

# Digital building logbook for improved asset management: The value of a linked data approach

Master Thesis

*B.J.F.P van der Hall*

*Student nr.: 1643681*

*Department of Built Environment, Eindhoven University of Technology, Eindhoven, Netherlands*



Educational institution | Eindhoven University of Technology  
Education program | MSc. Construction Management and Engineering

Academic year | 2022-2023

Study load | 45 ECTS

## Graduation Committee

Chairman | Prof. Dr. Ir. Bauke de Vries

First supervisor | Dr. Ir.-arch. Pieter Pauwels

Second supervisor | Dr. Ir. Rob Wolfs

Company Advisor | Maarten Mook, Heijmans N.V.

Date | 05-04-2023

*This master thesis has been carried out in accordance with the rules of the TU/e Code of Scientific Integrity.*



## Contents

1	Introduction .....	17
1.1	Background .....	17
1.2	Problem statement.....	17
1.3	Thesis objectives .....	18
1.4	Scientific Relevance .....	19
1.5	Outline .....	19
2	Literature Review .....	21
2.1	Building Information Modelling (BIM).....	21
2.1.1	ISO 19650 .....	22
2.2	Standards and Classifications used in the AEC industry .....	24
2.2.1	IFC .....	24
2.2.2	Information Delivery Standards.....	25
2.2.3	NL-SFB .....	25
2.2.4	ICDD .....	26
2.2.5	BuildingSMART Data Dictionary (bSDD) .....	26
2.3	Asset Management.....	27
2.3.1	Definition.....	27
2.3.2	Maintenance processes .....	28
2.3.3	Information in AM .....	29
2.4	Digital twins.....	30
2.4.1	Definition.....	30
2.4.2	Digital Twins in Different Industries.....	32
2.4.3	Digital Twins in the AEC Industry .....	32
2.4.4	Operation and Maintenance phase .....	35
2.4.5	DT Applications in Asset Management .....	36
2.5	Semantic Web and Linked Data Technologies .....	37
2.5.1	Introduction.....	37
2.5.2	Resource Description Framework.....	38
2.5.3	Ontologies .....	38
2.5.4	SHACL.....	39
2.5.5	SPARQL.....	40
2.5.6	Graph Databases over Relational Databases .....	40
2.6	Conclusion Literature Review .....	41
3	Methodology .....	43

3.1	Research Approach .....	43
3.2	Market Research .....	43
3.2.1	Interview Results .....	43
3.2.2	Use Case – Requirements / Competency questions .....	44
4	Development .....	47
4.1	Proposing System Architecture and Data Pipeline .....	47
4.2	Extracting data from current systems .....	49
4.2.1	BIM Models .....	49
4.2.2	Maintenance Management System (SAP).....	49
4.3	Transforming Data to RDF Format .....	50
4.3.1	Maintenance ontology .....	50
4.3.2	BOT ontology .....	52
4.3.3	Combined ontology.....	52
4.3.4	Converting SAP MMS data to RDF .....	54
4.3.5	Converting Building Information Models to RDF.....	59
4.4	Integrating data from MMS and IFC models.....	61
4.4.1	Integrating SAP with BIM .....	61
4.4.2	Automating the Data Pipeline .....	63
4.5	Presentation Layer / User Interface .....	64
4.5.1	Triple Store .....	65
4.5.2	Application for Data Analysis.....	65
4.5.3	Data Visualization Tool.....	66
4.6	Conclusion.....	67
5	Case Study and Results.....	69
5.1	Case study EMA.....	69
5.2	Starting Situation.....	69
5.2.1	TimeXtender output in excel .....	69
5.2.2	IFC files .....	69
5.3	Running Script .....	70
5.3.1	IFC2LBD .....	70
5.3.2	SAP Conversion.....	71
5.3.3	Integration.....	71
5.4	Answering the Competency Questions .....	72
5.4.1	Select all historic orders and measures related to a specified asset .....	72
5.4.2	Select all historic orders and measures that took place in a specified room.....	72
5.4.3	What kind of assets are prone to a specified damage picture? .....	73

5.4.4	What kind of rooms are prone to a specified damage picture?.....	73
5.4.5	Select all properties from BIM for the asset or room related to order .....	73
5.4.6	Select all measures that resolved similar malfunctions.....	73
5.4.7	What kind of orders have the longest lead times?.....	74
5.4.8	What is the prognosed lead time for this Notification?.....	74
5.4.9	What is the most common Cause for a specific problem? .....	75
5.4.10	Select all rooms where a specified malfunction occurred in a specific timeframe. ....	75
5.4.11	Visualize the context of data around a specific asset, room, or problem.....	75
5.5	LBDviz.....	77
5.5.1	Select all historic orders and measures related to a specified asset .....	77
5.5.2	Select all historic orders and measures that took place in a specified room.....	78
5.5.3	Select all properties from IFC for the asset or room related to order .....	78
6	Conclusion .....	80
6.1	Summary.....	80
6.2	Discussion .....	82
6.2.1	Implications .....	83
6.2.2	Limitations.....	83
6.2.3	Recommendations.....	84
7	References .....	87
8	Appendices .....	95
8.1	Maintenance Ontology .....	95
8.2	Python Code.....	96

## Preface

Dear Reader,

Six and a half years ago, I started my educational career in the built environment. After four years of studying the bachelor Architecture and Construction Engineering at the Avans University of Applied Sciences in Den Bosch, I was motivated to start the Master ‘Construction Management and Engineering’ at the Eindhoven University of Technology. The premaster was challenging, but during the master, I passed all courses without any resits. You are now reading the outcome of the final part, namely my master thesis “Digital building logbook for improved asset management: The value of a linked data approach”.

From early on, my interest was in 3D modeling and during my first internship I was introduced to the world of BIM and digital twins. After that, I did everything I could to learn more about digitalization of the built environment. I went to congresses and participated in the BIM Student Battle and Digital Twin Hackathon in Johannesburg (SA). In the course ‘Fundamentals of BIM’ lectured by Dr. Ir.-arch. Pieter Pauwels, I was introduced to linked data and graph databases. Although this topic was completely new to me, I was very much interested in learning more about it, so without any knowledge, I decided to do a deep dive in the world of linked data in the built environment. The last seven months came with highs and lows, but I gained a lot of experience in semantic web technologies, different programs, and even programming.

Without the guidance of a number of people, I would not have been able to finalize my thesis successfully. First of all, I would like to thank my first supervisor Dr. Ir.-arch. Pieter Pauwels for the guidance and support during the process. With all his knowledge on BIM, linked data and digital twins, I could not have wished for a better supervisor. I also want to thank the chairman of my graduation committee Dr. Ir. Bauke de Vries and my second supervisor Dr. Ir. Rob Wolfs for their feedback on, and assessment of my master thesis. Besides, I want to thank Joost van de Koppel, Maarten Mook, and all other colleagues at Heijmans for discussion about the topic and providing me with all the data for the case study. Finally, I want to thank my family, girlfriend and friends who supported me during the sometimes tough period of writing my thesis.

Lastly, to the ones that are reading my thesis: I hope you will be inspired. Enjoy!

Bob van der Hall



Eindhoven, April 2023

## Summary | English

The Architecture, Engineering, and Construction (AEC) industry is rapidly transforming, which is caused by the increasing use and development of digital technologies. The emergence of Building Information Modeling (BIM) allowed structured storage and use of data, making it essential for effective asset management. In almost all larger construction projects BIM has become the standard. However, despite advancements in the design, engineering and construction phases, there is room for improvement, in the way assets are managed by different types of maintenance parties. Compared to the costs during the initial design and construction phase, the operations and maintenance (O&M) costs are significantly higher. The potential advantages of digitizing the exploitation phase are significant because it is the most crucial stage in the life cycle of a building.

Despite the availability of Building Information Modeling (BIM) systems, reactive maintenance remains common, resulting in frequent malfunctions, complaints, and increased costs. To ensure efficient and effective building operation, it is essential to focus on information from the building's systems, equipment, and components, as well as maintenance schedules, performance metrics, and energy usage in the operations phase. Without comprehensive asset information, it is difficult for asset managers to make informed decisions regarding maintenance, repairs, and upgrades, leading to potential inefficiencies, increased costs, and even safety risks. Asset managers often lack a total overview of information and have to obtain asset information from various systems, which is a time-consuming and often unproductive task. At the same time, this means that they lack a comprehensive understanding of all the characteristics of the assets from all the systems. Potentially, this can lead to errors and inconsistencies in the data, which makes it difficult to take informed decisions, leading to delays and decreased productivity. As a result, technicians often lack the necessary tools and information to resolve issues in one time (first time fix ration), which again leads to delays and additional costs.

Linked Data and semantic web technologies are gaining more attention in the industry. The infrastructure sector has already recognized the benefits of linked data, with for instance the use of the Resource Description Framework (RDF) for interchanging data. However, building contractors have yet to realize the full potential of this technology for their asset management of buildings. The expectation is that Rijkswaterstaat (RWS) and municipalities will expect all projects to be delivered in RDF format in the future. Also, digital twins have been identified as a promising solution for improving the way in which assets are managed. By creating digital twins, a virtual replica of a physical asset, asset managers can gain better insights into how a building or asset operates, making it easier to make informed decisions.

This study aims to create a digital building logbook for a building and investigate the added value of a linked data approach for asset life cycle information management. The research will leverage historical data from an SAP maintenance management ERP database and building information from BIM models to create a graph database that integrates all asset information. By using a graph database, the study will explore the advantages of handling dynamic data and analyzing an asset's entire context. The study will evaluate the effectiveness of the linked data and digital twin technology in improving asset management and decision-making, including the potential for reducing maintenance costs, increasing asset performance, and improving safety. The results will contribute to the development of best practices for digital twins in asset management in the AEC industry, improving the efficiency and effectiveness of maintenance management systems and providing insights into the benefits of semantic web technologies for asset life cycle information management.

After interviews with different stakeholders within asset management it became clear that the system should help tackling the problems related to the total overview of information, insights into the maintenance history of specific assets and rooms, and the wayfinding of assets and rooms. Firstly,

instead of hosting data in different applications, the system should allow performing large-scale analysis on combined data to gain new insights which can help optimizing the preventive maintenance schedule. By optimizing the maintenance schedule, the amount of corrective maintenance orders is reduced. Of course, the amount of malfunctions will never be reduced to zero, so for the malfunctions that do occur, the system should instantly provide all potentially valuable information from historical data about an asset, room, or a specific problem. This will reduce the time planners and technicians need to search for the required information, which will also directly reduce the indirect productivity of technicians. This way, employees are supported in making better and quicker decisions on how to resolve malfunctions based on historical data, instead of only experience or even gut feeling. To find out what the system should be able to perform as a minimum, multiple meetings with maintenance engineers, planners, and data consultants were held, to create a list of eleven competency questions.

Subsequently, a system architecture and data pipeline are developed that can and integrate data from the existing systems in order to do analyses in two different applications. The pipeline starts with extracting the data from the current information systems. In the context of this research the existing systems are a maintenance management ERP system (SAP) and a common data environment with IFC BIM models. The next step is to transform the data into a format suitable for semantic web technologies. For the conversion of the maintenance management data to RDF, an ontology was created first, that defines the structure and relationships of the data, to ensure that the data is properly structured and organized for use in a graph database. Subsequently a python script was created transforms the tabular data to triples in TTL format according to the vocabulary of the maintenance ontology. To transform the information from the BIM model to RDF, the bot and PROPS ontologies are used. Using a Python based IFCtoLBD convertor, multiple IFC aspect models are converted to TTL files. Subsequently, in order to merge the data from the BIM model and the maintenance management system new relationships needed to be created between the Asset from the MMS and the Element from BIM and between the Room from the MMS and the Space from BIM. To create the two new relationships, based on element id's and room numbers, python was used in combination with the IfcOpenShell and RDFLib packages. Finally, to gather insights from the data, two applications are proposed. The first tool, with Ontotext GraphDB as a back end, is for asset managers, maintenance engineers or planners that are familiar with the data and want to do advanced querying in the data to derive insights that are not possible in a relational database. LBDviz is a web application for visualizing RDF data in an IFC viewer, for stakeholders, such as planners and technicians that are less familiar with the data and need predefined functionalities in a user-friendly interface to derive quick insights essential for completing their work.

Finally, the system and data pipeline were evaluated in a case study of the European medicines agency (EMA) and validated by answering the competency questions drawn up during the interviews. All eleven competency questions are answered by queries from the triple store Ontotext GraphDB. Three competency questions are also answered using the web application LBDviz. The combination of linked data and semantic digital twins in a digital building logbook enabled more intuitive searching for the entire context close to a single data point, easier tracking of complex relationships among assets, better ability to handle highly dynamic data, and suggestions on how to resolve new malfunctions, based on historical data as soon as they get notified. Asset managers and maintenance engineers have the ability to run complex queries that are not possible or really challenging in traditional relational databases. With a 3D user interface, also planners and technicians have access to quick insights on important data about assets and rooms. Applying linked data in a digital twin for asset management will increase efficiency, reduce costs, and improve the overall health and performance of buildings and equipment.



## Summary | Dutch

De architectuur-, ingenieurs- en bouwsector transformeert snel, wat wordt veroorzaakt door het toenemende gebruik en de ontwikkeling van digitale technologieën. De opkomst van Building Information Modeling (BIM) heeft de gestructureerde opslag en gebruik van gegevens mogelijk gemaakt, waardoor het al snel essentieel werd voor effectief asset management. Bij vrijwel alle grotere bouwprojecten is BIM de standaard geworden, maar ondanks de vooruitgang in de ontwerp-, engineering- en bouwfase is er ruimte voor verbetering in de manier waarop onderhoudspartijen hun assets beheren. In vergelijking met de kosten tijdens de initiële ontwerp- en bouwfase zijn de exploitatie- en onderhoudskosten aanzienlijk hoger, waardoor digitalisering in deze fase positieve financiële gevolgen kan hebben.

Ondanks de beschikbaarheid van verschillende digitale onderhoudssystemen en optimalisaties in de onderhoudsstrategie blijft correctief onderhoud gebruikelijk, met als gevolg frequente storingen, klachten en hoge kosten. Voor een efficiënte en effectieve exploitatie van gebouwen is het essentieel om in de exploitatiefase te focussen op een strategie om zo veel mogelijk gebruik te maken van beschikbare data. Zonder uitgebreide asset informatie is het moeilijk voor asset managers om weloverwogen beslissingen te nemen over onderhoud en reparaties, wat leidt tot potentiële inefficiëntie, hogere kosten en zelfs veiligheidsrisico's. Asset managers hebben vaak geen totaal overzicht van informatie en moeten asset informatie uit verschillende systemen opzoeken, wat een onproductieve en vaak tijdrovende taak is. Tegelijkertijd betekent dit dat zij geen volledig inzicht hebben in alle kenmerken en geschiedenis van bijvoorbeeld één bepaalde asset, waardoor het moeilijk wordt om goed geïnformeerde beslissingen te nemen, wat weer kan leiden tot extra kosten en verloren tijd. Tegelijkertijd beschikken monteurs hierdoor vaak niet over het juiste gereedschap en informatie om problemen in één keer op te lossen wat opnieuw leidt tot extra kosten.

Linked Data en semantic web technologieën krijgen steeds meer aandacht in de sector. De infrastructuursector heeft de voordelen van linked data al onderkend, met bijvoorbeeld het gebruik van het Resource Description Framework (RDF) voor de uitwisseling van gegevens. Daarentegen moet het asset management in de woning- en utiliteitsbouw het potentieel van de technologie nog onderkennen. De verwachting is dat het Rijksvastgoedbedrijf en gemeenten in de toekomst ook zullen verwachten dat projecten in RDF-formaat worden opgeleverd. Naast linked data is het digital twin concept ook een veelbelovende oplossing voor het verbeteren van de manier waarop assets worden beheerd. Door het creëren van digitale tweelingen, een virtuele replica van een fysiek object, kunnen asset managers beter inzicht krijgen in hoe een gebouw of asset functioneert, waardoor het gemakkelijker wordt om weloverwogen beslissingen te nemen.

Het doel van dit onderzoek is het creëren van een digitaal asset management logboek voor gebouw en om hierin de toegevoegde waarde van een "linked data" aanpak te onderzoeken. In dit onderzoek wordt gebruik gemaakt van historische data vanuit een SAP ERP-systeem en gebouwinformatie uit BIM-modellen voor het creëren van één geïntegreerde "graph database". Het onderzoek draagt hierdoor bij aan het verbeteren van het beheer en de besluitvorming in asset management, met het doel om onderhoudskosten te verminderen, het verhogen van de prestaties van assets en het verbeteren van de veiligheid.

Na gesprekken met verschillende stakeholders binnen het asset management werd duidelijk dat het systeem moet helpen bij het creëren van een totaaloverzicht van informatie, het inzicht in de onderhoudsgeschiedenis van specifieke assets en ruimten moet vergroten, en het lokaliseren en 3D visualiseren van assets en ruimtes. Ten eerste moet in plaats van data in verschillende systemen, grootschalige analyse van gecombineerde data mogelijk worden gemaakt om nieuwe inzichten te verkrijgen die kunnen helpen bij het optimaliseren van de preventieve onderhoudsplanning. Door het optimaliseren van de onderhoudsplanning wordt het aantal correctieve onderhoudshandelingen zoals

het oplossen van storingen verminderd. Natuurlijk zal het aantal storingen nooit tot nul worden gereduceerd, dus voor de storingen die zich toch voordoen, moet het systeem in staat zijn om zo veel mogelijk potentieel waardevolle informatie uit historische gegevens over een asset, ruimte of specifiek probleem kunnen verstrekken. Dit vermindert de tijd die werkvoorbereiders en monteurs nodig hebben om de vereiste informatie op te zoeken. Zo worden medewerkers ondersteund bij het nemen van betere en snellere beslissingen over het oplossen van storingen op basis van historische gegevens, in plaats van keuzes op basis van ervaring of intuïtie. Om te bepalen waar het systeem minimaal aan moest voldoen, zijn er meerdere gesprekken met maintenance engineers, werkvoorbereiders en data adviseurs gehouden wat resulteerde in een lijst van elf “competency questions”.

Vervolgens is een systeemarchitectuur voorgesteld en is er een data pipeline ontwikkeld die gegevens uit de bestaande systemen integreert om analyses te doen in twee verschillende toepassingen. De pipeline begint met het verkrijgen van de data uit de huidige informatiesystemen. In dit onderzoek gaat het over een SAP ERP-systeem voor het beheren van onderhoudsdata en een “common data environment” met BIM modellen in IFC formaat. De volgende stap is het omzetten van de gegevens in een formaat dat geschikt is voor toepassingen in het semantic web. Voor de conversie van de onderhoudsdata naar RDF is eerst de “maintenance ontology” ontwikkeld, die de structuur en relaties van de gegevens definieert, om ervoor te zorgen dat de gegevens goed gestructureerd en georganiseerd zijn voor gebruik in een graph database. Vervolgens is een python-script ontwikkeld dat de data vanuit SAP transformeert naar RDF-triples in het template van de maintenance ontology. Voor de transformatie van de IFC modellen naar RDF worden de BOT- en PROPS ontologies gebruikt. De IFCtoLBD-converter is gebruikt om de modellen te converteren naar TTL bestanden. Om vervolgens de alle data te integreren zijn er met behulp van een python script nieuwe relaties gevormd tussen assets en ruimtes uit SAP en elementen en ruimtes uit BIM. Ten slotte worden twee applicaties voorgesteld. De eerste tool, met Ontotext GraphDB als back-end, is bedoeld voor asset managers, maintenance engineers of planners die bekend zijn met de gegevens en geavanceerde query's willen uitvoeren die normaal gesproken niet mogelijk zijn in een relationele database. LBDviz is een webapplicatie voor het visualiseren van RDF-data in een IFC-viewer, voor planners en monteurs die minder bekend zijn met het achterliggende dataschema. De applicatie heeft verschillende voor-ingestelde functionaliteiten in een gebruiksvriendelijke interface om snel inzichten te genereren.

Als laatste stap zijn de systeem architectuur en data pipeline getest in een case study van het Europese medicijn agentschap (EMA) en gevalideerd door het beantwoorden van de competency questions die waren opgesteld tijdens de interviews. Alle elf competency questions konden worden beantwoord met query's uit de triple store Ontotext GraphDB. Aanvullend zijn drie competency questions ook beantwoord met behulp van de webapplicatie LBDviz. De linked data aanpak maakt het mogelijk intuïtiever te zoeken naar de gehele context van één data punt zoals een asset of ruimte en wordt het mogelijk suggesties te verkrijgen voor het oplossen van nieuwe storingen, gebaseerd op historische data zodra deze worden gemeld. Asset managers en maintenance engineers kunnen complexe query's uitvoeren die in traditionele relationele databases niet mogelijk zijn. Ook werkvoorbereiders en monteurs hebben met behulp van LBDviz toegang tot snelle inzichten over onderhoudsdata van bepaalde assets en ruimtes. Door het ondersteunen van besluitvorming wordt de prestatie en levensduur van assets verbeterd en wordt er uiteindelijk veel tijd en geld bespaard.

## Abstract

The Architecture, Engineering, and Construction (AEC) industry is rapidly transforming, which is caused by the increasing use and development of digital technologies. However, despite advancements in the design, engineering and construction phases, there is room for improvement, in the way assets are managed by different types of maintenance parties. Compared to the costs made during the initial design and construction phase, the operations and maintenance (O&M) costs are significantly higher, so potential advantages of new digital developments in this phase are significant. Linked Data and semantic web technologies are gaining more attention in the industry. The infrastructure sector has already recognized the benefits of linked data, with for instance the use of the Resource Description Framework (RDF) for interchanging data. However, building contractors have yet to realize the full potential of this technology for their asset management of buildings. This research aims to create a digital building logbook for a building and investigate the added value of a linked data approach for asset life cycle information management. The research will leverage historical data from an SAP maintenance management ERP database and building information from BIM models to create a graph database that integrates all asset information. In this research, a system architecture and data pipeline are proposed to integrate data from existing maintenance management and BIM systems in order to do derive new insights using two applications, which can help informed decision making in asset maintenance management. Asset managers and maintenance engineers have the ability to run complex queries that are not possible or really challenging in traditional relational databases. With a 3D user interface, also planners and technicians have access to quick insights on important data about assets and rooms. The implementation of linked data will increase efficiency, reduce costs, and improve the overall health and performance of buildings and equipment.

**Keywords:** Linked Data, Semantic Web Technologies, IFC, Asset Management.

## List of Abbreviations

AEC	Architecture Engineering and Construction
AEC-FM	Architecture Engineering Construction Facility Management
AIM	Asset Information Model
API	Application Programming Interface
AM	Asset management
BIM	Building Information Modelling
BOT	Building Topology Ontology
bsDD	buildingSMART Data Dictionary
CAD	Computer Aided Design
CDE	Common Data Environment
DB	Database
DT	Digital Twin
EMA	European Medicines Agency
ERP	Enterprise Resource Planning
GUID	Global Unique IDentifier
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
HVAC	Heating, Ventilation, and Air Conditioning
ICDD	Information Container for Linked Document Delivery
IDM	Information Delivery Manual
IDS	Information delivery manual
IFC	Industry Foundation Classes
IoT	Internet of Things
ISO	International Standards Organisation
JSON	JavaScript Object Notation
LBD	Linked Building Data
MMS	Maintenance Management System
MO	Maintenance Ontology
OTL	Object Type Library
OWL	Web Ontology Language
O&E	Ontwerp & Engineering
PIM	Project Information Model
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
RVB	Rijksvastgoedbedrijf
RWS	Rijkswaterstaat
SHACL	Shapes Constrained Languages
SPARQL	SPARQL Protocol and RDF Query Language
SPF	STEP Physical File Format
STEP	Standard for the Exchange of Product Data
SQL	Structured Query Language
Turtle	Terse RDF Triple Language
UI	User Interface
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
WIP	Work In Progress
W3C	World Wide Web Consortium
XML	Extensible Markup Language

## List of Figures

Figure 1: Research Outline .....	20
Figure 2: Information flow throughout the project lifecycle (EFCA, 2019) .....	21
Figure 3: BIM Maturity Model (copyrighted image Bew & Richards, 2008) .....	22
Figure 4: ISO 19650-2 Project Information Model Diagram .....	23
Figure 5: ISO 19650-3 Asset Information Model Diagram .....	24
Figure 6: ICDD Folder Structure .....	26
Figure 7: bSDD example of Doorset .....	27
Figure 8: The IAM's Conceptual Management Model .....	28
Figure 9: Maturity Levels of Asset Maintenance .....	28
Figure 10: Total of Maintenance Cost .....	29
Figure 11: Taxonomy for Required Information for Asset Management .....	30
Figure 12: Types of System Behavior .....	31
Figure 13: Levels of Integration for Digital Twins. ....	31
Figure 14: W3C Semantic web stack .....	38
Figure 15: Example of an RDF Graph .....	38
Figure 16: Schematic Visualization of System Architecture .....	47
Figure 17: Schematic Visualization of Data Pipeline .....	48
Figure 18: Schematic Visualization of Data from the Current Systems .....	49
Figure 19: Schematic Visualization of Transforming Data to RDF Format .....	50
Figure 20: Simplified visualization of the Maintenance Ontology .....	52
Figure 21: Simplified Visualization of the BOT Ontology .....	52
Figure 22: Combined Ontologies with an Example of Instance Data .....	53
Figure 23: Two Ways to Convert IFC to RDF .....	59
Figure 24: Schematic Visualization of the Integration of Data .....	61
Figure 25: Schematic Visualization of Presentation Layer .....	64
Figure 26: Photos and Factsheet of the European Medicines Agency in Amsterdam .....	69
Figure 27: 3D Visualization of the Congress IFC Models: Structural, Architectural, HVAC/MEP .....	70
Figure 28: Context of Asset_30000202560_Ruimtebedienerheid .....	76
Figure 29: Electrotechnical Orders with the Cause of Degradation .....	76
Figure 30: 3D Visualization of the Architectural, Electrical, and Interior IFC Models for LBDviz .....	77
Figure 31: Asset History Function In LBDviz .....	78
Figure 32: Room History Function in LBDviz .....	78
Figure 33: Element Properties Function in LBDviz .....	79

## List of Tables

Table 1: Seven Concepts specified in ISO 19650 .....	23
Table 2: Five Tables of the NL-SfB Classification .....	25
Table 3: Digital Twin Applications in Different Industries.....	32
Table 4: Overview of Articles related to Digital Twins in the AEC Industry. Categorized by application area .....	34
Table 5: Articles related to Digital Twins in Asset Management.....	36
Table 6: Query results .....	40
Table 7: Requirements/ Competency questions .....	45
Table 8: Definitions for Nodes in the Maintenance Ontology.....	51
Table 9: Definitions for Data Properties in Maintenance Ontology .....	51
Table 10: Modified Functionalities in the LBDviz Application .....	66
Table 11: IFC Subdivision with File Sizes .....	70
Table 12: Output TTL file with File Sizes and Number of Triples .....	70
Table 13: File Sizes and Number of Triples for BIM Models of XXL and Multifunctional Room .....	77

## List of Listings

Listing 1: SPARQL query .....	40
Listing 2: Pseudocode fragment showing part 1 and 2.1 .....	55
Listing 3: Pseudocode fragment showing part 2.2 .....	55
Listing 4: Example of Notification .....	55
Listing 5: Pseudocode fragment showing part 3 and 3.2 .....	56
Listing 6: Pseudocode fragment showing part 3.1 and 3.3. ....	56
Listing 7: Pseudocode fragment showing part 4 .....	56
Listing 8: Pseudocode fragment showing part 5.1, 5.2, and 5.3 .....	57
Listing 9: Pseudocode fragment showing part 6 until 10.....	58
Listing 10: Snippet of output from the script .....	58
Listing 11: Modification to convertIFCSFPtoTTL function .....	60
Listing 12: modification to cleanNameString function .....	61
Listing 13: Pseudocode Fragment showing the MO:hasbotElementOrigin relationships .....	62
Listing 14: Pseudocode Fragment showing the MO:hasbotSpaceOrigin relationships .....	63
Listing 15: Final part of combined python script where functions are called .....	63
Listing 16: Example of python script that checks for updated IFC models .....	64
Listing 17: Pseudocode fragment showing the queryComunicaGlobalIDProps functionality.....	67
Listing 18: SPARQL query for number of triples .....	70
Listing 19: SPARQL query for competency question 1 .....	72
Listing 20: SPARQL query for competency question 2 .....	72
Listing 21: SPARQL query for competency question 3 .....	73
Listing 22: SPARQL query for competency question 4 .....	73
Listing 23: SPARQL query for competency question 5 .....	73
Listing 24: SPARQL query for competency question 6 .....	73
Listing 25: SPARQL query for competency question 7 .....	74
Listing 26: SPARQL query for competency question 8 .....	74
Listing 27: SPARQL query for competency question 9 .....	75
Listing 28: SPARQL query for competency question 10 .....	75





## 1 Introduction

This introduction starts with introducing the background and specific subject for this research. subsequently, the problem statement for this thesis is introduced, after which multiple research questions to solve the problem are created. The introduction continues with defining thesis objectives and explaining the scientific relevance. The introduction ends with the outline for the rest of this document.

### 1.1 Background

The Architecture, Engineering, and Construction (AEC) industry is rapidly transforming, which is caused by the increasing use and development of digital technologies. The emergence of Building Information Modeling (BIM) (Eastman et al., 2011) allowed structured storage and use of data, making it essential for effective asset management. It has caused a paradigm shift of the industries in ways to define, tailor, and manage the semantics of product models closely linked to geometry. Industry domains and software developers are therefore becoming more interested in organizing and sharing the "semantics" of a building throughout design, construction, facility management, building engineering, HVAC design, simulation, renovation, and demolition. In the early stages of BIM adoption was only limited to software applications that display geometric views of buildings, combined with lengthy textual descriptions and unstructured spreadsheet data (Eastman, 1979). Later, the industry made significant progress towards creating a robust and structured semantic framework, as well as an organized map of semantic connections (Pauwels et al., 2017).

However, despite advancements in the design, engineering and construction phases, there is room for improvement, in the way assets are managed by different types of maintenance parties. Compared to the costs during the initial design and construction phase, the operations and maintenance (O&M) costs are significantly higher. In fact, these ongoing costs amount to several times the original construction costs (Becerik-Gerber et al., 2012). According to Hoang et al. (2020), even 85% of the total building life cycle costs are spent during the operational phase. The exploitation phase, which can last 50 years or even longer, is also the phase with the greatest duration. The potential advantages of digitizing the exploitation phase are significant because it is the most crucial stage in the life cycle of a building. To enable interoperability with systems beyond the AEC sector, the widely used Industry Foundation Classes (IFC) standard on its own is insufficient (Curry, et al., 2013). Other data sources related to asset management are usually stored locally and are not able to connect with each other. As a result, Linked Data and semantic web technologies are gaining more attention in the industry. The infrastructure sector has already recognized the benefits of linked data, with for instance the use of the Resource Description Framework (RDF) for interchanging data. However, building contractors have yet to realize the full potential of this technology for their asset management of buildings. The expectation is that Rijksvastgoedbedrijf (RVB) and municipalities will expect all projects in RDF format in the future. For the AEC industry to comply to these demands in the future, two things are needed. More research on the application of semantic web technologies in the AEC industry and some ambition by companies to actually make changes in the industry. Using semantic web technologies, interoperability problems can be tackled, by separating the actual data from its authoring tools and relying on data representation in a Linked (Open) Data format (Pauwels, 2014). Semantic digital twins have been identified as a promising solution for improving the way in which assets are managed. By creating digital twins, a virtual replica of a physical asset, asset managers can gain better insights into how a building or asset operates, making it easier to make informed decisions.

### 1.2 Problem statement

The operational costs of building ownership are high, with a significant portion attributed to the time and resources spent on recollecting information about the building for maintenance (Hoang et al.,

2020). The Architecture, Engineering, and Construction (AEC) industry faces challenges in managing large amounts of data from various sources in different formats, hindering effective asset management. Despite the availability of Building Information Modeling (BIM) systems, reactive maintenance remains common, resulting in frequent malfunctions, complaints, and increased costs. To ensure efficient and effective building operation, it is essential to focus on information from the building's systems, equipment, and components, as well as maintenance schedules, performance metrics, and energy usage in the operations phase. Without comprehensive asset information, it is difficult for asset managers to make informed decisions regarding maintenance, repairs, and upgrades, leading to potential inefficiencies, increased costs, and even safety risks.

However, asset managers often lack a total overview of information and have to obtain asset information from various systems, which is a time-consuming and often unproductive task. At the same time, this means that they lack a comprehensive understanding of all the characteristics of the assets from all the systems. Potentially, this can lead to errors and inconsistencies in the data, which makes it difficult to take informed decisions, leading to delays and decreased productivity. As a result, technicians often lack the necessary tools and information to resolve issues in one time (first time fix ration), which again leads to delays and additional costs.

To address these challenges, Linked Data and digital twins have emerged as promising technologies for asset management, offering a comprehensive view of an asset's life cycle and enabling more informed decision-making and proactive maintenance. This research aims to assess the current processes, data flows, and their correlations in the AEC sector and to improve the interoperability between maintenance management system (MMS) and BIM data, using digital twin and semantic web technologies to enhance asset management and reduce costs caused by indirect productivity. Considering the problem analysis, the following research question has been created:

*“How can the implementation of linked data and semantic digital twins improve decision-making in asset management during the operation and maintenance phase?”*

To answer the main research question, it is necessary to address the following sub-questions, which will provide guidance for organizing the thesis and serve as individual stages in the research process:

- RQ1 What is the current state-of-the-art literature with regards to semantic digital twins and linked data in asset management?
- RQ2 Which specific insights are currently challenging and time-consuming to obtain using existing systems?
- RQ3 What are the key characteristics and sources of data commonly used in asset life cycle information management?
- RQ4 How can the data originating from different systems be transformed into a format suitable for semantic web technologies?
- RQ5 What strategies can be utilized to merge data from multiple systems effectively?
- RQ6 What specific methods or tools can be utilized by different end users in the asset management team to generate insights from the data?

### 1.3 Thesis objectives

This study aims to create a digital building logbook or digital twin for a building and investigate the added value of a linked data approach for asset life cycle information management. The research will leverage historical data from an SAP maintenance management ERP database and building information from BIM models to create a graph database that integrates all asset information. By using a graph database, the study will explore the advantages of handling dynamic data and analyzing an asset's entire context. The study will evaluate the effectiveness of the digital twin in improving asset

management and decision-making, including the potential for reducing maintenance costs, increasing asset performance, and improving safety. The results will contribute to the development of best practices for digital twins in asset management in the AEC industry, improving the efficiency and effectiveness of maintenance management systems and providing insights into the benefits of semantic web technologies for asset life cycle information management. Finally, in order to improve the findability and accessibility of data, this study aims enhance the existing information structures. This will be achieved by proposing a system architecture and standardized data pipeline for the transformation and integration of data from the existing systems.

## **1.4 Scientific Relevance**

First of all, this research will contribute to the ongoing research about implementing semantic web technologies and digital twins in the built environment. Decision-making in the AEC industry is a complex task that involves a large number of stakeholders, which generate all kinds of data in their preferred formats, use their own domain languages, and store data in decentral storage locations. Therefore, it is difficult to communicate which results in decision-making based on incomplete and inconsistent information. This research will bridge the gap between data from the maintenance management system and data from BIM using linked data, enabling diagnostic analysis, which will set steps in the direction of predictive maintenance.

## **1.5 Outline**

This research starts with an extensive review of the state-of-the-art literature regarding semantic digital twins and linked data in asset management (Chapter 2) and will give answers to the first sub-question. The literature review begins with elaborating on Building Information Modelling (Section 2.1) and standards and classifications in the AEC industry (Section 2.2). Subsequently, different strategies and terminology in asset management are discussed (Section 2.3). furthermore, the literature on semantic digital twins is reviewed (Section 2.4) and finally, Section 2.5 elaborates on linked data and semantic web technologies.

The conclusions from the literature review are considered within the next chapter, namely the methodology (Chapter 3). This chapter involves the research approach (Section 3.1) and expert interviews to set up requirements and competency questions (Section 3.2) for the prototype development of a system architecture and data pipeline that integrate the SAP maintenance management system and BIM data (Section 4). For this development, ways to convert IFC to LBD and MMS to RDF are explored, after which python is used to integrate all data into one decentral location for data analysis. The chapter is finalized with a short conclusion (Section 4.6).

The system architecture and data pipeline are later tested and validated with a case study (Section 5). In this chapter the data from the maintenance project EMA (European Medicines Agency) is used to answer the competency questions from Section 3.2.

Finally, the research ends with the Conclusion (Chapter 6) and Discussion (Chapter 7), which starts with elaborating on the interpretations and implications (Section 7.1) on the results. The discussion ends with describing limitations (Section 7.2) and setting out Recommendations (Section 7.3). The full research outline is visualized schematically in Figure 1.

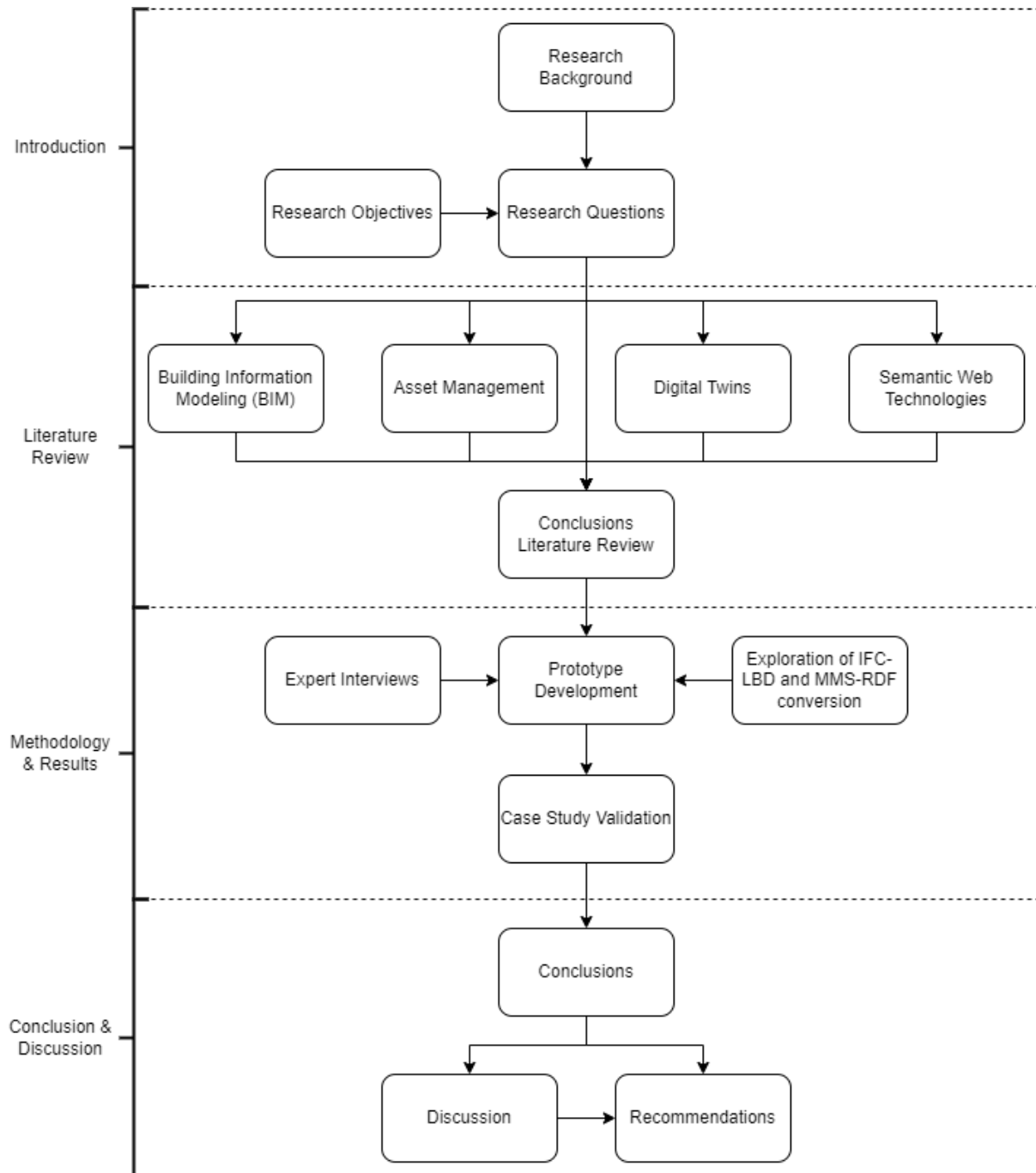


Figure 1: Research Outline

## 2 Literature Review

This chapter includes an extensive review of the state-of-the-art literature regarding semantic digital twins and linked data in asset management and will give answers to the first sub-question: *“What is the current state of the art literature with regards to semantic digital twins and linked data in asset management?”* The literature review begins with elaborating on Building Information Modelling (Section 2.1) and standards and classifications in the AEC industry (Section 2.2). Subsequently, different strategies and terminology in asset management are discussed (Section 2.3). Furthermore, the literature on semantic digital twins is reviewed (Section 2.4) and finally, Section 2.5 elaborates on linked data and semantic web technologies.

### 2.1 Building Information Modelling (BIM)

In building projects within the AEC industry, multiple parties or stakeholders collaborate to create buildings. In bigger projects, the number of parties can grow and change really fast. Furthermore, the interdisciplinary stakeholders all have other preferred software packages (Bertelsen, 2003). However, they all need to work together and exchange information over the entire life cycle of a building or asset. Throughout the entire life cycle of a building, significant amounts of data are generated, transmitted, and analysed, which causes the need for smart ways of collaborating. Building design and management tools have progressed for decades, starting from computer-aided design (CAD) in the 1970's. It is until the early 90's that the construction industry started to shift slowly from 2d to 3d design. In 1992, the phrase "Building Information Model" was first proposed by van Nederveen and Tolman (1992). Later it was referred to as "Building Information Modelling" (BIM) and has gained much popularity in recent years (Li et al. 2017). A building information model (BIM) is a digital representation of a facility that is data-rich, object-oriented, intelligent, and parametric. Views and data that are suitable for different users' needs can be extracted from the BIM model and analysed to produce information that can be used to make decisions and improve the process of delivering the facility (AGC, 2005). Figure 2 illustrates the information loss between stakeholders across several project phases using a traditional process compared to BIM, highlighting the significance of interoperability (EFCA, 2019). By using the BIM approach for construction projects, the information loss moving through different phases is reduced significantly. In an ideal information flow, there is no information loss, however, because information loss is a characteristic of information exchange, it will always remain. BIM as a method has become widely accepted in the construction sector in recent years, causing a significant shift toward full digitization (Quirk, 2012).

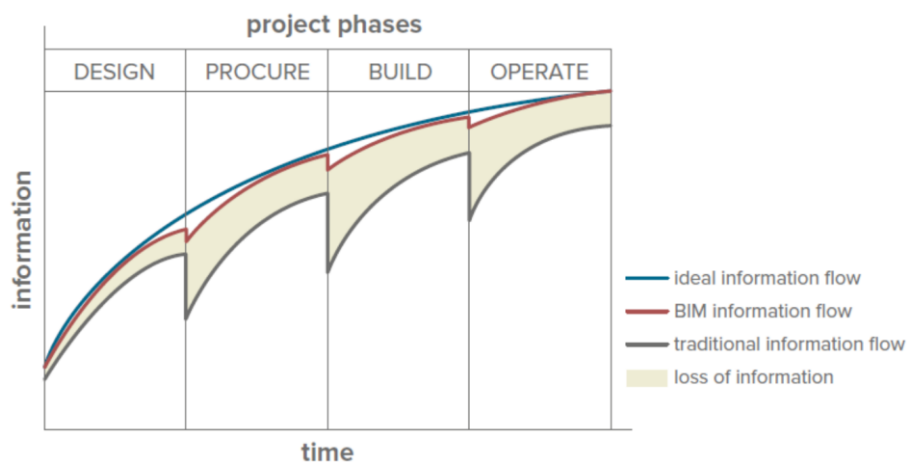


Figure 2: Information flow throughout the project lifecycle (EFCA, 2019)

The UK BIM maturity model, created by Bew & Richards (2008), is a frequently used model to evaluate the application of BIM. In this model, there are four stages of BIM maturity, with the higher levels describing situations with deeper BIM integration (See Figure 3). In maturity level 0, also referred as the "Pre-BIM"-phase, data exchange is not coordinated at all. In Level 1, different parties collaborate by exchanging on 2D and 3D geometric files, which is stored locally. BIM level 2 describes a situation where companies work with complex BIM models, which are shared with other parties and disciplines within a common data environment (CDE). The highest maturity level (3) proposes the use of an integrated solution built around open standards like Industry Foundation Classes (IFC) where a single server stores all the project data. All information is exchanged over the web and is integrated between all disciplines and parties. Interoperable and decentralised model servers allow collaborating on interoperable models. Instead of exchanging files organizations are shifted to object-based collaboration. Depending on the location of the world or the size of projects, the industry is currently located at level 0, 1, or 2. Today, BIM level 3 adoption is still very rare. The biggest obstacles to achieve BIM level 3 are cost, culture, expertise, technology and interoperability, lack of client demand, and legal issues (Ayinla & Adamu, 2017).

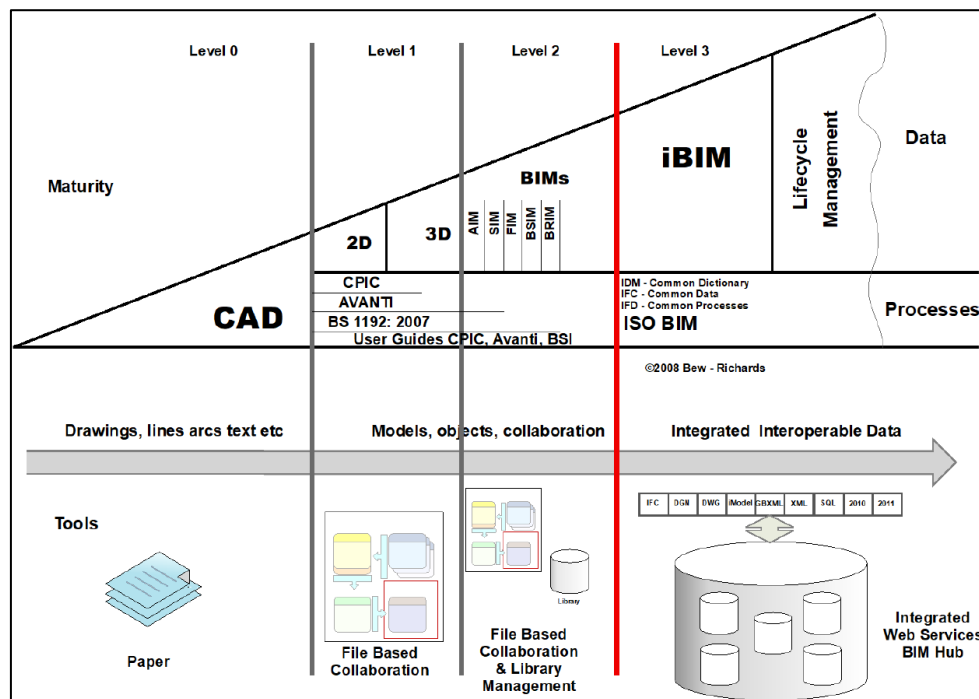


Figure 3: BIM Maturity Model (copyrighted image Bew & Richards, 2008)

### 2.1.1 ISO 19650

The use of BIM is becoming more and more standardized. A good example of this is the ISO 19650. The international standard ISO 19650 is regarded worldwide as the standard for the management of digital information in the life cycle of construction works, according to BIM level 2 of the UK BIM maturity model. The ISO 10650 specifies seven concepts which are displayed in Table 1. ISO 19650 is largely based on the older standards PAS-1192 and BS1192. Late in 2018, the European Standards Committee for BIM (CEN/TC 442) published the first two parts of this standard (ISO, 2019). ISO 19650 describes the collaborative processes for effective information management. This concerns the entire delivery and operational phase of a building and terms such as Level of Information Need (LOIN), Exchange Information Requirements (EIR), BIM Execution Plan (BEP) and Common Data Environment (CDE) are explained in detail (ISO,2019). According to the standard, information is stored in three ways: documentation, non-graphical data, and a graphical model. As the project stages advance, the

information in the CDE becomes more complex until the point of handover, when the entire data set is handed over to the building operator. Figure 4 displays the information and management process during the entire project. ISO 19650 describes multiple data drop or document delivery moments in which files and documentation is published and released. These data drops usually take place when a project progresses from one stage to another. The amount of data that is stored in the project information model (PIM) increases over time, until the project is handed over to the client or building operator.

Table 1: Seven Concepts specified in ISO 19650

1	How does information flow throughout the delivery and operational phase of an asset?
2	To which BIM level does the project comply?
3	Which requirements are set to information?
4	Where is information stored?
5	Which roles can be distinguished?
6	How does ISO 19650 relate to other norms?
7	Which level of information is necessary?

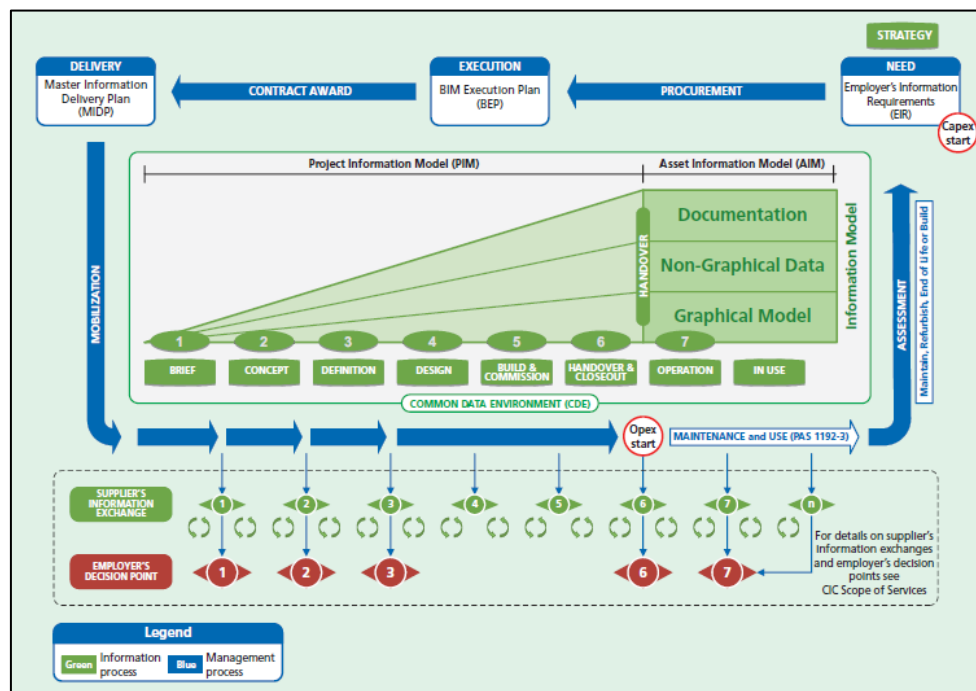


Figure 4: ISO 19650-2 Project Information Model Diagram

Compared to the delivery phase, the information model is called an Asset Information Model (AIM) in the operational phase. The information flow in the operations phase is displayed in Figure 5. The operations phase, which is less rigid than the project phase, covers a variety of scheduled and unforeseen events in the lifecycle of an asset, including maintenance, breakdowns, repairs, extensions, refurbishment or eventually demolition. Since they are non-consequential triggers, they can occur in any order. The asset information model must be updated whenever a trigger takes place. When all that information is available, building owners or building operators will be able to properly understand buildings (ISO, 2019). An essential aspect of this standard is that it implies that the biggest part (usually 80%) of a building's information is gathered and used in the operation and maintenance phase. Therefore, more emphasis should be on information on assets after projects are delivered to the client.



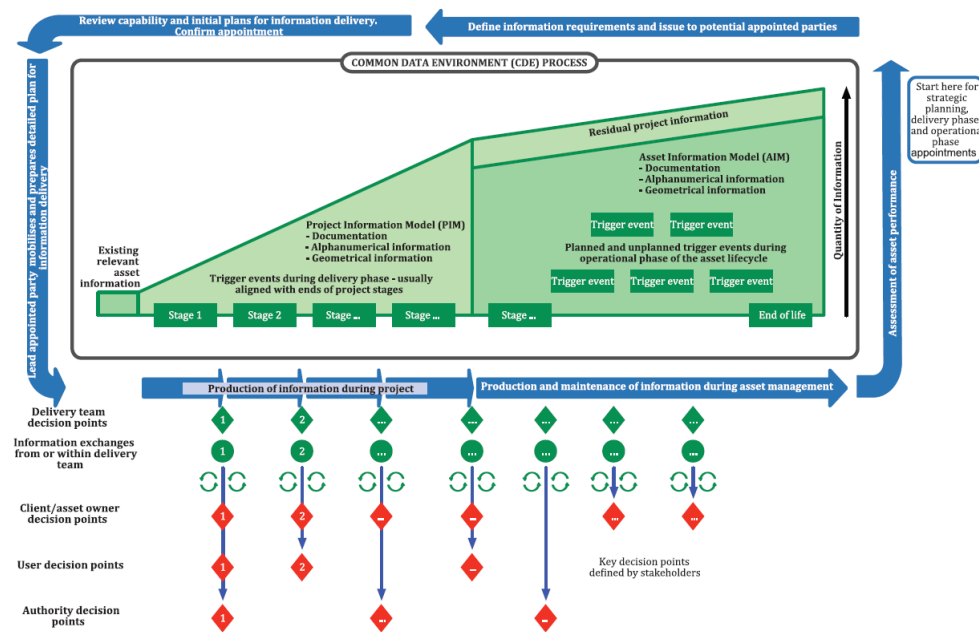


Figure 5: ISO 19650-3 Asset Information Model Diagram

## 2.2 Standards and Classifications used in the AEC industry

### 2.2.1 IFC

In terms of information exchange, the Industry Foundation Classes (IFC) is the accepted schema exchanging BIM data (ISO, 2018). "IFC is a standardized, digital description of the built asset industry. It is an open, international standard (ISO 16739-1:2018) and promotes vendor-neutral, or agnostic, and usable capabilities across a wide range of hardware devices, software platforms, and interfaces for many different use cases". The IFC standard is created and maintained by the buildingSMART organization and is open source. (buildingSMART International, 2019). The IFC data model allows complete information sharing between members of the project team as well as between the software programs they frequently use for design, construction, procurement, maintenance, and operations. For the exchange of building and facility management data among multiple AEC/FM industry applications, IFC serves as a data standard data schema. The schema is object-oriented and based on class definitions.

SPF (STEP Physical File Format) is the most common format, but it can also be encoded in other formats like XML, JSON, SPF, or RDF. STEP or "Standard for the Exchange of Product model data" is also referred to as ISO 10303-242 (ISO, 2014). The schema is described in the EXPRESS information modeling language (ISO, 2004). A STEP subdivided into two parts: A header with information about the file and the data where all entity instances are described. IFC can be handled in centralized or connected databases, imported, or exported as files, or sent using web services. IFC can also be used to archive project data, either progressively throughout the design, acquisition, and construction phases, or as an "as-built" repository of data for long-term preservation and operations.



### 2.2.2 Information Delivery Standards

In construction projects, effective collaboration between various companies is essential for successful project execution. Proper communication and coordination of information is crucial to maintain efficiency, particularly when utilizing digital tools that often have strict requirements for the interpretation of digital data. To ensure smooth operation, all stakeholders must be aware of the type and timing of information that must be communicated. The information delivery manual (IDM<sup>1</sup>) “ILS O&E” enables to communicate the information need for the design and engineering phase unambiguously and recognizably.

An information delivery manual for construction is a document that outlines the processes and procedures for delivering important information to stakeholders involved in a construction project. The manual includes guidelines for how to communicate project updates and how to share project plans and documents. The manual can help ensure that all stakeholders are kept informed and up to date throughout the construction process, which can help prevent misunderstandings and potential delays. Additionally, having a clear and comprehensive information delivery manual can help improve collaboration among all parties involved in the construction project, ultimately leading to a successful and efficient completion of the project.

An international and more recent alternative is the Information Delivery Specification (IdS), which is “a computer interpretable document that defines the Exchange Requirements of model-based exchange” (buildingSMART, 2023). It combines IFC with domain extensions and additional classifications and properties, ideally stored bSDD, which is explained in Section 2.2.5. Traditionally, information requirements are shared in ways that are not computer interpretable, such as excel sheets or PDFs. The benefit of using machine readable documents is the ability to automate processes, such as compliance checking. Similar to the IDM O&E, the IDS allows predictable and reliable data exchange workflows.

### 2.2.3 NL-SfB

NL/SfB<sup>2</sup> is a classification of installations and building parts, or “elements”, which is frequently used in the AEC industry when designing, realizing, and managing buildings. For many years, NL/SfB has been used to arrange data from building product suppliers as well as to encode layers and objects in BIM and CAD systems. NL/SfB is also utilized in many other standards for a variety of purposes, such as estimating construction costs and assessing the state of buildings and installations. With the introduction of BIM, the application of the NL/SfB classification has increased. Nowadays, almost all BIM projects encode their objects in accordance with the four-digit NL/SfB element coding, as it is part of the IDM D&E. This allows project members to filter and sort object information in BIM models. The NL-SfB is based the original Swedish SfB classification and dates from 1947. The current NL-SfB consists out of 5 tables which are displayed in Table 2:

Table 2: Five Tables of the NL-SfB Classification

Table 0	Spatial facilities
Table 1	Functional building elements
Table 2	Construction methods
Table 3	Construction resources
Table 4	Activities, characteristics, and properties

<sup>1</sup> [https://www.bimloket.nl/documents/Introductiepresentatie\\_ILS\\_Ontwerp\\_\\_Engineering.pdf](https://www.bimloket.nl/documents/Introductiepresentatie_ILS_Ontwerp__Engineering.pdf)

<sup>2</sup> [https://stprodeuwmystabu002stor.blob.core.windows.net/overig/NL-SfB\\_BNA\\_Boek\\_2005-ISBN-10%2090-807626-3-6.pdf](https://stprodeuwmystabu002stor.blob.core.windows.net/overig/NL-SfB_BNA_Boek_2005-ISBN-10%2090-807626-3-6.pdf)

In practice, only table 1 (Functional building elements) is used. Sometimes, an attempt is made to combine tables 2 and 3 with table 1. Objects are then encoded with 6 or more digits instead of the commonly used 4-digit code. Besides the application in BIM models, there are also NEN standards that use the tables. One of those standards is the NEN2699 “Investment and operating costs of real estate”. Another widely used standard is NEN2767 for condition measurements of objects in the exploitation phase, in which the NL-SfB coding is also used.

#### 2.2.4 ICDD

Information Container for Document Delivery (ICDD) is a container format created for data exchange in the built environment. Earlier, part 2 of the ISO 19650 described multiple document delivery moments. ICDD is the international successor of the COINS standard, and its purpose is to allow structured data exchange by connecting different models and to allow the connection with external sources. ICDD is described in ISO 21597 (ISO, 2020) and utilizes semantic web technologies to provide metadata about the contained data, and the linking between these documents delivered within. Ideally, the data is represented in the resource Description Framework (RDF) which will be explained in Section 2.5.2.

The standard consists out of two parts. “The first part defines the container structure and the general linking concepts through the definition of a container ontology, corresponding data types and object properties, along with a linkset ontology with corresponding data types and properties. Part 2 defines additional types of links which form the extended linkset” (Senthilvel et al., 2020). The structure is divided into three folders (see Figure 6). The “ontology” folder includes the data schema, the “linkset” folder contains all links within the ICDD, and finally the “payload” folder contains all the documents. Besides the three folders there is an Index file which summarizes the ICDD container documents and how they are interlinked.

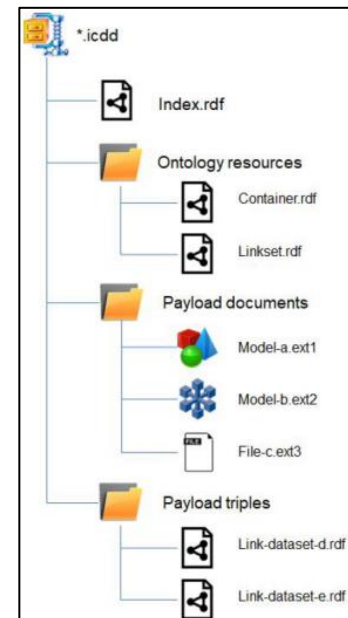


Figure 6: ICDD Folder Structure

#### 2.2.5 BuildingSMART Data Dictionary (bSDD)

The buildingSMART Data Dictionary (bSDD) is a terminology standard for BIM libraries and ontologies that aims to overcome implementation problems, by creating a central with frequently used objects in the AEC industry. This makes it possible to link existing databases to a BIM-model (buildingSMART International, 2022). bSDD is about concepts and not about the names or languages. This is because one concept might have different names in different languages which can cause misinterpretation and ambiguity. A good example of such ambiguities is the concept of a door, which is translated to dør in Norwegian. Norwegians refer to a dør, as a door within a frame, while the English word door refers to a door on its own. The concept of “the door in its frame” in English is referred to as doorset (See Figure 7). These ambiguities often cause problems in BIM projects. Therefore, the bSDD assigns Globally Unique IDs (GUID) to concepts. This way, all stakeholders in a building project can all have their own names in their own language, while they are still talking about the same concept.



Figure 7: bSDD example of Doorset

## 2.3 Asset Management

In this section, the domain of asset management (AM) is explored to gain insight in the goals, strategies, types of maintenance and relevant information in the operations and maintenance (O&M) phase. The section starts with defining the concept of asset management, after which the different maintenance processes are explained. The section ends with the necessary information for asset management.

### 2.3.1 Definition

Asset management can be defined as the systematic and coordinated management of physical assets throughout the entire lifecycle, to make sure their performance is maximized, downtime is reduced, maintenance costs are minimized, and their value is maximized. Asset management includes several activities such as identifying and cataloging assets, monitoring their condition and performance, planning, scheduling maintenance activities, and managing the asset lifecycle from acquisition to disposal (Grussing, 2014). Examples of “assets” are physical components of buildings, such as walls and doors, equipment, machinery, HVAC systems, plumbing, electrical systems, and infrastructure used in organizational operations.

The primary objective of asset management is to extract maximum value from assets. This can be accomplished by assessing opportunities, risks, and desired performance of each asset, to maximize the value of these assets over their lifetime, but also by monitoring their performance and making any necessary repairs or improvements. Finally, this is all done to plan for their maintenance and replacement. Overall, asset management is critical in aiding organizations to optimize asset performance, reduce maintenance costs, minimize downtime, and enhance efficiency, profitability, and customer satisfaction (Grussing, 2014).

A popular model describing asset management activities is the Conceptual Asset Management Model by the Institute of Asset Management (IAM), which is visualized in Figure 8 (Institute of Asset Management, 2014). The model includes 6 subject groups, covering a total of 39 asset management subjects. The model is designed to illustrate the activities within the scope of asset management, the relationships between the activities and the need to integrate them, and finally the critical role for asset management to align with and deliver the goals of an organization’s strategic plan. In the remainder of this research, the focus is on the two subject groups “asset management decision making” and “Asset Information”.

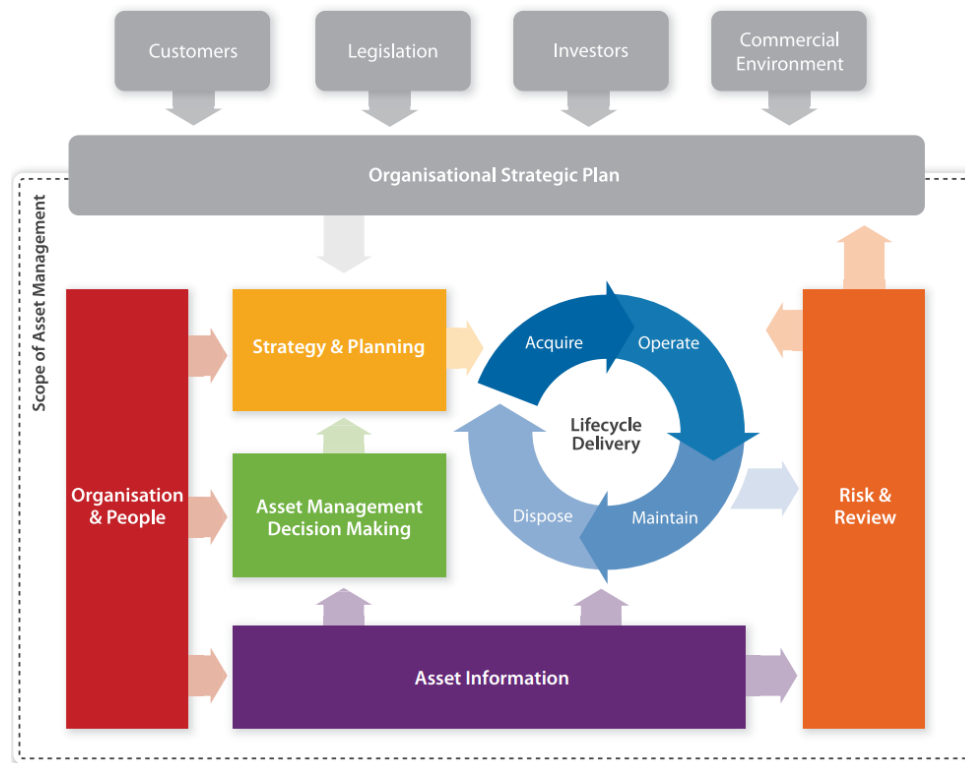


Figure 8: The IAM's Conceptual Management Model

### 2.3.2 Maintenance processes

The built asset industry uses several types or maturity levels of maintenance to ensure the proper functioning and preservation of assets (See Figure 9). Firstly, corrective maintenance involves fixing problems or defects as they arise, typically in response to a breakdown or malfunction (Ruparathna et al., 2018). In regressive maintenance assets are directly replaced when they break. In reactive maintenance assets are fixed in the case of problems or defects. Preventive maintenance, on the other hand, is fixing problems and defects before they arise. In planned maintenance breakdowns are prevented with regular repairs and inspections. Predictive maintenance utilizes data and technology to anticipate and diagnose potential problems, allowing for proactive maintenance and repairs. Finally, prescriptive maintenance goes a step further by using data and analytics to not only predict and diagnose problems, but also to recommend specific actions and maintenance strategies (Ruparathna et al., 2018).

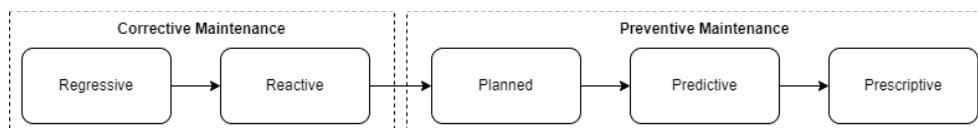


Figure 9: Maturity Levels of Asset Maintenance

The use of these different types of maintenance varies within the industry. Corrective maintenance can be costly and time-consuming. Preventive maintenance is widely recognized as a more effective approach, but it requires regular and thorough inspections and can be resource intensive. Predictive and prescriptive maintenance are growing in popularity, as they allow for more efficient and proactive maintenance planning. However, these approaches require advanced data analysis capabilities and can be complex to implement. Overall, the current state of the industry is moving towards a greater emphasis on predictive and prescriptive maintenance, as these approaches offer the potential for significant cost savings and improved asset performance. However, the adoption of these advanced

maintenance strategies is still in the early stages, and many organizations are still primarily using reactive and planned maintenance.

Douglas (2017) states that the overall expense of maintenance encompasses the sum of the preventive and corrective maintenance costs, as illustrated in Figure 10. To achieve an optimal maintenance zone, it is essential to have an effective preventive maintenance strategy. For instance, by optimizing the maintenance schedule, the corrective maintenance costs are reduced. However, excessive preventive maintenance measures come with high cost, so it is important to achieve a zone where the two expenses are balanced. After obtaining funds for the maintenance budget, it is crucial to allocate them efficiently within the organization to achieve this optimal zone at an operational level.

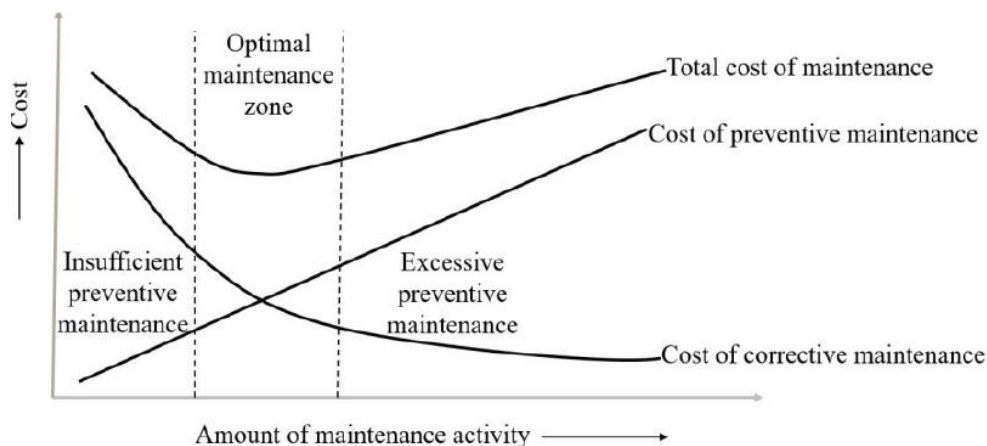


Figure 10: Total of Maintenance Cost

### 2.3.3 Information in AM

In asset management, it is really important to collect data about all assets. Usually, there is data on age, location, condition, and performance of the assets, but also about maintenance history and any potential risks or vulnerabilities. Asset management organizations always aim to enhance the performance and increase the useful life of their assets. To develop and implement effective maintenance strategies, ideally all available data is used. To gather and handle this data, asset management organizations use several procedures and instruments, such as asset registers, inspection reports, and maintenance logs. This data can be stored and analyzed using different software applications, such as maintenance management systems, which can help automating and streamlining several of the maintenance processes. Having complete and accurate data about assets is crucial for effective asset maintenance management. It allows organizations to make informed decisions about maintenance activities and prioritize their resources, ultimately leading to better asset performance and a more efficient and cost-effective maintenance program (Love et al, 2015).

Farghaly et al (2018) investigated how BIM could meet the needs of asset owners during the operation and maintenance phase, with a specific view on asset management. The purpose of their research was to integrate non-geometric BIM data necessary for asset maintenance and establish an appropriate taxonomy (see Figure 11). The taxonomy is divided into six parts, including location/space, classifications, specifications, warranty, assets capex, and maintenance. These six parts are then subdivided into sixty subclasses that describe the BIM data/parameters essential for asset maintenance during the handover stage.

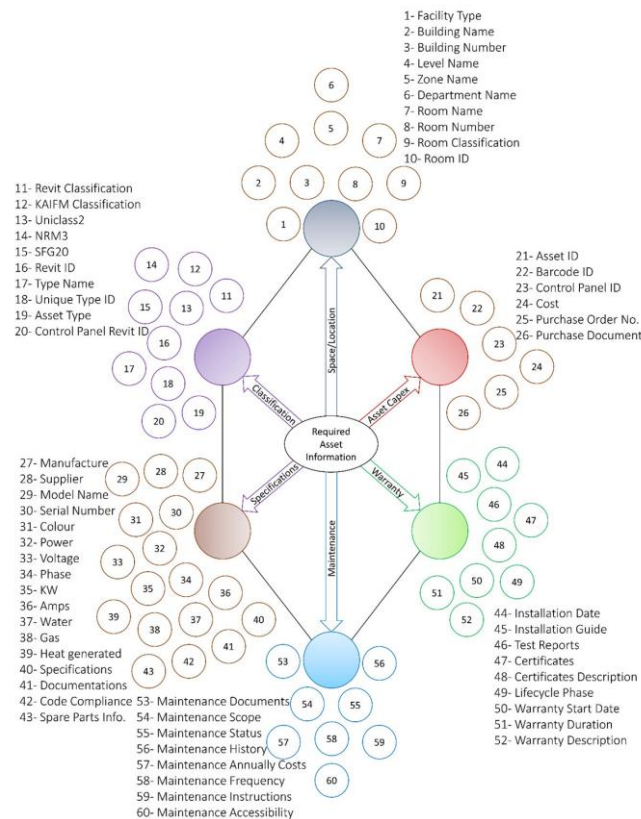


Figure 11: Taxonomy for Required Information for Asset Management

## 2.4 Digital twins

### 2.4.1 Definition

Physical twins such as mock-ups and prototypes have been around for a long time. They have been used for pre-recognizing errors in the final product and making products function and perform as good as possible. However, creating physical twins costs a lot of money and time, and adjustments are difficult. Therefore, from the year 2000, companies have been looking for ways to improve their products and processes through digital models. This became possible due to the growth of the internet, data storage and computing power (Mussomeli et al., 2020). The first definition of a Digital Twin (DT) in literature was established by Michael Grieves in 2003 as: *“The Digital Twin in its original form is described as a digital informational construct about a physical system, created as an entity on its own and linked with the physical system in question. Optimally, the Digital representation should include as much as information as possible concerning the system asset that could be potentially obtained from its thorough inspection in the real world”* (Grieves & Vickers, 2016). Later a more detailed and widely adopted definition was given by Glaessgen & Stargel: *“The Digital Twin is an integrated multi-physics, multi-scale, probabilistic simulation of an as-built vehicle or system that uses the best available physical models, sensor updates, etc., to mirror the life of its corresponding twin”* (Glaessgen & Stargel, 2012). The main principle of the Digital Twin is that continuous iterations are performed to improve an object or process. Therefore, it can be viewed as a cycle where a physical object, containing a lot of information, communicates dynamic data from the physical world to the digital world. The Digital Twin collects the dynamic data and combines it with the historical static data of the object. Analyses and simulations which use the data provide new insights that are used to adjust or improve the physical object.



By introducing Digital Twins, it becomes possible to manage system behaviour. For instance, the design process is focused on verifying and validating requirements (Predicted Desirable – PD) and eliminating problems and conflicts (Predicted Undesirable – PU). With the help of digital twins, it also becomes possible to identify unforeseen conflicts and problems (Unpredicted Undesirable – UU) and unforeseen positive consequences (Unpredicted Desirable – UD) (Grieves & Vickers, 2017) (see Figure 12). Most important is, if the “real world” is accurately modelled and simulated in virtual space, the number of UU’s can be drastically reduced.

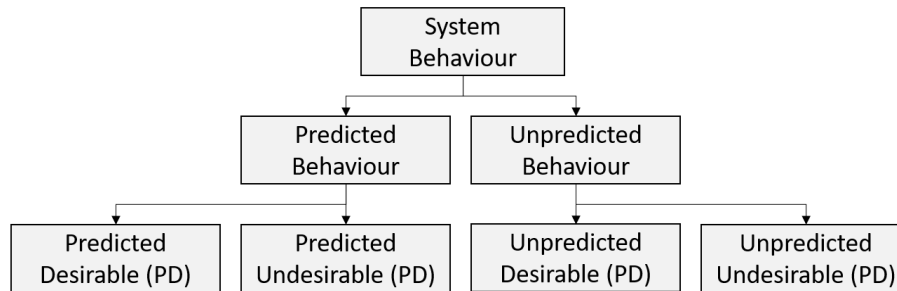


Figure 12: Types of System Behavior

Because there are many different digital twin implementations from different industries, the concept will always be defined differently. (Tao et al. 2017). According to Kritzinger et al. (2018), the terms Digital model, Digital Shadow and Digital Twin are often used interchangeably, while there is a difference between the degree to which data integration takes place between the physical and digital model. Kitzinger et al. (2018) already took a first step to categorize different publications about Digital Twins according to the level of integration. “The ‘Digital Model’ is a digital representation of an existing or planned physical object that does not use any form of automated data exchange between the physical object and the digital object”. In Figure 13, data flow 1 and 2 are manual. Digital data from existing physical systems can still be used to develop such models, but all data exchange is done manually. A change in the state of the physical object has no direct effect on the digital object and vice versa. “A ‘Digital Shadow’ exists when there is an automated one-way data flow between a physical object and a digital object”. A state change of the physical object leads to a state change in the digital object, but not the other way around. Typically, this is particularly useful for keeping logs for processes, such as digital building logbooks. In Figure 13, data flow 1 is automated and data flow 2 is still manual. “If the data flows between an existing physical object and a digital object are fully integrated in both directions, it could be described as ‘Digital Twin’”. In Figure 13, both data flows are automated, so the digital object can also function as a controlling system of the physical object (Kitzinger et al., 2018). An example is the use of smart buildings, with controlling devices in the building.

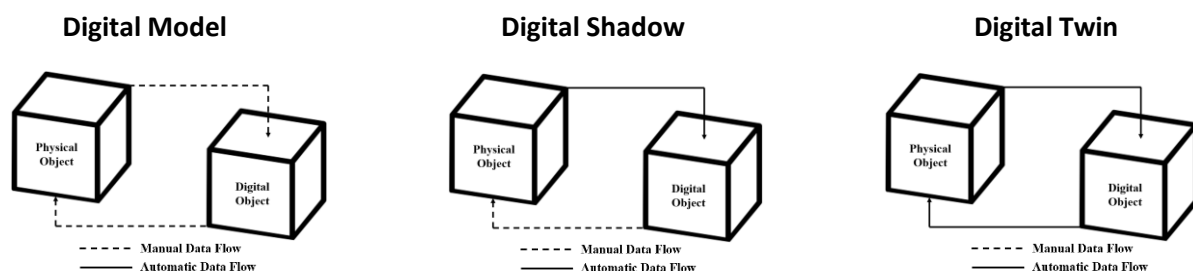
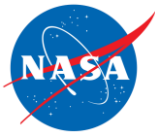





Figure 13: Levels of Integration for Digital Twins.

### 2.4.2 Digital Twins in Different Industries

Digital Twin applications have already been successfully implemented in different industries. Table 3 discusses several areas of application with the different life cycle phases. Examples are given from NASA (Glaessgen & Stargel, 2012), Airservices Australia (Mussomeli et al., 2020), Bridgestone (Mussomeli et al., 2020), US Air Force (Glaessgen & Stargel, 2012) (Brandyberry et al., 2012), and the manufacturing industry (Qi & Tao, 2018) (Tao et al., 2017) (Zhuang et al., 2017).

Table 3: Digital Twin Applications in Different Industries

Organisation	Phase	Applications
	Design & Engineering	<ul style="list-style-type: none"> <li>Virtual verification and validation</li> <li>Ultra-high-fidelity simulations</li> </ul>
	Operations	<ul style="list-style-type: none"> <li>Real-time vehicle health management</li> <li>Sensor updated fleet history to predict remaining lifespan of components</li> <li>Recommend mission profile changes to reduce load, increasing both longevity and chance of mission success.</li> </ul>
	Maintenance	<ul style="list-style-type: none"> <li>Read current state of different components</li> <li>Reducing damage or degradation by activating self-healing mechanisms</li> </ul>
	Operations	<ul style="list-style-type: none"> <li>Optimization of air traffic flow by means of machine learning based on historical data</li> <li>Distribute workload among air traffic controllers</li> <li>Increase in on-time arrivals and reduction in fuel through optimized routing</li> </ul>
	Design & Engineering	<ul style="list-style-type: none"> <li>Design based on recorded performance of older tire types</li> <li>Improving lifespan and performance</li> </ul>
	Production	<ul style="list-style-type: none"> <li>Determining price per kilometer by simulating various factors and circumstances</li> </ul>
	Operations	<ul style="list-style-type: none"> <li>Advising companies in selecting special tire types</li> <li>Advising companies on how to reduce tire wear and tear based on sensor data</li> </ul>
	Design & Engineering	<ul style="list-style-type: none"> <li>Virtual Certification</li> <li>Ultra-high-fidelity simulations</li> </ul>
	Operations	<ul style="list-style-type: none"> <li>Integrated vehicle health management (IVHM) based on sensor data</li> <li>Predict probability of mission success and remaining lifespan</li> <li>Evaluate effects on mission parameter changes</li> </ul>
	Design & Engineering	<ul style="list-style-type: none"> <li>Data on the entire life cycle of a product is later used in the innovation of the next generation product.</li> </ul>
	Production	<ul style="list-style-type: none"> <li>Identify problems during production and develop optimal solutions through simulation in the virtual world to adjust the production process</li> <li>Higher accuracy, stability, efficiency, and product quality</li> </ul>
	Operations	<ul style="list-style-type: none"> <li>Predict remaining life and predict possible defects</li> <li>Diagnose and analyse defects in the virtual model</li> </ul>

### 2.4.3 Digital Twins in the AEC Industry

In recent years, BIM has become the standard within larger construction projects and the concept of digital twin is becoming increasingly popular in literature. On a lower level, the AEC industry has been doing system integration for years. However, according to Deng et al. (2021), applications of real Digital Twins in the built environment are still in “nascent stages”. Because IoT devices are getting more accessible, the industry is now able to combine BIM models with dynamic (near) real-time information. Internet of Things (IoT) is defined as “the interconnection of sensing devices that are able to provide



*information exchange across different platforms”* (Ashton, 2009). The integration of IoT within BIM has therefore led to the emergence of the Digital Twin in the AEC industry (Lu et al. 2020, Tagliabue et al. 2021). Due to the adoption of BIM, the AEC industry is becoming more productive, collaboration is increased, project times are reduced, and many failure costs are prevented. The digital twin paradigm opens the door for many more possibilities.

The following section elaborates on publications related to digital twins and BIM-IoT integrations with different application areas. Using the search engines Google Scholar and ScienceDirect, a search was done based on all possible combinations between the keywords: BIM, IoT, Digital Twin, Digital Twins, AEC Industry, Built Environment, Construction, Predictive Maintenance, Asset Management, Facility management. After this step, the abstracts of all papers were carefully read, while quickly looking through the rest of the article, focusing on the following inclusion/exclusion criteria:

1. *Is it a scientific research article (i.e., not a commercial article, book, or thesis)?*
2. *Does the article discuss Digital Twins or a link between Internet of Things (IoT) and BIM?*
3. *Is the Digital Twin or BIM/IoT integration the main focus of the article (i.e., not only discussed in a small part of the research article?)*
4. *Does the article describe the application of a digital twin in a case study?*

On the resulting articles, backward and forward searching was done to find more relevant articles. After reading all thirty-three articles, it became clear that seven different application areas are discussed in the papers. The seven application areas include 1) Validating design decisions, 2) Construction process monitoring, 3) Space management / Occupancy analyses, 4) Hazards monitoring, 5) Equipment monitoring / Predictive maintenance, 6) Energy performance management, and finally 7) Indoor environment monitoring / thermal comfort. All seven are explained below:

**1 – Validating design decisions** – In the ideation stage, a digital twin allows evaluating different design choices. Design concepts that don’t meet the clients’ established criteria or other compliance checks can be abandoned. Therefore, digital twins allow testing design intent for compliance and functionality, which will eventually help in making better decisions.

**2 – Construction process monitoring** – With the advancement of BIM and IoT technology, digital twins are being used more frequently on construction sites since they are able to automatically compare the real process of construction with the designed planning and because they can help manage the site effectively. This real time overview helps improving the efficiency and daily operations, collaboration, decision making, and supervision.

**3 – Space management / Occupancy analyses** – A different aspect of digital twins is the localization of construction resources and people in and around buildings. With the use of sensor data, it becomes possible to gain insight into the use of certain spaces or walking routes. Based on this knowledge it becomes easier for facility managers to make decisions in the operations phase.

**4 – Hazards monitoring** – Additionally, real-time monitoring could also increase on and around building sites. Buildings and infrastructure are unfortunately subject to a number of hazards, such as fire, air pollution, gas leaks and accidents. Digital twin technology helps identifying these hazards and could advice building operators how and where to act in case of emergencies.

**5 – Equipment monitoring / Predictive maintenance** – Digital twins use all the available information that is connected to a specific system or asset. While assets are continuously being observed through sensor data, simulation models are able to predict the asset’s remaining life, by comparing the behavior to a database with historical data. In that way, maintenance is shifted from a reactive to a

more proactive and predictive approach. Therefore, maintenance is only executed, when necessary, what in most cases results in fewer expenditures on asset maintenance.

**6 – Energy performance management** – The key to creating effective and sustainable buildings is real-time energy evaluation and management of buildings. In digital twin literature, this is one of the most popular research areas. Sensor data can be input for building automation systems and can also support facility managers in creating better decision-making strategies. Digital twins are also able to spot inefficient behavior and act against it. In literature, this proves to be useful to assess how people use energy in the office. This encourages people to use less energy.

**7 – Indoor environment monitoring / thermal comfort** – The digital twin paradigm is also used for smart buildings in the context indoor environment and thermal comfort. The combination of real time sensor data and historical data enables digital twins to predict and visualize thermal comfort in buildings. Similar to the application area of energy performance management, the findings in the digital twin can be connected to the building automation system to correct specific rooms or area's before problems occur.

Table 4 displays the different application areas for each article. Some articles, discuss multiple fields of application, so they are indicated with multiple crosses.

Table 4: Overview of Articles related to Digital Twins in the AEC Industry. Categorized by application area

Authors	Title	1	2	3	4	5	6	7
Alonso et al. (2019)	SPHERE: BIM Digital Twin Platform.	X						
Alizadehsalehi & Yitmen (2021)	Digital twin-based progress monitoring management model through reality capture to extended reality technologies (DRX)		X					
Antonino et al. (2019)	Office building occupancy monitoring through image recognition sensors. International Journal of Safety and Security Engineering			X				
Austin et al. (2020)	Architecting Smart City Digital Twins: Combined Semantic Model and Machine Learning Approach.						X	
Bhargav et al. (2018)	A framework for integrating BIM and IoT through open standards.						X	X
Chen et al. (2018)	BIM-based framework for automatic scheduling of facility maintenance work orders.				X			
Cheng et al. (2020)	Data-driven predictive maintenance planning framework for MEP components based on BIM and IoT using machine learning algorithms.					X		
Cheung and Lin (2016)	Development of BIM-based Safety Monitoring System Integrated with WSN Technology				X			
Ciribini et al. (2017)	Tracking Users' Behaviors through Real-time Information in BIMs: Workflow for Interconnection in the Brescia Smart Campus Demonstrator.			X				X
Escandón et al. (2019)	Thermal comfort prediction in a building category: Artificial neural network generation from calibrated models for a social housing stock in southern Europe.							X
Guerra de Oliveira et al. (2020)	Airport pavement management systems: An open BIM approach.					X		
Gao et al. (2020)	A framework of developing machine learning models for facility life-cycle cost analysis.					X		
Hosamo et al. (2022)	A Digital Twin predictive maintenance framework of air handling units based on automatic fault detection and diagnostics.					X		
Huang et al. (2018)	Construction of intelligent fire management system based on BIM technology.				X			
Jing et al. (2019)	Development of BIM-Sensor Integrated Platform for MEP Piping Maintenance.			X		X		
Authors	Title	1	2	3	4	5	6	7

Kassem et al. (2018)	Measuring and improving the productivity of construction's site equipment Fleet: An integrated IoT and BIM system	X				
La Russa & Santagati (2020)	Historical Sentient – Building Information Model: a Digital Twin for the Management of Museum Collections in Historical Architectures.					X
Li et al. (2018)	An Internet of Things-enabled BIM platform for on-site assembly services in prefabricated construction.	X				
Liu et al. (2014)	Estimating and Visualizing Thermal Comfort Level via a Predicted Mean Vote in a BIM System.					X
Lu et al. (2020)	Developing a Digital Twin at Building and City Levels: Case Study of West Cambridge Campus.		X	X	X	X
Lu et al. (2020)	Digital twin-enabled anomaly detection for built asset monitoring in operation and maintenance.			X		
Ma et al. (2020)	Data-driven decision-making for equipment maintenance.			X		
Madni et al. (2019)	Leveraging Digital Twin Technology in Model-Based Systems Engineering.	X				
Mannino et al. (2019)	Office building occupancy monitoring through image recognition sensors.		X			
Moretti et al. (2020)	Maintenance service optimization in smart buildings through ultrasonic sensors network.		X			
Natephraa & Motamedib (2019)	Live data visualization of IoT sensors using Augmented Reality (AR) and BIM.	X				
Pour Rahimian et al. (2020)	On-demand monitoring of construction projects through a game-like hybrid application of BIM and machine learning.	X				
Quinn et al. (2020)	Building automation system – BIM integration using a linked data structure.		X			X
Qureshi et al. (2020)	Implications of Machine Learning Integrated Technologies for Construction Progress Detection Under Industry 4.0 (IR 4.0).	X				
Shim et al. (2019)	Development of a bridge maintenance system for prestressed concrete bridges using 3D digital twin model.			X		
Wang et al. (2020)	New Paradigm of Data-Driven Smart Customisation through Digital Twin.					X
Wehbe and Shahrour (2019)	Indoor hazards management using digital technology.		X			
Xie et al. (2020)	Visualised inspection system for monitoring environmental anomalies during daily operation and maintenance.		X	X		

#### 2.4.4 Operation and Maintenance phase

As described in Section 2.1.1, the operation and maintenance phase covers a variety of scheduled and unforeseen events in the lifecycle of an asset. Examples are maintenance, breakdowns, repairs, extensions, refurbishment or eventually demolition. The largest part (usually 80%) of a building's information is gathered and used in the operation and maintenance phase. Therefore, more emphasis should be on information on assets after projects are delivered to the client. This situation is supported in various research articles. For instance, Dixit et al. (2019) claim that the maintenance and operation of assets are the major application areas for digital twins. However, according to literature, there are and obviously will always be several challenges. In the operation and maintenance phase, the project is normally out of the control of the contractor. As a result, accessing and managing the object's data becomes challenging. Although a virtual model might be an exact representation of the object, it is not related to the actual constructed project (Anderl et al., 2018). In the operation and maintenance phase, projects experience operations by several stakeholders (Werbrout, 2020). These operations prevent the data exchange between the different lifecycle phases and the different stakeholders. When the physical building is connected to the digital world, digital twins can give facility or asset managers the chance to make daily operational decisions regarding building performance management, energy consumption optimization, and operation and maintenance. Using real time data, allowing predictive maintenance assures informed decision-making, which is why digital twins

increase the project's operational efficiency (Khajavi et al., 2019). Brilakis et al. (2019) confirms this by stating that a digital twin is intended to offer improved data-driven decision-making during asset operation and management.

#### 2.4.5 DT Applications in Asset Management

There are multiple papers that describe interesting use cases, where digital twins are used to improve asset management. In Table 4, those papers are categorized in different fields of application. The one that corresponds to the topic of asset management asset management is "Equipment monitoring / Predictive maintenance". In the following section the methods and results of the related articles are reviewed briefly. The papers are displayed in Table 5.

Table 5: Articles related to Digital Twins in Asset Management

Authors	Year	Title	Published in:	Where?
Cheng et al.	2020	Data-driven predictive maintenance planning framework for MEP components based on BIM and IoT using machine learning algorithms	Automation in Construction	Japan, Singapore
Guerra de Oliveira et al.	2020	Airport pavement management systems: An open BIM approach	Proceedings of the 5 <sup>th</sup> International Symposium on Asphalt Pavements & Environment	Italy, Slovenia
Gao & Pishdad-Bozorgi	2020	A framework of developing machine learning models for facility life-cycle cost analysis	Building Research & Information	USA
Hosamo et al.	2022	A Digital Twin predictive maintenance framework of air handling units based on automatic fault detection and diagnostics	Energy & Buildings	Norway, Denmark
Jing et al.	2019	Development of BIM-Sensor Integrated Platform for MEP Piping Maintenance	International Conference on Smart Infrastructure and Construction 2019	China
Lu et al.	2020	Developing a Digital Twin at Building and City Levels: Case Study of West Cambridge Campus	Journal of Management in Engineering	UK
Lu et al.	2020	Digital twin-enabled anomaly detection for built asset monitoring in operation and maintenance	Automation in Construction	UK
Ma et al.	2020	Data-driven decision-making for equipment maintenance	Automation in Construction	China, Slovenia
Shim et al.	2019	Development of a bridge maintenance system for prestressed concrete bridges using 3D digital twin model	Structure and Infrastructure Engineering	South-Korea
Xie et al.	2020	Visualised inspection system for monitoring environmental anomalies during daily operation and maintenance	Engineering, Construction and Architectural Management	UK

All papers share the goal of improving the planning and execution of maintenance activities through the integration of data from building and equipment with maintenance management systems. Many of the papers focus on real time monitoring of assets and equipment, which is enabled by integrating sensor data with 3d geometry from either BIM or GIS (Jing et al., 2019) (Guerra de Oliveira et al., 2020) (Xie et al, 2020). Results show that the time to access information is reduces significantly, which saved costs and optimized the management of assets.

Other papers integrate maintenance management systems with 3D models to improve decision making in asset management (Ma et al.,2020) (Lu et al., 2020) (Shim et al., 2019). The results show that the systems decreased labor costs, made equipment maintenance decisions easier, and promoted objectivity, while also supporting the project maintenance team as well as decision-makers

to timely perform appropriate maintenance through the ability to continuously monitor the state of building assets.

Finally, some papers also propose frameworks that utilize machine learning techniques to predict the future conditions of assets, allowing for better planning of maintenance schedules (Cheng et al. 2020) (Gao & Pishdad-Bozorgi, 2020) (Hosamo et al., 2022) (Lu et al., 2020). The results show that continuously updated data in conjunction with machine learning algorithms can identify defects and also that the future condition of MEP components could efficiently be predicted, minimizing mistakes by human involvement.

The interoperability of different systems and the standardization of data exchange is emphasized in all papers, as it is essential for optimized management of assets. Overall, these articles demonstrate that data-driven maintenance management systems will potentially increase efficiency, reduce costs, and improve the overall health and performance of buildings and equipment.

## **2.5 Semantic Web and Linked Data Technologies**

### **2.5.1 Introduction**

Today, more and more construction contracts also include operations and maintenance of buildings. Although the industry is quite traditional, dealing with digital information about assets in digital systems becomes increasingly important. This transition is leading to more development in digital twin technology. In recent years, the semantic web and linked data technologies are being implemented within the digital twin concept. “The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation” (Berners-Lee et al., 2001). The semantic web is a network of data, displayed in graphs, which consist of nodes and links, identified by Uniform Resource Identifiers (URI). These URI’s are all located by a Uniform Resource Locator (URL). The semantic web is based on the Resource Description Framework (RDF<sup>1</sup>)

Rijkswaterstaat (RWS), which is part of the Dutch Ministry of Infrastructure and Water Management has already started doing its asset management using a linked data approach with an own Object Type Library (OTL) and requires infrastructure contractors to hand over their projects in RDF format. Rijksvastgoedbedrijf (RVB) and municipalities also started trials with RDF data storage of assets. It is expected that data delivery and data storage using RDF will increase in the near future. For the AEC industry to comply to these demands in the future, two things are needed. More research on the application of semantic web technologies in the AEC industry and some ambition by companies to actually make changes in the industry.

In recent years, literature in the built environment has embraced semantic technologies and linked data. “The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation” (Berners-Lee et al., 2001). According to the W3C Semantic Web Activity Homepage (2013), “the Semantic Web provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries. The semantic web is a network of data, displayed in graphs, which consist of nodes and links, identified by Uniform Resource Identifiers (URI). These URI’s are all located by a Uniform Resource Locator (URL).” This means that URI’ only define entities, without displaying the location on the web and that A URL is the address of a given unique resource on the Web.

---

<sup>1</sup> [https://www.w3schools.com/xml/xml\\_rdf.asp](https://www.w3schools.com/xml/xml_rdf.asp)

The semantic web is built in such a way that the data cannot only be interpreted by humans, but also by computers. Before, on the traditional web, computers were only able to handle the data, but the meaning of things could not be interpreted. The semantic web aims to include the semantics (meaning) of data. To reach data objects located on the semantic web, the HTTP protocol is used. To make the semantic web practical, there are many technologies that work together as the semantic web stack (see Figure 14). In the following section, several parts of the semantic web stack are explained in detail.

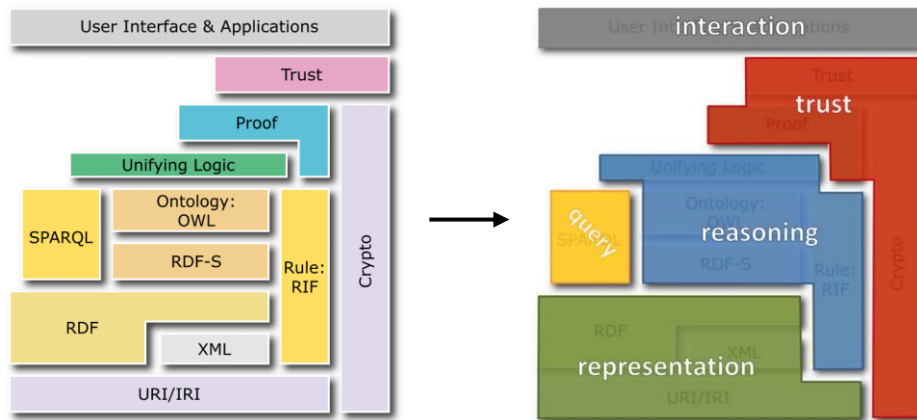


Figure 14: W3C Semantic web stack

### 2.5.2 Resource Description Framework

The semantic web is based on the Resource Description Framework (RDF). RDF is a language to define data so it can be read and understood by computer applications. Within the RDF language, all information is presented in triples, which consist out of a subject (URI), predicate (URI), and object (URI or literal). RDF documents are originally written in XML. The XML language used by RDF is called RDF/XML. By using XML, RDF information can easily be exchanged between several types of computers using different types of operating systems and application languages. Figure 15 displays an example of a small RDF graph, containing multiple triples. In the highlighted red triple, the subject is the building, the object is the room, and the relation between the two is the predicate: 'hasRoom'. All three parts have their own meaning. This provides semantic meaning to every information component on the semantic web, which is useful for automated inference. The green highlighted triple contains two URI's and one literal. The object in this triple is just a string. This means that it has not specific URI. RDF-triples can be notated using several syntaxes, such as N-triples, RDF/XML, JSON-LD, and Turtle (Beckett & Berners-Lee, 2011) (W3C, 2014).

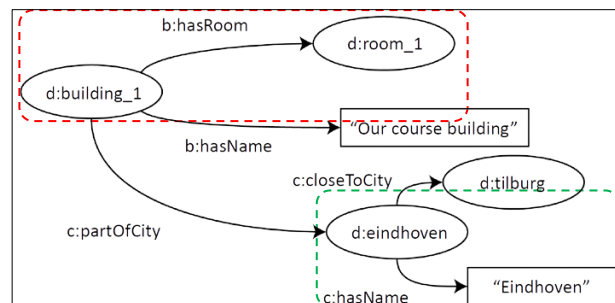


Figure 15: Example of an RDF Graph

### 2.5.3 Ontologies

To provide the web with semantic structure, ontologies are used. An ontology (or vocabulary) is a set of concepts and categories in a specific domain including properties and the relationships between them. The most fundamental elements are within the RDF schema (RDFS), and the more expressive elements are defined within Web Ontology Language (OWL) (Pauwels et al., 2017). OWL improves upon RDFS principles further to enable the creation of more complicated RDF statements, such as cardinality restrictions, type restrictions, and complex class expressions. This means RDF graphs constructed with OWL concepts are called OWL ontologies. An initiative called Linked Building Data



(LBD<sup>1</sup>) aims to make building data available for web-based BIM applications. Several OWL ontologies have been introduced to the Architecture, Engineering, and Construction industry, such as IfcOwl, BOT, and Brick.

IFC models contain 3D geometry and metadata regarding the object's relationships to other aspects of the object and the entire building, which makes them semantically rich. Despite the representation of numerous AEC industry domains, one major limitation of the IFC standard is the lack of mechanisms to extend the model's semantics (Costa & Madrazo, 2015). To address this issue, the IfcOwl web ontology language was developed by Beetz et al. (2019), which transforms the IFC schema into an RDF-based schema for Linked Data. This conversion makes building information available as an RDF graph while still adhering to the widely used IFC standard (Malcolm et al., 2021). Since IFC is already widely adopted in the AEC industry, IfcOwl can also function as a familiar linked-data ontology. However, IFC and therefore also IfcOwl has one downside that it is exhaustive in the types of data it can represent (Pauwels et al., 2022).

One of the W3C principles is to keep schemas light for easy reuse. Unlike the complex IfcOwl ontology, Rasmussen et al. (2017) proposes a simple Building Topology Ontology (BOT) for easy reuse across domain ontologies. BOT allows describing a building via zones and elements. A building can be defined as a bot:Building. A building is located on a site (bot:Site) and has levels (bot:Storey), which have spaces (bot:Space). These zones then have relationships with elements. For example, a space can be next to a bounding element (bot:adjacentElement), a space might contain an element (bot:containsElement), and a space might be intersected by an element (bot:intersectingElement) (Rasmussen et al., 2017). The use of a BOT can address the problem of redundancy, which is currently in violation of W3C's best practices. Additionally, this approach offers a flexible foundation that can be easily expanded to connect with both present and future domain ontologies.

Brick is also a schema for representing metadata in buildings. The schema defines sensors, subsystems, and relationships among them, which enables portable applications (Srivastava et al., 2016). To describe sensors and subsystems in a building, Brick uses clear tags and tagsets. For this list of tags and tagsets, it defines an ontology and a class hierarchy. Brick proposes functional blocks to abstract out complexity while also assisting with system composition and hierarchies. Sensors and subsystems that perform similar functions are compared using synonyms.

Besides IfcOwl, BOT, and Brick, there are many other ontologies or vocabularies that structure generic and domain specific data. Examples are PRODUCT, PROPS, Haystack, RealEstateCore, SAREF, DogOnt, SOSA, CityGML. In addition to these commonly used ontologies, certain organizations have developed their own exclusive vocabularies to organize their data. Notable instances of such ontologies include the OTL established by the municipality of Amsterdam (Gemeente Amsterdam, 2022), the AIRBIM-OTL created by the Dutch road and waterworks authority (Rijkswaterstaat, 2021), and the OTL spoor designed by the Dutch railway authority (ProRail, 2020). These unique ontologies can be utilized in conjunction with general ontologies to effectively structure data within the semantic web.

#### 2.5.4 SHACL

SHACL or Shape Constraint Language is a language that is used to set constraints to data structures. SHACL used the concept of shapes, which describe the constraints applying to different types of data. An example can be a shape that is used to define specific properties for a data type that should be present minimally (e.g. BIM properties such as loadbearing, material, and NL/SfB). The same can be done for defining constraints on the values of properties themselves (e.g., minimum, and maximum values or the allowed range of value). Typically, SHACL is used to validate data in a database, ensuring that it follows a defined schema and meets all specified constraints. SHACL helps improving data

---

<sup>1</sup> <https://www.w3.org/community/lbd/>

quality and consistency and is often used to identify and fix problems or inconsistencies in the data. Examples of applications where SHACL is used are data integration, data governance, and the semantic web, where it helps ensuring consistency and interoperability of data sources from multiple systems (W3C, 2017).

### 2.5.5 SPARQL

Previously the representation of data using RDF and ontologies is explained. Using triples, connections between two things are represented in the simplest way. The standard way of accessing RDF data is using the query language called SPARQL. The name stands for SPARQL Protocol And RDF Query Language (W3C, 2013). SPARQL allows retrieving and manipulating specific data stored in RDF format. As displayed in the semantic web stack SPARQL sits on top of RDF. Queries are built using a simple syntax, which unlike a few differences, looks very much like SQL. A notable feature of SPARQL is that it can perform complex queries on several data sources at the same time. This is very useful when dealing with data integration applications, where multiple sources of data are combined to answer specific questions. The key element of SPARQL queries is the graph pattern. In order to create query patterns, the TURTLE syntax is used, which is also used to represent RDF data initially. To specify what data is retrieved from a graph database, a query pattern is created with both known and unknown variables. (Allemang & Hendler, 2011). Within the SPARQL query language, there are four different ways of constructing queries, including SELECT, CONSTRUCT, ASK, and DESCRIBE. SELECT returns tabular results, CONSTRUCT creates a new RDF graph based on query results, ASK returns yes or no, if the query has a solution, and finally DESCRIBE returns RDF graph data about a resource (W3C, 2013). An example query is shown below. In this example of a combined HR database, the query selects employees that were hired before a specific date. The line “SELECT ?givenName ?familyName ?hd” specifies what variables should be displayed in the query result. Subsequently, the graph pattern specifies the relationships between the different variables and that the query only retrieves hiring date values before March 1<sup>st</sup>, 2022.

```
PREFIX vcard: <http://www.w3.org/2006/vcard/ns#>
PREFIX sn: <http://www.snee.com/hr/>
SELECT ?givenName ?familyName ?hd
WHERE
{
    ?person vcard:given-name ?givenName .
    ?person vcard:family-name ?familyName .
    ?person sn:hireDate ?hd .
    FILTER(?hd < "2022-03-01")
}
```

Listing 1: SPARQL query

Table 6: Query results

?givenName	?familyName	?hd
Francis	Smith	2022-02-13
John	Williams	2020-07-19
Heidi	Brown	2021-10-04

### 2.5.6 Graph Databases over Relational Databases

In the context of asset maintenance management, the key difference between graph databases and relational databases is how they store and retrieve data. In relational databases, data is stored in tables including rows and columns, which can be queried by using SQL commands. The different tables are connected with foreign keys. Tracking intricate relationships among different assets, can therefore be challenging. On the other hand, a graph database stores data as nodes and edges, which symbolize



entities and relationships. It is more intuitive to use when analyzing an entire context close to a single data point. Queries are executed by traversing the graph, making it easy to track complex relationships among different assets.

A notable difference between the two types of databases is that relational databases have a fixed schema. Therefore, the structure of data must be defined in advance, which can make it difficult to adapt to rapidly changing data or adding new assets to the system. In contrast, graph databases are easier to adapt and are able to handle highly dynamic data, which is very common in large asset maintenance contracts. Additionally, graph databases are highly scalable, because they can handle billions of nodes and relationships. Typically, this is very beneficial for asset maintenance management as the amount of data can be large and complex (Brunenberg, 2022).

Finding and analyzing data that has many connections, using traditional SQL databases can be difficult. Using SQL to follow connections between data often requires writing complex and hard-to-understand queries. Usually, these types of difficulties are a sign that a technology is not used in the best way for specific tasks. In traditional SQL databases, it is for instance difficult or impossible to perform queries like 1) Traversing a network of relationships: In graph database, it is easier to find all relationships starting from a particular node and all the nodes connected to those nodes. This principle is often used when finding the shortest path between two specified nodes, or to identify densely connected clusters of nodes. 2) Querying based on path patterns: in graph databases it is possible to query for nodes that relate to other nodes in a specific way. For example, selecting all the nodes that are two or three hops away from a specified node, or for all the nodes that have a specific pattern of incoming and outgoing relationships. 3) Handling highly dynamic data: within graph databases it is possible to dynamically add or remove relationships and nodes. As mentioned earlier, SQL databases require a fixed schema, which can create difficulties when dealing with rapidly changing data.

Translating this to the context of asset maintenance management, assets are often interconnected in various ways. Buildings may have several systems that need to be maintained and within those systems there are specific installations that can consist of multiple parts. A graph database allows tracking these relationships and understand the dependencies between assets. When providing a visual representation of assets and their interconnections, the asset management team will have a better understanding of the system. This also makes it possible to identify the most critical assets in a maintenance management context. By analyzing the relationships between assets and historic maintenance data, a graph database can understand which assets are most likely to cause problems if they fail and can provide a better general understanding of how to deal with these issues when they occur. Graph databases can also be used to optimize the scheduling of maintenance activities. Understanding dependencies between assets with the help of SPARQL queries can help scheduling maintenance activities, so that downtime is minimized and that the most critical assets are maintained first. The final step is predictive maintenance, where maintenance is performed before assets fail, by identifying patterns and predict when assets are likely to fail.

## **2.6 Conclusion Literature Review**

According to the BIM maturity model, the industry is currently located at level 0, 1, or 2. BIM level 3 adoption is still very rare, due to a number of obstacles like are cost, culture, expertise, technology and interoperability, lack of client demand, and legal issues. According to the ISO 19650 standard, the biggest part (usually 80%) of a building's information is gathered and used in the operation and maintenance phase. Therefore, more emphasis should be on information on assets after projects are delivered to the client. In recent years, many standards have been created in order to standardize building information and information delivery. Examples are IFC, IDS, NL-SfB, ICDD, and bSDD. In Asset management there are different types of maintenance, starting from corrective maintenance, followed by planned maintenance, growing into predictive maintenance, and finally prescriptive

maintenance where prediction is combined with recommendations for specific actions and maintenance strategies. In recent years, these principles have been discussed widely in digital twin literature, where the potential for data-driven maintenance management systems to increase efficiency, reduce costs, and improve the overall health and performance of buildings and equipment are discussed. Semantic web and linked data technologies are being implemented within the digital twin concept. The key difference between graph databases and relational databases is how they store and retrieve data. A graph database stores data as nodes and edges, which symbolize entities and relationships. It is more intuitive to use when analysing an entire context close to a single data point. Queries are executed by traversing the graph, making it easy to track complex relationships among assets. Also, graph databases are more adaptable and can handle highly dynamic data, which is quite common in large asset maintenance contracts. Nowadays, more and more ontologies are created, focussed on different parts of the AEC industry, so it is expected that the adoption of linked data will grow over time.

### 3 Methodology

This chapter includes the methodology for this research and will give answers to the sub-questions 2, 3, 4, and 5. The Methodology begins with explaining the research approach in Section 3.1. Subsequently, the findings of the interviews conducted with different stakeholders in the operation and maintenance sector are presented in Section 3.2.

#### 3.1 Research Approach

As mentioned in the introduction, this study aims to create a digital building logbook or digital twin for a building and investigate the added value of a linked data approach for asset life cycle information management. The research will leverage historical data from an SAP maintenance management ERP database and building information from BIM models to create a graph database that integrates all asset information. By using a graph database, the study will explore the advantages of handling dynamic data and analyzing an asset's entire context. The study will evaluate the effectiveness of the digital twin in improving asset management and decision-making, including the potential for reducing maintenance costs, increasing asset performance, and improving safety.

In order to achieve this, the research involves a number of methods, starting with market research to identify the struggles and problems encountered in the current information management process. Subsequently, the research continues with the development a system architecture and data pipeline, which mainly involves coding in Python. In this part of the methodology a framework is created for the transformation and integration of asset management data. Finally, the system and data pipeline are validated with a case study of a large maintenance project.

#### 3.2 Market Research

##### 3.2.1 Interview Results

The purpose of this section is to present the findings of the interviews conducted with different stakeholders in the operation and maintenance sector. The stakeholders involved three maintenance engineers, two asset managers, and two planners who are involved daily in the management of asset life cycle information. The objective of the interviews was to identify the struggles and problems encountered in the current information management process. The interviews revealed several key takeaways.

Firstly, employees miss a total overview of information. This means that they lack a comprehensive understanding of all the characteristics of the assets from all the systems. Normally an ERP system like SAP is used as the main system for managing maintenance data, but from the conversations it became clear that communication between the other systems is missing. Not only does this require a lot of administration, but it also costs employees too much time to find the information they need. First they need to know in which system they could find the information, then they have to log in, after which they have to do many filters to find the specific information they need. Sometimes after minutes of searching they find out that the information is stored in another system. All this can lead to errors and inconsistencies in the data, making it difficult to make informed decisions.

Another example is the mutation process after renovations or introduction of new assets, which is a complicated process that involves many manual actions, such as emailing, sending files, updating the BIM model, and asking for information, etc. It must be ensured that all information is processed in all systems correctly. Because it is mostly a manual process, human errors are made and sometimes the data is not complete. It also takes away time from employees and results in errors and inconsistencies in the data, causing problems in the later stages of an asset's life cycle.

The next issue is the often-low availability of information. Many service projects only have access to a list of assets and non-editable 2D drawings, which is insufficient for effective asset information management. Also, when new maintenance projects are contracted, the asset

management team sometimes only has a list of assets that is 70% correct. Assets are missing, while they are in fact in the building, data about the assets, such as properties, maintenance instructions, and warranties are missing. This low availability of information causes employees to send mails to or call other stakeholders to retrieve information. Because planners spend too much time on searching for required information, it sometimes leads to delays in maintenance and repairs.

Furthermore, not knowing the asset's maintenance history is mentioned as one of the biggest struggles by multiple stakeholders. This problem is closely related to the missing total overview of the data. In the current information system, it takes too much effort to find all relevant information related to a single asset or room. In the systems, data is stored in tables and across the different systems, not all tables are interlinked by unique id's. This makes it very difficult to find information close to one data point, for instance an asset. In the case of a newly reported malfunction on a specific asset, planners and technicians want to know everything that has happened to that asset immediately, since it could give insights on a possible cause or remind them of similar problems and how they were resolved in the past. On the other hand, for maintenance engineers, this information is crucial to make informed decisions about the appropriate preventive maintenance schedule.

Another consideration is wayfinding for technicians. In some situations, planners and technicians do not know where an asset is located. From the conversations it became clear that technicians who work night shifts even cannot find specific assets until the next day, when planners start working in the morning. Sometimes this is caused by missing information, but sometimes the data in the maintenance management system states that an asset is located in a specific room, but then the asset is above the ceiling, or the room is so big, that the asset cannot be found. The main reason for this is that the maintenance management system is not linked with the BIM model. Besides all asset information, the BIM model can also perfectly visualize where an asset is located. Also, assets that are located above the ceiling can be visualized so that technicians know what ceiling plate to lift to reach the asset. Localising assets as quickly as possible is important to minimize downtime and maximize efficiency.

Lastly, data quality will always be a problem. For instance, SharePoint is a commonly used system for document management. However, documents are not always put in the right folders, and metadata is not properly maintained. This can later lead to difficulties in finding and accessing the necessary information. From the conversations it also became clear that outdated information is very common in maintenance projects. Currently, there are many maintenance projects that were contracted in times when data was less well collected and maintained. Outdated information leads to incorrect decisions being made about maintenance, repairs, or replacements, which can have serious consequences for the assets' performance and longevity.

### *3.2.2 Use Case – Requirements / Competency questions*

After considering the issues above, the goal is to create a pipeline for creating a digital building logbook for buildings to improve asset management and decision-making, including the potential for reducing maintenance costs, increasing asset performance, and improving safety. The system should help tackling the problems related to the total overview of information, insights into the maintenance history of specific assets and rooms, and the wayfinding of assets and rooms. Firstly, instead of hosting data in different applications, the system should allow performing large-scale analysis on combined data to gain new insights which can help optimizing the preventive maintenance schedule. By optimizing the maintenance schedule, the amount of corrective maintenance orders is reduced. Of course, the amount of malfunctions will never be reduced to zero, so for the malfunctions that do occur, the system should instantly provide all potentially valuable information from historical data about an asset, room, or a specific problem. This will reduce the time planners and technicians need to search for the required information, which will also directly reduce the indirect productivity of

technicians. This way, employees are supported in making better and quicker decisions on how to resolve malfunctions based on historical data, instead of only experience or even gut feeling. The main goal of the system is to take steps towards predictive and prescriptive maintenance. The next chapter will propose a system architecture and will discuss the development of a data pipeline that can transform and integrate the existing data sources. To find out what the system should be able to perform as a minimum, multiple meetings with maintenance engineers, planners, and data consultants were held, to create a list of requirements or competency questions, which are listed below in Table 7.

Table 7: Requirements/ Competency questions

	<b>Req. / Competency Question</b>	<b>Argumentation</b>
1	Select all historic orders related to a specified asset	All employees should be able to see all historic orders related to a specified asset. The history contains all orders, descriptions, causes, damage pictures, and relating measures to resolve the issues. This reduces the indirect productivity for finding asset information. Normally, a user has to do a lot of filtering in different excels.
2	Select all historic orders that took place in a specified room	All employees should be able to see all historic orders that took place in a specified room. The history contains all orders, descriptions, causes, damage pictures, related asset, and relating measures to resolve the issues. This reduces the indirect productivity for finding asset information. Normally, a user has to do a lot of filtering in different excels.
3	What kind of assets are prone to a specified damage picture?	Maintenance engineers and asset managers should be able to retrieve a list of assets related to a specified damage picture, sorted by the amount of occurrences of the asset. This gives users insight in how to prevent certain issues.
4	What kind of rooms are prone to a specified damage picture?	Maintenance engineers and asset managers should be able to retrieve a list of room types related to a specified damage picture, sorted on the amount of occurrences of the room type. This gives users insight in how to prevent certain issues.
5	Select all properties from the IFC for the asset or room related to order	For each asset, the user should be able to retrieve all properties from the BIM model. This reduces the indirect productivity in which employees must login to the common data environment, find the right IFC model, and wait for it to load.
6	Select all measures that resolved similar malfunctions	Based on a specified set of properties, such as cause, damage picture, or asset type, planners and technicians should be able to retrieve a list of measures that resolved similar orders before. This gives the user suggestions on how to solve the issue, directly when the malfunction is notified.
7	What kind of orders have the longest lead times?	Subtracting the notification datetime from the datetime of the measure that resolved the issue, all employees should retrieve a list of all orders sorted by the duration of the orders. This way the user gets insight into what kind of orders should get more attention.
8	What is the prognosed lead time for this Notification?	Based on a specified cause, damage picture and asset/ room type the planners and technicians should retrieve the average duration of similar historic orders. This gives the user an estimation on how long the issue will probably take to resolve.

- |    |  |  |
|----|--|--|
| 9  | What is the most common Cause for a specific problem?                            | All employees should retrieve the most probable cause for a specified damage picture and asset or room type. This gives users insight in how to prevent certain issues.  |
| 10 | Select all rooms where a specific malfunction occurred in a specified timeframe. | Based on a specified month and year, maintenance engineers and asset managers should receive all rooms and corresponding GUID's where a specific malfunction (Discipline and Cause) happened. This gives users the ability to trace locations for specific malfunctions.                                     |
| 11 | Visualize the context of data around a specific asset, room, or problem          | Maintenance Engineers and asset managers should retrieve an overview of all data closely related to a specified room, asset, or problem. For instance, all nodes, 1 or two hops away from the asset or room node. This reduces the indirect productivity, when looking for information in different systems. |

## 4 Development

This section starts with proposing a system architecture and data pipeline (Section 4.1). Section 4.2 until 4.4 will discuss the development of the data pipeline that can transform and integrate data from the existing systems and will therefore answer sub-question 3, 4, and 5. In Section 4.5, the presentation layer of the system architecture is discussed which will give answer to sub question 6. Finally, the chapter is ended with a small conclusion (section 4.6)

### 4.1 Proposing System Architecture and Data Pipeline

The system architecture is composed of several layers of data, starting with the database layer, which contains all data from the existing databases. The next layer is the business logic layer which contains the main programs, code definitions, functions and rules that determine the behavior of the system. The final layer is the presentation layer or user interface (UI) which interacts with users through web applications. The system architecture is visualised schematically in Figure 17. The part of the system architecture that is highlighted in orange represents the data pipeline, which is explained in detail in the following section. The part that is highlighted in blue represents the further development of a web application called LBDviz, which will be explained in Section 4.5.3. The part that is not highlighted is explained briefly, but further development falls outside of the scope of this thesis.

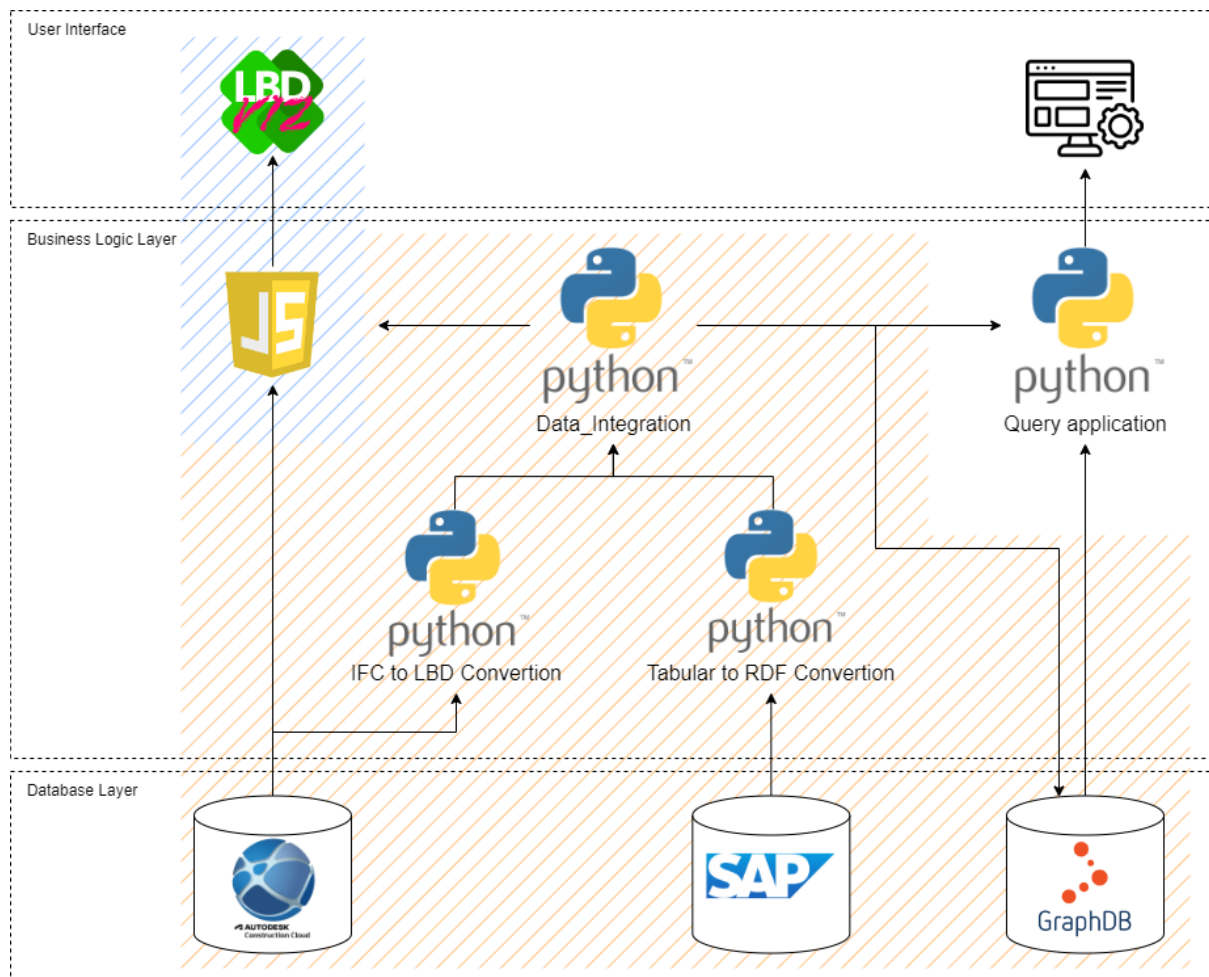


Figure 16: Schematic Visualization of System Architecture

The data pipeline is visualized schematically in Figure 17. The data pipeline starts with the two databases, namely the “SAP” maintenance management system and the common data environment “Autodesk Construction Cloud”. Subsequently, using Python, all data is transformed into a format that



is suitable for semantic web technologies and finally all data is integrated into one data file that contains all data and relationships between.

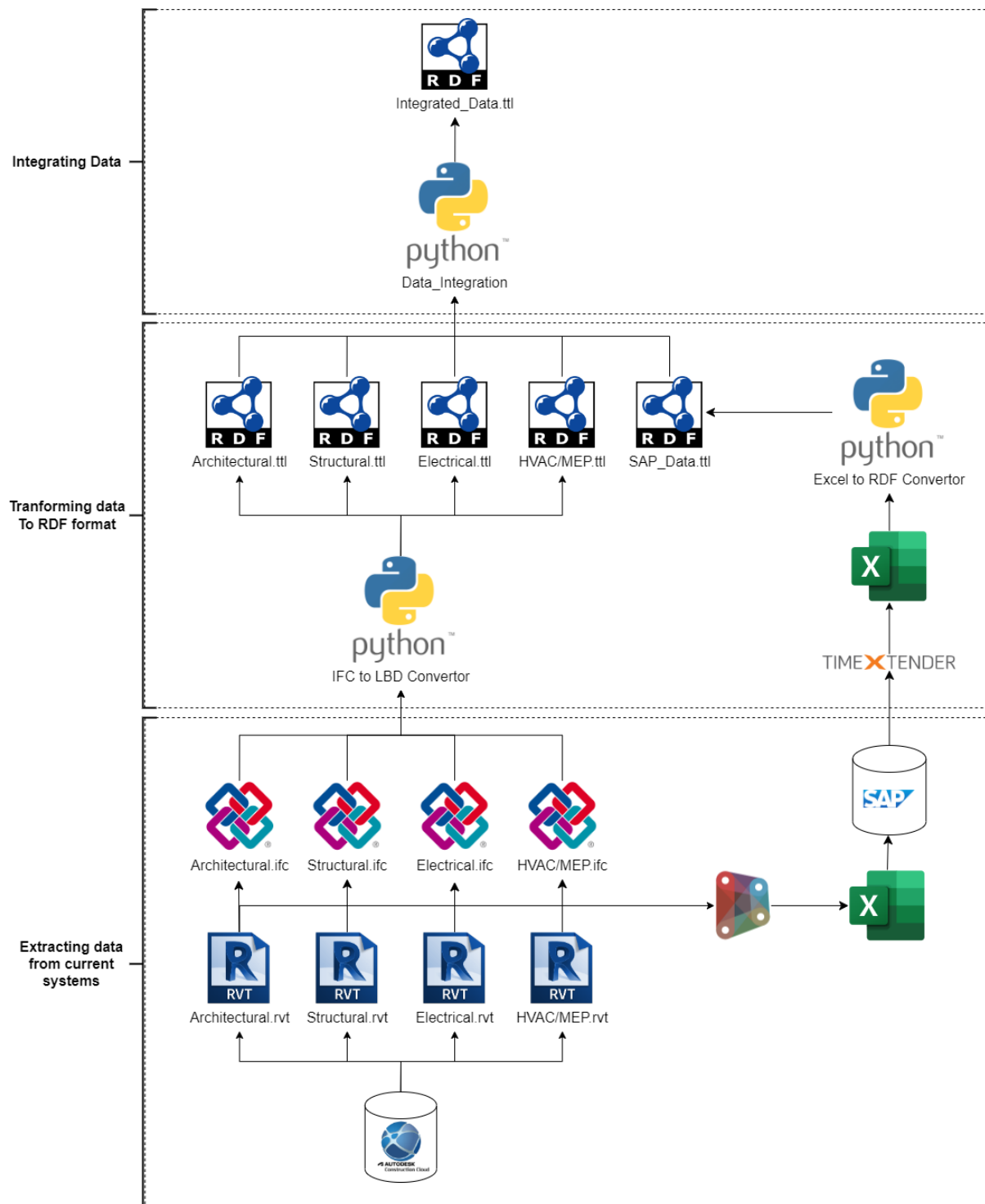


Figure 17: Schematic Visualization of Data Pipeline



## 4.2 Extracting data from current systems

In this section the data extraction of the data from the current systems is explained in detail. By further elaborating on this, sub-question 3: “What are the key characteristics and sources of data commonly used in asset life cycle information management?”, is answered. A project starts with the common data environment which contains multiple Revit BIM models for each discipline. The Revit models will be exported as IFC files and the elements from the Revit models are input for the MMS. This part of the data pipeline is visualized in Figure 18.

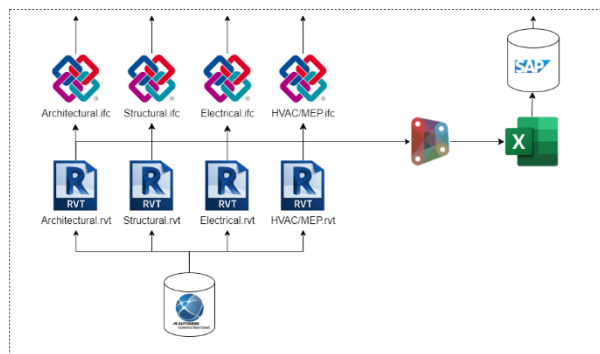


Figure 18: Schematic Visualization of Data from the Current Systems

### 4.2.1 BIM Models

Construction projects are typically created using a BIM authoring tool such as Autodesk Revit, Graphisoft ArchiCAD, or Tekla Structures. In this thesis, we assume that the projects are designed using a BIM protocol and BIM execution plan that mandates the use of separate aspect models for each discipline involved in the project, such as architectural, structural, electrical, and HVAC/MEP. All models use the same project origin, levels, and grids, and the architectural model includes information on the project's rooms. These models are combined to form the coordination model of the project. It is assumed that the project is minimally delivered in accordance with the BIM Basis ILS, which encapsulates the minimal standard data. Additionally, different companies will have a number of additional requirements that BIM models have to comply to. The Revit and IFC BIM models are stored in a common data environment such as Autodesk ACC or Trimble Connect. The literature review shows that the operations phase of the asset lifecycle, which includes maintenance, breakdowns, repairs, extensions, refurbishment, and eventually demolition, is less rigid than the project phase. In response to these non-consequential triggers, the asset information model needs to be updated periodically by BIM modellers (Nederlands Normalisatie-Instituut, 2019). The Revit models are stored as work-sharing files in a 'Work in Progress' (WIP) folder. Periodically the BIM models are exported as IFC and stored in a 'Published' Folder.

### 4.2.2 Maintenance Management System (SAP)

Typically, the management of project operations and maintenance is carried out using an enterprise resource planning (ERP) system. Using an ERP system, several business processes are centralized which enables automation so that companies can work more efficiently and reduce errors. The ERP system (in this graduation project SAP) contains data about orders, logistics, administration, planning, but also information about assets derived from the IFC files, such as "family name," "family type," "element ID," and "NL-SfB". In case of any errors or malfunctions, an official notification is sent to the service desk, which is then translated into an order in SAP, depending on the type of problem. The planners and technicians then work together to resolve the issue. For all orders, all valuable information about the cause, damage assessment, date, and related asset, as well as details on how the problem was resolved, such as measures and corresponding dates, are stored in SAP.

### 4.3 Transforming Data to RDF Format

This section will discuss the part of the data pipeline where all data from the existing systems is transformed to the RDF format and will give answer to sub-question 4: “How can the data originating from different systems be transformed into a format suitable for semantic web technologies?” Here the data from the existing systems is transformed to a format suitable for semantic web technologies. First, a maintenance ontology is created for structuring the MMS data. Subsequently the connection to the BOT ontology is explained. The next step is to create a Python script which allows the conversion of data from the originating systems to RDF format. This part of the data pipeline is visualized in Figure 19.

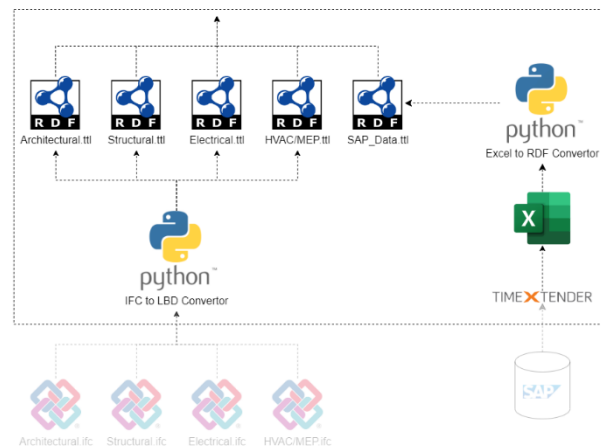


Figure 19: Schematic Visualization of Transforming Data to RDF Format

#### 4.3.1 Maintenance ontology

To transform the maintenance data from the SAP maintenance management system to data suitable for a graph database, it is necessary to first create an ontology that defines the structure and relationships of the data. By creating an ontology, it is ensured that the data is properly structured and organized for use in a graph database. Moreover, the ontology establishes a common understanding of the data across different maintenance projects, ensuring that the same data is represented in a consistent manner. This will improve data quality, facilitate data integration, and reduce the risk of errors in data interpretation. The corresponding prefix and namespace for this “maintenance Ontology” are defined as:

@prefix MO: <<https://heijmans.org/maintenanceontology#>>

At the core of the developed ontology is the concept of an "order," which is uniquely identified by its order number. An order can be categorized as a malfunction, wish, preventive action, complaint, or an information request, and these categories are distinguished using the "rdfs:subClassOf" relationship. Additionally, an order has several object properties. An order takes place in a specific room and is related to a particular asset that is located in that room, which is itself part of a larger building storey. An order also includes information on a damage picture, discipline, and cause, each of which can fall into one of 24, 9, and 16 predefined categories, respectively. These categories would normally be nodes in the ABox of the ontology. However, because they are predefined categories, which are representative for all maintenance projects, they can be part of the TBox, which is indicated by a different color in Figure 20. Finally, an order is resolved by one or more measures. Unlike a cause or damage picture, a measure is unique and has its own measure number and timestamp. Similar to an order, a measure can also fall into one of several predefined types, which are related using the "rdfs:subClassOf" relationship. All nodes are defined in Table 8.

Table 8: Definitions for Nodes in the Maintenance Ontology

Node	Definition
MO:Order	A specific work request or task that needs to be performed to maintain or repair an asset. After a problem gets notified, the notification is registered as an official order in the MMS.
MO:Malfunction	A failure or disruption in the operation of an asset or equipment, which results in a decrease or loss of performance or function.
MO:Wish	A request on a subject that falls out of the scope of the contract.
MO:Preventive	An order that relates to maintenance taken in advance to mitigate potential risks and prevent asset damage or loss.
MO:Complaint	An expression of dissatisfaction or criticism related to a product or service, made by building users.
MO:Information_Request	An inquiry or demand for specific details, data or knowledge on a particular topic or subject matter.
MO:Discipline	A specific area of expertise or knowledge that sub-divide the types of assets, such as Electrical, Architectural, or structural elements.
MO:DamagePicture	An assessment for sub-dividing the type and extent of damage or loss to an asset.
MO:Cause	the underlying reason or source of a failure or breakdown in an asset
MO:Measure	a specific action or set of actions that is taken to address issues or problems related to an asset, which may include tasks such as repair, replacement, or reconfiguration. One order may require one or more measures to be resolved completely.
MO:Asset	A physical component of a building, such as walls, doors, equipment, machinery, HVAC systems, plumbing, electrical systems, and infrastructure used in organizational operations.
MO:Room	An enclosed space within a building that is separated by walls, floors, ceilings, or roofs
MO:Storey	The space within a building, situated between two floor levels, or one floor level and the ceiling above

The ontology also includes thirteen data properties. These are defined in Table 9 and visualized in Figure 20.

Table 9: Definitions for Data Properties in Maintenance Ontology

Node	Data Property	Type	Definition
Order	OrderNumber	Integer	Uniquely identifies the “Order”
Order	OrderStatus	String	1 of 4 predefined categories (e.g., “resolved”)
Order	NotificationDateTime	Datetime	When is malfunction notified
Order	NotificationText	String	Detailed description of the notification
Order	OrderDescription	String	Short description of the “Order”
Order	DamagePictureText	String	Additional info on the “DamagePicture” category
Order	CauseText	String	Additional info on the “Cause” category
Measure	MeasureNumber	Integer	Uniquely identifies “Measure”
Measure	MeasureDateTime	Datetime	When a measure has been carried out
Measure	MeasureText	String	Detailed description about the “Measure”
Asset	AssetNumber	Integer	Uniquely identifies the “Asset”
Asset	ElementID	Integer	Unique ID from the BIM-Model
Room	RoomNumber	String	Uniquely identifies the “Room”

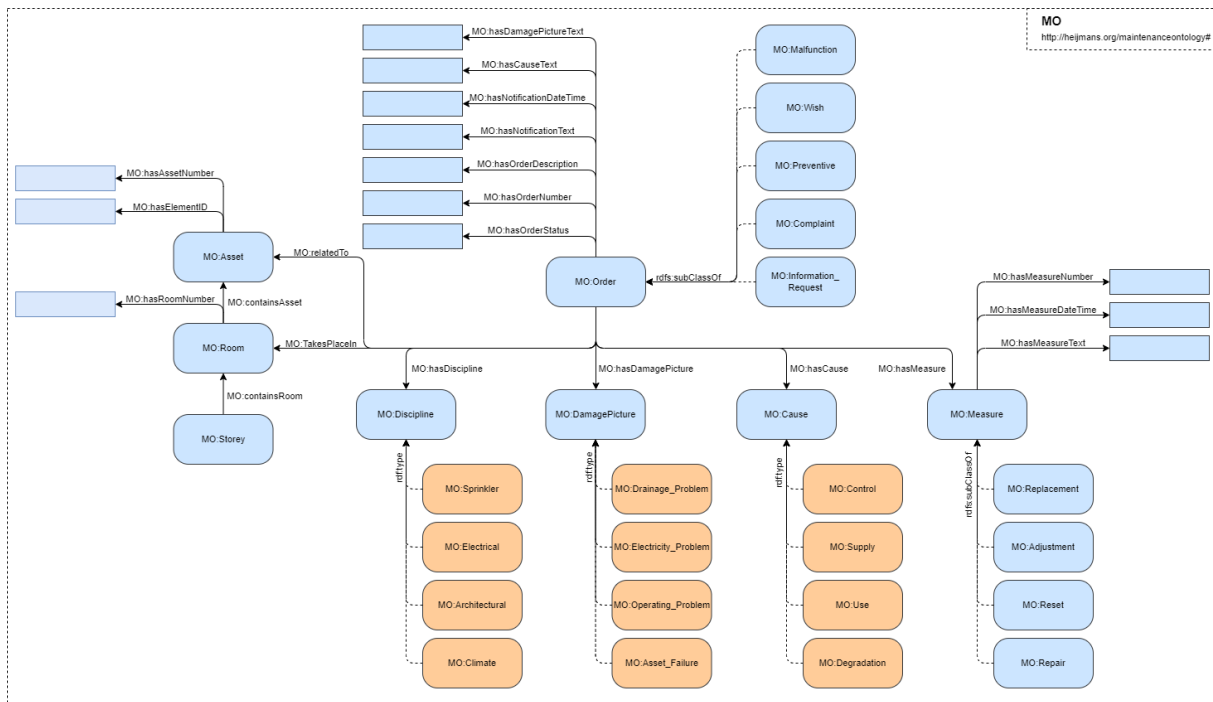


Figure 20: Simplified visualization of the Maintenance Ontology

The maintenance ontology is modelled in protégé, which is an open-source ontology editor. To model the ontology, RDF and OWL are used, where the classes are modelled as owl:thing. The eight relationships between classes are modelled as owl:ObjectProperties with a domain, range, and cardinality restrictions when applicable. The thirteen relations between a class and a literal are defined as owl:DataProperties, where the domain is the object of a specific class, and the range is a literal. The ontology is finally exported in Turtle-syntax and can be found in Appendix 8.1.

#### 4.3.2 BOT ontology

As explained in the literature review, BOT is a simple Building Topology Ontology for easy reuse across domain ontologies. BOT allows describing a building via zones and elements. A building can be defined as a bot:Building. A building is located on a site (bot:Site) and has levels (bot:Storey), which have spaces (bot:Space) (See Figure 21). These zones than have relationships with elements. For example, a space can be next to a bounding element (bot:adjacentElement), a space might contain an element (bot:containsElement), and a space might be intersected by an element (bot:intersectingElement) (Rasmussen et al., 2017).

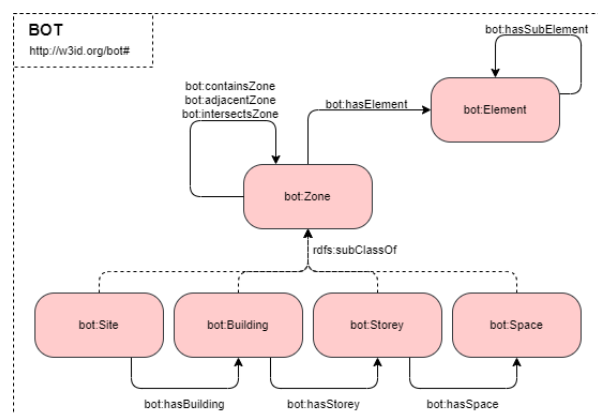


Figure 21: Simplified Visualization of the BOT Ontology

#### 4.3.3 Combined ontology

In the maintenance ontology, the term "Asset" refers to the same concept as the "Element" in the bot ontology, both of which represent a physical object within a building. Similarly, the "Room" concept in the maintenance ontology corresponds to the "Space" concept in the bot ontology. To establish a connection between the two ontologies, two new object properties have been created: "MO:hasbotElementOrigin" and "MO:hasbotSpaceOrigin." The two new relationships are displayed in

Figure 22. The same figure also displays an example of instance data in the ABox and how it relates to the TBox ontology.

The maintenance ontology establishes a common understanding of the data across different maintenance projects, ensuring that the same data is represented in a consistent manner. In practice, all maintenance projects will have their own prefix and namespace. Two examples are:

@prefix EMA: [https://heijmans.org/european\\_medicine\\_agency#](https://heijmans.org/european_medicine_agency#)  
@prefix NACH: [https://heijmans.org/new\\_amsterdam\\_court\\_house#](https://heijmans.org/new_amsterdam_court_house#)

All nodes must have a unique URI. Therefore, the order is identified by its order number, an asset by its asset number and description, a room by its room number and description, and a storey by its level number and description. However, a measure is a specific case. In the SAP maintenance management system, multiple measures can be associated with one order. However, all related measures have the same measure number, making it impossible to uniquely identify the measures only by their number. In order to make them unique, there are multiple possibilities such as indexing (i.e. EMA:Measure\_218668140.1 and EMA:Measure\_218668140.2) or adding a second identifier such as the timestamp (i.e., EMA:Measure\_218668140\_11-1-2023\_07:18:32 and EMA:Measure\_218668140\_11-1023\_13:56:16). After considering both alternatives, the second option with the timestamps was chosen since it does not require manual changes to the root data. Space And Element are explained in detail in Section 4.3.5. All nodes from the ABox have an `rdf:type` relation to the concepts in the TBox of the ontology.

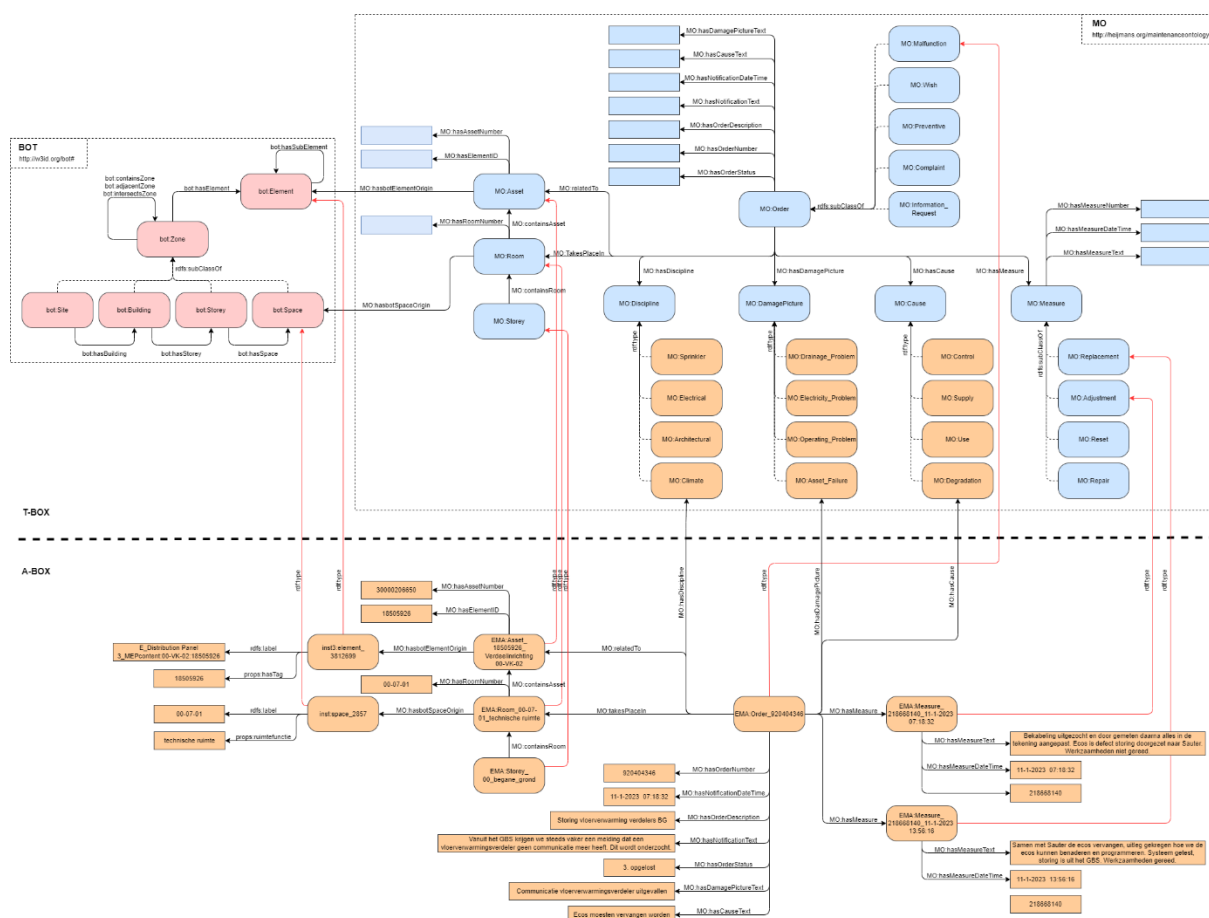


Figure 22: Combined Ontologies with an Example of Instance Data

#### 4.3.4 *Converting SAP MMS data to RDF*

The SAP maintenance management system is setup as a relational database which represents tabular data. Ideally, data should be kept in their own systems, but In order to connect the different systems on a higher level, this tabular data needs to be converted to RDF format according to the structure of the maintenance ontology. There are various ways to obtain data from the SAP maintenance management system. For instance, creating an SQL database that can be connected to or building an API directly from the system. In this thesis, it was chosen to use TimeXtender<sup>1</sup> (a data integration software for automated analytics processes) to export an Excel file to extract all the data from SAP, as it allows for quick and easy access to the data, without the need for extensive coding or customization of the SAP system. Although an SQL database or API directly from the SAP system would be preferable in a later phase of development because the transformation process needs to be automated and performed regularly, it requires significant time and resources to build and maintain. Therefore, for the purposes of this project, the Excel export method was chosen as a more practical solution which enables the project to be completed within the given timeframe.

After extracting the data into an Excel format, the data is imported into Python and converted to RDF format using a Python script. The script is structured in a number of steps, which will be explained in the following section. The full python script is enclosed in Appendix 8.2.

---

<sup>1</sup> <https://www.timextender.com>

## Import Excel and data cleaning

The first part of the code starts with creating the “ConvertExceltoTTL” function and importing the Excel file. The second part of the code performs a series of data cleaning tasks on the imported DataFrame. Data cleaning is necessary, because some of the characters in the original data may not be allowed in URIs when converting the data to RDF later on. Certain characters, such as spaces and special characters like parentheses and slashes, may cause errors or inconsistencies in the RDF data if they are not removed or replaced with allowed characters beforehand. This way, the script ensures that the data is in a format that can be properly converted to RDF data without any issues or errors. Listing 2 displays the pseudocode for part 1 and 2.1 of the full python script.

```
IMPORT the Pandas and Numpy libraries

// 1) Import Excel
CREATE function called ConvertExceltoTTL
    IMPORT Excel file
    PARSE dates in the "MeasureDateTime" and "NotificationDateTime" Columns as DateTime
// 2) Data Cleaning
// 2.1) Clean strings
    REPLACE given string characters by desired characters for multiple columns in dataframe
```

*Listing 2: Pseudocode fragment showing part 1 and 2.1*

## Remove inconsistencies in the “MeasureText” column

Sometimes, the “MeasureText” column is provided with an incorrect value. Instead of a description of the measure, a short version of the “NotificationText” is provided. These wrong values are therefore replaced with an empty string. The cells that are empty are replaced with the string “No text available.” To ensure that the data is in a format that can be properly converted to RDF data without any issues or errors, any double quotes (") in the “MeasureText” column are replaced by an empty string. Listing 3 displays the pseudocode for part 2.2 of the full python script.

```
// 2.2) Remove inconsistencies in "MeasureText" columns
SET MeasureText TO EMPTY string FOR rows where MeasureText is not null and starts with "Summary"
FILL null values in MeasureText with "No text available."
REPLACE double quotes in MeasureText with empty string
```

*Listing 3: Pseudocode fragment showing part 2.2*

## Extract description from full notification

In the SAP maintenance management system, notifications can be processed directly in the system, but preferably, notifications are officially handed in by building occupants via a specific format. An example of such a notification is shown in Listing 4.

```
0000000000000000000010001640438 Contract: 15489
Description Notification: Tr19-5 (19-07-04) generates an alarm in
Explanation Notification: Summary -
Tr19-5 (19-07- 04) generates an alarm in sky walker
Description -
The door of TR19-5 (19-07-07) is physically locked but generates an alarm in Skywalker. Could you please
assist with this issue?

Reporter Name: John Doe
Reporter Email: john.doe@example.com
Location ID: 101637G0119
Location Description: 19 - Nineteenth Floor
```

*Listing 4: Example of Notification*



When a notification is processed directly into SAP, only the description is included. However, in the case of an official notification, the descriptions need to be extracted from the entire text. The process of extracting the descriptions can be challenging because the last properties sometimes happen to be in a different order, or the description may end with personal information. The python script addresses these challenges and ensures that only the descriptions remain after running this section of the script.

The script starts with defining a starting word and a list of ending words. The function returns a substring of the text starting from the starting word and ending with the first occurrence of any of the ending words. The function is applied to the "NotificationText" column only for cells that start with "0000". The resulting text is stored in a new column "NotificationTextFiltered". Finally, double quotes are removed to prevent issues. The pseudocode for part 3 of the full python script is displayed in Listing 5 and Listing 6.

```
// 3) Extract descriptions from notifications
// 3.2) Shorten_text function
CREATE function called shorten_text
  // Define starting and ending word
  SET starting_word to "Description - "
  SET ending_words to ["Melder", "Naam:", "Oorzaak:", "Comment -", "ID:", "Melder:", "Best wishes,",
"Kinds regards,", "Many thanks," "Thank you"]
  FIND index of of starting word in text and STORE as "start_index"
  INITIALIZE ending index to the end of the text
  ITERATE over the list of ending words to FIND the closest ending index
    IF ending word is found after the starting index and before the current end index
      UPDATE the end index
  EXTRACT description from text starting with start_index +14 and ending with end_index
```

Listing 5: Pseudocode fragment showing part 3 and 3.2

```
// 3.1) Clean original "NotificationText" values
FILL null values in "NotificationText" column with "No text available."
REMOVE new lines
// 3.3) Apply shorten_text function
CREATE new column called "NotificationTextFiltered"
IF cells in "NotificationText" column that start with "0000"
  THEN APPLY shorten_text function
  ELSE KEEP original values
REPLACE double quotes and backslashes in NotificationTextFiltered with empty string
```

Listing 6: Pseudocode fragment showing part 3.1 and 3.3.

## MeasureDateTimes

The last step in the data cleaning process is to create a new temporary column for the MeasureDateTimes, where spaces are replaced by underscores. As mentioned in Section 4.3.3, all measures related to one order have the same measure number, making it impossible to uniquely identify the measures only by their number. Therefore, the measure datetime is added to the measure to uniquely identify it. Listing 7 displays the pseudocode for part 2.2 of the full python script.

```
// 4) Temporary "MeasureDateTime_" column
CREATE new column called "MeasureDateTime_"
REPLACE spaces in MeasureDateTime_ with underscores
```

Listing 7: Pseudocode fragment showing part 4

## Create triples according to the maintenance ontology

In this part of the code the cleaned tabular data is converted into triples in Turtle syntax. The script creates new columns that are named after the relationships from the Maintenance Ontology. This is



done for all object properties, all `rdf:type` and `rdfs:subClassOf` relations with the “MO”-Ontology, and all data properties. An example of an object property is the new column “Order\_relatedTo\_Asset”. The line in the script combines the values from the “OrderNumber”, “ElementID”, and “AssetDescription” columns to create a triple that specifies the relationship between the order and relating asset (e.g., “EMA:Order\_910030377 MO:relatedTo EMA:Asset\_17447241\_Trafo”).

The next part defines all triples representing `rdf:type` and `rdfs:subClassOf` relations with the “MO”-Ontology. An example of such a triple is the column “Order\_type\_MO:NotificationType”. This line in the script combines the values from the “OrderNumber” and “NotificationType” columns to create triples that specify the type of notification associated with the order (e.g., “EMA:Order\_920254071 rdf:type MO:Malfunction”).

The next part creates all data properties. An example of a data property is the new column “Order\_hasOrderStatus\_Literal”. This line in the script combines the values from the “OrderNumber” and “OrderStatus” columns to create triples that specify the order status associated with the order (e.g., “EMA:Order\_920254071 MO:hasOrderStatus “Resolved”). The pseudocode for part 5 of the full python script is displayed in Listing 8.

```
// 5.1) Object Properties
CREATE a new column "Order_relatedTo_Asset" in the DataFrame
CONCATENATE strings and values from the columns "OrderNumber", "ElementID", and "AssetDescription"
to create triple
// 5.2) rdf:type and rdfs:subClassOf relationships
CREATE a new column "Order_type_MO:NotificationType" in the DataFrame
CONCATENATE strings and values from the columns "OrderNumber" and "NotificationType" to create
triple
// 5.3) Data Properties
CREATE a new column "Order_hasOrderStatus_Literal" in the DataFrame
CONCATENATE strings and values from the columns "OrderNumber" and "OrderStatus" to create triple
```

Listing 8: Pseudocode fragment showing part 5.1, 5.2, and 5.3

### Transform triples in dataframe to Turtle file

The final part of the code transforms all created triples from the previous part into a Turtle file that can be used to represent the data in a graph-based format that is compatible with RDF-based tools and systems. First all empty and duplicate values are removed, after which the dataframe is transformed into one single list of triples. Subsequently, all values in the list are appended with a space and a period to indicate the end of the statement. Then, the “join” function is used to join all the statements with new lines (“\n”) to create a single string of triples. The last step is to export the converted data to a turtle file. Initially, all the necessary prefixes, including the maintenance ontology, a project placeholder, `rdf` and `rdfs`, are added to the variable named “Prefixes”. subsequently, a turtle file is created with the name “Converted Data.ttl”, after which both the prefixes and triples are written to the turtle file. Listing 9 displays the pseudocode for part 6 until 10 of the full python script.

```
// 6) Remove empty cells
CREATE list of string_parts that need to be removed, such as ["_nan", ":nan", "nan"].
ITERATE over each cell in the DataFrame.
    FOR each cell, CHECK if any of the string_parts are present in string
    IF any of the string_parts are present
        THEN REPLACE the cell's value with an empty string
        ELSE KEEP original value

// 7) Export Excel file (to check manually)
EXPORT dataframe to Excel

// 8) Create list from dataframe
CREATE list of column names
CREATE one list from all cells present in the defined columns called all_cells
IF value is empty or duplicate
    EXCLUDE from list

// 9) Transform values in list to tripples with a space and period
ITERATE over each item in the list of all cells
    CONCATENATE a space and period to the end of each item and append to the modified list
JOIN the modified list of cells into one string separated by newline characters

// 10) Export to TTL file
DEFINE prefixes as string
OPEN a file called "Converted Data.ttl" using UTF-8 encoding
WRITE the prefixes and triples to the file
```

Listing 9: Pseudocode fragment showing part 6 until 10

The full detailed script is enclosed in Appendix 8.2. Listing 10 displays a snippet of the output after running the full ExceltoTTL function:

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix MO: <https://heijmans.org/maintenanceontology#> .
@prefix EMA: <https://heijmans.org/EMA#> .

EMA:Order_920342044 MO:hasCauseText "vloertegel ( pvc ) ligt los" .
EMA:Measure_201169188_2022-07-19_11:25:11 rdf:type MO:003_Reset .
EMA:Order_920253434 MO:hasMeasure EMA:Measure_24478670_2020-08-05_11:52:35 .
EMA:Measure_183239373_2022-02-09_11:36:10 MO:hasMeasureDateTime "2022-02-09 11:36:10" .
EMA:Order_920235117 MO:hasMeasure EMA:Measure_24368039_2020-07-23_12:16:28 .
EMA:Measure_24361719_2021-06-21_15:12:49 MO:hasMeasureDateTime "2021-06-21 15:12:49" .
EMA:Order_920378993 MO:hasOrderDescription "Goods entrance out (00-33ab) door handle" .
EMA:Order_920375667 MO:hasCause MO:Toelevering .
EMA:Order_920262614 MO:hasOrderNumber "920262614" .
EMA:Room_00a-04-02_centrale_reproductie MO:hasRoomNumber "00a-04-02" .
EMA:Order_920272264 MO:hasNotificationDateTime "2020-12-03 08:35:55" .
EMA:Order_920320707 MO:relatedTo EMA:Asset_20114311_LG_Displays .
EMA:Order_920356739 MO:hasDamagePictureText "last van de zon." .
EMA:Order_920357349 MO:hasOrderNumber "920357349" .
EMA:Order_920354672 MO:hasOrderStatus "4. opgelost en afgehandeld" .
---
```

Listing 10: Snippet of output from the script

#### 4.3.5 Converting Building Information Models to RDF

As mentioned in the data extraction layer section, we assume that projects are designed using a BIM protocol and BIM execution plan that mandates the use of separate aspect models for each discipline involved in the project, such as architectural, structural, electrical, and HVAC/MEP, where the architectural model includes information on the project's rooms. These models are combined to form the coordination model of the project. The Revit work-sharing files are stored in a 'Work in Progress' (WIP) folder. Periodically the BIM models are exported as IFC and stored in a 'Published' Folder.

Section 2.5.3. of the literature review discussed different ontologies. To this date, several conversion tools have been developed to improve the conversion of building information models to the RDF format. A few examples are Pauwels & Terkaj (2016) and Hoang and Törmä (2015), who have created tools for converting IFC files to RDF using the IfcOwl ontology. In later stages, it became evident that IFC (thus IfcOwl) store an excessive amount of information, given their exhaustive representation of various data types. Therefore, research on more compact ontologies such as BOT raised popularity. Also in this research, the BOT ontology is used to define the core topological concepts of a building and PROPS<sup>1</sup> is used to describe all element properties. In more recent years, other converters were introduced that implement domain-specific ontologies, such as the IFC-to-LBD converter developed by Oraskari et al. (2018) and the Revit-bot-exporter<sup>2</sup> that creates RDF exports directly from Autodesk Revit. Both converters use the BOT ontology as the main ontology, along with other domain ontologies.

However, none of the conversion tools are perfect and different use cases involve manual steps to successfully convert all building information model data to RDF. Also in this research, customization was needed. In the following section all choices made, and steps taken to convert the IFC files to RDF are explained.

#### Converting Separate Files or One Merged File

Before converting the IFC files to RDF, it needs to be taken in consideration if the Revit files are merged first into a single Revit file or not. The two options are visualized in Figure 23. The reason for considering the files to be merged into one single Revit file is the amount of bot:hasElement relationships that are created after conversion. In the case of separate aspect files, these relations could only be made in the architectural model, since that is the only model that involves the Rooms. However, merging the different aspect models into one single Revit file can be time-consuming or could even cause problems. In an

ideal situation, all aspect models could be copied and pasted in the architectural model aligned to the same place. However, in practice, copying an entire Revit model into another, causes double elements and many other errors. Another problem is the presence of manual room separation lines in the architectural model. Instead of a wall from the structural or interior model, the room boundary is now defined by a line in Revit and there is no guarantee that the definition of the room boundary changes from the line to the new wall from, for instance the structural model. The bot:hasElement relationships are created when using the IFC property "IfcRelSpaceBoundary", so in order to create all relationships,

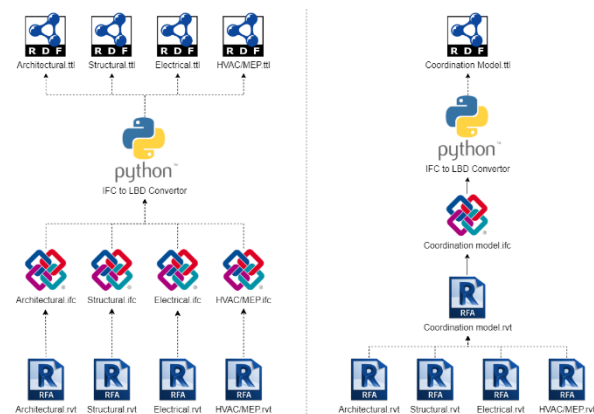


Figure 23: Two Ways to Convert IFC to RDF

<sup>1</sup> <https://github.com/w3c-lbd-cg/lbd/blob/gh-pages/presentations/props/>

<sup>2</sup> <https://github.com/pipauwel/revit-bot-exporter>

<sup>3</sup> <https://github.com/ISBE-TUe/IFCtoLBD>

all room separation lines that are replaced by the new walls must be deleted manually. To overcome this problem in new building projects, it is critical to define modelling guidelines that describe how the rooms should be modelled, so that all room bounding elements actually define the boundary of a room. Only that way, all space-element relationships can be transferred to the RDF/based maintenance management system. However as mentioned before, in this thesis the IFC models already exist, so the different aspect IFC models are converted separately.

### Convertor Choice

The convertors mentioned on the previous page are all in different phases of development. Initially, the IFC to LBD Convertor by Oraskari et al. (2018) was used for the first round of testing. This is a JAVA application that can be run locally on a computer. The testing involved converting several average sized IFC files at PROPS level 1, and the outcomes were satisfactory. However, the application crashed repeatedly when handling larger IFC files, which revealed that it lacked robustness. This can be explained by two factors: The application stores all data in the computer's random access memory (RAM) during conversion and the application first converts the IFC-SPF file to an IFC-RDF file using the IFC ontology, which is too exhaustive in the amount of triples that it produces, as mentioned earlier.

Another development is the Revit-bot-exporter, which is able to create RDF exports directly from Autodesk Revit. The last working version runs in an older version of Revit, but Revit is a software application which is not backwards compatible, further development is required in order to test models that are created in recent versions of Revit.

The third convertor is a python based IFctoLBD convertor<sup>1</sup>, which is available at the Information Systems in the Built Environment (ISBE) group at the TU/e. The conversion is done by a python script which uses the open source IfcOpenShell library to search for specific objects in the IFC file and translate it to bot:sites, buildings, storeys, spaces, and elements. In the last line of the code, users can input their IFC file name and desired output TTL file name, and by running the Python file, the IFC file is converted. Depending on the initial file size, this convertor is significantly quicker than the IFctoLBD java application, because it directly converts the IFC to bot instead of a workaround via IfcOwl. From the three alternatives, the python based IFctoLBD convertor was chosen.

### Modifications for Thesis Context

To avoid encoding errors, the “convertIFCSPFtoTTL” function was modified slightly, so that the new file is encoded using the UTF-8 character encoding. The change is displayed in Listing 11.

```
def convertIFCSPFtoTTL(inputFile, outputFile):
    inputFile = inputFile
    outputFile = outputFile
    now = datetime.now()
    current_time = now.strftime('%Y%m%d_%H%M%S')

    DEFAULT_PATH = "http://linkedbuildingdata.net/ifc/resources" + current_time + "/"
    global baseURI
    baseURI = DEFAULT_PATH

    #reading file - IfcSpfReader
    model = ios.open(inputFile)

    f = open(outputFile, "w", encoding='utf-8')
    writeTTLFileContent(model,f)
    f.close()
```

Listing 11: Modification to convertIFCSPFtoTTL function

---

<sup>1</sup> <https://github.com/ISBE-TUe/IFctoLBD>

After resolving the character encoding errors, the conversion went well. Smaller sized IFC models only lasted seconds, average sized IFC models between 5 and 20 minutes, and larger models took about 30 to 60 minutes. Similar to transforming the data from the maintenance management system, there are forbidden characters in literal strings. To prevent issues, the “cleanNameString” function was created, which replaces all occurrences of the forbidden characters into an empty string. Every other file that was tested afterwards was converted successfully. The change is displayed in Listing 12 and the full script is enclosed in Appendix 8.2.

```
def cleanNameString(name):
    name = ''.join(x for x in name.title() if not x.isspace())
    name = name.replace('\\', '').replace('/', '').replace('(', '').replace(')', '').replace('.', '')
    name = name.replace(',', '').replace('*', '').replace('!', '').replace('@', '').replace('~', '').replace('#', '')
    return name
```

Listing 12: modification to cleanNameString function

## 4.4 Integrating data from MMS and IFC models

This section will discuss the part of the data pipeline where all data in RDF format is integrated and will give answer to sub-question 5: “What strategies can be utilized to merge data from multiple systems effectively?” The TTL files that were created in the previous section are input for a Python script which creates the “MO:hasbotElementOrigin” and “MO:hasbotSpaceOrigin” relationships. As explained in Section 4.3.3, by creating these relationships the data from the IFC models is integrated in the MMS. The end result of after running the Python script is one TTL file, that contains all data from the MMS and the IFC models. In Section 4.4.2, strategies that enable automated updating of the data in the system are discussed. This part of the data pipeline is visualized in Figure 24.

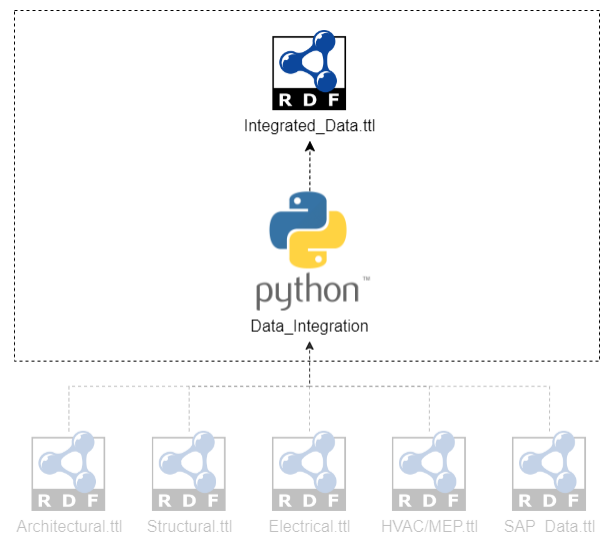


Figure 24: Schematic Visualization of the Integration of Data

### 4.4.1 Integrating SAP with BIM

As described in Section 4.3.3, to establish a connection between the maintenance ontology and bot ontology, and thus integrating all data, two new object properties must be created between the “Asset” and “Element” (“MO:hasbotElementOrigin”) and between the “Room” and “Space” (“MO:hasbotSpaceOrigin”). To create the two new relationships in the data, python is used in combination with the RDFLib package. The script is divided into two parts. The first part is the creation of the “MO:hasbotElementOrigin” relationship, which is explained in detail below.

#### Asset Element relationship

The script starts by defining a function called “CreateAssetRelationship”, which creates a new relationship between the Element and the Asset by adding a triple to the graph. Subsequently, the script includes a function called “Map\_ElementID\_Tag”, which involves loading multiple RDF files into a single graph. Then two SPARQL queries are performed to identify all the assets and corresponding ElementID’s and all elements and their corresponding Tags. Then, the script iterates over all rows from the two query results and if the “ElementID” property of the Asset matches the “Tag” property of the

Element, a new relationship is added to the graph. Finally, the graph is serialized to a TTL file that contains the original data from the input files as well as new relationships between Assets and Elements that were created by the script. Listing 13 displays the pseudocode for this part of the integration python script.

```
IMPORT the pandas and numpy libraries

CREATE function called CreateAssetRelationship
    DEFINE the relationship name "hasbotElementOrigin"
    CONSTRUCT the relationship URI using the namespace and relationship name
    CREATE a new URI reference object for the relationship URI
    ADD the relationship as a triple to the graph

CREATE function called MAP_ElementID_Tag
    // Create graph and load files
    Create an empty RDF graph called "g"
    PARSE the "Converted Data.ttl" file from the MMS into the graph
    PARSE the different converted IFC models into the graph
    BIND the prefix "MO" to the namespace "https://heijmans.org/maintenanceontology#"

    // Search Assets in MMS
    DEFINE SPARQL query that searches for EMA:Assets and their corresponding ElementID's
    QUERY the graph "g" using the SPARQL query "AssetQuery"
    STORE query result in a variable called "gres"

    // Search Elements in BIM data
    DEFINE SPARQL query that searches for bot:Elements and their associated Tag
    QUERY the graph "g" using the SPARQL query "ElementQuery"
    STORE query result in a variable called "gres1"

    // Create relationships between Assets and Elements
    FOR each row in the "gres" variable:
        FOR each row in the "gres1" variable:
            IF the ElementID of the Asset matches the Tag of the Element:
                CREATE relationship using the "CreateAssetRelationship" function

    // Write the TTL file
    SERIALIZE the graph "g" into the file "Merge_Assets_Elements.ttl"
```

*Listing 13: Pseudocode Fragment showing the creation of the MO:hasbotElementOrigin relationships*

## Room Space Relationship

When all “MO:hasbotElementOrigin” relationships are added to the graph, only the relationships between the Rooms in SAP and the Spaces in the BIM model have to be created. In the same script, this is done with a very similar approach. The “Merge\_Assets\_Elements.ttl” file is parsed into a new graph, after which again two SPARQL queries are performed that identify all Rooms and Spaces, with their corresponding number. Then, the script iterates over each row in the query results and if the “RoomNumber” property of the Room matches the “SpaceNumber” property of the Space a new relationship is added to the graph. If there is a match, the “CreateRoomRelationship” function is called. Finally, the graph including all original data, as well as new relationships between Assets and Elements and Rooms and Spaces is serialized to a final TTL file called: “Merge\_Assets\_Elements\_Rooms\_Spaces”. This part of the script is displayed in Listing 14.

```
CREATE function called CreateRoomRelationship
  DEFINE the relationship name "hasbotSpaceOrigin"
  CONSTRUCT the relationship URI using the namespace and relationship name
  CREATE a new URI reference object for the relationship URI
  ADD the relationship as a triple to the graph

CREATE function called MAP_RoomNumber_SpaceNumber
  // Create graph and load files
  Create an empty RDF graph called "g"
  PARSE the "Merge_Assets_Elements.ttl" file into the graph
  BIND the prefix "MO" to the namespace "https://heijmans.org/maintenanceontology#"

  // Search Rooms in MMS
  DEFINE SPARQL query that searches for EMA:Rooms and their corresponding RoomNumbers
  QUERY the graph "g" using the SPARQL query "RoomQuery"
  STORE query result in a variable called "qres2"

  // Search Spaces in BIM data
  DEFINE SPARQL query that searches for bot:Spaces and their associated SpaceNumber
  QUERY the graph "g" using the SPARQL query "SpaceQuery"
  STORE query result in a variable called "qres3"

  // Create relationships between Rooms and Spaces
  FOR each row in the "qres2" variable:
    FOR each row in the "qres3" variable:
      IF the RoomNumber of the Room matches the SpaceNumber of the Space
        CREATE relationship using the "CreateRoomRelationship" function

  // Write the TTL file
  SERIALIZE the graph "g" into the file "Merge_Assets_Elements_Rooms_Spaces.ttl"
```

Listing 14: Pseudocode Fragment showing the creation of the MO:hasbotSpaceOrigin relationships

#### 4.4.2 Automating the Data Pipeline

The web applications in the presentation layer of the system architecture should always have access to up-to-date data from the original systems. Therefore, the data pipeline is designed in such a way, that the process is automated. The 3 python scripts are merged together into one script. The final part of the script, where the functions are called is displayed in Listing 15

```
# IFC to LBD Convertor
convertIFCSPFtoTTL("Project_Architectural.ifc", "Project_Architectural.ttl")
convertIFCSPFtoTTL("Project_Structural.ifc", "Project_Structural.ttl")
convertIFCSPFtoTTL("Project_Electrical.ifc", "Project_Electrical.ttl")
convertIFCSPFtoTTL("Project_HVAC_MEP.ifc", "Project_HVAC_MEP.ttl")
# SAP to RDF Convertor
ConvertExceltoTTL()
#Data Integration
Map_ElementID_Tag()
Map_RoomNumber_SpaceNumber()
```

Listing 15: Final part of combined python script where functions are called

There are several possibilities for automating the process. One of the options is scheduling an automatic action to run the script on a regular interval. This can easily be done with for instance windows Task Scheduler.

As described earlier in Section 4.3.4, in a later phase of development the transformation process needs to be fully automated and performed regularly. Because the maintenance data is dynamic it needs to be converted on a regular interval, preferably daily or every 12 hours.

The IFC files are updated less frequently and converting larger BIM models can be time-consuming. Therefore, it is unnecessary to call the IFC to LBD converter functions every time the script runs. Instead, these functions should only be called when one or more IFC models are refreshed in the



folder. This can be accomplished by using an if statement that checks the date of the file against the date of the previous conversion. If the file has a newer date compared to the previous conversion, then the corresponding convertIFCSPFtoTTL function should be called. An example of a way to code this is shown in .

```
import os
import time

def check_files():
    # Define the path to the folder containing the IFC files
    folder_path = '/path/to/folder'

    # Define a list of the IFC files to check
    files_to_check = [
        'Project_Architectural.ifc',
        'Project_Structural.ifc',
        'Project_Electrical.ifc',
        'Project_HVAC_MEP.ifc'
    ]

    # Get the current time minus 24 hours
    cutoff_time = time.time() - (24 * 60 * 60)

    # Loop through each file in the folder
    for filename in os.listdir(folder_path):
        if filename in files_to_check:
            filepath = os.path.join(folder_path, filename)

            # Get the modification time of the file
            file_time = os.path.getmtime(filepath)

            # Compare the file time to the cutoff time
            if file_time > cutoff_time:
                # Call the conversion function for this file
                output_file = filename.replace('.ifc', '.ttl')
                convertIFCSPFtoTTL(filepath, os.path.join(folder_path, output_file))

def convertIFCSPFtoTTL(input_file, output_file):
    # Your conversion code goes here
    print(f'Converting {input_file} to {output_file}')

if __name__ == '__main__':
    check_files()
```

Listing 16: Example of python script that checks for updated IFC models

## 4.5 Presentation Layer / User Interface

The presentation layer or user interface of the system architecture (see Figure 25) consists out of two applications for different members in the asset management team. On the one hand, there is an application for data analysis with data visualization and querying functionalities, and on the other hand, LBDviz is a web-application that integrates the RDF data with an IFC viewer with predefined functionalities. The application for data analysis has more advanced querying and data visualization functionalities, while the LBDviz web-application is more intuitive to use for stakeholders who are less familiar with the underlying data in. This section will partially answer sub-question 6: “What specific methods or tools can be utilized by different end users in the asset management team to generate insights from the data?”

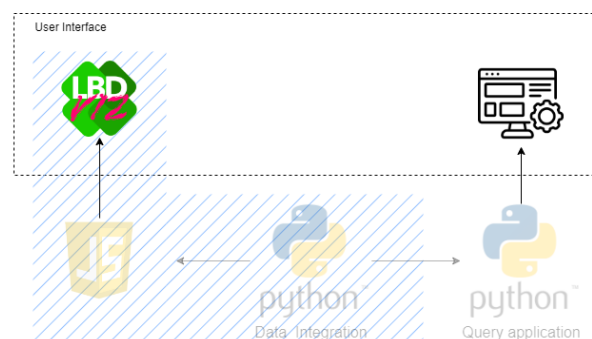


Figure 25: Schematic Visualization of Presentation Layer

#### 4.5.1 Triple Store

There is a number of available commercial and open-source triple stores that can be used to store the integrated maintenance data. A few examples are:

**Neo4j**<sup>1</sup> - a popular open-source graph database that is used for storing, querying, and manipulating graph data. It is known for its performance, scalability, and flexibility. It supports a wide range of use cases, such as fraud detection, social networking, recommendation engines, and network analysis.

**Amazon Neptune**<sup>2</sup> - a fully managed graph database service provided by Amazon Web Services (AWS). It is highly scalable, secure, and reliable, and is optimized for handling large-scale graph data. It supports open standards like Apache TinkerPop and SPARQL and can be used for a variety of use cases, including knowledge graphs, recommendation systems, and fraud detection.

**JanusGraph**<sup>3</sup> - an open-source graph database that is optimized for distributed computing environments. It is highly scalable and supports a variety of graph storage backends, including Apache Cassandra, Apache HBase, and Google Cloud Bigtable. It is designed to handle large-scale graphs and supports a range of use cases.

**AllegroGraph**<sup>4</sup> - a graph database that is designed for handling complex data models and semantic data. It supports a range of query languages, including SPARQL, Prolog, and Java, and is optimized for handling large-scale semantic data. It is used for a variety of use cases, including knowledge graphs, recommendation systems, and natural language processing.

**Ontotext GraphDB**<sup>5</sup> - a semantic graph database that is optimized for storing and querying large-scale semantic data. It supports RDF, OWL, and SPARQL standards, and provides powerful text analysis capabilities. It is used for a variety of use cases, including knowledge management, content enrichment, and fraud detection. It offers high availability, scalability, and performance, and can be deployed on-premises or in the cloud.

In this thesis, the choice is made for Ontotext GraphDB, for the reason that it is free, scalable, flexible, has high performance, is optimized for complex querying of data, has potential for integration, and has active communities that offer support, documentation, and tutorials. GraphDB has three versions: free, standard, and enterprise. The difference between the free and standard versions is that the standard version has more computational power. The enterprise version adds the functionality of semantic inference, which implies the possibility to derive new semantic facts from existing facts. In this research, the free version is used.

#### 4.5.2 Application for Data Analysis

As mentioned in Section 4.1, the Application for data analysis will be explained briefly, but further development falls outside of the scope of the thesis and is therefore directly input for future research. From the previous section it became clear that Ontotext GraphDB is used as a database, but in order to create a user interface that is workable for asset managers and maintenance engineers, a new development of code that retrieves and processes the data from GraphDB is needed in the business logic layer. For instance, when a button is clicked in the user interface, the application should retrieve and process the data from GraphDB in the background and display it to the user in a structured and appealing way.

---

<sup>1</sup> <https://neo4j.com/>

<sup>2</sup> <https://aws.amazon.com/neptune/>

<sup>3</sup> <https://janusgraph.org/>

<sup>4</sup> <https://allegrograph.com/>

<sup>5</sup> <https://www.ontotext.com/products/GraphDB/>

#### 4.5.3 Data Visualization Tool

Besides the Ontotext GraphDB triple store, a web app is created for stakeholders that are less familiar with the underlying data and lack the knowledge to query data using SPARQL. To demonstrate how such a web application would look like the framework from the research of Donkers (2023) is used. His paper presents a tool that integrates data from different stakeholders into a single viewpoint. The tool – LBDviz – combines a browser-based IFC viewer (using IFC.js<sup>1</sup>), and a knowledge graph querying framework to query RDF data (using Comunica<sup>2</sup>). The application is programmed using JavaScript.

As LBDviz is already a functional application, only a few changes were required in the code to adapt it to the needs of this research. This involved updating the queryComunicaGlobalIdProps JavaScript file and creating two new files that allow two new functionalities in the application (queryComunicaGlobalIdAssets and queryComunicaGlobalIdRooms). The three features are displayed in Table 10.

Table 10: Modified Functionalities in the LBDviz Application

Name	Functionality
queryComunicaGlobalIdProps()	Performs a SPARQL query using the Comunica query engine to retrieve all properties and corresponding values of a selected asset or room in the 3d model identified by its compressed GUID
queryComunicaGlobalIdAssets()	Performs a SPARQL query using the Comunica query engine to retrieve a full record of all historic orders and measures related to a selected asset in the 3d model identified by its compressed GUID
queryComunicaGlobalIdRooms()	Performs a SPARQL query using the Comunica query engine to retrieve a full record of all historic orders and measures that took place in a selected room in the 3d model identified by its compressed GUID

The three functionalities are programmed in a similar way. However, what sets them apart is the underlying SPARQL query that is executed when a particular asset or space is selected in the IFC viewer of the application. To illustrate how the functionalities work, the pseudocode for the queryComunicaGlobalIDProps functionality is displayed in Listing 17

The function starts with importing Comunica query engine after which the function is defined. When a user clicks on an element in the IFC viewer, the contents of the HTML element with an ID is logged. Subsequently, a predefined SPARQL query is executed where the GUID matches the value of the HTML element. After the query is executed, the results are processed and added to the results box in the user interface.

<sup>1</sup> <https://ifcjs.io/>

<sup>2</sup> <https://comunica.dev/>

```
IMPORT the comunica query engine
DEFINE function called queryComunicaGlobalIDProps
  CREATE new instance of query engine and STORE in variable "myEngine"
  LOG contents of HTML element with an ID of "selected-guid" to console
  GET value from "GRAPH-input" and SPLIT into array of graphs
  PERFORM SPARQL query where GUID matches the value of the HTML element with an ID of "selected-guid"
  STORE query results in the variable "bindings"
  CONSTRUCT HTML table to display query results
  LOOP through each binding in the bindings array
  FOR each binding
    LOOP through each result in the binding
      LOG property name and value to the console and ADD a table cell to the HTML table
  ADD HTML table to an HTML element with an ID of "results-box-content"
  EXPOSE the queryComunicaGlobalIdProps function to the global window object
  EXECUTE the queryComunicaGlobalIdProps function
```

*Listing 17: Pseudocode fragment showing the queryComunicaGlobalIDProps functionality*

The three new functionalities are bundled with the rest of the JavaScript files. Subsequently, the three functions are integrated into the HTML file (index.html) to enable users to access them through a button in the find tab. Whenever a user clicks on an element in the IFC viewer and then clicks on the function button in the find tab, the function is executed, and the results obtained from the query are presented in the results tab.

## 4.6 Conclusion

In this chapter, a system architecture and data pipeline including semantic web technologies are created which can be used by organizations that have access to one or more building information models and a maintenance management ERP system. To ensure that the data is properly structured for use in a graph database, an ontology was created first, after which a python script was created to clean and transform the data into RDF format according to the vocabulary of the maintenance ontology. The information from the BIM model was transformed using the BOT and PROPS ontologies. The different data sources were merged by creating new relationships between them using python in combination with the RDFLib package. Finally, different methods and tools were developed for end-users within the asset management team to derive insights from the data, such as Ontotext GraphDB for advanced querying and LBDviz for visualizing RDF data in a user-friendly 3D interface.



## 5 Case Study and Results

### 5.1 Case study EMA

The European Medicines Agency (EMA) project in Amsterdam was designed and built by a consortium including Heijmans and Dura Vermeer. The project involved the construction of a new office building for the EMA, which serves as the regulatory body responsible for the evaluation and supervision of medicines for human and veterinary use in the European Union. The project faced several challenges, including a tight timeline and strict safety requirements due to the nature of the EMA's work. The building had to meet high standards of security and safety, with features such as reinforced concrete walls, advanced fire protection systems, and secure access controls. Despite these challenges, the project was completed on schedule, and the EMA has since relocated to its new headquarters in Amsterdam. The building provides a modern and functional workspace for the EMA's employees, with state-of-the-art facilities such as meeting rooms, training facilities, and a large auditorium. Figure 26 displays a number of photos and the factsheet of the European Medicines Agency.

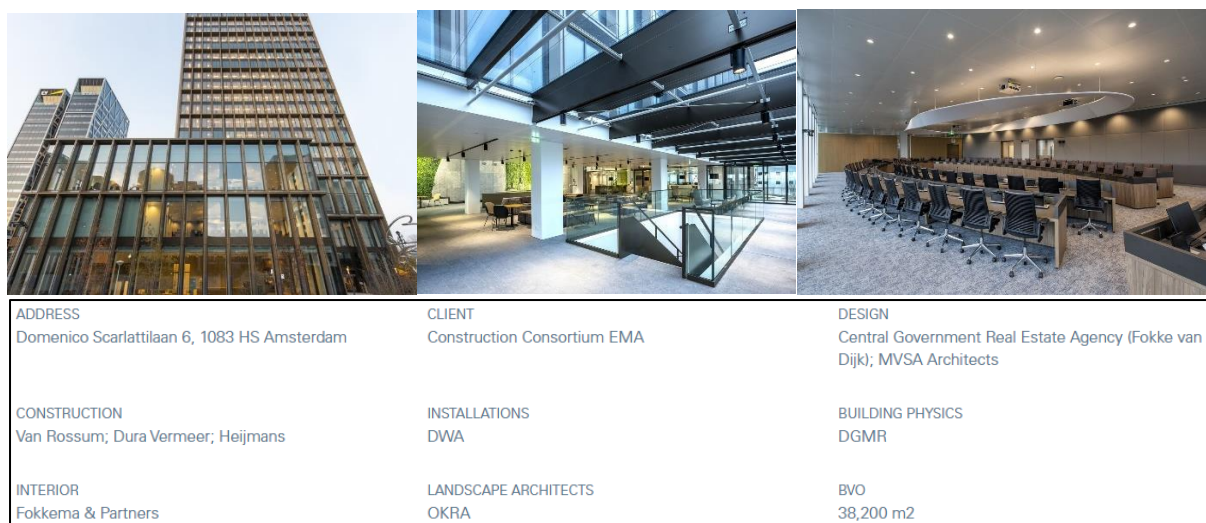


Figure 26: Photos and Factsheet of the European Medicines Agency in Amsterdam

After completion of the project, Heijmans is responsible for the operation and maintenance of the building for a period of 10 years. Due to the project's delivery with an extremely comprehensive Building Information Model, coupled with a maintenance management system including over 3000 orders collected within 2 years and 8 months, the building has been selected as an ideal case study for the system developed in the methodology section.

### 5.2 Starting Situation

#### 5.2.1 TimeXtender output in excel

On the twenty third of January 2023 an export was done from TimeXtender which resulted in an excel file called "2023-01-23 EMA Storingen snip YTD.xlsx". The 1382 kb file contains 22 columns and 5806 rows representing all orders and measures.

#### 5.2.2 IFC files

The project is divided into an office building section and congress building section and then sub divided in 8 disciplines. Together they form the total coordination model of the EMA. The individual models are listed in Table 11. Three examples of IFC models are visualized in Figure 27.

Table 11: IFC Subdivision with File Sizes

Model Discipline	Size Congress	Size Office
Architectural	486,550 kb	27,681 kb
Structural	53,160 kb	70,979 kb
Electrical	229,934 kb	226,261 kb
HVAC/MEP	629,767 kb	432,435 kb
Security	6,180 kb	2,757 kb
Sprinkler	220,703 kb	539,794 kb
Interior	165,007 kb	233,801 kb
Furniture	37,669 kb	60,859 kb
Total	1,828,970 kb	1,594,567 kb

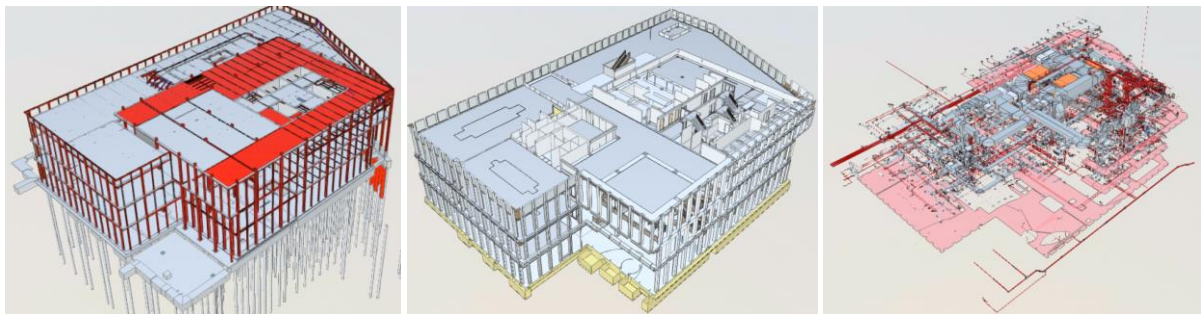


Figure 27: 3D Visualization of the Congress IFC Models: Structural, Architectural, HVAC/MEP

### 5.3 Running Script

In this part, the full data integration script is executed. The script is run in a virtual environment on a x64-based HP Pavilion Power Laptop 15-cb0xx, with an Intel(R) Core(TM) i7-7700HQ CPU @ 2.80GHz, 2808 MHz, 4 core('s) processor and a Random Access Memory (RAM) of 16,0 GB.

#### 5.3.1 IFC2LBD

The first part of the script that converts the IFC models into TTL files was executed successfully without any issues. The process was completed within a duration of 13 hours, 6 minutes, and 30 seconds. The individual times and amount of triples are listed in Table 12. The number of triples is extracted using the SPARQL query displayed in Listing 18.

```
SELECT (COUNT(*) AS ?count)
WHERE {
  ?subject ?predicate ?object .
}
```

Listing 18: SPARQL query for number of triples

Table 12: Output TTL file with File Sizes and Number of Triples

Model Discipline	Time Congress	Triples Congress	Time Office	Triples Office
Architectural	45 min, 33 sec	980,113	11 min, 11 sec	470,623
Structural	4 min, 57 sec	385,764	11 min, 38 sec	688,206
Electrical	41 min, 27 sec	1,119,194	116 min, 59 sec	1,983,687
HVAC/MEP	70 min, 16 sec	1,067,829	64 min, 20 sec	759,926
Security	23 sec	117,680	9 sec	70,668
Sprinkler	52 min, 5 sec	1,226,462	290 min, 21 sec	1,117,040
Interior	25 min, 34 sec	734,052	47 min, 57 sec	988,202
Furniture	25 sec	84,670	3 min, 40 sec	299,025



Total	4 hrs, 15 sec	5,715,764	9 hrs, 6 min, 15 sec	6,357,377
-------	---------------	-----------	----------------------	-----------

### 5.3.2 SAP Conversion

The conversion of SAP data to RDF format proceeded smoothly without any issues. The process was completed within a duration of 9.92 seconds. The resulting output file, named "Converted Data.ttl," has a size of 4,884 kb. To check the number of triples in the TTL file, the following SPARQL query was executed:

Query	Results
<pre>SELECT (COUNT(*) AS ?count) WHERE {   ?subject ?predicate ?object . }</pre>	"67,884"^^xsd:integer

To check the amount of orders in the TTL file, the following SPARQL query was done:

Query	Results
<pre>PREFIX rdf: &lt;http://www.w3.org/1999/02/22-rdf-syntax-ns#&gt; PREFIX MO: &lt;https://heijmans.org/maintenanceontology#&gt; SELECT (COUNT(?Order) as ?NumOrders) WHERE {   ?Order rdf:type MO:Order . }</pre>	"3,079"^^xsd:integer

### 5.3.3 Integration

The Integration between the SAP data and IFC files was also executed successfully without any issues. The "Map\_ElementID\_Tag" function was completed within a duration of 28 minutes and 41 seconds. The "Map\_RoomNumber\_SpaceNumber" function was completed within a duration of 22 minutes and 5 seconds. To check the number of triples in the TTL file, the following SPARQL query was executed:

Query	Results
<pre>SELECT (COUNT(*) AS ?count) WHERE {   ?subject ?predicate ?object . }</pre>	"11,651,759"^^xsd:integer

To check how many relationships are made between the Assets and Elements and between the Rooms and Spaces, multiple SPARQL queries are done. First the amount of assets and relationships between assets and elements is counted, after which the same is done for the rooms and relationships between rooms and spaces.

Purpose	Query	Results
Count Assets	<pre>PREFIX MO: &lt;https://heijmans.org/maintenanceontology#&gt;  SELECT (COUNT(?Asset) as ?NumAssets) where {   ?Asset a MO:Asset . }</pre>	"183"^^xsd:integer
Count relations Between Assets and Elements	<pre>PREFIX MO: &lt;https://heijmans.org/maintenanceontology#&gt; SELECT (COUNT(?s) AS ?count) WHERE {   ?s MO:hasbotElementOrigin ?o . }</pre>	"149"^^xsd:integer
Count Rooms	<pre>PREFIX MO: &lt;https://heijmans.org/maintenanceontology#&gt;</pre>	"399"^^xsd:integer

	<pre> SELECT (COUNT(?Room) as ?NumRooms) where {     ?Room a MO:Room . } </pre>	
Count relations Between Rooms and Spaces	<pre> PREFIX MO: &lt;https://heijmans.org/maintenanceontology#&gt; SELECT (COUNT(?s) AS ?count) WHERE {     ?s MO:hasbotSpaceOrigin ?o . } </pre>	"399"^^xsd:integer

## 5.4 Answering the Competency Questions

This section demonstrates how the system complies to the requirements / competency questions. For each competency question, a query is performed in GraphDB that demonstrates that the system allows acquiring the information asked in the competency question. Most queries directly answer the competency questions. Query seven and eight do not directly answer the competency question but do retrieve the needed data to answer the question. As mentioned in Section 4.5.2, the business logic layer from the system architecture should also include code that processes the data from the retrieved query results and display it to users in a structured and appealing way. In a later stage of development, the last calculations to answer the competency questions will be incorporated in the business logic layer. All queries are listed below.

### 5.4.1 Select all historic orders and measures related to a specified asset

```

PREFIX MO: <https://heijmans.org/maintenanceontology#>
PREFIX EMA: <https://heijmans.org/EMA#>

SELECT ?Order ?OrderDateTime ?Description ?Measure ?MeasureDateTime ?MeasureText
WHERE {
    ?Order MO:relatedTo EMA:Asset_4469122_Tourniquet_entree .
    ?Order MO:hasNotificationDateTime ?OrderDateTime .
    ?Order MO:hasOrderDescription ?Description .
    ?Order MO:hasMeasure ?Measure .
    ?Measure MO:hasMeasureDateTime ?MeasureDateTime .
    ?Measure MO:hasMeasureText ?MeasureText .
}
ORDER BY ?OrderDateTime ?MeasureDateTime

```

Listing 19: SPARQL query for competency question 1

### 5.4.2 Select all historic orders and measures that took place in a specified room

```

PREFIX MO: <https://heijmans.org/maintenanceontology#>
PREFIX EMA: <https://heijmans.org/EMA#>

SELECT ?Order ?OrderDateTime ?Description ?Asset ?Measure ?MeasureDateTime ?MeasureText
WHERE {
    ?Order MO:takesPlaceIn EMA:Room_01-02-04_conferentie XXL .
    ?Order MO:hasNotificationDateTime ?OrderDateTime .
    ?Order MO:hasOrderDescription ?Description .
    OPTIONAL { ?Order MO:relatedTo ?Asset } .
    ?Order MO:hasMeasure ?Measure .
    ?Measure MO:hasMeasureDateTime ?MeasureDateTime .
    ?Measure MO:hasMeasureText ?MeasureText .
}
ORDER BY ?OrderDateTime ?MeasureDateTime

```

Listing 20: SPARQL query for competency question 2

### 5.4.3 What kind of assets are prone to a specified damage picture?

```
PREFIX MO: <https://heijmans.org/maintenanceontology#>
PREFIX EMA: <https://heijmans.org/EMA#>
PREFIX props: <https://w3id.org/props#>

SELECT ?Description (COUNT(?Description) AS ?count)
WHERE {
    ?Order MO:hasDamagePicture MO:Uitval .
    ?Order MO:relatedTo ?Asset .
    ?Asset MO:hasbotElementOrigin ?Element .
    OPTIONAL { ?Element props:Description ?Description } .
}
GROUP BY ?Description
ORDER BY DESC(?count)
```

Listing 21: SPARQL query for competency question 3

### 5.4.4 What kind of rooms are prone to a specified damage picture?

```
PREFIX MO: <https://heijmans.org/maintenanceontology#>
PREFIX EMA: <https://heijmans.org/EMA#>
PREFIX props: <https://w3id.org/props#>

SELECT ?Description (COUNT(?Description) AS ?count)
WHERE {
    ?Order MO:hasDamagePicture MO:Bedieningsprobleem .
    ?Order MO:takesPlaceIn ?Room .
    ?Room MO:hasbotSpaceOrigin ?Space .
    OPTIONAL { ?Space props:4070_Ruimtefuncties ?Description } .
}
GROUP BY ?Description
ORDER BY DESC(?count)
```

Listing 22: SPARQL query for competency question 4

### 5.4.5 Select all properties from BIM for the asset or room related to order

```
PREFIX MO: <https://heijmans.org/maintenanceontology#>
PREFIX EMA: <https://heijmans.org/EMA#>

SELECT ?Property ?Value
WHERE {
    EMA:Asset_11379573_Hydrofoor_3_pomps MO:hasbotElementOrigin ?element .
    ?element ?Property ?Value .
}
}
```

Listing 23: SPARQL query for competency question 5

### 5.4.6 Select all measures that resolved similar malfunctions

```
PREFIX MO: <https://heijmans.org/maintenanceontology#>
PREFIX EMA: <https://heijmans.org/EMA#>

SELECT ?Order ?OrderDateTime ?Description ?Asset ?Measure ?MeasureDateTime ?MeasureText
WHERE {
    ?Order MO:hasCause MO:Degradatie .
    ?Order MO:hasDamagePicture MO:Uitval .
    ?Order MO:relatedTo ?Asset .
    ?Order MO:hasNotificationDateTime ?OrderDateTime .
    ?Order MO:hasOrderDescription ?Description .
    ?Order MO:hasMeasure ?Measure .
    ?Measure MO:hasMeasureDateTime ?MeasureDateTime .
    ?Measure MO:hasMeasureText ?MeasureText .
}
ORDER BY ?OrderDateTime ?MeasureDateTime
```

Listing 24: SPARQL query for competency question 6

#### 5.4.7 What kind of orders have the longest lead times?

This query retrieves all necessary information to calculate what kind of orders have the longest lead times, such as the datetime when the malfunction was notified and the datetime when the measure was taken. Subtracting the two gives the lead time. The logic that still needs to be added to the business logic layer is selecting the last measure for each order, calculating the average lead time per combination of damage picture and cause, and finally sorting the results by the 'lead time' duration. The final result will show what combination of damage picture and cause has the longest average lead time.

```
PREFIX MO: <https://heijmans.org/maintenanceontology#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

SELECT ?Order ?DamagePicture ?Cause ?NotificationDateTime ?Measure ?MeasureDateTime ?LeadTime
WHERE {
    ?Order MO:hasDamagePicture ?DamagePicture.
    ?Order MO:hasCause ?Cause.
    ?Order MO:hasNotificationDateTime ?NotificationDateTime.
    ?Order MO:hasMeasure ?Measure .
    ?Measure MO:hasMeasureDateTime ?MeasureDateTime .
    BIND((xsd:dateTime(replace(?MeasureDateTime, " ", "T")) -
xsd:dateTime(replace(?NotificationDateTime, " ", "T"))) AS ?LeadTime)
}
```

Listing 25: SPARQL query for competency question 7

#### 5.4.8 What is the prognosed lead time for this Notification?

This query retrieves all necessary information to prognose the lead time for a specific problem, such as the datetime when the malfunction was notified and the datetime when the measure was taken. Subtracting the two gives the lead time. The logic that still needs to be added to the business logic layer is selecting the last measure for each order and calculating the average lead time. For all orders.

```
PREFIX MO: <https://heijmans.org/maintenanceontology#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX props: <https://w3id.org/props#>

SELECT ?Order ?NotificationDateTime ?Measure ?MeasureDateTime ?LeadTime
WHERE {
    ?Order MO:hasDamagePicture MO:Bedieningsprobleem .
    ?Order MO:relatedTo ?Asset .
    ?Asset MO:hasbotElementOrigin ?Element .
    ?Element props:Description "Ruimtebedien apparaat" .
    ?Order MO:hasNotificationDateTime ?NotificationDateTime.
    ?Order MO:hasMeasure ?Measure .
    ?Measure MO:hasMeasureDateTime ?MeasureDateTime .
    BIND((xsd:dateTime(replace(?MeasureDateTime, " ", "T")) -
xsd:dateTime(replace(?NotificationDateTime, " ", "T"))) AS ?LeadTime)
}
ORDER BY ?Order ?MeasureDateTime
```

Listing 26: SPARQL query for competency question 8

#### 5.4.9 What is the most common Cause for a specific problem?

```
PREFIX MO: <https://heijmans.org/maintenanceontology#>
PREFIX props: <https://w3id.org/props#>

SELECT ?Cause (COUNT(?Cause) AS ?count)
WHERE {
    ?Order MO:hasDamagePicture MO:Communicatie_audio_video-probleem .
    ?Order MO:hasCause ?Cause .
    ?Order MO:relatedTo ?Asset .
    ?Asset MO:hasbotElementOrigin ?Element .
    ?Element props:Description "AV, Spottercamer" .
}
GROUP BY ?Cause
ORDER BY DESC(?count)
```

Listing 27: SPARQL query for competency question 9

#### 5.4.10 Select all rooms where a specified malfunction occurred in a specific timeframe.

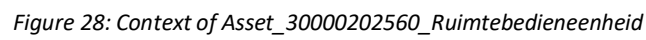
```
PREFIX MO: <https://heijmans.org/maintenanceontology#>
PREFIX props: <https://w3id.org/props#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX EMA: <https://heijmans.org/EMA#>

SELECT ?Room ?GUID
WHERE {
    ?Order MO:hasNotificationDateTime ?DateTime .
    ?Order MO:hasDiscipline MO:Elektrotechnisch .
    ?Order MO:hasCause MO:Gebruik_Bediening .
    BIND(xsd:dateTime(replace(?DateTime, " ", "T"))AS ?DateTime) .
    ?Order MO:takesPlaceIn ?Room .
    ?Room MO:hasbotSpaceOrigin ?Space .
    ?Space props:hasCompressedGuid ?GUID .
    FILTER(MONTH(xsd:dateTime(?DateTime)) = 7 && YEAR(xsd:dateTime(?DateTime)) = 2022) .
}
```

Listing 28: SPARQL query for competency question 10

#### 5.4.11 Visualize the context of data around a specific asset, room, or problem

There is no specific SPARQL query for this competency question, but the Graphs overview in GraphDB is perfect for this question. Figure 28 and Figure 29 show examples of graphs with the context of a room and the context of electrotechnics orders that have the cause “Degradation”.



## 5.5 LBDviz

This section presents the new and modified features of LBDviz and how they address requirements/competency questions 1, 2, and 5. These features provide easy and quick access to information, even for users who lack knowledge of the underlying data or SPARQL usage. According to the methodology, LBDviz takes a TTL file and one or more IFC models as input. However, when attempting to upload the EMA IFC files, the application crashed due to their large size of 3.42 GB, which was too much for the IFC.js web IFC viewer to handle. Although uploading individual files was feasible, the application would still crash when attempting to load a second or third model in the viewer.

To overcome this issue, a solution was proposed to export new IFC models for only a small portion of the building. Specifically, the XXL conference and multifunctional room were selected since they had experienced numerous malfunctions. After analyzing the orders that have a relationship to an asset, it was determined that these assets were related to the architectural, electrical, and interior model. For these Revit models, a section box was created around the two rooms and the models were exported to IFC, using the “only export elements visible in view” option. Subsequently, the IFC models were converted to RDF using the IFC to LBD convertor and integrated with the maintenance data. The conversion script executed successfully and was completed in 4 minutes and 56 seconds. The resulting integrated data file (lbdviz2.ttl) had a size of 9,163 kb. Table 13 lists the file sizes before and after conversion as well as the number of triples. In Figure 30, three IFC files are visualized.

Table 13: File Sizes and Number of Triples for BIM Models of XXL and Multifunctional Room

Model Discipline	Size IFC File	Size TTL File	Nr. Of Triples
Architectural	2,149 kb	3,975 kb	93,795
Electrical	3,050 kb	838 kb	21,678
Interior	3,979 kb	473 kb	9,989
Total	9,178 kb	5,286 kb	125,462

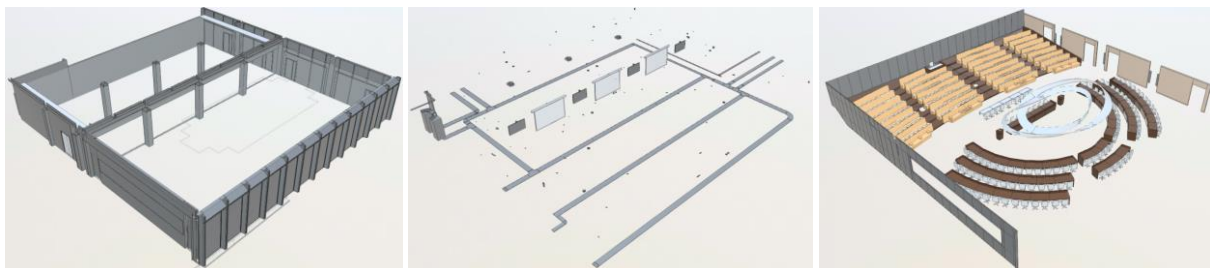


Figure 30: 3D Visualization of the Architectural, Electrical, and Interior IFC Models for LBDviz

After loading the IFC files into LBDviz and connecting the TTL file, the functionalities created in Section 4.5.3 were tested. The three functionalities are visualized and explained below:

### 5.5.1 Select all historic orders and measures related to a specified asset

After selecting the projector in the 3D viewer, the “Asset History” button was selected. In the background the function is called, and the query results are displayed in the Results tab. The underlying query is also tested in GraphDB, which gave the same results. A screenshot of the LBD web app is visualized in Figure 31.



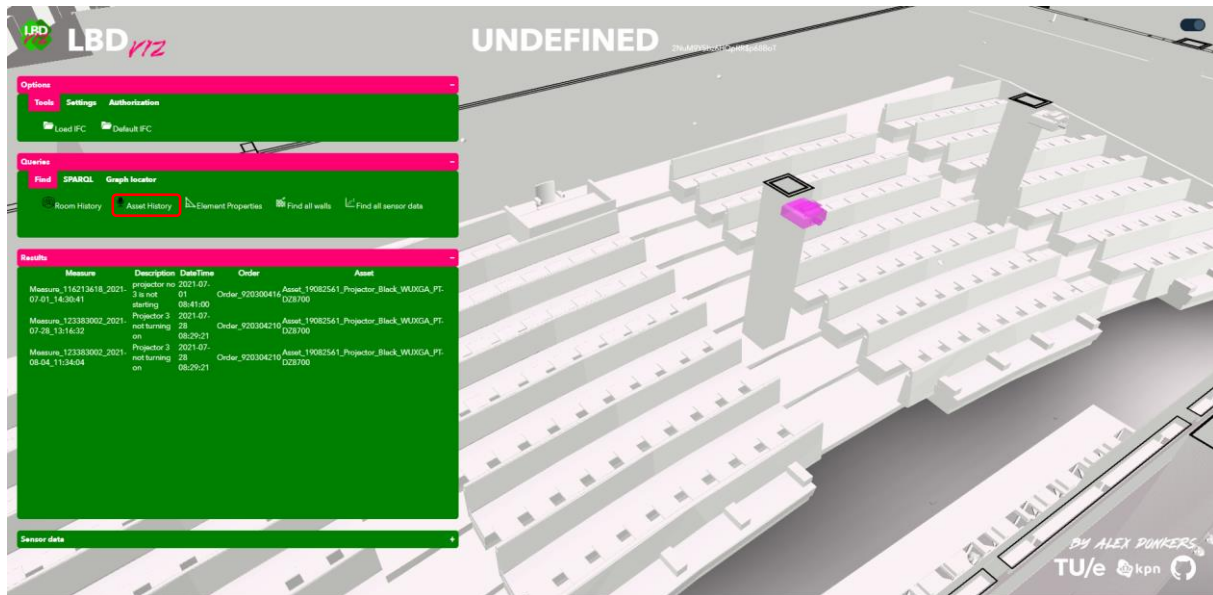


Figure 31: Asset History Function In LBDviz

### 5.5.2 Select all historic orders and measures that took place in a specified room

After selecting the multifunctional room in the 3D viewer, the “Room History” button was selected. In the background the function is called, and the query results are displayed in the Results tab. The underlying query is also tested in GraphDB, which gave the same results. A screenshot of the LBD web app is visualized in Figure 32.

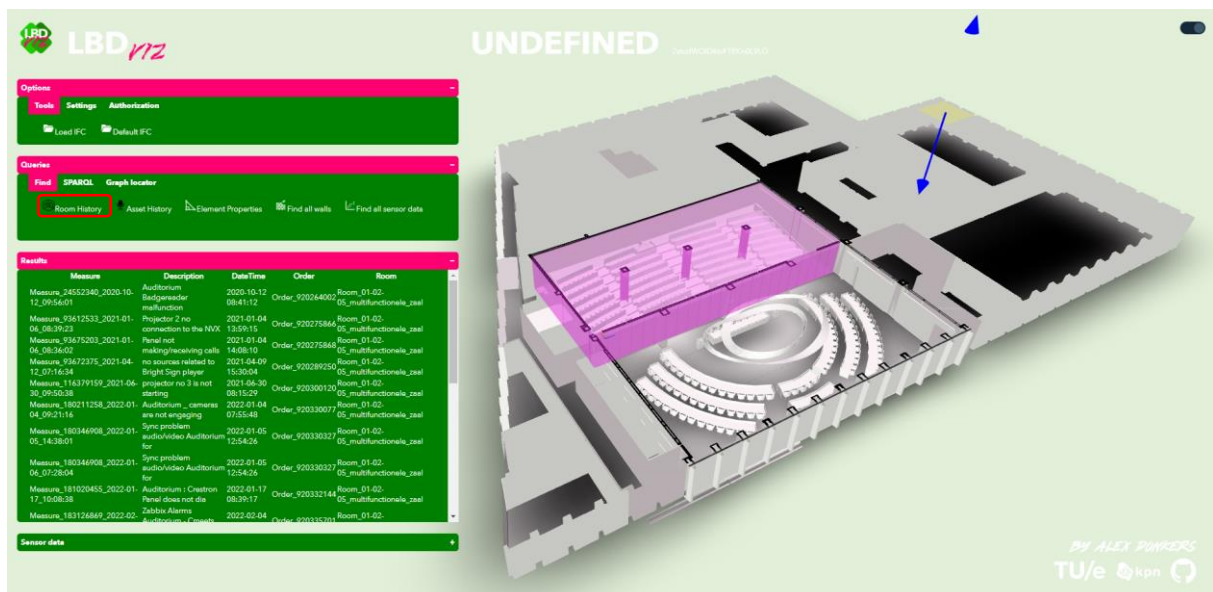


Figure 32: Room History Function in LBDviz

### 5.5.3 Select all properties from IFC for the asset or room related to order

After selecting the projector in the 3D viewer, the “Element Properties” button was selected. In the background the function is called, and the query results are displayed in the Results tab. The underlying query is also tested in GraphDB, which gave the same results. A screenshot of the LBD web app is visualized in Figure 33.

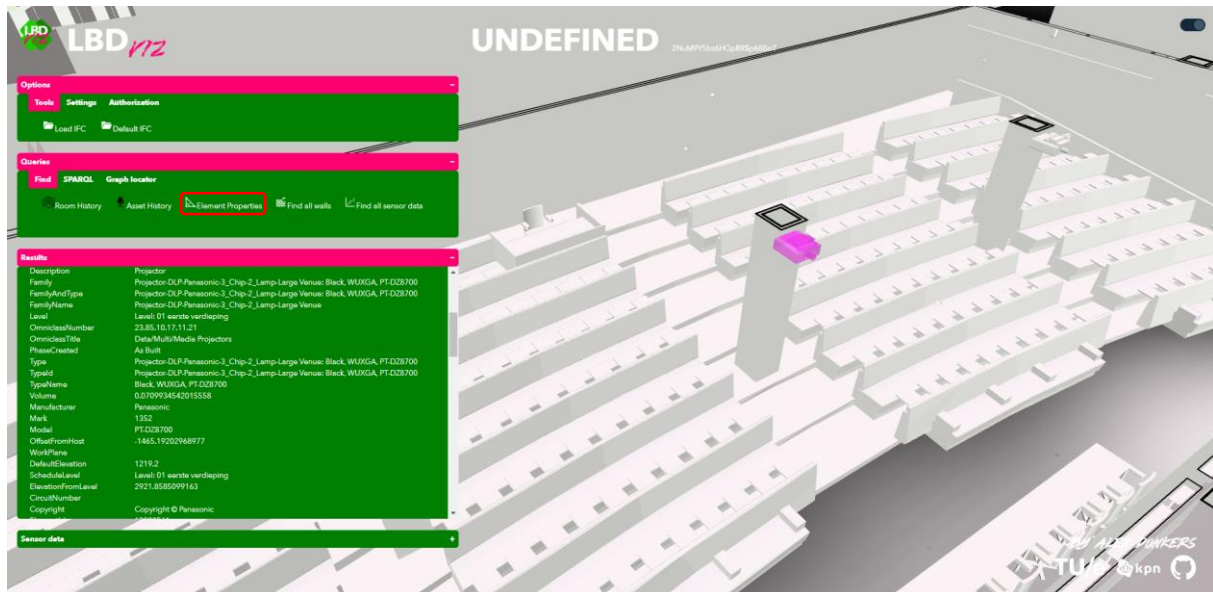


Figure 33: Element Properties Function in LBDviz

## 6 Conclusion

### 6.1 Summary

This research focussed on how the implementation of linked data and semantic digital twins could improve decision making in asset management. This is done by reviewing relevant literature in a literature study, composing a theoretical framework, identifying key problems in current information management systems, developing a method to automatically convert and integrate data in a graph database, and validating the system architecture and data pipeline in a case study. In order to fulfil the research objective a main research question is created. The main question comprises several corresponding sub-questions. This section will give answers these sub-questions and will therefore automatically give answer to the main research question, formulated in the introduction:

*“How can the implementation of linked data and semantic digital twins improve decision-making in asset management during the operation and maintenance phase?”*

What is the current state-of-the-art literature with regards to semantic digital twins and linked data in asset management?

According to the BIM maturity model, the industry is currently located at level 0, 1, or 2. BIM level 3 adoption is still very rare. The biggest obstacles to achieve BIM level 3 are cost, culture, expertise, technology and interoperability, lack of client demand, and legal issues. According to the ISO 19650 standard, the biggest part (usually 80%) of a building’s information is gathered and used in the operation and maintenance phase. Therefore, more emphasis should be on information on assets after projects are delivered to the client. In recent years, many standards have been created in order to standardize building information and information delivery. Examples are IFC, IDS, NL-SfB, ICDD, and bSDD. In Asset management there are different types of maintenance, starting from corrective maintenance, followed by planned maintenance, growing into predictive maintenance, and finally prescriptive maintenance where prediction is combined with recommendations for specific actions and maintenance strategies. In recent years, these principles have been discussed widely in digital twin literature, where the potential for data-driven maintenance management systems to increase efficiency, reduce costs, and improve the overall health and performance of buildings and equipment are discussed. Semantic web and linked data technologies are being implemented within the digital twin concept. The key difference between graph databases and relational databases is how they store and retrieve data. A graph database stores data as nodes and edges, which symbolize entities and relationships. It is more intuitive to use when analysing an entire context close to a single data point. Queries are executed by traversing the graph, making it easy to track complex relationships among assets. Also, graph databases are more adaptable and can handle highly dynamic data, which is quite common in large asset maintenance contracts.

Which specific insights are currently challenging and time-consuming to obtain using existing systems?

After interviews with multiple stakeholders in the field of asset management, it was found that in the current information system, it takes too much effort to find all relevant information related to a single asset or room. No comprehensive understanding of all the characteristics of the assets from all the systems, and thus not knowing an asset's complete maintenance history is mentioned as one of the biggest struggles in the information management during the operations phase, as it can lead to errors and inconsistencies in the data, making it difficult to make informed decisions.

*What are the key characteristics and sources of data commonly used in asset life cycle information management?*

Within maintenance projects, there usually is information on the building and its assets (Ideally in a building information model) and a maintenance management ERP system, where maintenance activities are defined as orders, including data about causes, damage pictures, and measures.

*How can the data originating from different systems be transformed into a format suitable for semantic web technologies?*

To transform the maintenance management data to RDF, an ontology was created first, that defines the structure and relationships of the data, to ensure that the data is properly structured and organized for use in a graph database. Subsequently a python script was created transforms the tabular data to triples in TTL format according to the vocabulary of the maintenance ontology. To transform the information from the BIM model to RDF, the bot and PROPS ontologies are used. Using a Python based IFCtoLBD convertor, multiple IFC aspect models are converted to TTL files.

*What strategies can be utilized to merge data from multiple systems effectively?*

In order to merge the data from the BIM model and the maintenance management system new relationships needed to be created between the Asset from the MMS and the Element from BIM and between the Room from the MMS and the Space from BIM. To create the two new relationships, based on element id's and room numbers, python was used in combination with the IfcOpenShell and RDFLib packages.

*What specific methods or tools can be utilized by different end users in the asset management team to generate insights from the data?*

The first tool, with Ontotext GraphDB as a back end, is for asset managers, maintenance engineers or planners that are familiar with the data and want to do advanced querying in the data to derive insights that are not possible in a relational database. LBDviz is a web application for visualizing RDF data in an IFC viewer, for stakeholders, such as planners and technicians that are less familiar with the data and need predefined functionalities in a user-friendly interface to derive quick insights essential for completing their work.

*To finally answer the main research question:*

The combination of linked data and semantic digital twins in a digital building logbook enables more intuitive searching for the entire context close to a single data point, easier tracking of complex relationships among assets, better ability to handle highly dynamic data, and suggestions on how to resolve new malfunctions, based on historical data as soon as they get notified. Asset managers and maintenance engineers have the ability to run complex queries that are not possible or really challenging in traditional relational databases. With a 3D user interface, also planners and technicians have access to quick insights on important data about assets and rooms. Applying linked data in a digital twin for asset management will increase efficiency, reduce costs, and improve the overall health and performance of buildings and equipment. The proposed system architecture and data pipeline were evaluated in a case study of the European medicines agency in Amsterdam which has a comprehensive Building Information Model, coupled with a maintenance management system including over three thousand orders collected within 2 years and 8 months. Results show that the conversion from IFC to RDF proceeded successfully and resulted in an output file containing 67,884 triples including all 3,079 orders from the SAP MMS. Also, the conversion from IFC to LBD and the integration between the MMS and IFC was also executed successfully, resulting in an output file

containing 1,165,759 triples. 149 of 183 (81%) of assets and all of 399 (100%) of rooms from the maintenance management system were related to elements and spaces from the BIM model, respectively. The system and data pipeline were also validated by answering the competency questions drawn up after the interviews in Section 3.2.2. All eleven competency questions are answered by queries from the triple store Ontotext GraphDB. Three competency questions are also answered using the web application LBDviz.

## 6.2 Discussion

ISO 19650 implies that the biggest part (usually 80%) of a building's information is gathered and used in the operation and maintenance phase and that the emphasis should therefore be on information on assets after projects are delivered to the client (ISO, 2019). The case study building in this research is only operational for 2 years and 8 months and looking at the data from the maintenance management system there are already 3,079 orders that include multiple data properties. Transforming this to triples in the compact maintenance ontology already resulted in 67,884 triples. If this pattern is extrapolated over the life span of 50 years of it will already result in around 1,250,000 triples. Let alone all the other sources of data, like documentation, external asset information, sensor data, invoices, warranties, and modifications in the building information model. It already gives a good indication that the largest amount of data is indeed generated and used in the operations phase.

Within this research, it is shown that linked data is increasingly being used within the AEC industry and many different built environment specific ontologies are being created (e.g., IfcOwl, BOT, BRICK, PRODUCT, PROPS, Haystack, RealEstateCore, SAREF, DogOnt, SOSA, CityGML). Besides, an expectation that not only Rijkswaterstaat, but also Rijksvastgoedbedrijf and municipalities will expect all projects to be delivered in RDF format in the future. For the AEC industry to comply to these demands in the future, much more research like this is needed. Furthermore, the BIM maturity model by Bew & Richards (2008) describes that organizations can only comply to level 3 if data from different sources is integrated and interoperable and if information is stored in the form of linked information objects instead of documents. Linked data is a technology that makes this possible.

Multiple research papers regarding digital twins integrate maintenance management systems with 3D models to improve decision making in asset management (Ma et al., 2020) (Lu et al., 2020) (Shim et al., 2019). These papers show that the systems decreased labor costs, made equipment maintenance decisions easier, and promoted objectivity, while also supporting the maintenance team and decision-makers to perform maintenance appropriately. The aim of this study was to find out how the implementation of linked data and semantic digital twins could improve decision making in asset management. In the literature review, the different maturity levels of asset maintenance were described (Ruparathna et al., 2018). The system architecture developed in this study uses historical corrective maintenance data to derive insights that can improve planned maintenance in the form of a more efficient maintenance planning and insights that already give stakeholders suggestions on how similar problems were resolved earlier as soon as a malfunction gets notified. This is already moving the towards predictive maintenance, which is the fourth of five maturity levels.

Overall, the combination of linked data and semantic digital twins in a digital building logbook enables more intuitive searching for the entire context close to a single data point, easier tracking of complex relationships among assets, better ability to handle highly dynamic data, and suggestions on how to resolve issues based on historical data. Asset managers and maintenance engineers have the ability to run complex queries that are not possible or really challenging in traditional relational databases. With a user interface, also planners and technicians have access to quick insights on important data about assets and rooms.



### 6.2.1 *Implications*

First of all, this research has contributed to the ongoing research about implementing semantic web technologies and digital twins in the built environment. This research has bridged the gap between data from the maintenance management system and data from BIM using linked data. In practice, these systems are often used separately and by different stakeholders. With the implementation of the developed system, it is possible to start analysing maintenance data immediately. Different organizations may use different maintenance management systems, but in general, the variables from the maintenance ontology are available in most of them. Therefore, maybe sometimes with small modifications to the code, the system is very generalizable.

The intended users of the developed system architecture and data pipeline are parties that are involved with maintaining assets during the operations phase like construction companies, asset management organizations, and maintenance companies. For these parties to use the system efficiently, it is implied that they will need to organize their data in a structured way. No matter what variables organizations want to analyze, it is essential to use the right vocabularies or ontologies, make data available, and to combine information from different systems to create as much value as possible.

The created maintenance ontology and data pipeline are really important for the industry to move towards predictive and prescriptive maintenance. By optimizing the maintenance schedule, the number of corrective maintenance orders is reduced. Of course, the number of malfunctions will never be reduced to zero, so for the malfunctions that do occur, the system can instantly provide all potentially valuable information from historical data about an asset, room, or a specific problem. This will reduce the time planners and technicians need to search for the required information, which will also directly reduce the indirect productivity of technicians. This way, employees are supported in making better and quicker decisions on how to resolve malfunctions based on historical data, instead of only experience or even gut feeling. Taking steps towards predictive and prescriptive maintenance can only be done when companies within the industry combine their data in a system similar to the one proposed in this thesis. The next step in being able to predict will be possible to add machine learning techniques in the business logic layer.

### 6.2.2 *Limitations*

This section aims to acknowledge the areas where the study may have fallen short in achieving its objectives or where its findings may not be entirely accurate. In this section, the limitations of this study and their potential impact on the study's outcomes are discussed. While this study was conducted with great care and attention to detail, it is important to acknowledge that no research is perfect, and there are limitations that may have influenced the results. By identifying and addressing these limitations, a comprehensive evaluation of the study's strengths and weaknesses is provided.

First of all there are several subjects relating data quality. The overall data quality in both the maintenance management system and the building information model caused limitations in the accuracy of the query results. In the competency questions the cause, damage picture, discipline, and measures were especially important variables. However, the orders in the maintenance management system, often lacked values for the corresponding variables, which makes results in less instances to be analysed. Also in the building information model, several empty values and inconsistencies existed, which makes the overall analyses less accurate. "Garbage in, garbage out" is a phrase often used to remind people that software and control systems can only provide useful information when they are fed with correct and consistent data. Besides, because the BIM models are subdivided into different discipline specific aspect models, the spaces are only present in the architectural model, which causes the absence of `bot:hasElement` relations between spaces and elements in all other models after conversion. In the proposed data pipeline, the room-asset relationships are defined in the maintenance management system where an order takes place in a room and is related to a specific

asset. If both relations exist, there is a relationship between the asset and room. The substantial downside of this, is that it is not defined what other assets are present in a specific room. This knowledge could give answer to other competency questions in the future. Another problem is the number of links between the MMS and instances from the BIM models in the case study of EMA. The first problem is the number of orders that are initially linked to a specific asset in SAP. At first, people who notify the malfunctions do not have information on what specific asset the malfunction is related to. Subsequently, planners and technicians that solve the malfunctions, often do not define the related asset after an order is resolved. This results in a number of asset-related orders of only 15%. The next problem is that not all assets are connected with a related BIM element after the integration process. More specifically, doors, outlets, and lighting fixtures are asset types that are clustered in the project of EMA. Because per storey or room, there are sometimes so many instances of one type, that it is too time consuming for the technicians to assign the specific corresponding asset instance. Therefore, the asset management team of EMA decided to cluster these asset types, which resulted in equipment numbers without a corresponding element id. The last consideration about data quality is that three thousand orders collected within 2 years and 8 months, would be a large enough sample to do meaningful analyses on the data. However, because the overall availability of data relating assets is relatively low, the database is not large enough for meaningful queries that ask for what type of assets are most vulnerable to specific problems. In the contrary, data quality about rooms and spaces is near perfect, so results on queries relating rooms are reliable.

The second limitation is that there could be alternatives ways to develop the data pipeline and system architecture. For instance, the python IFCtoLBD convertor in the current data pipeline is not provided with the PRODUCT or BEO ontologies. Therefore, when users want to query for specific building elements, they will need to use two lines of code, namely “?Element rdf:type bot:Element” and “?Element props:Description ?Description,” instead of just one line in the PRODUCT OR BEO ontology. Another consideration is that there could be alternatives for the MO:hasbotElementOrigin and MO:hasbotSpaceOrigin, like for instance the owl:sameAs relationship, which is used to state that two URI references refer to the same individual. Instead of accessing the BIM properties from the bot:Element through the hasbotElementOrigin relationship from the MO:Asset, it is then possible to query properties from the Asset node directly. This will allow for smaller SPARQL queries, which will make the system more user friendly. However, the owl:sameAs relationship also has several downsides, such as possible loss of granularity, transitivity issues, inference issues, and maintenance issues.

Finally, there were some limitations in the implementation of the LBDviz web application. First of all, the IFC.js IFC viewer lacked robustness when dealing with larger sizes IFC models. For instance, in the EMA case study, loading individual aspect models was possible, but when loading two or more models the application crashed. This has consequences for larger projects, such as the EMA, where one building storey already is the max capacity. Another limitation is the Comunica query engine, which was not able to run complex SPARQL queries and was relatively slow, compared to Ontotext GraphDB, when dealing with large graphs. The query engine sometimes gave no results when queries included optional triple patterns and query modifiers such as GROUP BY and ORDER BY.

### 6.2.3 Recommendations

This section will discuss recommendations based on the results of this research. A distinction is made between suggestions for future research that can build on anything this research was unable to address within the given time frame a graduation and recommendations related to practical application, which are aimed at the graduation company and all other construction companies that do similar maintenance projects.



## Suggestions for Future Research

**Building Element Taxonomy** - as mentioned in the limitations section, the PRODUCT or BEO ontologies which describe the taxonomy of building elements is not integrated in the python IFCtoLBD convertor that is used in the proposed system. In similar converters, these ontologies are already included, so future research could focus on integrating one of the two in the python IFCtoLBD convertor.

**owl:sameAs** - Future studies could explore the implications of using the owl:sameAs relationship as an alternative for the MO:hasbotOrigin relationships in the data pipeline of this research. As already mentioned it can have big advantages in simplifying SPARQL queries. On the other hand, in literature, this relationship is considered to be risky, because it has a number of potential downsides. Future studies could replace the MO:hasbotOrigin relationships with owl:sameAs and explore the differences.

**Bot:hasElement** - As mentioned in the limitations, the BIM models are subdivided into different discipline specific aspect models, so the spaces are only present in the architectural model, which causes the absence of bot:hasElement relations between spaces and elements in all other models after conversion. Future research should focus on a method to integrate the spaces in other models to create a complete picture what assets are located in each room. This will allow end users to do more reliable and comprehensive queries on the data.

**LBDviz** - Future research should focus on further developing the application's user interface based on a larger number of requirements drawn up by real practitioners. By co-creating the application with industry experts, LBDviz might reduce the barriers to using linked data in practice. Besides, when the application will be used in the future, future research should also focus on further developing the Comunica query engine and IFC.js, since they lack robustness when dealing with a large amount of data.

**Extracting data from Text** – data from the MMS includes the “NotificationText” and “MeasureText” variables. The notification text often contains information on a possible damage picture or relating asset. The measure text often includes data about what tools were used, or what settings were used to resolve the issue. Future research could focus on extracting this information from the texts to create new variables such as “tools used” which can help suggesting how to resolve specific malfunctions.

**NLP Implementation** - With the emergence of natural language processing (NLP) models such as GPT-3 (and 4) and BERT, future research could focus on implementing this technology in the user interface of digital twin applications. The impact of implementing this technology can be enormous, since it will allow effortless interaction with a database for all stakeholders in a project (from management to technicians). End users will not have to require any knowledge about the SPARQL query language or underlying data schema to interact with the data and get answers to questions that are relevant in their work.

## Practical Application

**Data Quality** - First of all, by far the most important recommendation is to start improving the data quality immediately. First SHACL should be used to check the dataset for inconsistent and missing data. In order to convince management to get access to budget and resources, the numbers should be presented concrete and specific. In this study a working system architecture has been created. Despite the fact that the data quality is not up to standards, it is already important to show the asset management team that the product works and that only better insights could be generate, when there is access to complete and consistent data. This will create a pull. Not only in the case study of EMA, but also in other projects.

The easiest step to improve on the data is to look back at the existing data. In the MMS, many orders are missing a related asset, but often the notification text contains information on the relating asset. Looking at the room and the information in the notification text, the relating asset often can be traced pretty easily. The same concept applies to the cause and damage picture. Sometimes these can be derived from the measure texts.

To improve the way data is provided to the system in the future, technicians and planners should have access to a new functionality in the MMS. The main reason for planners or technicians not to fill in the relating asset, is that the process often is time consuming. Instead of endlessly scrolling through a list of equipment numbers, the process should be very straightforward. The new functionality should therefore include the buildings taxonomy of spaces and 2D drawings, where the relating asset can be selected, so that it is filled in automatically. This will increase the data quality significantly.

Finally, it is recommended to mandate that all variables in the maintenance management system are filled in before a malfunction is officially resolved. Not knowing the cause initially is okay, but too often, there are empty values in the data. So, before a malfunction is officially resolved, all values must be filled in. The overall attitude should be: “The better the data, the more meaningful the insights of the analysis.”

**Clustering of Assets** - As mentioned in the limitations section, in the current MMS, several asset types are clustered to simplify the task of linking orders to assets. A way to integrate this concept in the data pipeline of this study is to integrate a mapping table between cluster id's and individual element id's. One cluster id could host multiple element id's, so for instance, when an asset is selected, the relating cluster id could be used in a SPARQL query, to extract the maintenance history from the data. However, as mentioned in the recommendation about data quality there is a feasible way to fully scrap the clustering of assets. The main goal of a digital twin is to do asset specific analysis, and when moving to a predictive maintenance approach, it is essential to know an individual assets maintenance history. Only in that way, meaningful analyses and predictions can be done to the data in the graph.

**Adding more Datasets** - With a solid foundation created in this research, organizations should consider integrating this data with as much as other available datasets, such as costs, maintenance intervals, threshold times, manufacturer documentation, warranties, and other relevant data. By integrating all relevant data, it is possible to gain a better understanding of maintenance needs, which can help organizations optimize their maintenance budget and reduce downtime. Other datasets can be transformed and integrated using a similar method described in this research. Only minor changes in the code will be necessary.

**Software Development** - A crucial step in implementing linked data technologies is to make the data accessible to all stakeholders. Therefore, investing in software development is critical. Graph data is often an effective way to represent complex relationships, but it can be difficult for non-experts to interpret without knowledge of the underlying data. By investing in software development, organizations can create user-friendly tools that allow stakeholders to interact with graph data in a meaningful way.

## 7 References

- Alonso, R., Borrás, M., Koppelaar, R. H. E. M., Lodigiani, A., Loscos, E., & Yöntem, E. (2019). SPHERE: BIM Digital Twin Platform. Sustainable Places 2019. MDPI. Retrieved from <http://dx.doi.org/10.3390/proceedings2019020009>
- Alizadehsalehi, S., & Yitmen, I. (2021). Digital twin-based progress monitoring management model through reality capture to extended reality technologies (DRX) | Emerald Insight. <https://doi.org/10.1108/SASBE-01-2021-0016>
- Anderl, R., Haag, S., Schützer, K., & Zancul, E. (2018). Digital twin technology – An approach for Industrie 4.0 vertical and horizontal lifecycle integration. *It - Information Technology*, 60(3), 125–132. <https://doi.org/10.1515/itit-2017-0038>
- Antonino, M., Nicola, M., Claudio, D. M., Luciano, B., & Fulvio, R. C. (2019). Office building occupancy monitoring through image recognition sensors. *International Journal of Safety and Security Engineering*, 9(3), 371–380. <https://doi.org/10.2495/safe-v9-n4-371-380>
- Ashton, K. (2009). That 'Internet of Things' Thing. *RFID Journal*, 22(7), 97-114.
- Associated General Contractors of America. (2005). *The Contractor's Guide to BIM*, 1st ed. AGC Research Foundation, Las Vegas, NV.
- Austin, M., Delgoshaei, P., Coelho, M., & Heidarinejad, M. (2020). Architecting Smart City Digital Twins: Combined Semantic Model and Machine Learning Approach. *Journal of Management in Engineering*, 36(4). [https://doi.org/10.1061/\(asce\)me.1943-5479.0000774](https://doi.org/10.1061/(asce)me.1943-5479.0000774)
- Ayinla, K. O., & Adamu, Z. (2017). Bridging the digital divide gap in BIM technology adoption. <https://doi.org/10.1108/ECAM-05-2017-0091>
- Bhargav, D., Buda, A., Nurminen, A., & Främling, K. (2018). A framework for integrating BIM and IoT through open standards. *Automation in Construction*, 95(August), 35–45. <https://doi.org/10.1016/j.autcon.2018.07.022>
- Becerik-Gerber, B., Jazizadeh, F., Li, N., & Calis, G. (2012). Application Areas and Data Requirements for BIM-Enabled Facilities Management. *Journal of Construction Engineering and Management*, 431-442.
- Beckett, D., & Berners-Lee, T. (2011). Turtle - Terse RDF Triple Language. [w3.org. https://www.w3.org/TeamSubmission/turtle/](https://www.w3.org/TeamSubmission/turtle/)
- Beetz, J., van Leeuwen, J., & de Vries, B. (2009). IfcOwl: A case of transforming EXPRESS schemas into ontologies. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 89-101.
- Bertelsen, Sven & Sven,. (2003). Construction as a Complex System. Proceedings for the 11th Annual Conference of the International Group for Lean Construction.
- Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The Semantic Web. *Scientific American*, 284(5), 34–43. <http://www.jstor.org/stable/26059207>
- Bew, M., & Richards, M. (2008). BIM Maturity Model. In Paper presented at the Construct IT Autumn 2008 Members' Meeting.
- Bonduel, M., Oraskari, J., Pauwels, P., Vergauwen, M., & Klein, R. (2018). The IFC to linked building data converter - current status. LDAC.

- Brandyberry, M., Penmetsa, R. & Tuegel, E. (2012). Challenges with Structural Life Forecasting Using Realistic Mission Profiles. 53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference&lt;BR&gt;20th AIAA/ASME/AHS Adaptive Structures Conference&lt;BR&gt;14th AIAA. <https://doi.org/10.2514/6.2012-1813>
- Brilakis, I.; Fischer, H.; Fellow, S. Built Environment Digital Twinning; Technical University of Munich: Munich, Germany, 2019.
- Brunenberg, P. (2022, January 4). At Its Core: How Is a Graph Database Different from a Relational One? Medium. <https://towardsdatascience.com/at-its-core-hows-a-graph-database-different-from-a-relational-8297ca99cb8f>
- buildingSMART International. (2019, June 20). Industry Foundation Classes (IFC). Retrieved 12 October 2022, from <https://www.buildingsmart.org/standards/bsi-standards/industry-foundation-classes/>
- buildingSMART International. (2022b, August 15). buildingSMART Data Dictionary. Retrieved December 12, 2022, from <https://www.buildingsmart.org/users/services/buildingsmart-data-dictionary/>
- buildingSMART International. (2023). Information Delivery Specification IDS - buildingSMART Technical. buildingSMART Technical. Retrieved March 7, 2023, from <https://technical.buildingsmart.org/projects/information-delivery-specification-ids/>
- Chen, W., Chen, K., Cheng, J. C., Wang, Q., & Gan, V. J. (2018). BIM-based framework for automatic scheduling of facility maintenance work orders. *Automation in Construction*, 91, 15–30. <https://doi.org/10.1016/j.autcon.2018.03.007>
- Cheng, J. C., Chen, W., Chen, K., & Wang, Q. (2020). Data-driven predictive maintenance planning framework for MEP components based on BIM and IoT using machine learning algorithms. *Automation in Construction*, 112, 103087. <https://doi.org/10.1016/j.autcon.2020.103087>
- Cheung, W. F., & Lin, Y. C. (2016). Development of BIM-based Safety Monitoring System Integrated with WSN Technology."
- Ciribini, A. L. C., Pasini, D., Tagliabue, L. C., Manfren, M., Daniotti, B., Rinaldi, S., & De Angelis, E. (2017). Tracking Users' Behaviors through Real-time Information in BIMs: Workflow for Interconnection in the Brescia Smart Campus Demonstrator. *Procedia Engineering*, 180, 1484–1494. <https://doi.org/10.1016/j.proeng.2017.04.311>
- Costa, G., & Madrazo, L. (2015). Connecting building component catalogues with BIM models using semantic technologies: An application for precast concrete components. *Automation in Construction*. 57. 239–248. [10.1016/j.autcon.2015.05.007](https://doi.org/10.1016/j.autcon.2015.05.007).
- Curry, E., O'Donnell, J., Corry, E., Hasan, S., Keane, M., & O'Riain, S. (2013). Linking building data in the cloud: Integrating cross-domain building data using linked data. *Advanced Engineering Informatics*, 206-219.
- Deng, M., Menassa, C. C., & Kamat, V. R. (2021). From BIM to digital twins: a systematic review of the evolution of intelligent building representations in the AEC-FM industry. *Journal of Information Technology in Construction*, 26, 58–83. <https://doi.org/10.36680/j.itcon.2021.005>

- Dixit, M. K., Venkatraj, V., Ostadalimakhmalbaf, M., Pariafsai, F., & Lavy, S. (2019). Integration of facility management and building information modeling (BIM). *Facilities*, 37(7/8), 455–483. <https://doi.org/10.1108/f-03-2018-0043>
- Donkers, A. (2023). A Visual Support Tool for Decision-Making over Federated Building Information. [Unpublished Manuscript]
- Douglas, C (2017) EMIT optimisation: Getting more out of existing equipment for less. *RISK World* (31), 4-5. Available from <https://risktec.tuv.com/wp-content/uploads/2018/10/riskworld-newsletter-med-resolution-single-pages.pdf>
- Eastman, C. (1979), The representation of design problems and maintenance of their structure, Technical Report, Carnegie Mellon University.
- Eastman, C. M., Eastman, C., Teicholz, P., Sacks, R., & Liston, K. (2011). BIM handbook: A guide to building information modeling for owners, managers, designers, engineers and contractors. John Wiley & Sons.
- EFCA, 2019. BIM and ISO 19650 from a project management perspective. European Federation of Engineering Consultancy Associations.
- Escandón, R., Ascione, F., Bianco, N., Mauro, G. M., Suárez, R., & Sendra, J. J. (2019). Thermal comfort prediction in a building category: Artificial neural network generation from calibrated models for a social housing stock in southern Europe. *Applied Thermal Engineering*, 150, 492–505. <https://doi.org/10.1016/j.applthermaleng.2019.01.013>
- Executive Board TU/e. (2019). TU/e Code of Scientific Conduct.
- Gao, X., & Pishdad-Bozorgi, P. (2019). A framework of developing machine learning models for facility life-cycle cost analysis. *Building Research & Information*, 48(5), 501–525. <https://doi.org/10.1080/09613218.2019.1691488>
- Gemeente Amsterdam. (2022). OTL Gemeente Amsterdam. <https://amsterdam.otl-viewer.com/>
- Glaessgen, E., & Stargel, D. (2012). The Digital Twin Paradigm for Future NASA and U.S. Air Force Vehicles. 53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference<BR>20th AIAA/ASME/AHS Adaptive Structures Conference<BR>14th AIAA. <https://doi.org/10.2514/6.2012-1818>
- Grieves, M., & Vickers, J. (2016). Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems. *Transdisciplinary Perspectives on Complex Systems: New Findings and Approaches*, August 2017, 85–113. <https://doi.org/10.1007/978-3-319-38756-7>
- Grussing, M. N. (2014). Life Cycle Asset Management Methodologies for Buildings. *Journal of Infrastructure Systems*, 20(1). [https://doi.org/10.1061/\(asce\)is.1943-555x.0000157](https://doi.org/10.1061/(asce)is.1943-555x.0000157)
- Guerra de Oliveira, S., Tibaut, A., & Dell’Acqua, G. (2020). Airport pavement management systems: An open BIM approach. *Lecture Notes in Civil Engineering*, 48(January), 450–459. [https://doi.org/10.1007/978-3-030-29779-4\\_44](https://doi.org/10.1007/978-3-030-29779-4_44)
- Hosamo, H. H., Svennevig, P. R., Svidt, K., Han, D., & Nielsen, H. K. (2022). A Digital Twin predictive maintenance framework of air handling units based on automatic fault detection and diagnostics. *Energy and Buildings*, 261, 111988. <https://doi.org/10.1016/j.enbuild.2022.111988>

- Hoang, N.V., & Törmä, S. (2015). Implementation and Experiments with an IFC-to-Linked Data Converter.
- Hoang, G. v., Vu, D. K. T., Le, N. H., & Nguyen, T. P. (2020). Benefits and challenges of BIM implementation for facility management in operation and maintenance face of buildings in Vietnam. IOP Conference Series: Materials Science and Engineering, 869(2). <https://doi.org/10.1088/1757-899X/869/2/022032>
- Huang, L., Xu, Y., Liao, X., Qin, L., & Qiu, J. (2018). Construction of intelligent fire management system based on BIM technology. Proceedings of the 2018 7th International Conference on Energy, Environment and Sustainable Development (ICEESD 2018). <https://doi.org/10.2991/iceesd-18.2018.130>
- I., & Teicholz, P. (2013). BIM for Facility Managers. Wiley.
- Institute of Asset Management (2014). Asset Management – an anatomy, Version 2, July 2014.
- ISO. (2004). ISO 10303-11:2004 Industrial automation systems and integration — Product data representation and exchange — Part 11: Description methods: The EXPRESS language reference manual. ISO.org. <https://www.iso.org/standard/38047.html>
- ISO. (2014). ISO 10303-242:2014 Industrial automation systems and integration — Product data representation and exchange — Part 242: Application protocol: Managed model-based 3D engineering. ISO.org. <https://www.iso.org/standard/57620.html>
- ISO. (2018). ISO 16739-1:2018 Industry Foundation Classes (IFC) for data sharing in the construction and facility management industries — Part 1: Data schema. ISO.org. <https://www.iso.org/standard/70303.html>
- ISO. (2019). ISO 19650-1:2018, Organization and digitization of information about buildings and civil engineering works, including building information modelling (BIM) — Information management using building information modelling — Part 1: Concepts and principles. ISO.
- ISO. (2019). ISO 19650-2:2018, Organization and digitization of information about buildings and civil engineering works, including building information modelling (BIM) — Information management using building information modelling — Part 2: Delivery phase of the assets. ISO.
- ISO. (2020). ISO 21597-1:2020 Information container for linked document delivery — Exchange specification — Part 1: Container. ISO.org. <https://www.iso.org/standard/74389.html>
- Jing, Y., Chen, C., Tang, L., Xiong, H., & Wang, Y. X. (2019). Development of BIM-Sensor Integrated Platform for MEP Piping Maintenance. 2019, 55– 60. <https://doi.org/10.1680/icsic.64669.055>
- Kassem, M., Rogage, K., Huntingdon, J., Durojaye, G., Arena, N., Kelly, G., ... & ClarNe, T. (2019). Measuring and improving the productivity of construction's site equipment Fleet: An integrated IoT and BIM system. In 36th CIB W78 2019 Conference Advances in ICT Design, Construction & Management in Architecture, Engineering, Construction and Operations (AECO) (pp. 901-911).
- Khajavi, S. H., Motlagh, N. H., Jaribion, A., Werner, L. C., & Holmstrom, J. (2019). Digital Twin: Vision, Benefits, Boundaries, and Creation for Buildings. IEEE Access, 7, 147406–147419. <https://doi.org/10.1109/access.2019.2946515>



- La Russa, F. M., & Santagati, C. (2020). Historical Sentient – Building Information Model: a Digital Twin for the Management of Museum Collections in Historical Architectures. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLIII-B4-2, 755–762. <https://doi.org/10.5194/isprs-archives-xliii-b4-2020-755-2020>
- Li, X., Wu, P., Shen, G. Q., Wang, X., & Teng, Y. (2017). Mapping the knowledge domains of Building Information Modeling (BIM): A bibliometric approach. *Automation in Construction*, 84, 195–206. <https://doi.org/10.1016/j.autcon.2017.09.011>
- Li, C. Z., Xue, F., Li, X., Hong, J., & Shen, G. Q. (2018). An Internet of Things-enabled BIM platform for on-site assembly services in prefabricated construction. *Automation in Construction*, 89, 146–161. <https://doi.org/10.1016/j.autcon.2018.01.001>
- Liu, C.-C., W.-L. Kuo, R.-S. Shiu and I.-C. Wu (2014). "Estimating and Visualizing Thermal Comfort Level via a Predicted Mean Vote in a BIM System."
- Love, P. E., Matthews, J. & Lockley, S. (2015) BIM for built asset management. *Built Environment Project and Asset Management*, 5(3).
- Lu, Q., Chen, L., Li, S., & Pitt, M. (2020). Semi-automatic geometric digital twinning for existing buildings based on images and CAD drawings. *Automation in Construction*, 115, 103183. <https://doi.org/10.1016/j.autcon.2020.103183>
- Lu, Q., Parlikad, A. K., Woodall, P., Don Ranasinghe, G., Xie, X., Liang, Z., Konstantinou, E., Heaton, J., & Schooling, J. (2020). Developing a Digital Twin at Building and City Levels: Case Study of West Cambridge Campus. *Journal of Management in Engineering*, 36(3), 1–19. [https://doi.org/10.1061/\(ASCE\)ME.1943-5479.0000763](https://doi.org/10.1061/(ASCE)ME.1943-5479.0000763)
- Lu, Q., Xie, X., Parlikad, A. K., & Schooling, J. M. (2020). Digital twin-enabled anomaly detection for built asset monitoring in operation and maintenance. *Automation in Construction*, 118, 103277. <https://doi.org/10.1016/j.autcon.2020.103277>
- Kritzinger, W., Karner, M., Traar, G., Henjes, J., & Sihn, W. (2018). Digital Twin in manufacturing: A categorical literature review and classification. *IFAC-PapersOnLine*, 51(11), 1016–1022. <https://doi.org/10.1016/j.ifacol.2018.08.474>
- Ma, Z., Ren, Y., Xiang, X., & Turk, Z. (2020). Data-driven decision-making for equipment maintenance. *Automation in Construction*, 112(July 2019), 103103. <https://doi.org/10.1016/j.autcon.2020.103103>
- Malcolm, A., Werbrouck, J., & Pauwels, P. (2021). LBD Server: Visualising Building Graphs in Web-Based Environments Using Semantic Graphs and GTF-Models. *Formal Methods in Architecture*, 287–293. [https://doi.org/10.1007/978-3-030-57509-0\\_26](https://doi.org/10.1007/978-3-030-57509-0_26)
- Macchi, M., Roda, I., Negri, E., & Fumagalli, L. (2018). Exploring the role of Digital Twin for Asset Lifecycle Management. *IFAC-PapersOnLine*, 51(11), 790–795. <https://doi.org/10.1016/j.ifacol.2018.08.415>
- Madni, A., Madni, C., & Lucero, S. (2019). Leveraging Digital Twin Technology in Model-Based Systems Engineering. *Systems*, 7(1), 7. MDPI AG. Retrieved from <http://dx.doi.org/10.3390/systems7010007>



- Mannino, A., Moretti, N., Dejacco, C., Baresi, L., & Re Cecconi, F. (2019). Office building occupancy monitoring through image recognition sensors. *International Journal of Safety and Security Engineering*, 9(4), 371–380. <https://doi.org/10.2495/SAFE-V9-N4-371-380>
- Moretti, N., Blanco Cadena, J. D., Mannino, A., Poli, T., & Re Cecconi, F. (2020). Maintenance service optimization in smart buildings through ultrasonic sensors network. *Intelligent Buildings International*, 0(0), 1–13. <https://doi.org/10.1080/17508975.2020.1765723>
- Mussomeli, A., Parrot, A., Umberhauer, B., & Warshaw, L. (2020). Digital twins: Bridging the physical and digital. *Tech Trends* 2020, 59–77. Retrieved from <https://www2.deloitte.com/us/en/insights/focus/tech-trends/2020/digital-twin-applications-bridging-the-physical-and-digital.html>
- Natephraa, W. and A. Motamedib (2019). Live data visualization of IoT sensors using Augmented Reality (AR) and BIM. ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction, IAARC Publications.
- Pauwels, P. (2014). Supporting Decision-Making in the Building Life-Cycle Using Linked Building Data. *Buildings*, 4(3), 549–579. <https://doi.org/10.3390/buildings4030549>
- Pauwels, P., & Terkaj, W. (2016). EXPRESS to OWL for construction industry: Towards a recommendable and usable IfcOwl ontology. *Automation in Construction*, 63, 100–133. <https://doi.org/10.1016/j.autcon.2015.12.003>
- Pauwels, P., Zhang, S., & Lee, Y. (2017) Semantic web technologies in AEC industry: A literature overview. *Automation in Construction*, 73, 145-165. <http://dx.doi.org/10.1016/j.autcon.2016.10.003>
- Pour Rahimian, F., Seyedzadeh, S., Oliver, S., Rodriguez, S., & Dawood, N. (2020). On-demand monitoring of construction projects through a game-like hybrid application of BIM and machine learning. *Automation in Construction*, 110, 103012. <https://doi.org/10.1016/j.autcon.2019.103012>
- ProRail. (2020, June 5). OTL Spoor. <https://otl.prorail.nl/otl/>
- Qi, Q., Tao, F., Zuo, Y. & Zhao, D. (2018). Digital Twin Service towards Smart Manufacturing. *Procedia CIRP*, 72, 237–242. <https://doi.org/10.1016/j.procir.2018.03.103>
- Quinn, C., Shabestari, A. Z., Mistic, T., Gilani, S., Litoiu, M., & McArthur, J. J. (2020). Building automation system - BIM integration using a linked data structure. *Automation in Construction*, 118(June), 103257. <https://doi.org/10.1016/j.autcon.2020.103257>
- Qureshi, A. H., W. S. Alaloul, B. Manzoor, M. A. Musarat, S. Saad and S. Ammad (2020). Implications of Machine Learning Integrated Technologies for Construction Progress Detection Under Industry 4.0 (IR 4.0). 2020 Second International Sustainability and Resilience Conference: Technology and Innovation in Building Designs(51154).
- Quirk, V. (2012, December 7). A Brief History of BIM. <https://www.archdaily.com/302490/a-brief-history-of-bim>
- Rasmussen, M. H., Pauwels, P., Hviid, C. A., & Karlshoj, J. (2017). Proposing a central AEC ontology that allows for domain specific extensions. In *Proceedings of the Joint Conference on Computing in Construction (JC3)* (Vol. 1, pp. 237–244). Heraklion, Crete. <https://doi.org/10.24928/JC3-2017/0153>.

- Rijksvastgoedbedrijf. (2013, January). RVB BIM Norm v1.1. Rijksvastgoedbedrijf.nl. <https://www.rijksvastgoedbedrijf.nl/documenten/richtlijn/2014/06/20/rvb-bim-norm-v1.1>
- Rijkswaterstaat. (2021). Rijkswaterstaat ARIBIM-OTL. <https://github.com/RWS-NL/airbim-otl>
- Ruparathna, R, Hewage, K and Sadiq, R (2018) Multi-period maintenance planning for public buildings: A risk-based approach for climate conscious operation. *Journal of Cleaner Production*, 170(1), 1338-1353.
- Senthilvel, M., Oraskari, J., & Beetz, J. (2020, July). Common Data Environments for the Information Container for linked Document Delivery. In *Proceedings of the 8th Linked Data in Architecture and Construction Workshop-LDAC* (pp. 132-145).
- Shim, C. S., Dang, N. S., Lon, S., & Jeon, C. H. (2019). Development of a bridge maintenance system for prestressed concrete bridges using 3D digital twin model. *Structure and Infrastructure Engineering*, 15(10), 1319–1332. <https://doi.org/10.1080/15732479.2019.1620789>
- Srivastava, M. , & Whitehouse, K. (2016). Brick: Towards a Unified Metadata Schema For Buildings. 41–50. <https://doi.org/10.1145/2993422.2993577>
- Tagliabue, L. C., Cecconi, F. R., Maltese, S., Rinaldi, S., Ciribini, A. L. C., & Flammini, A. (2021). Leveraging Digital Twin for Sustainability Assessment of an Educational Building. *Sustainability*, 13(2), 480. <https://doi.org/10.3390/su13020480>
- Tao, F., Cheng, J., Qi, Q., Zhang, M., Zhang, H. & Sui, F. (2017). Digital twin-driven product design, manufacturing and service with big data. In *Int J Adv Manuf Technol*, 10 (4), p. 2233.
- van Nederveen, G., & Tolman, F. (1992). Modelling multiple views on buildings. *Automation in Construction*, 1(3), 215–224. [https://doi.org/10.1016/0926-5805\(92\)90014-b](https://doi.org/10.1016/0926-5805(92)90014-b)
- Volk, R., Stengel, J., & Schultmann, F. (2014). Building Information Modeling (BIM) for existing buildings — Literature review and future needs. *Automation in Construction*, 38, 109-127.
- W3C. (2013). SPARQL 1.1 Query Language. W3.org. <https://www.w3.org/TR/sparql11-query/>
- W3C. (2014). RDF Schema 1.1. w3.org. <https://www.w3.org/TR/rdf-schema/>
- W3C. (2017). Shapes Constraint Language (SHACL). W3.org. <https://www.w3.org/TR/shacl/>
- Wang, X., Wang, Y., Tao, F., & Liu, A. (2020). New Paradigm of Data-Driven Smart Customisation through Digital Twin. *Journal of Manufacturing Systems*, July, 1–11. <https://doi.org/10.1016/j.jmsy.2020.07.023>
- Wehbe, R., & Shahrour, I. (2019). Indoor hazards management using digital technology. *MATEC Web of Conferences*, 281, 01013. <https://doi.org/10.1051/mateconf/201928101013>
- Werbrouck, J., Pauwels, P., Bonduel, M., Beetz, J., & Bekers, W. (2020). Scan-to-graph: Semantic enrichment of existing building geometry. *Automation in Construction*, 119, 103286. <https://doi.org/10.1016/j.autcon.2020.103286>
- Xie, X., Lu, Q., Rodenas-Herraz, D., Parlikad, A. K., & Schooling, J. M. (2020). Visualised inspection system for monitoring environmental anomalies during daily operation and maintenance. *Engineering, Construction and Architectural Management*, 27(8), 1835–1852. <https://doi.org/10.1108/ECAM-11-2019-0640>

Zhuang C., Liu J., Xiong H., Ding X., Liu S. & Wang G. (2017). “Connotation, architecture and trends of product digital twin,” (in Chinese), Comput. Integr. Manuf. Syst., vol. 23, no. 4, pp. 753–768.

## **8 Appendices**

### **8.1 Maintenance Ontology**

Link to GitHub: [https://github.com/Bobvanderhall/Maintenance\\_Ontology](https://github.com/Bobvanderhall/Maintenance_Ontology)

## 8.2 Python Code

```
import pandas as pd
import numpy as np
from RDFLib import Graph
from RDFLib import term
from RDFLib import Namespace
from RDFLib import RDF
from datetime import datetime
import ifcopenshell as ios
import Namespace

global baseURI
inputFile = ""
targetFile = ""

includeBuildingElements = True
includeBuildingProperties = True
includeQuantities = True
includeGeometry = False

def convertIFCSPFtoTTL(inputFile, outputFile):
    inputFile = inputFile
    outputFile = outputFile
    now = datetime.now()
    current_time = now.strftime('%Y%m%d_%H%M%S')

    DEFAULT_PATH = "http://linkedbuildingdata.net/ifc/resources" + current_time + "/"
    global baseURI
    baseURI = DEFAULT_PATH

    #reading file - IfcSpfReader
    model = ios.open(inputFile)

    f = open(outputFile, "w", encoding='utf-8')
    writeTTLFileContent(model, f)
    f.close()

def writeTTLFileContent(model, file):
    file.write(writeTTLHeader())
    file.write(writeLBDinstances(model, file))

def print_properties(properties, output):
    for name, value in properties.items():
        if name == "id":
            continue
        name = cleanNameString(name)
        output += "; \n"
        if isinstance(value, int):
            output += "\tprops:"+name+" \""+ str(value) +"\"^^xsd:int "
        elif isinstance(value, float):
            output += "\tprops:"+name+" \""+ str(value) +"\"^^xsd:double "
        else:
            output += "\tprops:"+name+" \""+ str(value) +"\"^^xsd:string "
    return output

""" def print_quantities(quantities, output):
    for name, value in quantities.items():
        if name == "id":
            continue
        name = cleanNameString(name)
        output += "; \n"
        if isinstance(value, int):
            output += "\tprops:"+name+" \""+ str(value) +"\"^^xsd:int "
        elif isinstance(value, float):
            output += "\tprops:"+name+" \""+ str(value) +"\"^^xsd:double "
        else:
            output += "\tprops:"+name+" \""+ str(value) +"\"^^xsd:string "
    return output """

def cleanNameString(name):
    name = ''.join(x for x in name.title() if not x.isspace())
```

97

98



```
#         output = print_quantities(quantities, output)

    output += ". \n\n"
    return output

def writeElements(model,f):
    output = ""
    for b in model.by_type("IfcElement"):
        output += "inst:element_"+str(b.id()) + "\n"
        output += "\ta bot:Element ;" + "\n"
        if(b.Name):
            output += "\trdfs:label \""+b.Name+"\"^^xsd:string ;" + "\n"
        if(b.Tag):
            output += "\tprops:hasTag \""+b.Tag+"\"^^xsd:string ;" + "\n"
        if(b.Description):
            output += "\trdfs:comment \""+b.Description+"\"^^xsd:string ;" + "\n"
        output += "\tbot:hasGuid \""+ ios.guid.expand(b.GlobalId) +"\"^^xsd:string ;" + "\n"
        output += "\tprops:hasCompressedGuid \""+ b.GlobalId +"\"^^xsd:string "

        for relvoids in b.HasOpenings:
            if relvoids is not None:
                st = relvoids.RelatedOpeningElement
                for relfills in st.HasFillings:
                    if relfills is not None:
                        filler = relfills.RelatedBuildingElement
                        output += ";\n"
                        output += "\tbot:hostsElement inst:element_"+ str(filler.id()) + " "

        for relvoids in b.HasOpenings:
            if relvoids is not None:
                st = relvoids.RelatedOpeningElement
                for relfills in st.HasFillings:
                    if relfills is not None:
                        filler = relfills.RelatedBuildingElement
                        output += ";\n"
                        output += "\tbot:hostsElement inst:element_"+ str(filler.id()) + " "

    if(includeBuildingProperties):
        psets = ios.util.element.get_psets(b)
        for name, properties in psets.items():
            output = print_properties(properties, output)

    output += ". \n\n"
    return output

def writeInterfaces(model,f):
    output = ""
    for b in model.by_type("IfcRelSpaceBoundary"):
        output += "inst:interface_"+str(b.id()) + "\n"
        output += "\ta bot:Interface ;" + "\n"
        if(b.Name):
            output += "\trdfs:label \""+b.Name+"\"^^xsd:string ;" + "\n"
        if(b.Description):
            output += "\trdfs:comment \""+b.Description+"\"^^xsd:string ;" + "\n"
        output += "\tbot:hasGuid \""+ ios.guid.expand(b.GlobalId) +"\"^^xsd:string ;" + "\n"
        output += "\tprops:hasCompressedGuid \""+ b.GlobalId +"\"^^xsd:string "

        sp = b.RelatingSpace
        el = b.RelatedBuildingElement
        if sp is not None:
            output += ";\n"
            output += "\tbot:interfaceOf inst:space_"+ str(sp.id()) + " "
        if el is not None:
            output += ";\n"
            output += "\tbot:interfaceOf inst:element_"+ str(el.id()) + " "

    output += ". \n\n"
    return output

def shorten_text(text):
    # 3.2) Extract the description from the string, starting from starting_word (+14 because the word
    "Description - " is 14 indeces.) and ending with the first occurrence of one of the words in
    ending_words.
```

```

    starting_word = "Description - "
    ending_words = ["Melder:", "Naam:", "Oorzaak:", "Comment -", "ID:", "Melder:", "Best wishes,",
    "Kinds regards,", "Many thanks," "Thank you"]

    start_index = text.find(starting_word)
    end_index = len(text)

    for ending_word in ending_words:
        temp_end_index = text.find(ending_word)
        if temp_end_index > start_index and temp_end_index < end_index:
            end_index = temp_end_index

    result = text[start_index+14:end_index]
    return result

def ConvertExceltoTTL():
    # 1) Import Excel file and parse Dates as DateTimes
    df = pd.read_excel('2023-01-23 EMA Storingen snip YTD.xlsx', parse_dates=['MeasureDateTime',
    'NotificationDateTime'])

    # 2) Data cleaning
    # 2.1) Replace characters with desired value (for instance: spaces to underscores)
    df['Cause'] = df['Cause'].str.replace(' / ', '_').str.replace(' ', '_')
    df['Discipline'] = df['Discipline'].str.replace(' ', '_').str.replace('-', '').str.replace('(', '',
    regex=False).str.replace(')', '', regex=False)
    df['DamagePicture'] = df['DamagePicture'].str.replace(' ', '_').str.replace('/',
    '_').str.replace('__', '_').str.replace('-', '_').str.replace('_', '_')
    df['MeasureType'] = df['MeasureType'].str.replace(' - ', '_').str.replace(' ',
    '_').str.replace('/', '_').str.replace('_', '_').str.replace('-', '_')
    df['Storey'] = df['Storey'].str.replace(' ', '_')
    df['ElementID'] = df['ElementID'].astype(str).str.replace('.0', '', regex=False)
    df['AssetNumber'] = df['AssetNumber'].astype(str).str.replace('.0', '', regex=False)
    df['AssetDescription'] = df['AssetDescription'].str.replace(' ', '_').str.replace('(', '',
    regex=False).str.replace(')', '', regex=False).str.replace('__', '_').str.replace('-',
    '_').str.replace('_/_', '_').str.replace('""', '').str.replace('""', '').str.replace('; ',
    '').str.replace(' ', '').str.replace('.', '_ ', regex=False).str.replace('²', '2').str.replace('°',
    'graden-').str.replace('+', 'en', regex=False).str.replace('/', '_en_')
    df['OrderDescription'] = df['OrderDescription'].str.replace('""', '').str.replace('.', '',
    regex=False)
    df['RoomNumber'] = df['Room'].astype(str)
    df['RoomNumber'] = df['RoomNumber'].str.split(' ').str[0]
    df['Room'] = df['Room'].str.replace(' | ', '_ ', regex=False).str.replace(' ', '_').str.replace('(',
    '', regex=False).str.replace(')', '', regex=False).str.replace('_/_', '_').str.replace('/',
    '').str.replace('.', '', regex=False).replace('\n', '', regex=True).str.replace('&',
    '').str.replace(' ', '')

    # 2.2) Sometimes the measure text is provided with a wrong answer. These cells need to be replaced
    by an empty string.
    df.loc[pd.isna(df['MeasureText']) & df['MeasureText'].str.startswith('Summary'), 'MeasureText'] =
    ''

    df['MeasureText'].fillna("No text available.", inplace=True)
    df['MeasureText'] = df['MeasureText'].str.replace('""', '')

    # 3) There are direct notifications and notifications in a specific format. The formatted
    notifications must be transformed to description only
    # 3.1) Remove new lines and replace empty cells with the text "No text available."
    df['NotificationText'].fillna("No text available.", inplace=True)
    df['NotificationText'] = df['NotificationText'].replace('\n', ' ', regex=True)

    # 3.2? See def shorten_text(text)
    # 3.3) Finally only apply the function to cells that start with "0000"
    df["NotificationTextFiltered"] = df.apply(lambda row: shorten_text(row["NotificationText"]) if
    row["NotificationText"].startswith("0000") else row["NotificationText"], axis=1)
    df['NotificationTextFiltered'] = df['NotificationTextFiltered'].str.replace('""',
    '').str.replace('\n', '', regex=False)

    # 4) Create new temporary column for MeasureDateTime in string format with spaces replaced to
    underscores (this is done to make the measures unique later).
    df['MeasureDateTime_'] = df['MeasureDateTime']
    df['MeasureDateTime_'] = df['MeasureDateTime_'].astype(str).str.replace(' ', '_')

    # 5) Create new columns for the triples with:
    # 5.1) Object Properties:

```

```

df['Order_takesPlaceIn_Room'] = 'EMA:Order_' + df['OrderNumber'].astype(str) + ' MO:takesPlaceIn
EMA:Room_' + df['Room'].astype(str)
df['Order_relatedTo_Asset'] = 'EMA:Order_' + df['OrderNumber'].astype(str) + ' MO:relatedTo
EMA:Asset_' + df['ElementID'].astype(str) + '_' + df['AssetDescription'].astype(str)
df['Order_hasCause_Cause'] = 'EMA:Order_' + df['OrderNumber'].astype(str) + ' MO:hasCause MO:' +
df['Cause'].astype(str)
df['Order_hasDiscipline_Discipline'] = 'EMA:Order_' + df['OrderNumber'].astype(str) + '
MO:hasDiscipline MO:' + df['Discipline'].astype(str)
df['Order_hasDamagePicture_DamagePicture'] = 'EMA:Order_' + df['OrderNumber'].astype(str) + '
MO:hasDamagePicture MO:' + df['DamagePicture'].astype(str)
df['Order_hasMeasure_Measure'] = 'EMA:Order_' + df['OrderNumber'].astype(str) + ' MO:hasMeasure
EMA:Measure_' + df['MeasureNumber'].astype(str) + '_' + df['MeasureDateTime'].astype(str)
df['Room_containsAsset_Asset'] = 'EMA:Room_' + df['Room'].astype(str) + ' MO:containsAsset
EMA:Asset_' + df['ElementID'].astype(str) + '_' + df['AssetDescription'].astype(str)
df['Storey_containsRoom_Room'] = 'EMA:Storey_' + df['Storey'].astype(str) + ' MO:containsRoom
EMA:Room_' + df['Room'].astype(str)

# 5.2) rdf:type and rdfs:subClassOf relations with the 'MO'Ontology:
df['Order_type_MO:NotificationType'] = 'EMA:Order_' + df['OrderNumber'].astype(str) + ' rdf:type
MO:' + df['NotificationType'].astype(str)
df['NotificationType_subClassOf_MO:Order'] = 'MO:' + df['NotificationType'].astype(str) + '
rdfs:subClassOf MO:Order'
df['Asset_type_MO:Asset'] = 'EMA:Asset_' + df['ElementID'].astype(str) + '_' +
df['AssetDescription'].astype(str) + ' rdf:type MO:Asset'
df['Room_type_MO:Room'] = 'EMA:Room_' + df['Room'].astype(str) + ' rdf:type MO:Room'
df['Storey_type_MO:Storey'] = 'EMA:Storey_' + df['Storey'].astype(str) + ' rdf:type MO:Storey'
df['DamagePicture_type_MO:DamagePicture'] = 'MO:' + df['DamagePicture'].astype(str) + ' rdf:type
MO:DamagePicture'
df['Cause_type_MO:Cause'] = 'MO:' + df['Cause'].astype(str) + ' rdf:type MO:Cause'
df['Discipline_type_MO:Discipline'] = 'MO:' + df['Discipline'].astype(str) + ' rdf:type
MO:Discipline'
df['Measure_type_MO:MeasureType'] = 'EMA:Measure_' + df['MeasureNumber'].astype(str) + '_' +
df['MeasureDateTime'].astype(str) + ' rdf:type MO:' + df['MeasureType'].astype(str)
df['MeasureType_subClassOf_MO:Measure'] = 'MO:' + df['MeasureType'].astype(str) + ' rdfs:subClassOf
MO:Measure'

# 5.3) Data Properties:
df['Order_hasCauseText_Literal'] = 'EMA:Order_' + df['OrderNumber'].astype(str) + ' MO:hasCauseText
''' + df['CauseText'].astype(str) + '''
df['Order_hasDamagePictureText_Literal'] = 'EMA:Order_' + df['OrderNumber'].astype(str) + '
MO:hasDamagePictureText ''' + df['DamagePictureText'].astype(str) + '''
df['Order_hasNotificationDateTime_Literal'] = 'EMA:Order_' + df['OrderNumber'].astype(str) + '
MO:hasNotificationDateTime ''' + df['NotificationDateTime'].astype(str) + '''
df['Order_hasNotificationText_Literal'] = 'EMA:Order_' + df['OrderNumber'].astype(str) + '
MO:hasNotificationText ''' + df['NotificationTextFiltered'].astype(str) + '''
df['Order_hasOrderDescription_Literal'] = 'EMA:Order_' + df['OrderNumber'].astype(str) + '
MO:hasOrderDescription ''' + df['OrderDescription'].astype(str) + '''
df['Order_hasOrderNumber_Literal'] = 'EMA:Order_' + df['OrderNumber'].astype(str) + '
MO:hasOrderNumber ''' + df['OrderNumber'].astype(str) + '''
df['Order_hasOrderStatus_Literal'] = 'EMA:Order_' + df['OrderNumber'].astype(str) + '
MO:hasOrderStatus ''' + df['OrderStatus'].astype(str) + '''
df['Measure_hasMeasureText_Literal'] = 'EMA:Measure_' + df['MeasureNumber'].astype(str) + '_' +
df['MeasureDateTime'].astype(str) + ' MO:hasMeasureText ''' + df['MeasureText'].astype(str) + '''
df['Measure_hasMeasureDateTime_Literal'] = 'EMA:Measure_' + df['MeasureNumber'].astype(str) + '_' +
df['MeasureDateTime'].astype(str) + ' MO:hasMeasureDateTime ''' + df['MeasureDateTime'].astype(str) +
'''
df['Asset_hasElementID_Literal'] = 'EMA:Asset_' + df['ElementID'].astype(str) + '_' +
df['AssetDescription'].astype(str) + ' MO:hasElementID ''' + df['ElementID'].astype(str) + '''
df['Asset_hasAssetNumber_Literal'] = 'EMA:Asset_' + df['ElementID'].astype(str) + '_' +
df['AssetDescription'].astype(str) + ' MO:hasAssetNumber ''' + df['AssetNumber'].astype(str) + '''
df['Room_hasRoomNumber_Literal'] = 'EMA:Room_' + df['Room'].astype(str) + ' MO:hasRoomNumber ''' +
df['RoomNumber'].astype(str) + '''

# 6) Remove empty cells (containing "nan")
string_parts = ["_nan", ":nan", "nan"]
df = df.applymap(lambda x: "" if any(string_part in str(x) for string_part in string_parts) else x)

# 7) Export to new Excel file with .1 added to the file name. not necicary for script, but just to
check if the conversion went well.
df.to_excel('2023-01-23 EMA Storingen snip YTD.1.xlsx', index=False)

# 8) Transform all cells from the triple columns to a single list, while excluding empty cells and
dropping duplicate values.

```

```

    columns = ['Order_takesPlaceIn_Room', 'Order_relatedTo_Asset', 'Order_hasCause_Cause',
'Order_hasDiscipline_Discipline', 'Order_hasDamagePicture_DamagePicture', 'Order_hasMeasure_Measure',
'Storey_containsRoom_Room', 'Room_containsAsset_Asset', 'Order_type_MO:NotificationType',
'NotificationType_subClassOf_MO:Order', 'Asset_type_MO:Asset', 'Storey_type_MO:Storey',
'Room_type_MO:Room', 'DamagePicture_type_MO:DamagePicture', 'Cause_type_MO:Cause',
'Discipline_type_MO:Discipline', 'Measure_type_MO:MeasureType', 'MeasureType_subClassOf_MO:Measure',
'Order_hasCauseText_Literal', 'Order_hasDamagePictureText_Literal',
'Order_hasNotificationDateTime_Literal', 'Order_hasOrderDescription_Literal',
'Order_hasOrderNumber_Literal', 'Order_hasOrderStatus_Literal', 'Measure_hasMeasureDateTime_Literal',
'Asset_hasElementID_Literal', 'Asset_hasAssetNumber_Literal', 'Room_hasRoomNumber_Literal',
'Measure_hasMeasureText_Literal', 'Order_hasNotificationText_Literal']
    all_cells = list(set(df[columns].apply(lambda x: x[x!=''].tolist()).sum()))

    # 9) Transform the list into triples in turtle format: adding a space and a point to each string,
    and then transform the list to triples separated by new lines
    all_cells = [item + " ." for item in all_cells]
    Triples = "\n".join(all_cells)

    #10) Export the triples to a ttl file with the following prefixes:
    Prefixes = "@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .\n@prefix rdfs:
<http://www.w3.org/2000/01/rdf-schema#> .\n@prefix MO: <https://heijmans.org/maintenanceontology#>
.\n@prefix EMA: <https://heijmans.org/EMA#> .\n\n"
    with open("Converted Data.ttl", "w", encoding='utf-8') as f:
        f.write(Prefixes + Triples)

def CreateAssetRelationship(g, Element, Asset):
    relationshipname = "hasbotElementOrigin"
    g.add((Asset, term.URIRef("https://heijmans.org/maintenanceontology#" + relationshipname),
Element))
    return

def Map_ElementID_Tag():
    # 1) load files
    g = Graph()
    # Maintenance data
    g.parse("Converted Data.ttl")

    # BIM Models
    g.parse("EMA_Architectural.ttl")
    g.parse("EMA_Structural.ttl")
    g.parse("EMA_Electrical.ttl")
    g.parse("EMA_HVAC_MEP.ttl")

    # Bind prefix and namespace
    g.bind("MO", "https://heijmans.org/maintenanceontology#")

    # 2) search EMA:Assets
    AssetQuery = """select ?Asset ?ElementID
where{
    ?Asset rdf:type MO:Asset .
    ?Asset MO:hasElementID ?ElementID .
}"""
    qres = g.query(AssetQuery)
    # for row in qres:
    #     print(f"{row.Asset} hasElementID {row.ElementID}")

    # 3) search bot:Elements
    ElementQuery = """select ?Element ?Tag
where {
    ?Element a bot:Element .
    ?Element props:hasTag ?Tag
}"""
    qres1 = g.query(ElementQuery)
    # for row in qres1:
    #     print(f"{row.Element} hasTag {row.Tag}")

    # 4) Create relationships
    for Assetrow in qres:
        for Elementrow in qres1:
            if Assetrow.ElementID == Elementrow.Tag:
                CreateAssetRelationship(g, Elementrow.Element, Assetrow.Asset)

    # 5) Write TTL file

```

```
g.serialize(destination="Merge_Assets_Elements.ttl")

def CreateRoomRelationship(g, Space, Room):
    relationshipname = "hasbotSpaceOrigin"
    g.add((Room, term.URIRef("https://heijmans.org/maintenanceontology#" + relationshipname), Space))
    return

def Map_RoomNumber_SpaceNumber():
    # 1) load files
    g = Graph()
    g.parse("Merge_Assets_Elements.ttl")
    g.bind("MO", "https://heijmans.org/maintenanceontology#")

    # 2) search EMA:Rooms
    RoomQuery = """select ?Room ?RoomNumber
    where{
        ?Room rdf:type MO:Room .
        ?Room MO:hasRoomNumber ?RoomNumber .
    }"""
    qres2 = g.query(RoomQuery)
    #for row in qres2:
        #print(f"{row.Room} hasRoomNumber {row.RoomNumber}")

    # 3) search bot:Spaces
    SpaceQuery = """select ?Space ?SpaceNumber
    where{
        ?Space rdf:type bot:Space .
        ?Space rdfs:label ?SpaceNumber .
    }"""
    qres3 = g.query(SpaceQuery)
    #for row in qres3:
        #print(f"{row.Space} hasSpaceNumber {row.SpaceNumber}")

    # 4) Create relationships
    for Roomrow in qres2:
        for Spacerow in qres3:
            if Roomrow.RoomNumber == Spacerow.SpaceNumber:
                CreateRoomRelationship(g, Spacerow.Space, Roomrow.Room)

    # 5) Write TTL file
    g.serialize(destination="Merge_Assets_Elements_Rooms_Spaces.ttl", use_plain=True)

# Run functions

# IFC to LBD Converter
convertIFCSPFtoTTL("Project_Architectural.ifc", "Project_Architectural.ttl")
convertIFCSPFtoTTL("Project_Structural.ifc", "Project_Structural.ttl")
convertIFCSPFtoTTL("Project_Electrical.ifc", "Project_Electrical.ttl")
convertIFCSPFtoTTL("Project_HVAC_MEP.ifc", "Project_HVAC_MEP.ttl")
# SAP to RDF Converter
ConvertExceltoTTL()
#Data Integration
Map_ElementID_Tag()
Map_RoomNumber_SpaceNumber()
```