. <https://www.aecbytes.com/feature/2005/CORENETePlanCheck.html> (accessed May 05, 2023).
- [19] W. Solihin, 'Lessons learned from experience of code-checking implementation in Singapore Using BIM to support performance-based compliant building engineering designs View project Simplified BIM model server in a big data platform View project', 2004. [Online]. Available: <https://www.researchgate.net/publication/280599027>
- [20] P. Cole, 'REScheck TM Basics Energy-Building Energy Codes Program AIA Provider #: I014 AIA Course #: EC16-M03', 2016.
- [21] J. Dimyadi, W. Solihin, C. Eastman, and R. Amor, 'Integrating the BIM Rule Language into Compliant Design Audit Processes', 2016.

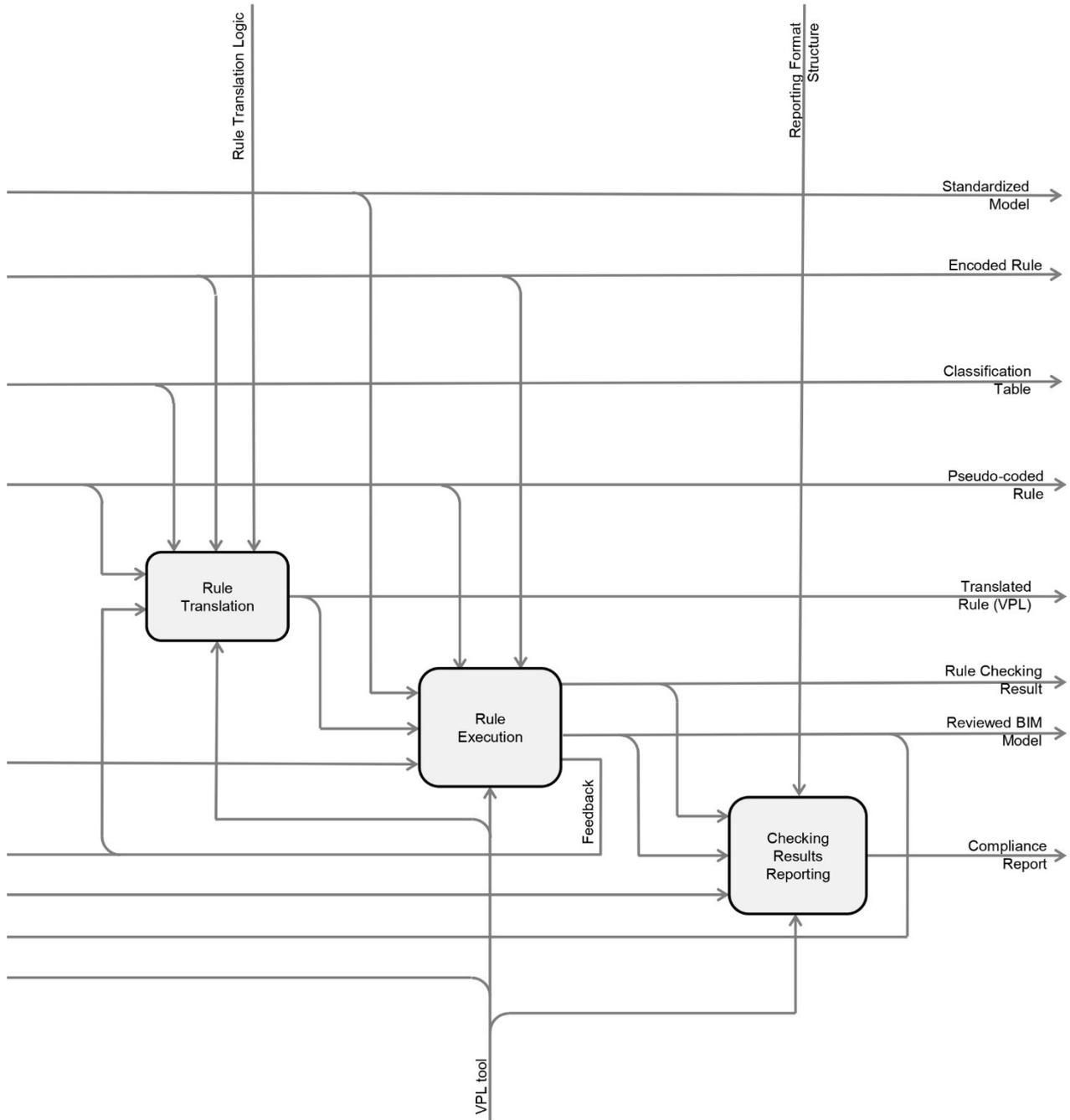
- [22] 'RegBIM: BIM-based Regulatory Compliance Design Environment', <https://gtr.ukri.org/projects?ref=101152#/tabOverview>.
- [23] J. Beetz, J. Van Leeuwen, and B. De Vries, 'IfcOWL: A case of transforming EXPRESS schemas into ontologies', *Artificial Intelligence for Engineering Design, Analysis and Manufacturing: AIEDAM*, vol. 23, no. 1, pp. 89–101, Feb. 2009, doi: 10.1017/S0890060409000122.
- [24] S. Zhang, J. Teizer, J. K. Lee, C. M. Eastman, and M. Venugopal, 'Building Information Modeling (BIM) and Safety: Automatic Safety Checking of Construction Models and Schedules', *Autom Constr*, vol. 29, pp. 183–195, Jan. 2013, doi: 10.1016/j.autcon.2012.05.006.
- [25] L. Jovanovic, L. Meng, B. Doherty, N. Gardner, M. Hank Haeusler, and K. Daniel Yu, 'Automated Code Compliance Checking A computational workflow for verifying model, parameter and regulatory compliance', 2022.
- [26] H. Narayanswamy, H. Liu, and M. Al-Hussein, 'BIM-based Automated Design Checking for Building Permit in the Light-Frame Building Industry', 2019.
- [27] E. Hjelseth, 'Foundations for BIM-based model checking systems Transforming regulations into computable rules in BIM-based model checking systems Prinsipper for BIM basert modell sjekking Omforming av bestemmelser i regelverket til digitaliserte regler i BIM-baserte systemer for modellsjekking', 2015, p. 54.
- [28] P. Pauwels *et al.*, 'A semantic rule checking environment for building performance checking', *Autom Constr*, vol. 20, no. 5, pp. 506–518, Aug. 2011, doi: 10.1016/j.autcon.2010.11.017.
- [29] E. Hjelseth, 'Classification of BIM-based Model Checking Concepts', 2016. [Online]. Available: <http://www.itcon.org/2016/23>
- [30] E. Hjelseth and N. Nisbet, 'Capturing Normative Constraints by Use of the Semantic Mark-Up RASE Methodology', 2011.
- [31] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, 'Introduction to Algorithms, Third Edition'.

- [32] E. Hjelseth, 'Foundation for development of computable rules CIB TG90 Norwegian book chapters on Information Integration in Construction View project', 2009. [Online]. Available: <https://www.researchgate.net/publication/303205970>
- [33] Í. Guedes and M. Andrade, 'Automatic Rule-Based Checking for the Approval of Building Architectural Designs of Airport Passenger Terminals based on BIM', 2019.
- [34] M. Aydın, 'A BIM Based Automated Code Compliance Checking Conceptual Model Proposal for Housing Projects in Turkey', 2018. [Online]. Available: <https://www.researchgate.net/publication/357254473>
- [35] S. Malsane, J. Matthews, S. Lockley, P. E. D. Love, and D. Greenwood, 'Development of an object model for automated compliance checking', *Autom Constr*, vol. 49, no. PA, pp. 51–58, 2015, doi: 10.1016/j.autcon.2014.10.004.
- [36] 'Rules on minimum technical requirements for the construction of residential buildings and apartments, Slovenia. Translated from: "Pravilnik o minimalnih tehničnih zahtevah za graditev stanovanjskih stavb in stanovanj".', <http://www.pisrs.si/Pis.web/pregledPredpisa?id=PRAV10213>, 2022.

9. ANNEXES

Annex A: ACCC Framework's Application IDEF0 decomposition.





Annex B: Slovenian Regulation for Residential Building (SRRB).

(Translation to English)

Rules on the minimum technical requirements for the construction of residential buildings and apartments

- **Article 1: Editing area.**
 - (1) These regulations determine the minimum technical requirements for the construction of apartments and residential buildings, except for buildings and apartments intended to temporarily solve the housing needs of socially vulnerable persons.
 - (2) Construction, according to these regulations, is the construction of buildings in accordance with the regulations governing the construction of buildings.
- **Article 2: deleted.**
- **Article 3: deleted.**
- **Article 4: Entrance to multi-apartment buildings.**
 - (1) The entrance area to a building with three to nine apartments has a clear width of at least 1.50 meters, and in buildings with ten or more apartments, at least 1.80 meters.
 - (2) The entrance area to a building with three to nine apartments has a clear width of at least 1.50 meters, and in buildings with ten or more apartments, at least 1.80 meters.
 - (3) Clear width is the width measured between installed equipment and installations up to a clear height of 2.10 meters.
- **Article 5: Common areas in multi-apartment buildings.**
 - (1) An easily accessible covered storage space for bicycles is provided in or next to the building. The size of the bicycle shed is determined by providing 0.75 bicycle storage space for each planned bed in the residential building. If the room is also intended to store devices with an internal combustion engine, the same technical conditions as apply to motor vehicle parking spaces must be met.
 - (2) Places for garbage cans are designated in a building or near a building that allows unobstructed access to the municipal service.
 - (3) In a multi-apartment building with more than four apartments, there is a storage area for cleaners for cleaning common areas, in which there is a water supply connection and a wastewater outlet. The surfaces of the floor and walls of the room for storing cleaners must allow wet cleaning.

- **Article 6: Safety of use.**
 - (1) Stairways with more than three steps and ramps with a height difference of more than 0.48 meters are equipped with a handrail on at least one side.
 - (2) Staircases, balconies, loggias, terraces and the like, raised above the surrounding area by more than 0.48 meters, are protected by a fence at least one meter high. Openings in the fence make it impossible for objects with a diameter of more than 12 centimetres to pass through and for children to climb.

- **Article 7: Measuring points in multi-apartment buildings.**
 - (1) Meters for monitoring the consumption of energy products, energy and water enable separate monitoring for each apartment and common areas of the building and are accessible from its common areas. If the meters are connected to a remote reading system, their installation is also allowed in the apartment.
 - (2) The previous paragraph also applies in cases of reconstruction.

- **Article 8: Elevators in multi-apartment buildings.**
 - (1) In buildings with five floors or more, at least one personal elevator with clear cabin dimensions of at least 110 cm x 210 cm x 220 cm (width x depth x height) is installed.
 - (2) At least two personal elevators are installed in buildings with nine floors or more, with at least one meeting the conditions from the previous paragraph.
 - (3) To the extent that the building has connected access to different elevators on all floors, at least one elevator meets the conditions from the first paragraph of this article.
 - (4) The clear width of the entrance to the elevator cabin is at least 90 centimetres.

- **Article 9: Size of garages and parking spaces.**
 - (1) The clear height of garages and other surfaces on which motor vehicles are intended to be driven must not be less than 2.20 meters, except at the rear walls of rectangular or sloping parking spaces where, due to built-in equipment and installations, up to a width of 0.70 meters the clear height must not be less than 1.80 meters. These reductions in clear ceiling height are properly marked and secured.
 - (2) Depending on the organization of parking, the size of driveways and parking spaces for passenger vehicles must comply with the requirements from the regulation governing traffic signals and traffic equipment on roads, and their number must comply with the requirements from point 6 of the SIST standard ISO 21542 Construction of buildings - accessibility and usability of built environment.

- **Article 10: Parts of apartments.**

(1) An apartment is a complete unit consisting of a room or part of a room:

- Intended for living (e.g., living room, work corner, day corner, study cabinet);
- Intended for sleeping (e.g., bedroom, children's room, sleeping area);
- Intended for food preparation (e.g., kitchen, kitchenette);
- Intended for eating food (e.g., dining room, dining area);
- intended for personal hygiene (e.g., bathroom, toilet), and from the vestibule (e.g., hallway, hall).

(2) At least one room dedicated to personal hygiene is separate from other rooms.

(3) Each apartment has an individual storage space, with a floor area of at least 3 m², which can be inside or outside the apartment, but on the plot intended for the construction of a building”.

(4) Apartments must be built in such a way that furniture of standard sizes from Article 24 of these regulations can be installed in them and the clearances from Article 25 of these regulations can be provided.

- **Article 11: Installations and equipment in the apartment.**

The apartment has installations and the necessary equipment for the supply of cold and hot drinking water, electricity, heating, and at least pre-installation has been carried out, which enables the subsequent installation of installations and equipment for telecommunications. The room in which the water connection is located has a regulated wastewater outlet. Adequate ventilation with constant air exchange must be ensured in the apartment.

- **Article 12: Height of rooms.**

(1) The clear height of residential premises is at least 2.50 meters, except for the vestibule and rooms from the second and third paragraphs of Article 10 of these regulations, where the clear height must not be lower than 2.20 meters.

(2) In the reconstruction of residential buildings, the clear height of rooms intended for living, sleeping and preparing and eating food, if it is less than 2.50 meters, may not be further reduced.

(3) The provision on the minimum clear height of residential premises must be fulfilled on at least two thirds of the floor area of each room in the apartment, and in residential premises with sloping ceilings on two-thirds of that floor area where the clear height of the room is greater than 1.80 meters.

- **Article 13: Room width.**

Rooms intended for living and sleeping are not narrower than 1.90 meters and, in the case of lighting only on the shorter side, not narrower than half of their length.

- **Article 14: Apartment lighting.**

(1) Rooms or parts of rooms intended for consumption and preparation of food, sleeping and living are naturally lit. Natural lighting can be direct or indirect.

(2) Direct lighting is achieved if the total area of treated masonry openings (in this case, only that part of the opening that is more than 0.50 meters above the finished floor) intended for lighting reaches at least 20 percent of the net floor area of these parts of the apartment

(3) Indirect lighting of rooms is permissible through another directly lit room. In this case, the total area of the openings of the directly lit room, through which other rooms are indirectly lit, is determined in accordance with the previous paragraph, considering the total net floor area of the directly lit room and the rooms indirectly lit through it. The total area of treated masonry openings intended for indirect lighting of indirectly lit rooms must not be less than the total area of the required openings of the directly lit room.

(4) If a part of the apartment is directly illuminated by natural light from only one side, its depth must not measure more than three light heights of the room. The total depth of the directly and indirectly lit room, which are lit from only one side, must not measure more than three clear heights of the directly lit room.

(5) At least one opening for natural lighting of the room, which allows living during the day, is designed, and built in such a way that the height of the parapet is no more than 0.90 meters above the finished floor and allows a view to the outside space.

(6) All openings for natural lighting have built-in elements to prevent excessive influence of the sun's rays and to darken.

(7) Natural lighting of the parts of the apartment referred to in the first paragraph of this article, with more than three beds, must not be provided exclusively through openings oriented in the area from the northeast to the northwest direction.

(8) To the extent that the parts of the apartment referred to in the first paragraph of this article are illuminated through openings oriented into light shafts, they are designed so that the sun shines on part of the bottom of the shafts at least once a year. Rooms that are illuminated via light shafts do not serve to indirectly illuminate other rooms.

- **Article 15: Opening windows and doors.**
The location, size and shape of windows and doors and the installation of equipment in each room allow the unobstructed opening of at least one window and door sash up to the clear width of their opening, unless otherwise specified by regulations governing fire safety.
- **Article 16: Security of window openings.**
All windows that can be opened and where the height of the parapet on the inside is less than 0.90 meters and on the outside more than one meter have an additional guardrail with an upper holding edge that extends at least one meter above the internal finish under.
- **Article 15: Entrance to the apartment**
The entrance door to the apartment ensures a safe separation of private spaces from common ones. The entrance door, the design and the equipment of the apartment allow visual control of the space in front of the entrance door of the apartment.
- **Article 18: Transitions.**
The clear width of passages intended for communication between rooms is at least one meter, even when equipment is installed.
- **Article 19: Width of the staircase in the apartment.**
Staircases inside the apartment that lead to rooms or parts of rooms for living, preparing, and eating food or personal hygiene have a clear width of at least 0.80 meters. Still, it can be less if the apartment on the second floor has an additional entrance from the common areas of the apartment building or another option for entering furniture.
- **Article 20: Beds.**
A maximum of two beds may be provided in a room or part of a room intended for living or sleeping. An exception is a bed for an infant, which is considered a temporary bed.
- **Article 21: Food preparation area.**
The room or part of the room for food preparation has wiring and the necessary devices for the supply of cold and hot drinking water and the drainage of wastewater.
- **Article 22: Rooms for Personal hygiene.**
 - (1) The bathroom has installations and the necessary equipment for the supply of cold and hot drinking water with wastewater drains.
 - (2) The toilet has an installation for at least the supply of cold water and the drainage of wastewater.

- (3) The bathroom has at least a sink, a bathtub or a shower and a toilet bowl with a flusher if this is not installed in a separate room (toilet) where there is also a smaller sink. Sinks and bathtubs have built-in plumbing fittings.
 - (4) The apartment with four beds has a toilet bowl with a flusher in a separate room (toilet). An apartment with five or more beds has two toilet bowls with a flush, separately installed in the bathroom and a separate room (toilet).
 - (5) At least one room with a toilet bowl has an entrance from a corridor, hall or similar area of the apartment.
- **Article 23: Surface treatment.**

The surface of the walls up to a height of 1.50 meters and the floor in the bathroom and toilet, as well as the surface of the walls next to the counters in the food preparation area, must not allow water to pass through and must allow wet cleaning.
 - **Article 24: Minimum room equipment.**
 - (1) Space is provided in the food preparation area for the installation and use of the following equipment:
 - In a one-bed apartment: kitchen block 100 cm x 60 cm and storage cupboard 60 cm x 60 cm
 - In an apartment with two beds: stove 60 cm x 60 cm, refrigerator 60 cm x 60 cm, sink 60 cm x 60 cm, storage cupboard 60 cm x 60 cm, work-storage surface at least 60 centimetres.
 - In an apartment with more than two beds: equipment as in the previous indent, with a work-storage surface at least 40 centimetres wide added for each additional bed in the apartment.
 - (2) In the dining area, space is provided for the installation and use of a dining table and the same number of chairs as the number of beds in the apartment; each chair must be at least 60 centimetres long and 40 centimetres deep at the table.
 - (3) In the space intended for living, space is provided for the installation and use of a sofa set with a seat of 80 cm x 80 cm for each bed of the apartment; TV bookcases with a width of at least 60 centimetres and a depth of 40 centimetres for each bed of the apartment and coffee tables 60 cm x 60 cm.
 - (4) In the room intended for sleeping, space is provided for the installation and use of a bed 90 cm x 210 cm or a bed for two people 160 cm x 210 cm and a worktable 120 cm x 70 cm for three beds (which can also be in the living room).
 - (5) In the room dedicated to personal hygiene, space is provided for the installation and use of a sink with a depth of at least 40 cm and a total area

of at least 0.33 m², a bathtub with an area of 1.20 m² or a shower with an area of 0.60 m² and a toilet bowl 40 cm x 60 cm.

- (6) Space is provided in the toilet for the installation and use of a 40 cm x 60 cm toilet bowl and a small sink.
- (7) Each apartment has a guaranteed space for the installation and use of a tall wardrobe with a total width of at least 120 centimetres and a depth of 60 centimetres for each bed of the apartment.
- (8) Space for installing and connecting a washing machine measuring 60 cm x 60 cm is provided in the apartment if the building does not have a common laundry room.
- (9) The entrance and all interior doors of the apartment have a clear opening of at least 80 cm x 208 cm.
- (10) The clear width of the door to the toilet can be 70 centimetres if the apartment has at least one other room with a toilet bowl with a clear door width of at least 80 centimetres.

- **Article 25: Equipment offsets.**

- (1) The following offsets and heights are designed in the apartment:
 - At least 70 centimetres in front of cupboards and washing machine.
 - In front of a set of kitchen equipment and between two sets of this equipment, there must be a passage at least 100 centimetres wide, except in apartments with three or fewer beds, where the passage is at least 90 centimetres wide.
 - At least 75 centimetres on those sides of the table where chairs are provided.
 - At least 70 centimetres in front of the accessible sides of the bed, whereby the bed for two people must be accessible from both long sides.
 - At least 70 centimetres in front of the equipment in the personal hygiene areas and at least 60 centimetres in front of the sink in the toilet.
 - Lateral deviations between equipment in personal hygiene rooms must be at least 20 centimetres.
 - Where electrical switches or sockets are planned to be installed next to the door frames, a minimum distance of 15 centimetres must be provided for the equipment.
- (2) When the equipment is arranged in such a way that the offsets from the previous paragraph could be added together, the minimum necessary offset is calculated by multiplying the sum of the minimum offsets from the previous paragraph by a factor of 0.65.

- (3) Deviations from the first paragraph of this article apply to rooms with finished surfaces.
 - (4) The apartment is designed in such a way that, in addition to the built-in equipment, it ensures the usability of the fixtures and the opening of doors and windows.
- **Article 26: Design Requirements.**
 - (1) Compliance with the requirements of this rulebook can be seen from the project documentation for the construction (hereinafter: PZI). For each type of apartment in the building, the architecture plan in the PZI must also contain a quoted drawing on a scale of at least 1:100, in which the equipment is drawn from Article 24 of this rulebook, with the deviations from Article 25 of this rulebook taken into account.
 - (2) Except for built-in equipment, a distance of at least 5 centimetres from the walls must be considered when drawing the equipment.
 - (3) The drawing referred to in the first paragraph of this article shows all areas of the apartment that do not meet the requirement regarding the clear height of the premises, specified in the first paragraph of Article 12 of this rulebook.
 - **Article 27: Supervision.**

Supervision over the implementation of the provisions of this rulebook is carried out by building inspectors.