Integrating sensor- and building dataflows

A case study of indoor environmental quality assessment of an office building in the Netherlands”

Sjors van Gool

March 2020

MSc Construction Management and Engineering

Colophon

General

Thesis title	Integrating sensor- and building dataflows
Thesis subtitle	A case study of indoor environmental quality assessment of an office building in the Netherlands
Date	4th of March 2020
Date final colloquium	11th of March 2020
Place	Eindhoven

Student

Author	Sjors van Gool
Student number	0859231
E-mail	s.v.gool@student.tue.nl / sjorsvangool@gmail.com
University	Eindhoven University of Technology
Master track	Construction Management and Engineering (CME)
Chair	Information Systems in the Built Environment (ISBE)

Collaborating party

Company	Radboudumc
Department	Vastgoed & Infrastructuur

Graduation committee

Chairman	Prof. dr. ir. B. (Bauke) de Vries *Full Professor, Department of the Built Environment, Information Systems in the Built Environment*
University supervisor	ir. L.I.H. (Luuk) Wijnholts *University Lecturer, Department of the Built Environment, Information Systems in the Built Environment*
University supervisor	dr. D. (Dujuan) Yang *Assistant Professor, Department of the Built Environment, Information Systems in the Built Environment*
University supervisor	dr. ir. P. (Pieter) Pauwels *Associate Professor, Department of the Built Environment, Information Systems in the Built Environment*
Radboudumc supervisor	ing. N. (Norbert) Janssen *Manager Vastgoedmanagement, Department of Vastgoed & Infrastructuur*

Table of contents

Preface

This report is the result of the graduation research carried out in collaboration with the Eindhoven University of Technology and the Radboudumc. This research finalizes the master Construction Management and Engineering, and thereby also marks the end of my student career. I have changed a lot since I started at the TU/e in 2013 and I am very proud of what I have accomplished during this time. I could never have done this without the great education and support that I received at the Eindhoven University of Technology.

My interests also changed numerous times during my student years. In 2013 I started with the ambition to become an architect. Looking at where we are now you can imagine that this ambition was adjusted quickly. However, in the latter years it changed more frequently as I started to experience the countless possibilities that technology could bring to our sector.

This made choosing a suitable topic for my graduation a lot more difficult. I wanted to research step 3 while our sector is only at step 1. I would truly like to thank Bauke de Vries, Dujuan Yang and Luuk Wijnholts for their great support, which made me realise that the basis must first be laid prior to engaging in such future initiatives. Such wisdom comes with experience I guess, at least I hope so…

Once the topic of integrating building and sensor dataflows was chosen, I was confronted with a whole new area of research of which I had no clue how it worked. Luckily our department is strengthened with the arrival of our new assistant professor Pieter Pauwels, who is an expert in this area. I would like to thank Pieter, Dujuan and Luuk for their strong support during this graduation. This result would never have been possible without your guidance.

Also, I would like to thank everyone at the Radboudumc for their support and trust in me, especially my supervisor Norbert Janssen. Norbert supported me throughout this graduation, let me engage in related pilots that were running within V&I and gave me a lot of freedom during this graduation. Without his trust and guidance this result would never have been possible.

As a former Radboudumc employee, I am very proud of the progress that we are making, especially in the area of real estate. The Radboudumc dares to experience with new technologies and ways of working without an assurance of success and profit. I would love to see our sector digitize and evolve technologically, and I aim to dedicate my career to this goal. However, without companies that dare to experiment, this goal shall not be reached. I hope, and I believe, that the Radboudumc shall keep up this progress and keep achieving great things in the future.

Finally, I would like to thank my friends, my mom, dad, brother, sister and my other family members for the support that they gave me over the years. Mom, dad, you have always given me the autonomy and motivation that helped me both shape and realise my ambitions. I am very grateful for that.

I hope reading this report will give you as much energy as it gave me.

Sjors van Gool
Eindhoven, March 2020

Summary

We are the indoor generation. Approximately 90% of our time is being spent indoors (YouGov, 2018). Ensuring a good indoor climate is therefore essential, especially as a bad indoor climate strongly affects the productivity, health and wellbeing of the occupants (Jia et al., 2018; Klepeis et al., 2001). In a world where big data is one of the most important and revolutionary developments in the past half century, it could be argued that the built environment should be able to identify and solve unsatisfactory conditions by collecting usage data in buildings. However, the amount of data collected and the intensity of it being used in the built environment industry is significantly lower than in other industries (Loyola, 2018). To change this, the paradigm of using BIM-related technology is increasingly being used. Though, by integrating other technologies, the capabilities of BIM technology can be extended throughout the full building lifecycle. One of the most promising integrations is the Internet of Things (IoT), especially sensor data (Dave, Kubler, Främling, & Koskela, 2016).

Problem definition

The integration of building/BIM and sensor dataflows has the potential to alter the way people interact with the built environment and the involvement of all parties (Dave, Buda, Nurminen, & Främling, 2018a). By effectively utilizing the full potential of the data gathered in the current building stock, the building performance can be improved and, consequently, the productivity and wellbeing of occupants as well. Therefore, the built environment is considered as the most important sector for sensor-related research (Dave et al., 2018a). However, failure in effectively integrating sensor data with building data will hinder its possibilities (K. M. Chang, Dzeng, & Wu, 2018; Tang, Shelden, Eastman, Pishdad-Bozorgi, & Gao, 2019).

Without information integration and management standards, data analytics with large, heterogeneous datasets is costly and time-inefficient (Gerrish et al., 2017). BIM-authoring tools often use common data standards such as IFC, but these are unable to support the more complex datatypes (Loyola, 2018; Patacas, Dawood, Vukovic, & Kassem, 2015).

Research goal

Considering the increased emphasis on improving building performance as well as the potential of combining building and sensor dataflows to provide actionable insights, this research shall focus on effectively combining these dataflows to provide actionable insights in improving the Indoor Environmental Quality (IEQ). The following main research question shall be addressed:

"How can building and sensor dataflows be effectively combined to provide actionable insights in improving indoor environmental quality and occupancy efficiency in an existing building stock?"

This research shall consist of two main parts: a literature review and a case study on a UMC office building. As the aimed integration is not a final goal but a method of achieving new goals, an extensive discussion shall elaborate on what future developments such an integration can contribute to.

Dataflow integration

From the literature review it followed that using a hybrid integration approach would be most promising: semantic modelling for building data using domain-specific ontologies and keeping the sensor (time-series) data in its native format.

To conduct the IEQ analyses, the linked building data sources must be easily queried in a Python script. Using GraphDB to store the Turtle graphs, the built in GraphDB API can be used. This can be done in either GraphDB's SPARQL Endpoint or by locally parsing the whole graph using

Python’s RDFLib package. Using Python’s RDFLib package, the full graph is parsed into the local Python environment. Depending on the use case, this might be faster than executing countless direct SPARQL Endpoint queries. If only a few queries are to be made, a direct SPARQL Endpoint query might be preferred. The sensor data in the MongoDB NoSQL database can be queried using the Python package ‘pymongo’. By naming the MongoDB collections after a certain datapoint in the PROPS graph, a link can be made.

For a domain-specific ontology integration to function, ontologies must be carefully chosen. In this thesis several ontologies have been considered and were indeed carefully selected. Aiming for an industry-wide standardization in the usage of ontologies, three main building ontologies developed by the W3C Linked Building Data Group have been selected: the BOT as foundation ontology, the PRODUCT as building element ontology and the PROPS as element property ontology. Additionally, to integrate geometry elements in the IEQ analyses, a dedicated gbXML ontology has been written and COLLADA mesh geometry ID’s have been linked. The link between the geometry elements and the other building ontologies has been made through mutual datapoints in the PROPS graph and gbXML/COLLADA data.

To illustrate the dataflow conversion from the initial formats to the integrated situation, Figure 1 shows a Yourdon and Coad dataflow diagram of the entire integration.

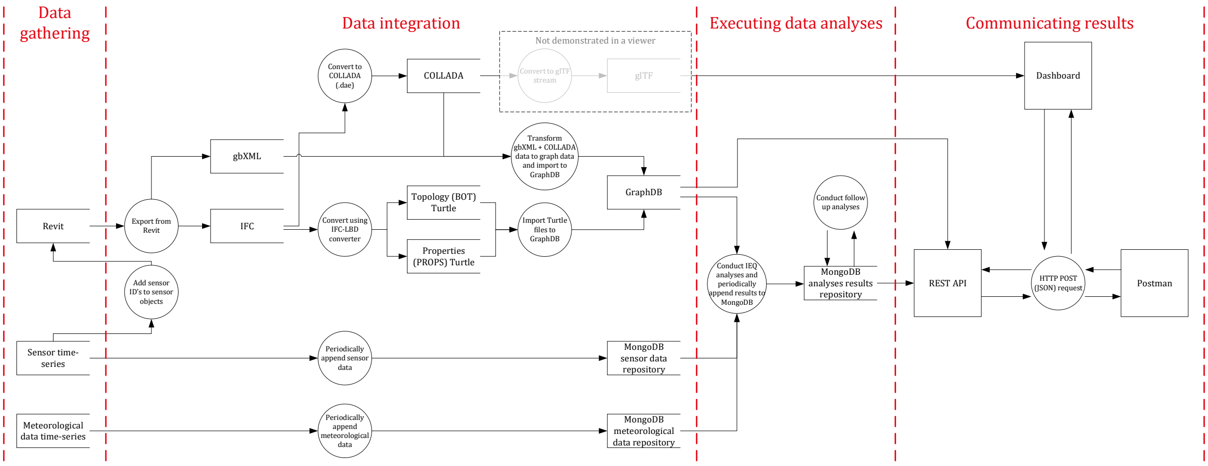


Figure 1: Dataflow diagram of the proposed building/sensor dataflow integration

Several ways exist to make the link between a sensor and the building model. All sensors with a specific property or naming can automatically be appended to a room, or specific sensors can be drawn in the building model. As shown in Figure 1, the latter has been chosen for this thesis. A sensor can be drawn as an object and attached to another object, for example a table or a wall, making the exact location of the sensor known. A condition must be fulfilled though: the sensor name in the time-series database must be equal to a property of the sensor in Revit. In this thesis the property ‘Comments’ has been used, but any property that can define a string works. However, during the integration to LBD graphs this property must be converted as well, which is currently not the case for custom properties.

Each step shown in Figure 1 has been elaborated on in this thesis, including how to interact with each format. To ease the integration, this thesis has created some linking shortcuts between the data formats. As numerous datapoints are involved, Figure 2 demonstrates the fully integrated situation in a Neo4J-styled diagram. Even though the actual integration has not been created using Neo4J, a Neo4J-styled diagram can still clearly demonstrate the developed integration.

This diagram aims to clarify each used point including how the link between the formats are made. Getting the data right is the first and foremost step in conducting effective analyses. To make sure of this, a script has been written that checks several common data quality issues during this integration.

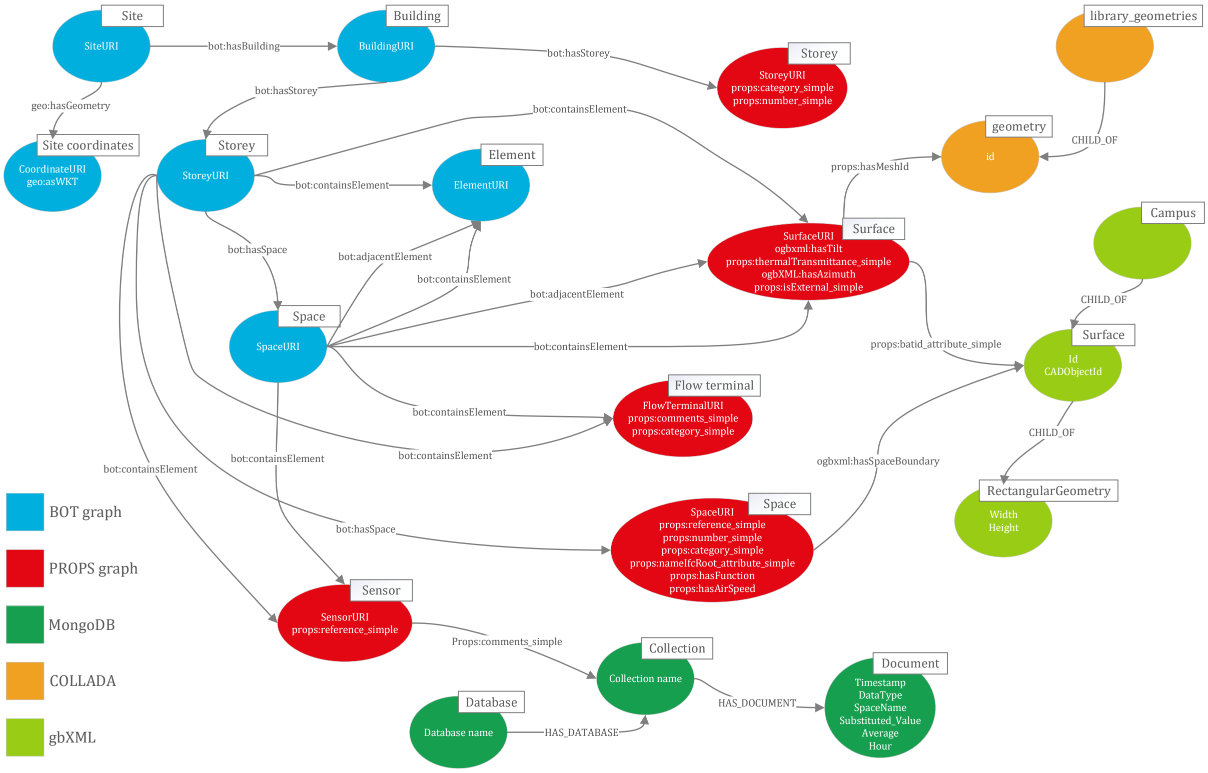


Figure 2: A Neo4J-styled diagram with the LBD enrichment

Proving the integration functionality with IEQ analyses

Using the created integration, numerous new possibilities in combining both building and sensor dataflows arise. To prove its functionality, the following Indoor Environmental Quality (IEQ) assessments have been automatically conducted over the measured rooms: indoor air quality (IAQ), thermal comfort (including mean radiant heat analysis) and occupancy efficiency. To take it a step further, these results are combined for actionable insights in an optimized long-term ventilation capacity distribution to redistribute the constant air volume systems at the pilot building.

Communicating the results

One of the main strengths of semantic modelling is its easy integration and sharing of data. Executing everything in a local Python environment does not empower this strength. Therefore, to easily communicate the sensor data and analysis results, a REST API and a dashboard have been created.

Scientific contribution

The results of this thesis demonstrate an integration that does indeed successfully combine building and sensor dataflows. Additionally, it proves that building data can complement time-series dataflows in achieving more useful IEQ results than only portraying the measured sensor values. It shows that several linked building data approaches, which are thoroughly described in literature, work well in practice. Also, it demonstrated code to easily work with these various data sources. Using the developed integration, follow up research can fully focus on the next steps of effectively utilizing the synergy between numerous data sources within buildings.

Societal contribution

One of the most essential issues in building and sensor dataflow integration is convincing companies to engage in gathering such data in the first place. Clear business cases are needed to set such projects in motion, including at the Radboudumc. The societal importance of this thesis focusses on the possibilities that such a data integration can bring forth. It demonstrates that combining of dataflows is possible, that analyses can be conducted and that insights can be gathered relatively easily.

Conclusion

The integration created in this thesis provides a solution for, but is not limited to, suggesting actionable insights on improving IEQ. The potential of the presented building dataflows to be used in numerous application areas has not been realised yet. However, as the frequency of applying IoT devices, among others sensors, increases, continuous building monitoring and analyses are enabled. Through the integration demonstrated in this thesis, heterogenous data sources can be effectively utilized. Instead of considering buildings as a final product after delivery, involved parties can consider buildings to be dynamic objects that can be continuously improved, benefitted from and learned from.

Samenvatting

Wij zijn een indoor generatie. Ongeveer 90% van onze tijd wordt binnen besteed (YouGov, 2018). Het garanderen van een goed binnenklimaat is daarmee essentieel, zeker aangezien een slecht binnenklimaat negatieve gevolgen kan hebben voor de productiviteit, gezondheid en welzijn van de gebruikers (Jia et al., 2018; Klepeis et al., 2001). In een wereld waar big data een van de belangrijkste en revolutionaire ontwikkelingen van de afgelopen eeuw vormt, zou verwacht kunnen worden dat de gebouwde omgeving veelal ontoereikende binnenklimaat condities automatisch identificeert en oplost. Echter de hoeveelheid data dat binnen de gebouwde omgeving verzameld wordt en de intensiteit dat het gebruikt wordt is significant lager dan in andere industrieën (Loyola, 2018).

Om hier verandering in te brengen wordt BIM-gerelateerde technologie steeds meer toegepast. Door integratie met andere technologieën zouden de mogelijkheden uitgebreid kunnen worden voor de gehele levenscyclus van gebouwen. Een van de meest veelbelovende integraties is met de Internet of Things (IoT), met name sensor data (Dave et al., 2016).

Probleemstelling

De integratie van gebouw/BIM data met sensor datastromen heeft de potentie om de manier dat mensen omgaan met de gebouwde omgeving aan te passen (Dave et al., 2018a). Door de verzamelde data in de bestaande bouw effectief in te zetten, kunnen de prestaties van een gebouw worden verbeterd, en als gevolg ook de productiviteit, de gezondheid en het welzijn van de gebruikers. Hierom wordt de gebouwde omgeving gezien als een van de meest belangrijke sectoren voor sensor-gerelateerd onderzoek (Dave et al., 2018a). Echter het missen van een succesvolle integratie tussen gebouwdata en sensordata zal deze mogelijkheden verminderen (K. M. Chang et al., 2018; Tang et al., 2019). Zonder informatie integratie en management standaarden zullen data analyses met grote heterogene datasets duur en tijds-inefficiënt zijn (Gerrish et al., 2017).

Onderzoeksdoel

De nadruk op het verbeteren van gebouwprestaties en het potentieel van het gecombineerd gebruik van sensor- en gebouwdatastromen groeit. Hierom heeft dit onderzoek zich gericht op het effectief combineren van deze datastromen, om vervolgens toepasbare inzichten te generen voor het verbeteren van het binnenklimaat. De volgende centrale onderzoeksvraag is gesteld:

"How can building and sensor dataflows be effectively combined to provide actionable insights in improving indoor environmental quality and occupancy efficiency in an existing building stock?"

Dit onderzoek bestaat uit twee onderdelen: een literatuur review en een case study op een UMC-kantoorgebouw. Aangezien de beoogde integratie niet een uiteindelijk doel op zich is maar meer een methode om verdere doelen te bereiken, is een uitgebreide discussie geleverd over de doelen waaraan een dergelijke integratie bij kan dragen.

Integratie van datastromen

Uit de literatuur review bleek dat een hybride integratie oplossing de meest veelbelovende aanpak was. Deze bestond uit semantisch modelleren voor gebouw data gebruik makend van domein-specifieke ontologieën, en de sensor (time-series) data in het oorspronkelijke formaat laten.

Om de binnenklimaat analyses uit te voeren zullen de verschillende databronnen gemakkelijk in een Python script benaderd moeten kunnen worden. Door gebruik te maken van GraphDB om de Turtle grafieken in op te slaan, kan de ingebouwde GraphDB API in worden gezet. Een

query kan dan beide via de directe GraphDB SPARQL Endpoint uit worden gevoerd, of door de gehele grafiek in de lokale Python omgeving te parsen. Afhankelijk van de use case zal de ene optie sneller zijn dan de andere. Wanneer er veel queries gemaakt moeten worden zal het sneller zijn om de gehele graph te parsen, echter als er minder dan ongeveer 30 queries gemaakt moeten worden dan zal de directe SPARQL Endpoint sneller zijn. De sensordata in de MongoDB NoSQL database kan opgevraagd worden via de Python package ‘pymongo’. Door de benaming van de MongoDB collecties gelijk te stellen aan een bepaald datapunt in de PROPS graph, bijvoorbeeld het ruimte ID, kan een link gemaakt worden.

Om een domein-specifieke ontologie integratie werkend te krijgen, moeten de ontologieën zorgvuldig gekozen worden. In deze thesis zijn enkele ontologieën afgewogen en beargumenteerd gekozen. Met een industrie-brede standaardisatie in het gebruik van ontologieën als doel zijn drie hoofdzakelijke gebouw ontologieën van de W3C Linked Building Data Group gekozen: de BOT als gebouw topologie ontologie, de PRODUCT als gebouw element ontologie en de PROPS als element eigenschap ontologie. Om de geometrie elementen in de binnenklimaat analyses te integreren is een specifieke gbXML ontologie geschreven en zijn de COLLADA mesh geometrie ID’s gelinkt. De link tussen de geometrie elementen en de andere gebouw ontologieën is gemaakt via overeenkomende datapunten in de PROPS graph en de gbXML/COLLADA data.

Om de datastroom conversie van het originele formaat naar de geïntegreerde situatie te illustreren, toont Figure 3 een Yourdon and Coad datastroom diagram van de algehele integratie.

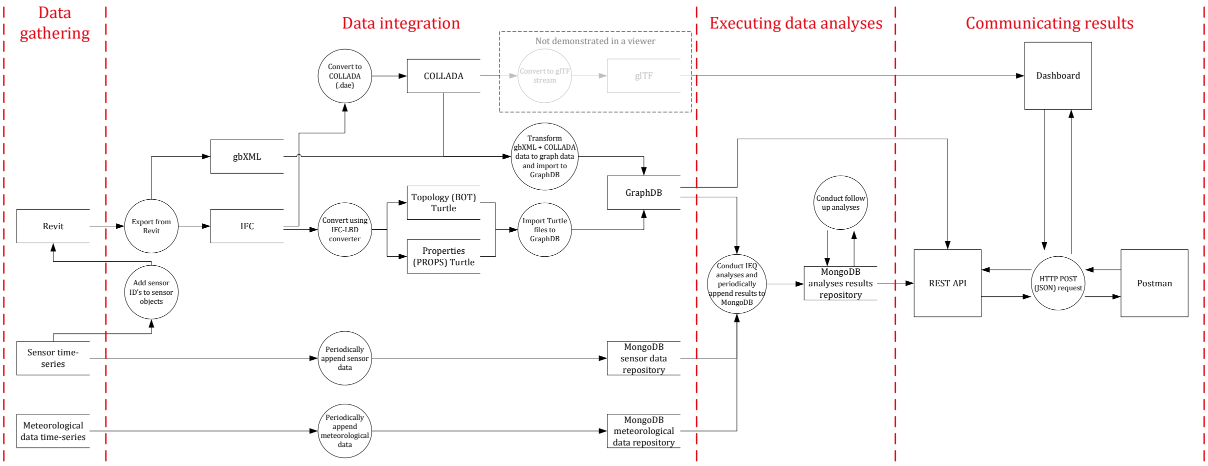


Figure 3: Datastroom diagram van de voorgestelde gebouw/sensor datastroom integratie

Meerdere manieren bestaan voor het maken van de link tussen een sensor en een gebouw model. Alle sensoren met een specifieke eigenschap of benaming kunnen automatisch aan een ruimte toe worden gewezen. Een alternatieve aanpak is het intekenen van specifieke sensoren in het gebouw model. Zoals Figure 3 laat zien is er in dit onderzoek voor de tweede optie gekozen. Een sensor kan als een object in worden getekend en op een ander object, zoals een tafel of muur, worden geplaatst, waardoor de exacte locatie van de sensor duidelijk is. Een voorwaarde hierbij is dat de sensor naam in de time-series database gelijk is aan een sensor eigenschap in Revit. In dit onderzoek is ‘Comments’ gebruikt, maar een andere eigenschap met een string input kan hier ook voor worden gekozen.

Elke stap in Figure 3 is toegelicht in dit rapport, inclusief de interactiemethodes voor elk formaat. Om de integratie te vergemakkelijken zijn enkele extra links gemaakt tussen de dataformaten. Aangezien vele datapunten hierbij betrokken zijn, is in Figure 4 een Neo4J grafiek opgesteld van de hele geïntegreerde situatie. Ondanks dat de integratie niet in Neo4J gemaakt is, kan een Neo4J grafiek wel duidelijk de links weergeven zoals die in deze thesis op zijn gesteld. Deze diagram heeft als doel om elk gebruikt datapunt, inclusief de onderliggende links, duidelijk te maken. De data goed geïntegreerd krijgen is de eerste en belangrijkste stap in het effectief uitvoeren van analyses. Om zeker te zijn van de datakwaliteit is een script geschreven die verschillende veelvoorkomende datakwaliteit issues in deze integratie checkt.

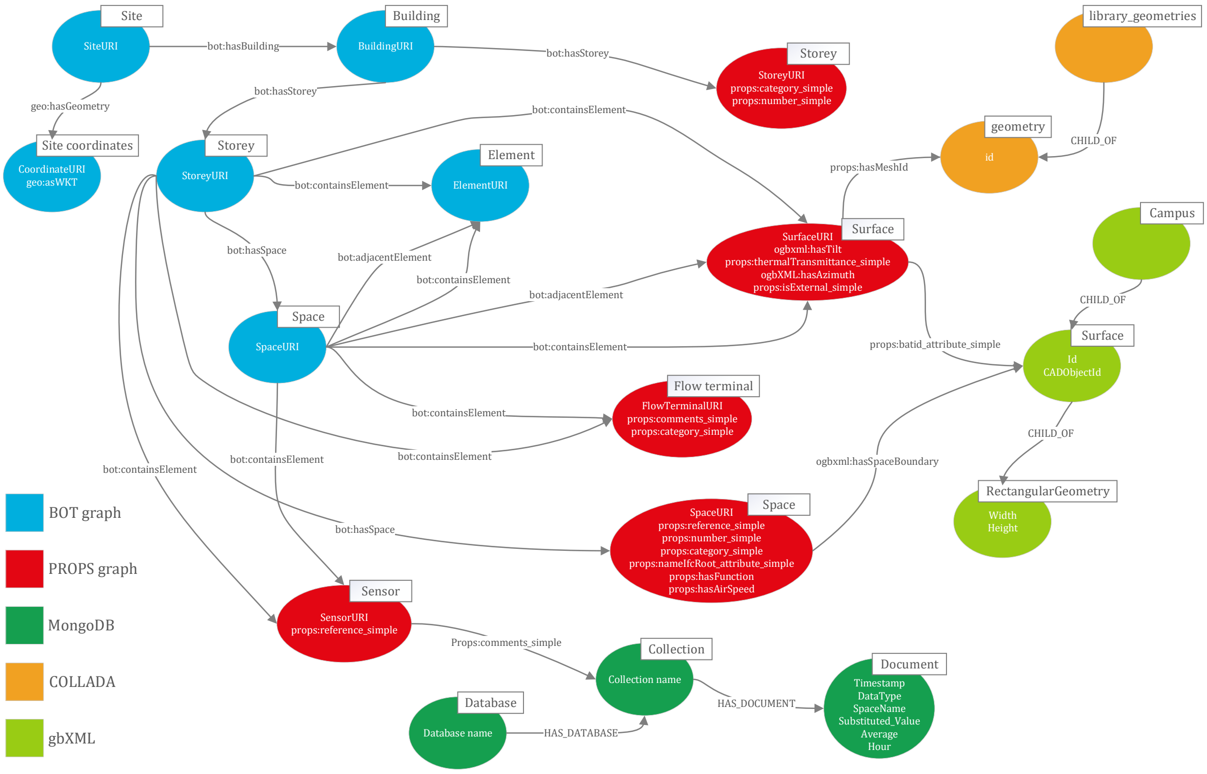


Figure 4: Een Neo4J grafiek met de toegevoegde links

De werking van de integratie bewijzen met de binnenklimaat analyses

Met de gecreëerde integratie zijn verschillende nieuwe mogelijkheden in het gecombineerd gebruik van beide gebouw- en sensordatastromen ondersteund. Om de werking te bewijzen zijn de volgende binnenklimaat analyses automatisch uitgevoerd voor alle gemeten ruimtes: binnenklimaat, thermisch comfort (inclusief stralingswarmte) en bezetting. Om een stap verder te gaan zijn deze resultaten gecombineerd in toepasbare inzichten voor het optimaal verdelen van aanwezige ventilatiecapaciteit.

De resultaten communiceren

De kracht van semantisch modelleren zit in het flexibel integreren en delen van data. Alles in een lokale Python omgeving uitvoeren zou deze kracht niet tot stand brengen. Om de sensordata, gebouwdata en analyseresultaten goed te communiceren is een REST API en een dashboard opgezet.

Academische bijdrage

De resultaten van dit onderzoek bewijzen dat een succesvolle integratie van sensor- en gebouwdatastromen praktisch mogelijk is. Daarnaast bewijst het dat gebouwdata time-series data kan ondersteunen in het verkrijgen van nuttigere binnenklimaat resultaten dan enkel het weergeven van de gemeten waardes. Het toont aan dat verschillende linked building data aanpakken, die veelal zijn beschreven in literatuur, in de praktijk goed blijken te werken. Ook laat het zien dat het gemakkelijk is om met verschillende van deze heterogene databronnen te werken. Met de integratie ontwikkeld in dit onderzoek kan vervolgonderzoek zich volledig richten op de volgende stappen in het effectief inzetten van de synergie tussen verschillende databronnen binnen de gebouwde omgeving.

Maatschappelijke bijdrage

Een van de meest essentiële zaken in de integratie van sensor- en gebouwdatastromen is het overtuigen van bedrijven om mee te gaan in het überhaupt verzamelen van deze data. Duidelijke business cases zijn nodig om zulke projecten in gang te zetten, ook bij het Radboudumc. Het maatschappelijke belang van dit onderzoek focust zich op de mogelijkheden dat dergelijke data integratie mee kan brengen. Het laat zien dat het combineren van de onderzochte heterogene datastromen mogelijk is, dat analyses hiermee uitgevoerd kunnen worden en dat toepasbare inzichten relatief makkelijk te verkrijgen zijn.

Conclusie

De integratie ontwikkeld in dit onderzoek levert een oplossing voor, maar is niet gelimiteerd tot, het voorstellen van toepasbare inzichten voor het verbeteren van het binnenklimaat in gebouwen. Het potentieel van de toepassing van de geïntegreerde datastromen in andere gebieden is nog lang niet bereikt. Toch, aangezien de hoeveelheid van toegepaste IoT devices, onder andere sensoren, sterk stijgt, zal continue gebouwmonitoring en analyses steeds makkelijker gaan. Met de integratie uit dit onderzoek kunnen heterogene databronnen effectief in worden gezet. In plaats van gebouwen als eindproduct te beschouwen, kunnen betrokken partijen gebouwen zien als dynamische objecten die continu verbeterd kunnen worden en waarvan beide geprofiteerd en geleerd kan worden.

Abstract

The need of using data gathered in the full building life cycle is growing. To realise this need, the paradigm of Building Information Modelling (BIM) is increasingly being used. Though, by integrating other technologies with building dataflows, the capabilities of BIM technologies can be extended. One promising integration is sensor data. The integration provides the potential to alter the way various stakeholders interact with the built environment (Dave, Buda, Nurminen, & Främling, 2018b).

This potential synergy between building and sensor dataflows has been researched, but successfully integrating building data with sensor dataflows remains a challenge (Dave et al., 2018; Tang et al., 2019). The current research on the integration of building and sensor dataflows focused almost exclusively on the automation of transmitting sensor data to building models (Tang et al., 2019), often not enabling operational and analysis possibilities with the integrated data.

This research aims to fill this gap by focussing on effectively integrating building and sensor dataflows for analysis purposes. As a case study it will use the integrated data to provide actionable insights in improving the indoor environmental quality (IEQ). Using the linked building data approach, building topology and property graphs are integrated with gbXML for spatial data, and MongoDB is used for integration of sensor- and meteorological time-series data. Using the created dataflow integration, various sources are used to analyse the hourly IEQ and occupancy efficiency of an office building.

These results prove that building data can complement time-series dataflows in achieving more useful results than only portraying the measured sensor values. Using building information, measured sensor values can be placed into context and actionable insights can automatically be derived instead of manually interpreting the sensor results. The integration created in this thesis provides a solution for, but is not limited to, suggesting actionable insights on improving IEQ. The potential of the presented building dataflows to be used in areas such as IEQ or adaptive floor plans has not been realised yet. However, prior to such applications, the data sources must be well organized and integrated, for which this thesis demonstrates an approach.

Keywords
BIM, Building data, Building data integration, IEQ, IoT, Linked Building Data, Ontologies, Semantic modelling, Sensor data

List of Abbreviations

API	Application Programming Interface
ASHRAE	American Society of Heating, Refrigerating and Air-Conditioning
BIM	Building Information Modelling
BOT	Building Topology Ontology
CAD	Computer Aided Design
COLLADA	COLLAborative Design Activity
gbXML	Green Building eXtensible Markup Language
IAQ	Indoor Air Quality
IEQ	Indoor Environmental Quality
IFC	Industry Foundation Classes
IoT	Internet of Things
JSON	JavaScript Object Notation
KPI	Key Performance Indicators
LBD	Linked Building Data
LBDC	Linked Building Data Community
LDWG	Linked Data Working Group
NaN	Not a Number
NoSQL	Not Only SQL
OPM	Ontology for Property Management
OWL	Web Ontology Language
PoR	Program of Requirements
PPD	Predicted Percentage Dissatisfied
PropTech	Property Technology
RDF	Resource Description Framework
REST API	Representational State Transfer Application Programming Interface
RH	Relative Humidity
SOSA	Semantic Sensor Network Ontology
SPARQL	SPARQL Protocol and RDF Query Language
SQL	Structured Query Language
SSN	Semantic Sensor Network
TTL	Turtle (Terse RDF Triple Language)
UMC	University Medical Centre
UML	Unified Modelling Language
URI	Universal Resource Identifier
W3C	World Wide Web Consortium
WKT	Well Known Text
XML	eXtensible Markup Language

List of figures

List of tables

Section 1

Introduction

This section shall start by introducing the issue that shall be tackled in this thesis with its context. After, the problem definition shall be stated as well as the research questions. Finally, a research design is presented which has been followed throughout the thesis development

We are the indoor generation. Approximately 90% of our time is being spent indoors (European Comission, 2003; Klepeis et al., 2001; Velux; YouGov, 2018). Ensuring a good indoor climate is therefore essential, especially as a bad indoor climate strongly affects the productivity, health and wellbeing of the occupants (Jia et al., 2018; Klepeis et al., 2001).

In a world where big data is one of the most important and revolutionary developments in the past half century, it could be argued that the built environment should be able to identify and solve unsatisfactory conditions by collecting usage data in buildings. However, even though Computer-Aided Design (CAD) and Building Information Modelling (BIM) technologies has been a paradigm shift in the design and construction phase of the building lifecycle, the amount of data collected and the intensity of it being used in the built environment industry is significantly lower than in other industries (Loyola, 2018). Not surprisingly, the artificial intelligence developments within the built environment have been far behind on other industries. The built environment is the world's largest industry by asset value but the second lowest on the Morgan Stanley Digitization Index (Navitas Capital, 2019). Still, in the academic world as well as in the industry it is argued that big data and artificial intelligence shall strongly impact the building industry (Bilal et al., 2016; Deutsch, 2015; Navitas Capital, 2019).

But the industry is changing, leading to the need of innovation. Commercial Real Estate (CRE) teams strongly invest in the goal of achieving satisfied tenants. Tenants keep raising their demands, requiring better buildings. These days tenants use their offices to attract and retain employees with talent (Building Engines, 2019). Consequently, the demand for smart buildings has globally doubled every three years in the last decade (Petrullo et al., 2016). Real estate developers anticipate on this trend by distinguishing their buildings through applying new technology (Deloitte, 2018). Following this trend, start-ups entering the property technology (PropTech) market have rapidly accelerated and capital infusion has reached unprecedented amounts (Navitas Capital, 2019).

These technological developments prove that the built environment industry is open to change in all phases of the building lifecycle. However, most of these PropTech applications are focussed on asset management and user experience, leaving the actual performance of buildings behind. For instance, a large gap often exists between calculated and actual energy consumption (Corry, Pauwels, Hu, Keane, & O'Donnell, 2015; H. L. Rasmussen, Jensen, Nielsen, & Kristiansen, 2019). Apart from energy consumption, lack of operational friendliness and poor indoor climate are examples of problems that are receiving more attention lately (H. L. Rasmussen et al., 2019).

These problems often arise not because of faulty calculations but because of changed space usage. In this rapidly changing world, space usage does not stay a constant over the course of the building exploitation phase. Capacity changes are being requested more often, leading to difficulty in achieving high occupation rates. Companies such as SpaceIQ (SpaceIQ, n.d.) or Archibus (Archibus, n.d.) anticipate on this need by providing a clear overview of the building occupancy and assisting in reconfiguration of floor plans.

Building/sensor data integration

In the current big data era, research points out that the need of using data gathered in the exploitation phase of a building during the design and construction of new buildings or redevelopment of existing buildings is growing (Ganisen, Nesan, Mohammad, Mohamed, & Kanniyapan, 2015; H. L. Rasmussen et al., 2019). To realise this need, the paradigm of using BIM-related technology is increasingly being used. However, BIM is not one technology but rather a catch-all term for technologies that manage and create building data. Generally, such 'BIM' technologies are widely used in the design and construction phase, but its adoption in the operations & maintenance and renewal phase is still limited (Bilal et al., 2016; Heaton & Parlikad, 2019). However, these days such technologies are starting to receive specific interest from property managers that desire to share and utilize information in new ways (Carbonari, Corneli, Di Giuda, Ridolfi, & Villa, 2018). Thus, BIM technologies now begin to extend their capabilities over the full building lifecycle, making room for integration of other technologies. One of the most promising interactions is sensor deployment (Dave et al., 2016).

Sensor dataflows

Using a variety of sensors, facility managers and building owners can be relieved of several tasks and can create differentiation throughout the whole building lifecycle (Kejriwal & Mahajan, 2016). Sensor application is being utilized and researched in areas such as lowering building operating costs (Deloitte, 2018; Kirk, 2017), lowering building energy consumption (Deloitte, 2018; EDNA, 2018; García Kerdan, Raslan, Ruyssevelt, & Morillón Gálvez, 2017; K. S. Lee, Han, & Lee, 2016; Mcglinn, Yuce, Wicaksono, Howell, & Rezgui, 2017), predictive maintenance (Deloitte, 2018; Kirk, 2017), predicting occupancy for energy use and demand (Shi, Yu, & Yao, 2017; W. Wang, Chen, & Hong, 2018; W. Wang, Chen, & Song, 2017; Z. Wang & Srinivasan, 2017), predicting building occupancy and occupant behaviour (Birt & Newsham, 2009; Chen, Jiang, & Xie, 2018; D'Oca & Hong, 2014; Saha, Florita, Henze, & Sarkar, 2019; Salimi, Liu, & Hammad, 2019; W. Wang et al., 2017; Zhao, Lasternas, Lam, Yun, & Loftness, 2014) and capturing end-user behaviours to customize building design and experience (Deloitte, 2018; Kirk, 2017).

By constantly monitoring a building on multiple fronts, problems can be pinpointed real time allowing a building to perform during its whole lifecycle (Deloitte, 2018). Consequently, sensor-enabled buildings can strongly contribute to efficiency and quality during the full building lifecycle. However, to realise its full potential in the built environment, the physical and digital world should be blended (Ploennigs, Ba, & Barry, 2018). Without clear integration of sensors in the building dataflows, it is difficult to use sensor data effectively for multiple purposes (Tang et al., 2019).

So, advanced data analytics has proven its potential in every industry. By effectively utilizing the full potential of the data gathered in the current building stock, the building performance can be improved and, consequently, the productivity and wellbeing of occupants as well.

"It is time to allow buildings to have their say." (Frics, Watson, & Management, 2019)

1.1. Problem definition

Several researchers have been exploring this potential synergy between BIM technology and sensor data integration, but it is a relatively new development (Dave et al., 2018a; Tang et al., 2019). The current research on the building and sensor dataflow integration focused almost exclusively on the automation of transmitting sensor data to BIM models (K. M. Chang et al., 2018). Therefore, even though the built environment is considered as the most important sector for sensor-related research (Dave et al., 2018a), failure in effectively integrating building models with sensor data will hinder its possibilities (K. M. Chang et al., 2018; Tang et al., 2019).

Without information integration and management standards, data analytics with large, heterogeneous datasets is costly and time-inefficient (Gerrish et al., 2017). BIM-authoring tools often use common data standards such as IFC, but these are unable to support the more complex datatypes (Loyola, 2018; Patacas et al., 2015). An essential question in the further development of using BIM during the whole building lifecycle, is how to effectively utilize existing open source software for facility management purposes (Tang et al., 2019; J. K. W. Wong, Ge, & He, 2018).

Considering the increased emphasis on improving building performance as well as the potential of combining building and sensor dataflows to provide actionable insights, this research shall focus on effectively combining these dataflows to provide actionable insights in improving the Indoor Environmental Quality (IEQ), making use of existing open source technologies.

1.2. Research questions

To realise this goal, a literature review shall be conducted as well as a case study on a university medical centre (UMC) office building. The following main research question shall be addressed:

"How can building and sensor dataflows be effectively combined to provide actionable insights in improving indoor environmental quality and occupancy efficiency in an existing building stock?"

To answer this main research question, the following sub-research questions shall be addressed:

Literature review

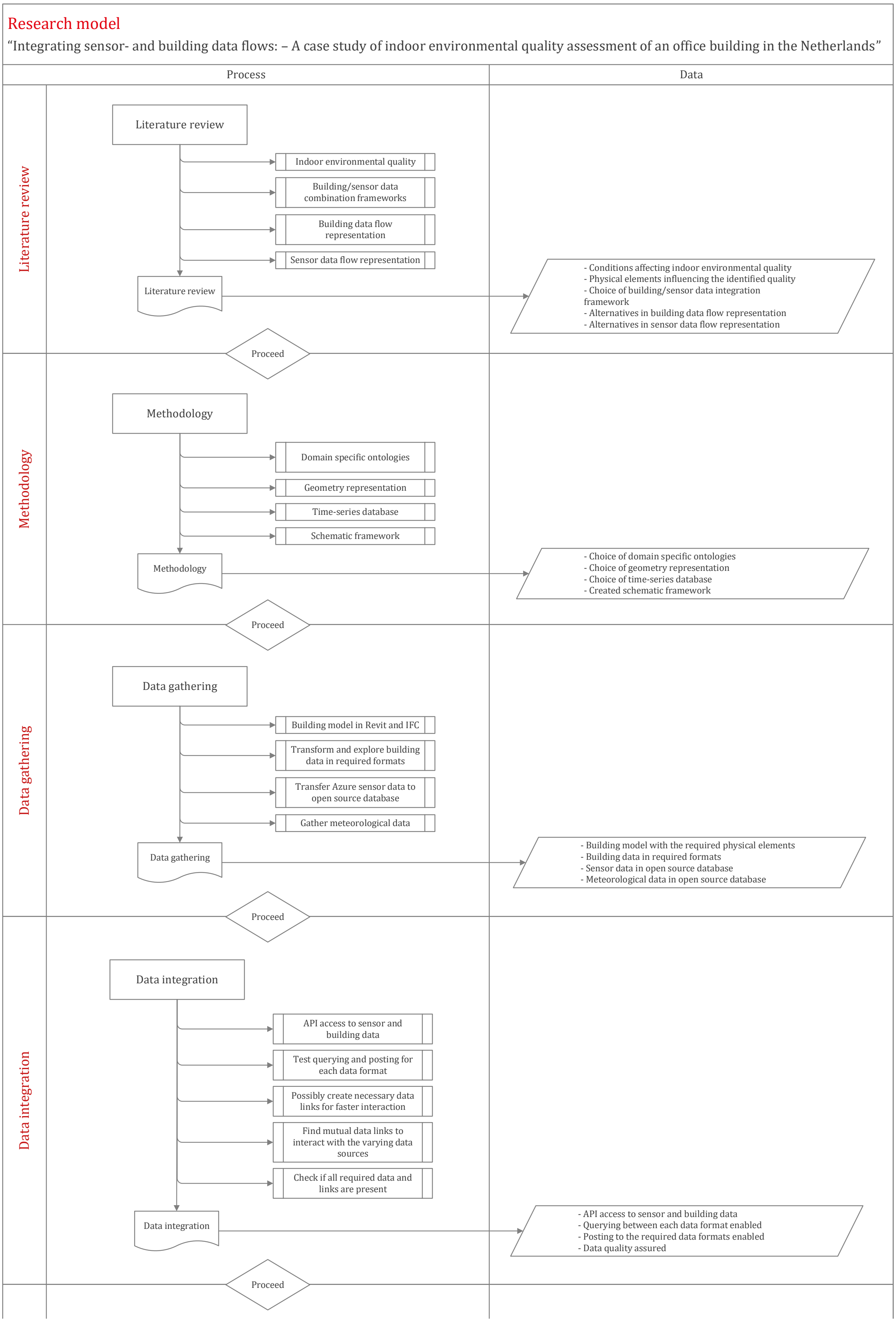
1. *What building/sensor data integration frameworks are currently suggested in literature, how do they differ, and which are most promising for monitoring building efficiency and IEQ conditions?*
2. *How can building dataflows be effectively represented and stored following the chosen integration framework?*
3. *How can sensor dataflows be effectively gathered, stored and linked with the building dataflows, and which technologies are most suitable for this case?*

Radboudumc case study

4. *What form or forms of geometry representation is most suitable for this case, and how do they differ?*
5. *How can the Radboudumc building and sensor dataflows be transformed to the required formats?*
6. *How can the different formats be queried and how does each format perform?*
7. *How can the chosen formats be linked to easily query the data for analytics?*
8. *How can the indoor environmental quality be analysed using the queries?*
9. *How can the analysis results be effectively accessed by interested parties at the Radboudumc?*
10. *Can the created method of combining building and sensor dataflows be seamlessly replicated?*
11. *Using the created integration, what future directions in realising and maintaining healthy buildings should be explored?*

1.3. Research design

This research shall consist of two main parts: a literature review and a case study on a UMC office building. After, a discussion shall assess how the integration fits in the overall road to building/sensor dataflow integration. The schematic research model is shown in Figure 5.


Research model
"Integrating sensor- and building data flows: – A case study of indoor environmental quality assessment of an office building in the Netherlands"
Process
Data
Literature review
Literature review
Indoor environmental quality
Building/sensor data combination frameworks
Building data flow representation
Sensor data flow representation
Literature review
- Conditions affecting indoor environmental quality
- Physical elements influencing the identified quality
- Choice of building/sensor data integration framework
- Alternatives in building data flow representation
- Alternatives in sensor data flow representation
Proceed
Methodology
Methodology
Domain specific ontologies
Geometry representation
Time-series database
Schematic framework
Methodology
- Choice of domain specific ontologies
- Choice of geometry representation
- Choice of time-series database
- Created schematic framework
Proceed
Data gathering
Data gathering
Building model in Revit and IFC
Transform and explore building data in required formats
Transfer Azure sensor data to open source database
Gather meteorological data
Data gathering
- Building model with the required physical elements
- Building data in required formats
- Sensor data in open source database
- Meteorological data in open source database
Proceed
Data integration
Data integration
API access to sensor and building data
Test querying and posting for each data format
Possibly create necessary data links for faster interaction
Find mutual data links to interact with the varying data sources
Check if all required data and links are present
Data integration
- API access to sensor and building data
- Querying between each data format enabled
- Posting to the required data formats enabled
- Data quality assured
Proceed

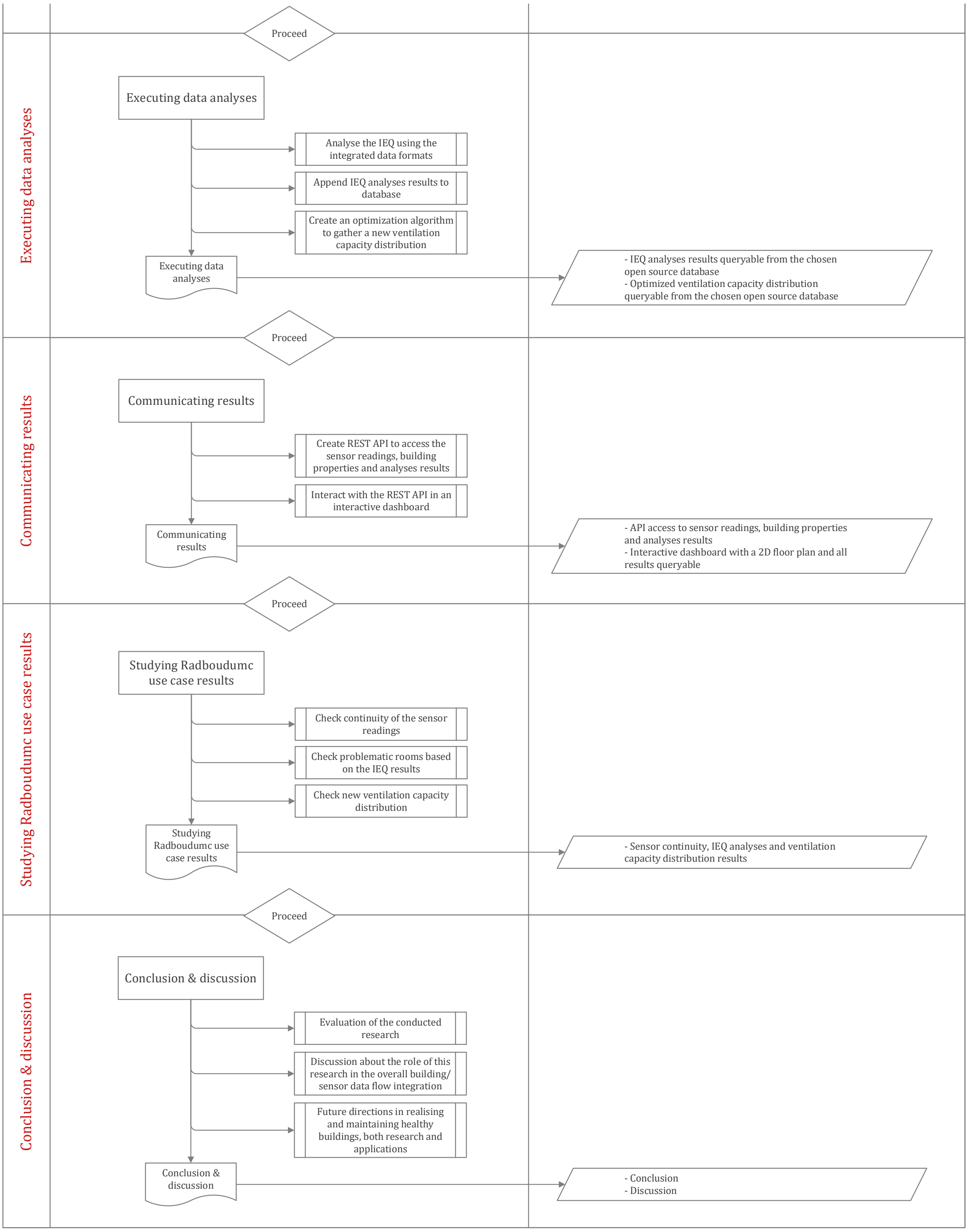


Figure 5: Schematic research model

The first part, the literature review, shall start by assessing the IEQ in (office) buildings. Once these qualities are determined, the physical elements in the building of influence to the IEQ can be identified. As this is not part of the actual integration of building and sensor dataflows, this chapter shall be placed in the appendix. However, the information is required as context to know what parts of building and sensor dataflows are to be combined. After, current framework categories in combining building and sensor dataflows shall be investigated. By assessing the differences between these categories, an argued choice can be made in what framework category shall be chosen for this case. Later, the chosen framework shall be assessed in more detail with all complexities and options.

Once the literary basis has been laid, the case study on the Radboud University Medical Centre can be started. First the literary basis shall be used to transform the collected building and sensor data to the desired formats. The querying should be made possible through an API to prevent local data storage.

Using the prepared data, querying of each data format can be tested. Also, data posting possibilities should be tested for some of the data formats to possibly add new datapoints to this data. Using the (extended) data, links between the formats should be found to use multiple formats in one analysis. Finding and utilizing such links should form the basis of the integration. Once the integration is done, the IEQ analyses should be set up, the results should be gathered, and these should be made accessible through an API. To effectively communicate the results and to show the 2D geometry interaction through geometry data gathered from the building data, a dashboard should be developed. Lastly, combining building and sensor dataflows shall be placed in the road to realising and maintaining healthy buildings. This shall involve future directions in multiple related areas.

Section 2

Literature review

Building- and sensor dataflow integration can be realised in numerous ways. To understand the current state of research a literature review is conducted. Starting on a larger scale, integration frameworks are compared. To reduce the scope of further research, a framework category is chosen. Considering the chosen framework, more detailed research is conducted in the relevant areas.

2.1. Building and sensor dataflow integration frameworks

Integrating building and sensor dataflows can be conducted in numerous ways. To address this challenge, existing frameworks shall first be explored prior to going into more detail.

2.1.1. Framework requirements for building and sensor dataflow integration

The choice of framework category is of importance for not only the case study goals, but for a wider range of future applications as well. Using the combined building/sensor dataflow developed in this thesis, the creation of fully monitored buildings should be supported. Such monitoring should function for a whole (UMC) campus as well, leading to the desire of scalability for larger locations. Simultaneously, the usage of existing open source tools is aimed for to prevent reinventing the wheel. Consequently, the following framework requirements are defined:

1) Is flexible for usage of other datatypes;
2) Is scalable for larger, more complex projects;
3) Is suitable for multi-purpose usage;
4) Updates sensor data automatically real time;
5) Uses existing sensor data query languages;
6) Uses existing building data query languages.

2.1.2. Existing integration frameworks

Extensive literature research on building/sensor dataflow integration frameworks has already been conducted. A recent and extensive example is Tang et al. (2019). They concluded that building/sensor dataflow integration consists of three parts: (1) building context data, (2) time-series data and (3) an integration method. Realising these three parts can be done in multiple ways, of which Tang et al. (2019) identified several main categories of integration frameworks, each having their benefits and limitations. As this review is extensive and recent, these framework categories shall be used in assessing the existing frameworks for this thesis. This chapter shall first argue the most suitable framework category, followed by a more detailed investigation in the chosen framework category. Tang et al. (2019) use the term 'BIM data' to indicate the 'Building Context Data' as shown in their diagrams. As 'BIM' can be considered as a catchall term that entails, among others, authoring and simulation tools, this thesis uses the term 'building dataflows' instead of 'BIM data'.

Tang et al. (2019) differentiated the following framework categories: (1) BIM tools' APIs + relational database, (2) transform building data into relational database using new data schema, (3) create a new query language, (4) semantic web approach and (5) hybrid approach: semantic web + relational database. The benefits and limitations of each framework category shall be compared in this chapter.

BIM tools' APIs + relational database

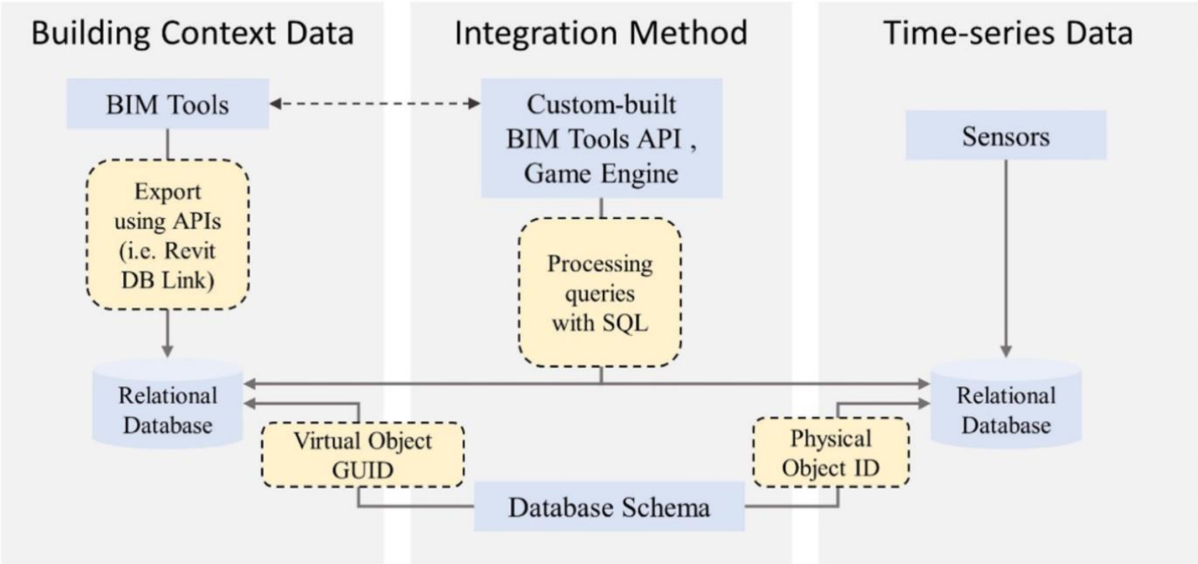


Figure 6: Steps within category 'BIM tools' APIs + relational database' (Tang et al., 2019)

As shown in Figure 6, sensor data is directly translated to a relational database via an API and shall thus be constantly updated. The API might be realised using the sensor provider's solution, or open source API technologies can be used after having downloaded the data. The type of database can be relational, as suggested by Tang et al., but non-relational databases are well suited for time-series data as well. Also, the building model is exported to a relational database via an API. Several BIM tool APIs are available such as Revit DB Link, Dynamo or Grasshopper.

Also, to link the sensors to the building elements, it is suggested to define a new database schema with unique identifications (such as a Global Unique Identifier (GUID). Similarly, Dave et al. (2018) created IfcSensorType objects in the building model to which the sensors are individually linked.

The last step is to realise effective queries between a user interface and the relational databases. Configuring this step strongly depends on the goal of the application. Tang et al. (2019) suggests using a third-party processing engine (such as Unity) and direct queries over the relational database (for object properties).

Evaluation of framework suitability

This framework makes use of existing building and sensor data query languages, namely the stable relational databases. Therefore, the querying of both dataflows and the real time-updating should not prove to be an issue. However, using a tabular structure for building data has strong limitations in the scalability to more complex projects, as well as multi-purpose usage. Building data is generally too complex and heterogeneous to be stored in (relational) tables. Thus, this framework shall not be used.

Transform BIM data into relational database using new data schema

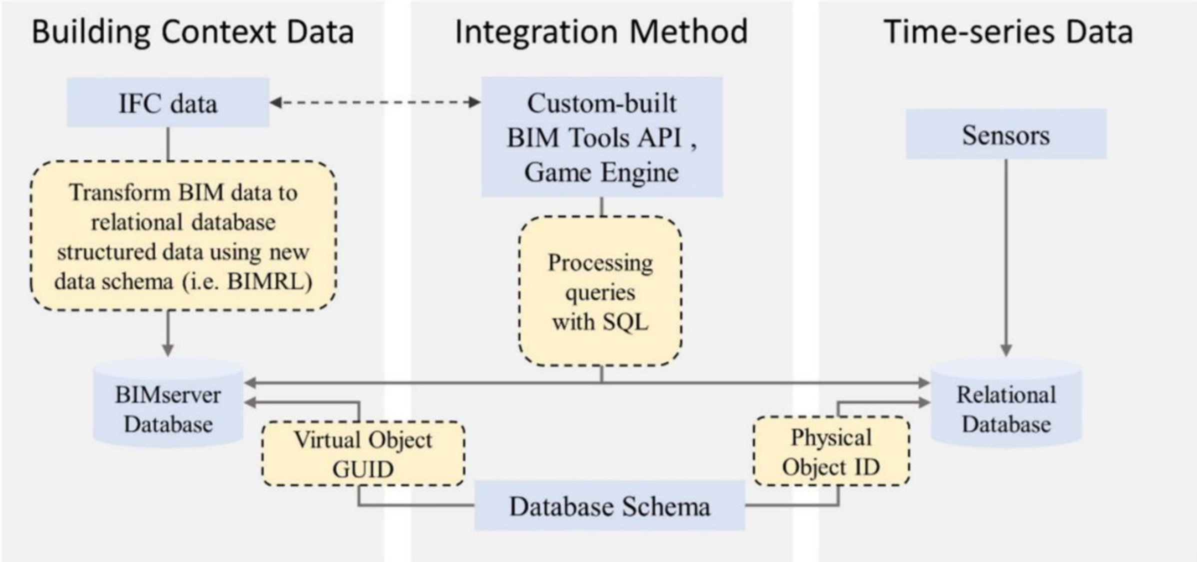


Figure 7: Steps within category 'Transform BIM data into relational database using new data schema' (Tang et al., 2019)

This framework category is similar to the first one, but the storage of building data is different. In this framework category, the building data is transformed to relational database structured data. Database options include BIMserver (open source), making querying from the perspective of different users easier. The building data is made SQL queryable, allowing easy integration of the sensor data as a property of virtual sensor objects.

Evaluation of framework suitability

Using BIM server to store building data is a better option than using tables, however, BIM server uses the IFC format. IFC is a strongly hierarchical data format which is less suitable for making easy changes and for interlinking other datatypes. Additionally, this framework chooses to develop a new building data query language instead of using existing ones. This leads to a less stable and interoperable language instead of reusing widely accepted alternatives.

Create a new query language

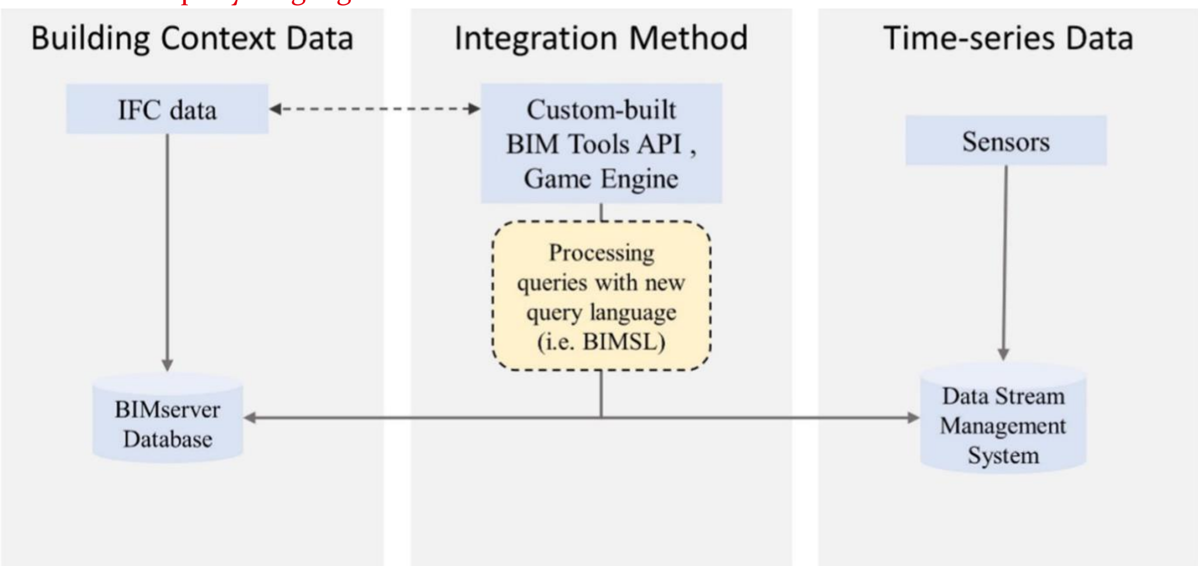


Figure 8: Steps within category 'Create a new query language' (Tang et al., 2019)

Several papers throughout literature introduced a new query language to integrate building and sensor dataflows. An example of this approach is BIMSL (Alves, Carreira, & Costa, 2017). BIMSL gathers data from both a BIMserver Database as well as a Data Stream Management System, as shown in Figure 8. Consequently, it does not use standardized languages such as SQL and SPARQL.

Evaluation of framework suitability

Creating a new query language provides strong flexibility and can be fully customized to the client's situation. However, similar to the second framework, by not reusing languages that have developed over time, one can expect an unstable language that is less reproducible for different situations.

Semantic web approach

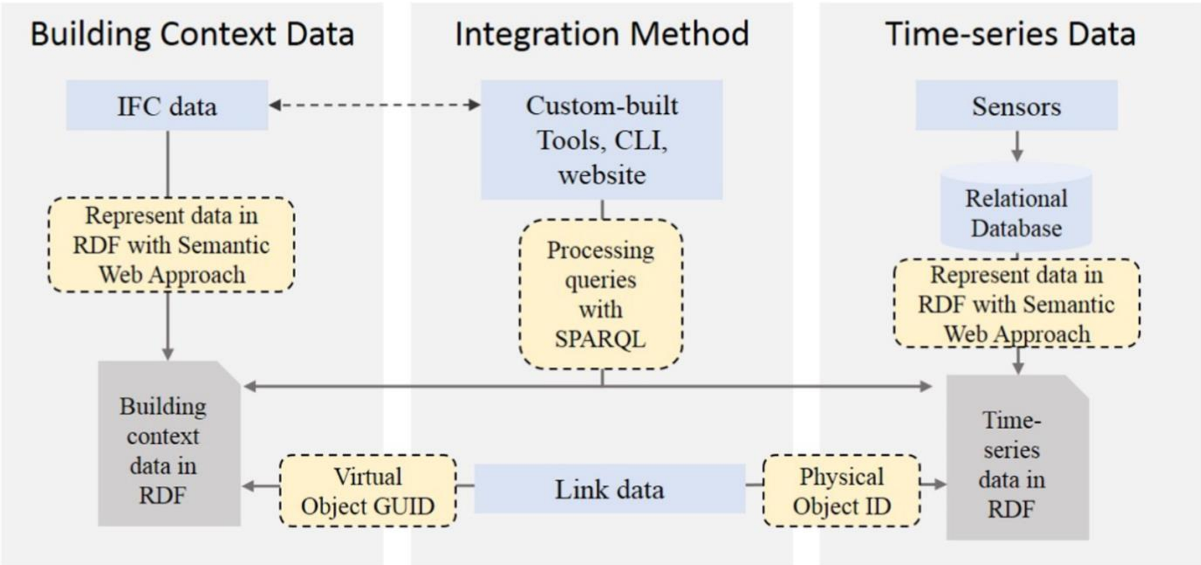


Figure 9: Steps within category 'Semantic web approach' (Tang et al., 2019)

Another possibility is the semantic web approach. In buildings numerous heterogeneous data sets are created that should function together in multiple applications. The semantic web approach can solve problems related to using product-centric data models (for example IFC) in such heterogenous data sets (Hu, Corry, Curry, Turner, & O'Donnell, 2016). By storing data in either one or multiple RDF stores, data can be easily queried using the standardized query language SPARQL.

Evaluation of framework suitability

The semantic web approach has a high flexibility in datatypes, allows multi-purpose usage, and has the possibility of reusing the existing query language SPARQL. However, the semantic web approach is less suitable for storing time-series data, as the semantic capabilities are not useful while still causing large RDF files.

Hybrid approach: semantic web + relational database

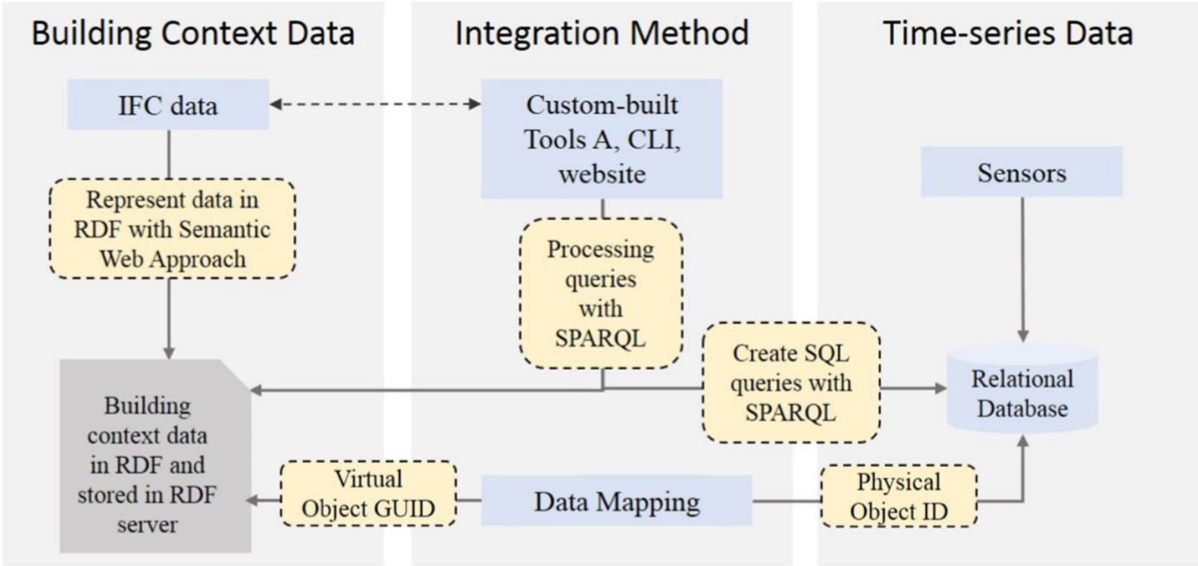


Figure 10: Steps within category 'Hybrid approach: semantic web + relational database'

As the name suggests, this framework category uses both the (relational) database approach from category 1 and 2, while also using the Semantic Web Approach for the building data as used in category 4. The relational database with the time-series data can be queried using SQL with SPARQL.

Evaluation of framework suitability

This framework uses the promising semantic web approach of the previous framework option, while keeping the time-series data in its native format. However, instead of a relational database a non-relational database may be used. The discussion of non-relational and relational databases is ongoing these days with each their own benefits and limitations. It seems too hasty to choose relational databases without investigating its non-relational counterparts.

2.1.3. Choice of existing framework as starting point

Considering the large variety of options in the identified existing frameworks, this literature review shall narrow its scope to one framework. Following from the comparison above, the hybrid approach 'semantic web + relational database' seems to fulfil the requirements best. The hybrid approach has the flexibility of linking numerous data types using the semantic web approach, making it possible to extend the project scope. Additionally, it uses the standard query languages SPARQL and SQL. Lastly, by retaining the sensor time-series data in its native format instead of transforming it to RDF, query performance should be faster.

2.2. Building dataflows

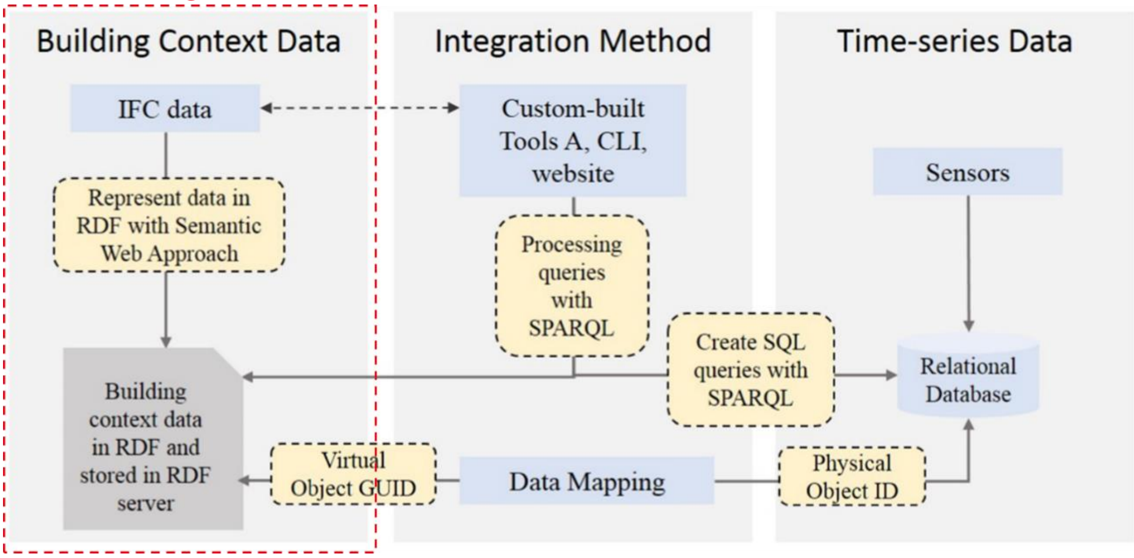


Figure 11: The hybrid integration framework described by Tang et al. (2019) with the building context data highlighted

The first part to elaborate on is the building dataflows, or as Tang et al. (2019) described it 'Building Context Data' (Figure 11). Due to the increasing importance of data integration in the built environment industry, BIM technologies have had a strong adoption throughout the industry and has led to significant advances in BIM data standards. The standard Industry Foundation Classes (IFC) is one of the most widely recognized and well accepted model for building information exchange throughout the built environment industry (Y. C. Lee, Eastman, Solihin, & See, 2016; Mazairac & Beetz, 2013; Pieter Pauwels, Poveda-Villalón, Sicilia, & Euzenat, 2018; Terkaj, Schneider, & Pauwels, 2017). Enabling open data exchange throughout all different disciplines in the built environment industry, as well as different authoring tools, is the main goal of IFC. This data exchange might be both non-geometrical- and geometrical data (McGlinn et al., 2019).

However, despite its wide adoption, acceptance and open data structure, several challenges exist in using IFC. Secondly, IFC lacks capabilities of defining formal semantics, leading to difficulties in querying and reasoning tasks (Pieter Pauwels & Terkaj, 2016). Thirdly, IFC schemas are not easily extendible in a user-friendly manner (Pieter Pauwels & Terkaj, 2016), and the IFC schema is not flexible when combining different data sources required for new situations (Beetz, Van Leeuwen, & De Vries, 2009; P. Pauwels et al., 2011; Pieter Pauwels et al., 2018; Pieter Pauwels & Terkaj, 2016). Lastly, the IFC schema is large and complex – the latest version of the IFC schema (IFC4) contains 1200 classes (Pieter Pauwels & Terkaj, 2016).

Consequently, many relationships and properties within a building that are required in daily processes are difficult to use in IFC models, and, despite its goal of covering the entire built environment industry of information exchange, combining information from related domains is difficult (Zhang, Beetz, & De Vries, 2018). Such data combining includes data from different domains, such as sensor data, and is therefore a cross-industry effort, making developing a robust standard very challenging (Loyola, 2018).

However, IFC format models can be serialized into different general schemas, including XML-based (.ifcXML) and RDF-based (.rdf or .ttl) (McGlinn et al., 2019). Though, any data transformation, including the transformation to and from IFC, can lead to data loss or undesired adjustments. Therefore, any combined dataflow should aim for as little data transformation steps as possible. For example, when an RDF format is desired, it might be advised to export directly from a BIM authoring tool instead of first transforming the data to IFC. This might be an important issue as, using RDF, Semantic Web technologies have often been proposed to represent building data (Beetz et al., 2009; Building & Modeling, 2015; Pieter Pauwels et al., 2018; Pieter Pauwels & Terkaj, 2016). Semantic Web Technologies could overcome the challenges in standard data models related to flexible interlinking of data from different domains and scales, and thereby enabling interoperability among systems and actors (Pieter Pauwels et al., 2018).

2.2.1. Semantic web for building dataflows

After the technological shift to using more BIM authoring tools, it is argued that the next technological shift shall be about exchanging and managing building data over the web (Terkaj et al., 2017). With the technological advancements within buildings, built environment use cases are getting increasingly complex and multi-disciplinary, requiring multi-disciplinary sets of data (Ferguson, Krisnadhi, & Cheatham, 2017). Not all use cases require the full detail of the data. For example, geometry might only be required for visualization purposes in combination with other data (McGlinn et al., 2019).

With the emphasis on the multiplicity of datasets, the increasing interest in semantic web technologies as well as linked data technologies is logical. Using these technologies, a variety of datasets can be integrated with building information represented in different formats (e.g. urban and building models) (Pieter Pauwels, Zhang, & Lee, 2017). Using formal semantics, disparate data can be linked together to draw new conclusions (Ontotext, n.d.), as envisioned by Tim Berners-Lee (Berners-Lee, 2000):

> *"The Semantic Web is the web of connections between different forms of data that allow a machine to do something it wasn't able to do directly." – Tim Berners-Lee*

As a semantic counterpart of the IFC model, ifcOWL has been introduced by the buildingSMART organisation, using Web Ontology Language (OWL) and RDF (Pieter Pauwels & Terkaj, 2016). RDF is a flexible and generic language to show and integrate information from diverse domains (Pieter Pauwels, Zhang, et al., 2017). RDF can evolve schemas without having to transform or reload all the data (Ontotext, 2014). This provides existing systems to easily be extended or adapted to use new semantic structures (Pieter Pauwels, Zhang, et al., 2017).

Using RDF models, standard query languages such as SPARQL can be used (Harris & Seaborne, 2013). This linked data, ontology-based modelling approach is considered highly promising due to, among others, the common language these technologies rely on as well as its apparent global scale deployment (Curry et al., 2013; Pieter Pauwels, 2014; Pieter Pauwels, Zhang, et al., 2017). Ontologies are required to define the organization of building information, but the multitude of ontologies that partly overlap hinder the industry-wide adoption. Universal common agreements on the combination of geometric and non-geometric data lack (McGlinn et al., 2019).

Consequently, using the semantic web in the multi-disciplinary built environment industry shows large potential, but agreement and standards must be found in combining multiple data types. Two main potential solutions for building data using semantic web capabilities are:

1) The one-file approach: ifcOWL, an OWL version of IFC models (Pieter Pauwels, Krijnen, Terkaj, & Beetz, 2017; Pieter Pauwels & Terkaj, 2016);
2) The domain-specific ontologies approach (Terkaj & Pauwels, 2017; Terkaj et al., 2017).

The next two chapters shall elaborate on these two potential semantic web solutions for building data.

2.2.1.1. One-file approach

Transforming an IFC model to an RDF file was firstly introduced by Beetz et al. in 2009 to, among others, enhance querying and reasoning capabilities (Beetz et al., 2009). Using the RDF and OWL capabilities, data processing in existing algorithms would be easier than using the IFC format (Zhang et al., 2018). The most noteworthy examples of representing IFC models in RDF are ifcOWL (Pieter Pauwels & Terkaj, 2016) and ifcWoD (Farias & Nicolle, 2015).

Adopting a standardized method throughout the industry is the first step in collectively tackling the interoperability problems using the semantic web (Pieter Pauwels, Zhang, et al., 2017). Throughout literature, ifcOWL seems to be the most adopted method for further development (McGlinn et al., 2019; Terkaj & Pauwels, 2017). Therefore, the Linked Data Working Group (LDWG), initiated by BuildingSMART International, is making efforts to formalize a standard ifcOWL ontology (Pieter Pauwels & Terkaj, 2016). Doing this, the EXPRESS schema is still considered the 'master' data model, as shown in Figure 12 (Pieter Pauwels, Zhang, et al., 2017). As IFC files are becoming the standard of exchanging building data during the design and build process, this should not prove to be an issue. To convert IFC to RDF using ifcOWL, a tool has been developed[1] that transforms an IFC model to an RDF Abox graph, following the ifcOWL structure (Bonduel, Oraskari, Pauwels, Vergauwen, & Klein, 2018).

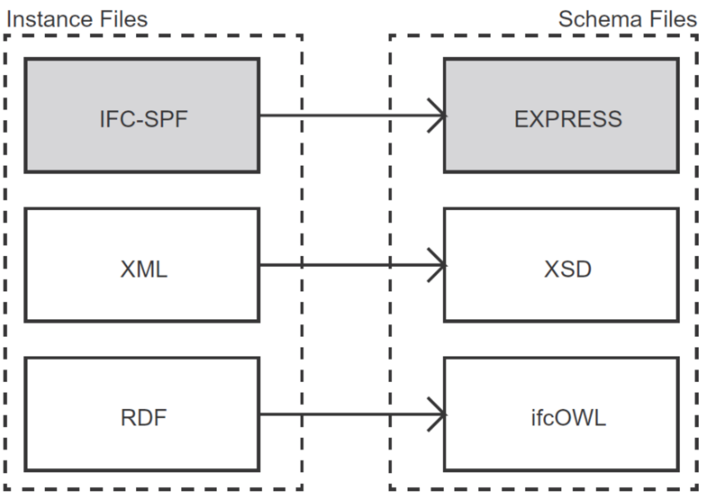


Figure 12 (Pieter Pauwels, Zhang, et al., 2017)*: Instance files with associated schema files*

Limitations of the one-file approach

Despite the indicated benefits, using an RDF file instead of an IFC file still has its limitations. For example, as the ifcOWL schema aimed to be equivalent to the IFC EXPRESS schema with backwards compatibility (Bonduel et al., 2018), the ifcOWL still resembles more to the IFC EXPRESS and STEP concepts rather than the concepts of the semantic web (McGlinn et al., 2019; P. Pauwels & Roxin, 2016). Even though this was one of the criteria in Pieter Pauwels & Terkaj (2016), this does lead to some challenges, as noted by Pauwels & Roxin (2016):

1) Several semantic constructs of the EXPRESS schema (such as LIST data types) are adopted in OWL and RDF, leading to complex and unintuitive constructs (Farias & Nicolle, 2015; P. Pauwels & Roxin, 2016; Pieter Pauwels, Krijnen, & Beetz, 2015);

[1] https://github.com/jyrkioraskari/IFCtoRDF-Desktop

2) The ABox instance graphs, as resulting from the aforementioned conversion tool, do not reduce the complexity and size of the IFC models, possibly even enlarging them (P. Pauwels & Roxin, 2016).

Consequently, an alternative approach is desired. One promising alternative, related to the one-file approach, is the domain-specific ontologies approach.

2.2.1.2. Domain-specific ontologies approach

Instead of having a generic ontology, such as the one-file approach described in chapter 2.2.1.1, domain-specific ontologies enable the decentralized combining of data and standards (Pieter Pauwels, Zhang, et al., 2017). These domain-specific ontologies/modular approach have several benefits over the one-file approach, including:

1) It allows domain specialists to maintain and enhance ontologies for their specific demands (McGlinn et al., 2019; M. H. Rasmussen, Pauwels, Hviid, & Karlshoj, 2017);
2) It enables specialists in other industries to achieve an improved interface to the AEC industry (M. H. Rasmussen et al., 2017);
3) It improves the usability of the subsets of the full ontology (Hutchison & Mitchell, 2009), and reduces overlapping (Terkaj & Pauwels, 2017);
4) It improves the scalability and reduces the complexity (Hutchison & Mitchell, 2009);
5) It eases the sharing of cross-domain information of a building (Curry et al., 2013);
6) It makes it easier to specify and respect the legal responsibilities and Intellectual Property Rights (IPR) surrounding objects (M. H. Rasmussen et al., 2017).

Linking the separate but structured datasets, data must be either represented in RDF or linked to it. Using Uniform Resource Identifiers (URIs) in combination with RDF files, data can be instantly represented on the web of data (McGlinn et al., 2019). Still, standardization and consensus should be achieved among the ontologies to realise industry wide adoption (Terkaj et al., 2017).

Similar to formalizing a standard ifcOWL ontology, the aforementioned LDWG aims to achieve this standardization and consensus on ontologies using a few deliverables (Terkaj et al., 2017). It does so together with the Linked Building Data (LBD) group of the World Wide Web Consortium (W3C), forming the Linked Building Data Community (LBDC) (Terkaj et al., 2017).

Foundation ontology

Central in these deliverables is a foundation (or core) ontology that contains the topology of the building. The LBDC promotes the Building Topology Ontology (BOT) as foundation ontology (M. H. Rasmussen et al., 2017). Apart from BOT, three additional ontologies are defined: a PRODUCT, GEOM and PSET ontology, representing product data, geometric representations and properties respectively (Terkaj et al., 2017). Having these four ontologies as the centre of the linked data approach for building data, adjacent domains can be added, such as SAREF, SSN, SOSA, DogOnt and geospatial ontologies.

Several other foundation ontologies have been proposed in literature. For example, the BIM Shared Ontology (BIMSO) was proposed by Niknam & Karshenas (2017), representing a foundation ontology to which BIM Design Ontologies (BIMDO) can be linked. The BIMDOs express the properties of building elements. The idea behind BIMSO seems similar to the idea behind BOT, as defined by the LBDC.

The BOT aims to reduce the current redundancy in the building dataflows and is extensible with the other LBD ontologies, as indicated in Figure 13. The BOT should only define the core building topology using physical and conceptual objects as well as their relations, and limits itself to these concepts (M. H. Rasmussen et al., 2017):

- A building is divided into storeys and spaces;
- A space may be bounded by building elements;
- A space may contain building elements.

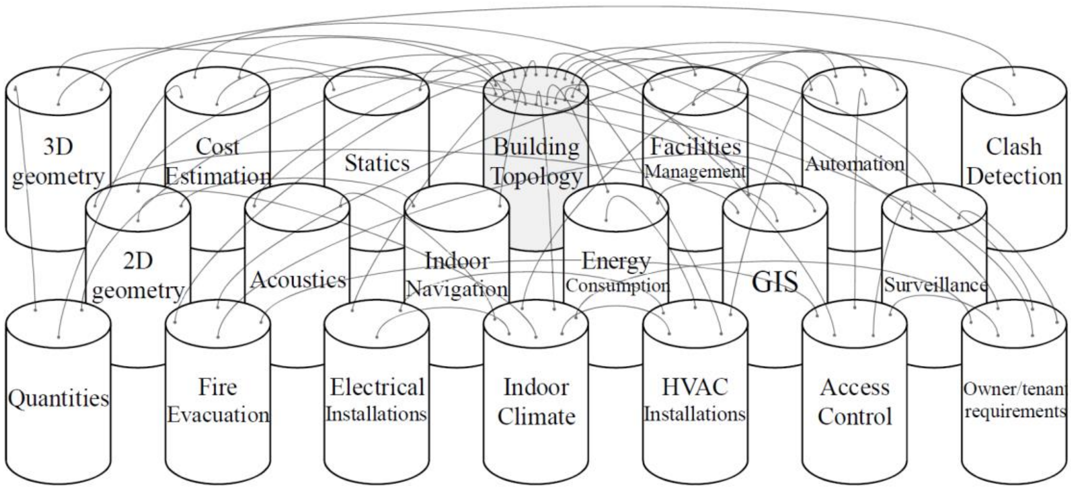


Figure 13 (M. H. Rasmussen et al., 2017)*: Indication of the links between BOT and related ontologies*

The BOT ontology is available on the Github page[2] of W3C Linked Building Data Community Group. As shall be shown in chapter 4.1.2.2, an IFC to LBD converter can be used to transform IFC to a BOT graph.

Product data ontology

Several ontologies exist to represent building product data of which two potential alternatives are considered here. The first is the aforementioned PRODUCT ontology[3] defined by the W3C LBDC. This ontology can seamlessly link a specific product to a BOT element, as shown in Figure 14. It allows to make simple aggregations between building products (Wagner & Rüppel, 2019).

---

[2] https://github.com/w3c-lbd-cg/bot

[3] https://github.com/w3c-lbd-cg/product

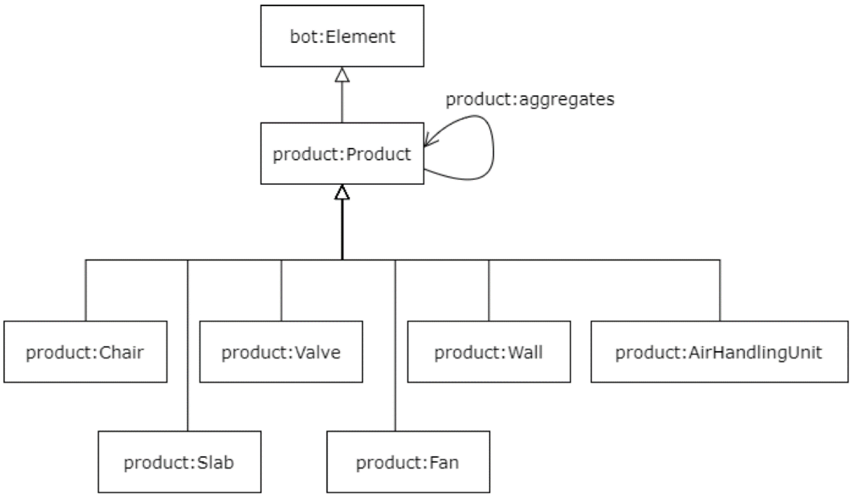


Figure 14 (Pieter Pauwels, 2017)*: Example of linking the product ontology to a bot:element*

An alternative is the Building Product Ontology (BPO)[4], which builds forth on the SolConPro ontology. SolConPro is an ontology built for, and limited to, multi-functional façade elements (Wagner & Rüppel, 2019). As BPO builds forth on the SolConPro ontology, it re-uses some of the concepts, but enhanced them with new axioms and properties (Wagner & Rüppel, 2019). BPO is not limited to multi-functional façade elements but only describes the schematic description of products, without material compositions and geometry (Wagner, Moeller, Leifgen, & Eller, 2019).

Property data ontology

Similar to the ontologies for building topology and product data, the W3C LBDC aims to standardize the property data ontology (Bonduel, 2018). To realise this goal, they created the PROPS ontology and the Ontology for Property Management (OPM). This chapter shall first elaborate on PROPS and OPM, after which it shall elaborate on a related ontology as well: Smart Energy Aware Systems (SEAS).

PROPS and Ontology for Property Management (OPM)

Prior to OPM, the W3C LBDC created the PROPS ontology. However, PROPS was not aligned to commonly used vocabularies and was unable to add to property meta data (Wagner & Rüppel, 2019). Additionally, as described by M. H. Rasmussen, Lefrançois, Bonduel, Hviid, & Karlshø (2018), after having had several interviews with the AEC industry experts, several competency questions were identified. Therefore, to overcome these limitations and to fulfil the identified competency questions, OPM was introduced as an extension to PROPS.

Unlike PROPS, OPM defines three levels of complexity: (Level 1) no objectification, (Level 2) one-time objectification and (Level 3) double objectification. Level 1 does not allow to provide extra information about a property as well as grouping of properties, which is both possible from Level 2 onwards. Level 3 allows versioning of properties which was one of the main goals of introducing OPM, as argued in M. H. Rasmussen, Lefrançois, et al. (2018). Versioning could for example be used to report the as planned, as designed, as realised properties. PROPS works similar to OPM but only contains Level 1.

[4] https://www.projekt-scope.de/ontologies/bpo/

M. H. Rasmussen, Lefrançois, et al. (2018) describe OPM as an extension of the SEAS ontology, because it follows the evolution of properties over time. OPM would therefore be applicable in any other domain in which properties change over time.

Smart Energy Aware Systems (SEAS)

As OPM is considered an extension of the SEAS ontology, this ontology shall be briefly elaborated on as well. The SEAS ontology describes physical systems and their interrelations, with mainly the modules *seas:FeatureOfInterestOntology* and *seas:EvaluationOntology* related to property management (M. H. Rasmussen, Lefrançois, et al., 2018). The *seas:FeatureOfInterestOntology* module is borrowed from the core concepts of SSN/SOSA (Lefrançois, 2017). A feature of interest is for example 'light', and the *seas:Property* is the observable quantity. According to Lefrançois (2017), *seas:Property* can be linked to *ssn:Property*, and thereby the measured sensor values.

Additionally, using the seas:hasProperty predicate and the seas:Evaluation class, varying evaluations can be assigned to a property of a product. Describing the changing evaluations is strongly related to the versioning functionality of OPM.

2.2.1.3. Geometric representation ontology

Geometric representation is central in building data models. However, most geometric descriptions rely on aggregate data and ordered lists (Pieter Pauwels, Krijnen, et al., 2017; Wagner, n.d.). For example, the LIST data types are used a lot in IFC files. Though, ordered lists in RDF are complex to describe and lead to an inefficient representation (Wagner, n.d.). Additionally, geometric information contains little semantic meaning, leading to the question if describing geometry in non-RDF format might be a better option (Pieter Pauwels, Zhang, et al., 2017). The latter leads to the first of two options in simplifying RDF graphs according to Pieter Pauwels, Krijnen, et al. (2017): (1) remove the geometry from RDF representation or (2) reduce the size and complexity of RDF geometry representation.

The first option would be logical, especially considering that geometric representations often directly flow to interpreter programs which have their own semantics. However, ifcOWL aims to be the reference standard in sync with IFC and its EXPRESS schema, in which geometry plays a large role (Pieter Pauwels, Krijnen, et al., 2017). Pauwels & Roxin (2016) propose to remove the geometry depending on the use case.

To realise the second option, reducing the size and complexity of RDF geometry representation, a more efficient representation must be found than the current lists in RDF. As argued by McGlinn et al. (2019), the linked data approach allows to use multiple geometry representations, eliminating the need to define one geometrical representation method. This way, the geometrical representation can be based on the use case, as proposed by P. Pauwels & Roxin (2016).

In this thesis geometry is required for two purposes: (1) assess several IEQ-related influences and (2) visualization (both 2D and 3D). The former requires semantically usable geometry, while the second does not. This chapter shall evaluate several geometry options for both purposes.

Semantically usable geometry

To realise semantically usable geometry, multiple options were found throughout literature. The most notable are GEOM, WKT and gbXML. These options can be easily used for reasoning purposes but are less suitable for visualisation purposes.

GEOM

The GEOM ontology[5] only describes geometric representations. Therefore, the user is free to choose among all required non-geometric ontologies, following the linked data approach. GEOM defines the shapes as instances, without having any direct information on absolute coordinates. This is one of the main differences between GEOM and alternative ontologies such as WKT, gbXML, etc., that focus on direct geometry with absolute coordinates (McGlinn et al., 2019). The GEOM ontology represents geometry with a rich taxonomy of concepts for representing geometric entities mathematically with numerous transformation matrices and Boolean operations. However, this causes the GEOM ontology to overload an RDF graph (McGlinn et al., 2019), which is the opposite of what the linked data approach aims to do.

Green Building XML (gbXML)

Green Building XML was originally developed by a small consulting firm GeoPraxis for data that was used in energy analysis software (An, 2017). These days it is mostly used for Autodesk Green Building Studio, but it is still open sourced. Similar to IFC, it contains component tagging such as building, space, surface, material, etc., but the relational structure is not the same as in IFC (Ferguson et al., 2017). Also, like IFC, it allows building analysis software to mutually share information (gbXML, n.d.). Having the XML structure, gbXML is easy to conduct analyses with and to link to the other ontologies, such as BOT and PROPS data.

Well-Known Text (WKT)

WKT provides possibilities of representing both 2D and 3D geometry. WKT can represent geometry both semantically meaningful as well as efficiently with a short representation (Pieter Pauwels, Krijnen, et al., 2017). WKT is human-readable, efficient and the semantics related to spatial reasoning are well understood (Pieter Pauwels, Krijnen, et al., 2017). WKT contains equivalents for the higher order geometrical representations, making mapping of IFC to WKT a suitable option (Pieter Pauwels, Krijnen, et al., 2017). For example, IfcPolyline can be represented as a LINESTRING WKT.

Using GeoSPARQL, 2D WKT literals can be easily queried from a RDF graph (McGlinn et al., 2019). However, representing 3D geometries remains more of a challenge as GeoSPARQL does not support 3D geometries. WKT 3D geometric representation can be linked to the BOT ontology though, as BOT can encode a 3D model as a datatype property (bot:hasSimple3DModel) or object property (bot:has3DModel) (M. H. Rasmussen, Frausing, Hviid, & Karlshoj, 2018).

Building visualisation

Using tessellated geometry is suitable for 3D building visualisation but do not have the rich building semantics such as IFC does (McGlinn et al., 2019). Several options exist regarding tessellated geometry for building visualisation: (1) linking each object to a geometric representation in a standard such as COLLADA or (2) link the building to a geometry representation that is only for visualisation purposes and contains no additional semantics, for example using a Wavefront OBJ file. The former approach has the limitation of having an overhead as each component has its own set of meta data. Additionally, the files can be quite verbose even for simple representations. The OBJ files in the latter approach are of a simple format that only represents the 3D geometry.

[5] http://rdf.bg/geometry.ttl

IFC files can contain detailed geometry which can, among others, be used for mesh geometry representation. However, to keep the query efficient, it is advised to remove as much information as possible which is not relevant for the mesh geometry. Only mesh geometry and a link to the graph elements are needed.

Another option containing only this information is COLLADA. COLLADA is a xml-based file format that contains building element meshes. As it is xml-based, queries can be easily executed. COLLADA files contain geometry elements which in turn contain the meshes. Each geometry element contains a unique id of which the last characters represent the bat ID that can also be found in the graphs. So, considering the easy query possibilities and the mutual identifiers, also COLLADA is a well-suited file format for mesh geometry integration.

Ontologies for uniform geometry linking

Using multiple geometric representations and experimenting with its integration in the other ontologies, a linking ontology is required. Two most notable alternatives exist: (1) Ontology for Managing Geometry (OMG) and (2) File Ontology for Geometry formats (FOG).

Using OMG, geometry descriptions can be attached to the corresponding building objects (Wagner, Bonduel, & Pauwels, n.d.). FOG extends OMG but provides geometry-schema-specific relations between geometry descriptions and the corresponding building objects (Bonduel, Wagner, & Pauwels, 2019; Wagner, n.d.).

2.3. Sensor data

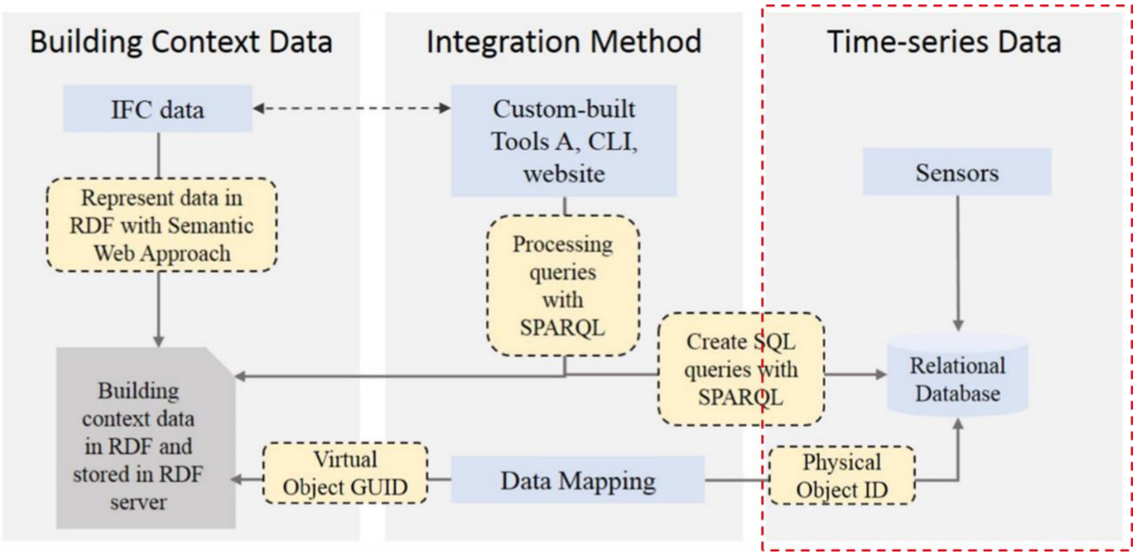


Figure 15: The hybrid integration framework described by Tang et al. (2019) with the time-series data highlighted

Having covered the building data, including the usage of semantic modelling, the next part to be explored is sensor data (or time-series data in general) (Figure 15). Using sensors in, among others, buildings is a fast-growing market. Efficient data storage and management of the non-structured data generated by the sensors is therefore one of the main challenges surrounding this market (Braulio, Moreno, De MacEdo, Kreutz, & Dantas, 2018). This chapter shall first explore the semantic modelling possibilities surrounding sensor data, especially for creating a combined dataflow with the building data. After, representation of sensor data in a variety of databases shall be explored.

2.3.1. Semantic modelling for sensor data

As concluded in chapter 2.1.3, sensor (time-series) data should not be stored in graphs, but instead kept in their native format. Even though transforming time-series data to RDF would enable easy linking to other RDF data silos, it will cause several problems as well, as argued by (Bornea et al., 2013):

1) By transforming it to RDF, data is duplicated, causing redundancy and thereby inefficiencies;
2) RDF is not made to store fixed-structured data, leading to unnecessarily large files;
3) Execution of functions such as lookups are less efficient in triple-stores as they would be in their native format, for example in relational databases.

Nevertheless, interconnectivity, such as that realised by the semantic web, is still essential to the vision of IoT (and thereby sensors): creating a 'dynamic global network of infrastructure with self-configuring capabilities' (Dave et al., 2018b). To increase the interconnectivity, IoT research started focussing on reducing the amount of vertical siloes and increasing the system interconnectivity within all IoT devices and platforms, such as through universal messaging standards (Framling, Kubler, & Buda, 2014) and flexible APIs (Briggs, 2015). Without such standardization, it is likely that several schemes shall be developed separately and each for its own use, leading to IoT fragmentation, the opposite of the IoT vision (Dave et al., 2018b).

Using ontologies to enable easy linking of sensor data

Aiming to fulfil the original IoT vision for the application on buildings, sensors need a shared communication method with the building data. A possibility is to use the linked data approach to formally specify the sensor data. Ontologies have been developed to realise this, among others the Semantic Sensor Network (SSN) ontology developed by the W3C (Compton et al., 2012). Using the SSN, it is possible to describe sensors, their accuracy and capabilities, observations and methods used for sensing (Compton et al., 2012). The SSN also includes the Sensor, Observation, Sample, and Actuator (SOSA) module, allowing modelling of the sensor, actuator, observations, observable properties and results (Terkaj et al., 2017). Elaboration on SSN and SOSA can be found on the W3C SSN page[6]. Other ontologies for describing and linking sensor data have been developed. However, as it is already decided that sensor data shall be kept in their native format, it is not relevant to discuss more options of semantic modelling for sensor data.

Representing sensor data in a database

Choosing a database querying method used to be straightforward, use Structured Query Language (SQL). Using SQL was the standard and the data was generally tables with columns and relations among the tables (Flynn, 2015). These days there are two extremes which have gotten an increased amount of attention in the past few years: full SQL support and fully custom languages (NoSQL) (Arye, 2018). SQL databases are still a great solution for numerous situations, but these days types of data often vary strongly and, consequently, database technologies have changed (Flynn, 2015). Even though nonrelational databases have existed for a long time, these changes in data types and new technological developments allow the NoSQL databases to finally gain market traction (Leavitt, 2010). Leavitt (2010) therefore argues that relational databases shall not be fully replaced by non-relational databases anytime soon, but its usage shall decrease.

[6] https://www.w3.org/TR/vocab-ssn/

This chapter shall briefly describe SQL and NoSQL, as well as elaborate on their benefits and limitations. The Methodology section shall use this information to argue what type to use in this thesis.

2.3.1.1. Relational (SQL) databases

Traditional relational databases have fixed table structures and use Structured Query Language (SQL) to select data from these tables. Relational databases make it possible to easily select data originating from different tables (Van Der Veen, Van Der Waaij, & Meijer, 2012). SQL has provided a widely adapted standardized database management language. Using a standardized method is valuable for two main reasons: code portability and skills re-use (Arye, 2018). The SQL standard could be extended for new or improved functionalities, reducing the code portability but keeping the standardized language, allowing to leverage pre-existing knowledge of the language (Arye, 2018). Consequently, some clear pro's and con's surrounding the use of SQL exist, which shall be briefly listed here:

Pro's of relational databases

- It was developed a long time ago, creating a well-developed, consistent and safe language (Vongsingthong & Smanchat, 2015);
- Developers are generally familiar with SQL (Hsieh & Perry, 2019; Van Der Veen et al., 2012);
- It is stable and safe (Hsieh & Perry, 2019; Van Der Veen et al., 2012; Vongsingthong & Smanchat, 2015);
- Good for structured data (Agarwal & Rajan, 2016);
- Easy to work with (Hsieh & Perry, 2019);
- It is well integrated/compatible with many existing systems (Hsieh & Perry, 2019; Vongsingthong & Smanchat, 2015);
- Update incorrect information (Hsieh & Perry, 2019).

Con's of relational databases

- Joining tables across distributed systems is difficult (Agarwal & Rajan, 2016; Hsieh & Perry, 2019; Leavitt, 2010; Van Der Veen et al., 2012);
- Database structures can be complex and difficult (Leavitt, 2010);
- Relational databases have a big feature set of which usually only a few are needed, leading to unnecessary costs and complexity (Leavitt, 2010);
- It was invented in the 1970s (Dix, 2018).

The latter may seem strange but is often considered a disadvantage as people do not want 'outdated' technologies. Since the 1970s SQL has strongly evolved but the essence is still the same, causing it to be considered traditional and old, even though non-relational databases exist since the late 1960s as well (Leavitt, 2010).

Relational database options

PostgreSQL

PostgreSQL is a widely diffused open source SQL database. It has flexible query capabilities and a high read performance (Van Der Veen et al., 2012). Its writes are supposedly slower. PostgreSQL has been around for a long time and has placed itself in the centre of relational databases. Partly due to the ageing of the database, it is highly stable and has a large feature set (Hsieh & Perry, 2019).

TimescaleDB

TimescaleDB is a SQL database option that is built on top of PostgreSQL. TimescaleDB provides all features required for time-series data using the stable PostgreSQL as basis. TimescaleDB thereby supports SQL and has a good PostgreSQL integration (Solnichkin, 2018). It uses hypertables to store data in chunks of time, called buckets (Pifferi, 2018). However, there seems to be a downside to using hypertables: it does not support any foreign key constraint on itself. Therefore, referential integrity is strongly reduced which is one of the main benefits of using SQL. Some argue that this is less relevant for time-series data, while others argue that this is essential in their database choice (Pifferi, 2018).

2.3.1.2. Non-relational (NoSQL) databases

NoSQL is an umbrella term used to express all non-relational databases, better described as 'not only SQL' (Agarwal & Rajan, 2016). NoSQL databases do not have to follow the strict data models of relational databases (Y. S. Kang, Park, Rhee, & Lee, 2016). The NoSQL databases have existed for decades but they recently gained popularity during the growth of the Web 2.0 and adoption of companies like Google, Facebook and Amazon (Flynn, 2015). These companies have had changing database needs over time, leading to the necessity of a SQL alternative. This necessity strongly boosted the growth of NoSQL developments and thereby popularity.

There are numerous different NoSQL databases but they are grouped into four categories (Singh, 2019): (1) Key/Value Store (such as Redis or BerkelyDB), (2) Object or Document DBs (such as MongoDB and RavenDB), (3) Graph DBs (semantic web) (such as Neo4j and GraphDB) and (4) Column-oriented DBs (such as HBase and Cassandra). Each NoSQL database product is built for a specific purpose (Y. S. Kang et al., 2016) and serves a different data model (Braulio et al., 2018). Instead of choosing one NoSQL product, the solution might be to choose a set of them (Braulio et al., 2018). Generally, the following pro's and con's are attributed to non-relational databases:

Pro's of non-relational databases

- Generally a higher write and read performance, especially for larger amounts of data (Agarwal & Rajan, 2016; Leavitt, 2010; MongoDB, n.d.; Moniruzzaman & Hossain, 2013; Singh, 2019; Van Der Veen et al., 2012);
- Higher (horizontal) scalability (Y. S. Kang et al., 2016; MongoDB, n.d.; Moniruzzaman & Hossain, 2013; Singh, 2019; Van Der Veen et al., 2012);
- Can deal with unstructured data heterogeneity (Moniruzzaman & Hossain, 2013; Singh, 2019).

Cons of non-relational databases

- Can require manual query programming (Leavitt, 2010);
- Complex query programming can be difficult (Leavitt, 2010);
- Lack robustness of relational databases (Agarwal & Rajan, 2016);
- The absence of schemas often results in later technical difficulties (Hsieh & Perry, 2019);
- Less powerful query language is available (Van Der Veen et al., 2012).

Non-relational database options

Numerous suitable open-source NoSQL options are available. As the main goal of this thesis is to realise effective building and sensor data integration, and as most sensor providers use their own database choices to store sensor data, this thesis shall not conduct an extensive comparison study between a large variety of options. Instead a few main options that seem to be the most suitable shall be briefly compared.

Cassandra
Cassandra is a powerful open source database, ideal for large sensor applications as it was built for horizontal scaling. It was created in 2008 and has strongly diffused among numerous applications since (Van Der Veen et al., 2012). Cassandra uses a structured key-value store and uses Thift for its external API (Lakshman & Malik, 2010). Cassandra seems to be a good choice for relatively big sensor data sets (Y. S. Kang et al., 2016).

MongoDB
Similar to Cassandra, MongoDB is a powerful open source NoSQL database, and is argued to be the best choice for medium-sized non-critical sensor applications (Van Der Veen et al., 2012). It has also been around for some time, since 2007, and runs a key-value store for binary JSON documents (Creative Commons, n.d.).

InfluxDB
InfluxDB is an open source NoSQL database specialized in time-series data. It was created in 2013 and has built up good technical documentation since (Solnichkin, 2018). InfluxDB was built by InfluxData which also provides a visualization tool and a data processing engine that can collect real-time metrics from numerous different sources (Solnichkin, 2018). InfluxDB is strong in collecting data from various locations but not to keep business-critical data (Pifferi, 2018).

ElasticSearch
ElasticSearch is a distributed search and analytics engine (Ramo, Stagni, Tomassetti, & Mathe, n.d.). According to a performance test by Ramo et al. (n.d.), ElasticSearch is faster than among others InfluxDB and is easy to maintain. Using ElasticSearch in combination with for example Kibana, ingesting and analysing multiple datasets should be easy (Barnsteiner, 2015).

Section 3

Methodology

The literature review investigated current research on combining building and sensor dataflows. Using the conducted research, this section shall choose specific ontologies, geometry formats and a time-series database. It shall create a custom framework that will be developed in sections 4 and 5, combining the building and sensor dataflows to assess the building efficiency and IEQ.

When choosing ontology options as well as storage options for time-series data, it is important to keep the purpose of combining the dataflows in mind. In this thesis the purpose is to assess the IEQ within buildings during the exploitation phase and to compare it with the ARBO criteria. To find the necessary dataflows to integrate, Appendix 0 investigated the IEQ aspects and how to assess them.

3.1. Choice of domain specific ontologies

From the literature review it followed that the scope of this thesis shall be limited to the domain-specific ontologies approach for building data. Within this approach several domain specific formats are to be chosen from the options investigated in the literature review.

This thesis aims for standardization and consensus on ontologies in the built environment industry, so it shall focus on the deliverables of the LBDC group. Therefore, it shall not choose the BIMSO approach but build forth on the foundation ontology BOT, as presented in M. H. Rasmussen et al. (2017).

Additionally, the integration requires linkage of products (such as a window) to the general building topology. As this linkage is created using mutual element URI's, this can also be done in the property data ontology. Therefore, the product ontology is not necessary for the integration in this thesis, but it should be considered for possible later applications. As both considered product ontologies contain a product representation, as well as easy integration possibilities with BOT, it is difficult to make this choice based on functionalities. Therefore, the choice shall be based on the position of the ontology in the industry standardization. As the PRODUCT ontology is part of the W3C LBDC, this ontology shall be used in this thesis.

Apart from the foundation and product ontology, a property ontology is required. Even though the functionality of SEAS seems very promising, the focus of SEAS on physical systems and their interrelations seems unnecessary for the integration purposes of this thesis. Also, the different complexity levels of the OPM ontology can be very useful, but it is less relevant to this thesis. Therefore, a simple property ontology is preferred, for which OPM ontology[7] Level 1 seems to be the most suitable. The graphs within complexity Level 1 all have the prefix 'props'. So, from this point onwards, the property ontology shall be referred to as the PROPS ontology.

Lastly, the different geometry types should be identified in the graphs. The literature review described both the FOG and OMG ontology. Even though FOG extends OMG, this thesis shall use OMG to identify the geometry type to reduce unnecessary complexity.

[7] https://github.com/w3c-lbd-cg/opm

3.2. Choice of geometry representation

This thesis looks for a semantically rich geometry representation for analysis purposes, specifically for calculating the mean radiant heat. However, this thesis aims to integrate all building data that might be required for IEQ purposes. Visualizing 3D geometry can, among others, assist in communicating the IEQ results to a user or for interaction possibilities. Even though no mesh visualiser has been created in the dashboard, this thesis aims to anticipate on future developments of building analyses. These future developments might require interacting with a specific element in a viewer. Therefore, it might be desirable to understand the link between certain elements in the viewer and the graphs.

Regarding semantically usable geometry, two important criteria in choosing a format are (1) limited complexity and (2) easy querying possibilities. As gbXML contains all required geometric components, contains mutual datapoints with the PROPS ontology and is XML-based, gbXML shall be used.

Regarding visualisation geometry, integrating mesh geometry requires a mutual identifier in the graphs and the mesh geometry file. As OBJ files do not have a unique element ID to link a tessellated geometry representation to a certain object, the possibilities of OBJ files are limited. Using the original IFC file might provide a solution here, as elaborated on in chapter 2.2.1.3. However, considering the linked building data approach with dedicated purposes per file format, this integration aims to exclude IFC from the integration. COLLADA does have a unique identifier in each element, only contains mesh geometry and is XML-based, enabling easy querying and interaction possibilities. Therefore, this thesis shall consider COLLADA for 3D building visualisation.

3.3. Choice of time-series database

Sensors periodically (e.g. 10 min.) write tiny parts of information to the database, after which analyses and visualizations read large parts of data from the database. Therefore, a database should be good at single writes as well as at multiple reads. MongoDB has the performance and scalability of a non-relational database, and it has been around for some time during which it has developed stability. Additionally, MongoDB has 'MongoDB Compass Community' to easily navigate through the databases.

3.4. Schematic framework resulting from the methodology

This chapter shall conclude the methodology section with a custom framework based on the chosen formats, schematically portraying the dataflows in the overall integration.

3.4.1. Semantic integration layer

To conduct the IEQ analyses, the linked building data sources must be easily queried in a Python script. In literature such integration frameworks/layers are suggested but might be unnecessarily complex. For example, in both Zhang et al. (2018) and Pieter Pauwels, de Farias, et al. (2017) the open source Apache Jena framework and the SPARQL Inferencing Notation (SPIN) API are used. However, using GraphDB to store the Turtle graphs, the built in GraphDB API can be used. This can be done in either GraphDB's SPARQL Endpoint or by locally parsing the whole graph using Python's RDFLib package. These two methods shall be explored in chapter 5.1.1 to interact with the building dataflow. As argued in chapter 3.3, the sensor data shall be stored in a MongoDB NoSQL database. Using the Python package 'pymongo'[8], data can

[8] https://api.mongodb.com/python/current/

be easily queried into the local Python environment. By naming the MongoDB collections after a certain datapoint in the PROPS graph, a link can be easily established.

3.4.2. Linking ontologies

In chapter 3.1, the BOT has been chosen as foundation ontology to which the domain-specific ontologies shall be linked. As the domain-specific ontologies PRODUCT and PROPS are also used, linkage between these ontologies is not difficult. The URI's that are used in each of the graphs are identical, easing the interaction between multiple graphs. Using mutual datapoints from the PROPS graph, a link can be established between the graphs and both the gbXML and COLLADA geometry.

3.4.3. Dataflow diagram

Using the argued methodology as input, Figure 16 shows a Yourdon and Coad dataflow diagram of the entire integration. An enlarged version of the diagram is shown in Appendix A1. Even though most other diagrams in this thesis are of the type Unified Modelling Language (UML), this diagram is not. The dataflow diagram in this chapter should both show the links between the formats, as well as the chronological order of transformation and integration. After consideration of UML alternatives, it was decided that the Yourdon and Coad dataflow diagram fulfils these criteria better. The execution of this integration is discussed in the next sections.

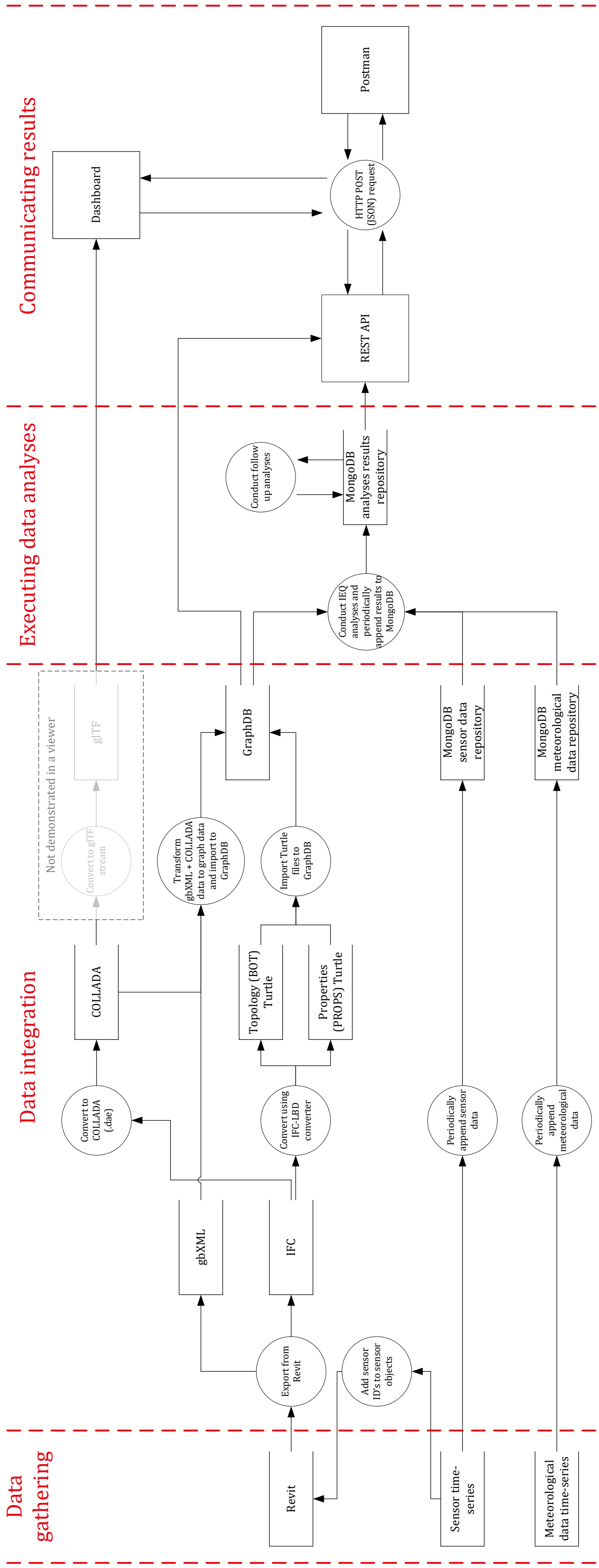


Figure 16: Dataflow diagram of the proposed building/sensor dataflow integration

Section 4

Data gathering

Considering the framework created in section 3, this section shall gather the data required to set up the integration and conduct the IEQ analyses. A building model, sensor data and meteorological time-series data shall be gathered. The building model requires some adjustments to be made which are described in the first chapter of this section. Additionally, the required conversions in these data sources shall be described.

To develop the integration between building dataflows and sensor dataflows, a case study is conducted at the Radboudumc. The Radboudumc is increasingly placing an emphasis on gathering and effectively using data in their operations, among others in their real estate. Two of the initiatives are experimenting with transforming their CAD models to Revit models and conducting a sensor deployment pilot with BeSense. Both initiatives shall be used in this thesis.

4.1. Building model

The created Revit model shall be used for the building dataflow. The Revit model shall be fragmented into several main pieces, each serving a specific purpose.

4.1.1. Adding sensors to the building model

Before exporting the Revit model to IFC, it must be enriched in the Revit environment. Several ways exist to make the link between a sensor and the building model. All sensors with a specific property or naming can automatically be appended to a room, or specific sensors can be drawn in the building model. The latter has been done in this thesis. A sensor can be drawn as an object and attached to another object, for example a table or a wall, making the exact location of the sensor known. A condition must be fulfilled though: the sensor name must be equal to a property of the sensor in Revit. In this thesis the property 'Comments' has been used but any property that can define a string works. Figure 17 shows an example of a sensor that is represented into both the Revit model and in MongoDB.

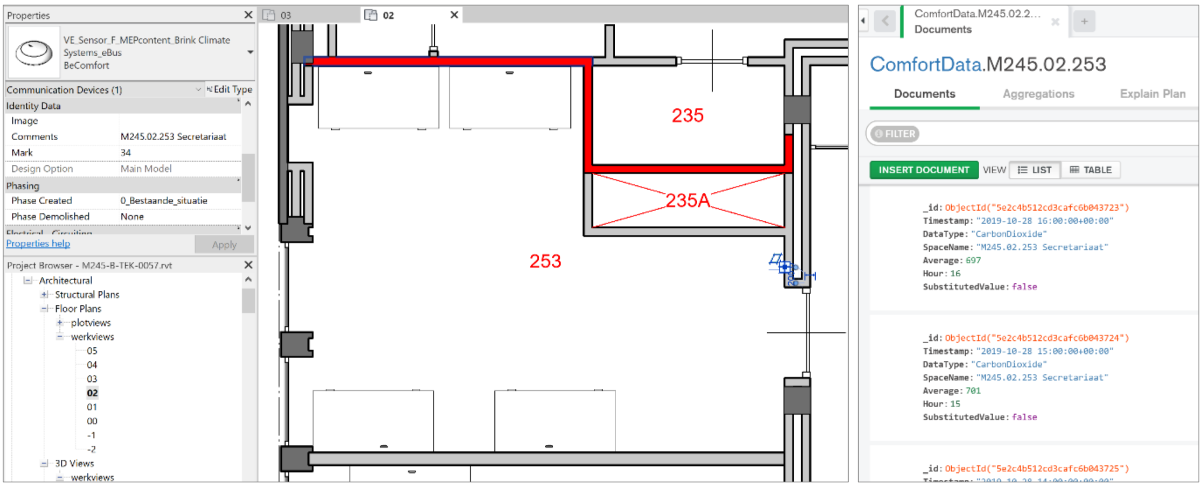


Figure 17: Defining a sensor object in Revit (left) and representing the data of that sensor in MongoDB (right)

4.1.2. Required formats of building data

To fully utilize the building data that a Revit file provides according to the linked building data approach, it must be broken down into several dedicated file formats. Figure 18 represents a tree breakdown structure of how the Revit file is converted to the new file formats. This chapter shall elaborate on how these conversions can be executed.

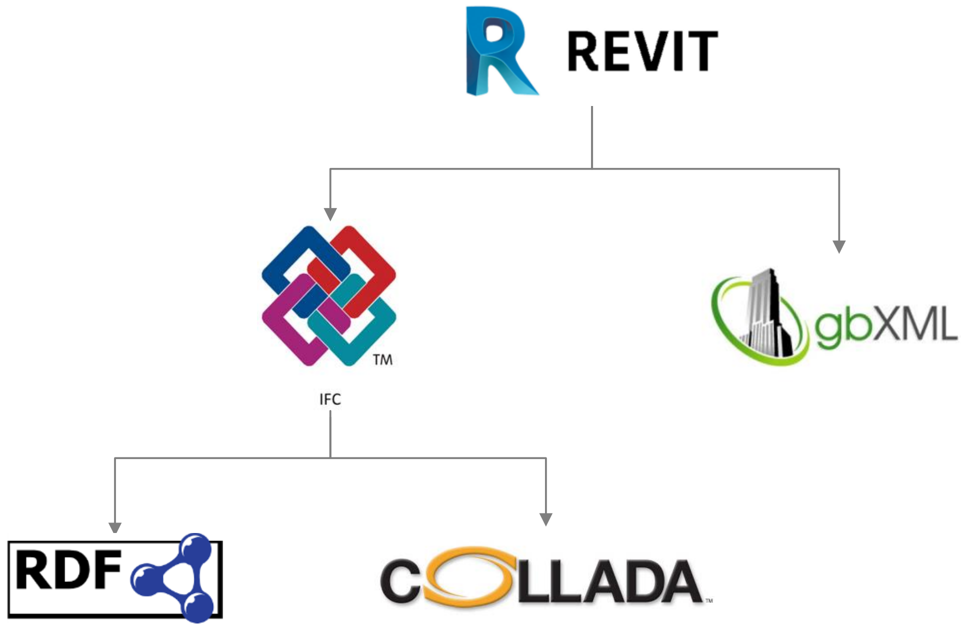


Figure 18: Tree breakdown structure of converting a Revit file to linked building data

4.1.2.1. Converting a Revit file to IFC and gbXML

Revit is a well-developed program. Therefore, it also has a well-developed export functionality, which can convert a Revit file to both IFC and gbXML. Even though exporting can be done with ease, choosing the right settings is essential. When exporting Revit to IFC, the space boundaries option must be set to '2nd Level', and the correct phase to export must be chosen, both in the 'General' tab, as shown in Figure 19. Additionally, in the 'Property Sets' tab, both the Revit property sets and the IFC common property sets must be exported.

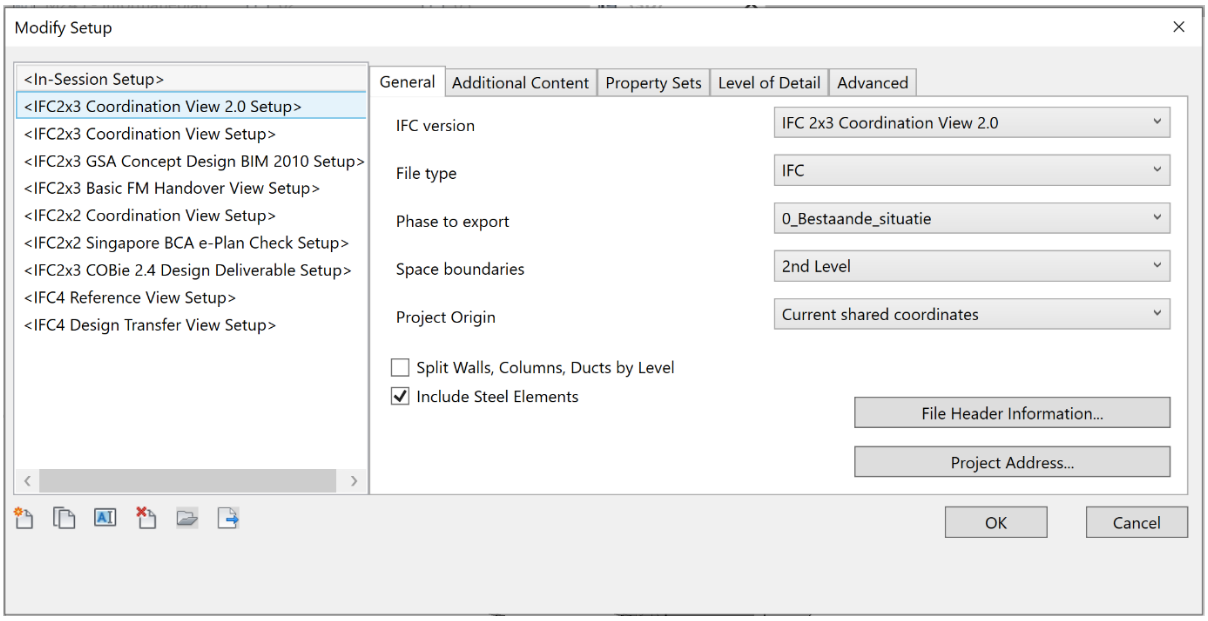


Figure 19: IFC export setup

When converting Revit to gbXML, it is important to make sure that the option 'Use Room/Space Volumes is chosen and that exterior elements are identified, as shown in Figure 20.

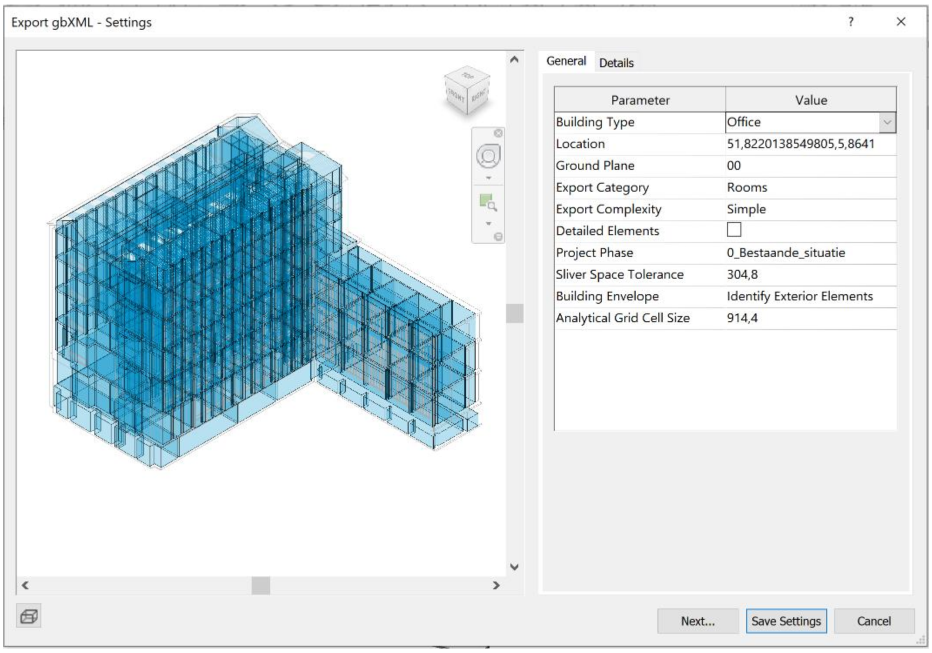


Figure 20: gbXML export setup

4.1.2.2. Converting IFC to LBD (Turtle)

As Revit does not have a Linked Building Data (LBD) export option (yet), the Revit model must first be exported to IFC and then to LBD turtle graphs. An easy to use desktop converter exists, but its latest version does not export all required components. However, an older version is forked on GitHub and can be used to export all required components[9]. When using the converter, make sure to use PROPS level 1, not higher than that. This would lead to different predicates causing the scripts not to work. The reason for the different predicates is because the converter appends different predicates to distinguish the level that is used in the conversion. For example, using the level 1 props places '_simple' behind each predicate, while level 2 does not add such a string behind the predicates. Additionally, level 2 defines a separate node (and thus a URI) for each property. For instance, the property 'isExternal' in level 1 is given as 'props:isExternal_simple' with the property value as object. In level 2 the predicate is 'props:isExternal' with a URI of the property value as object. This URI refers to a node with the properties 'props:partOfPset' and 'schema:value'. The latter contains the property value as object. As this makes the readability and navigation more complicated which is not necessary in this thesis, level 1 is used. Level 3 adds the time element to the equation which might be relevant if for example temporary properties are used. The settings used for this thesis are shown in Figure 21.

[9] https://github.com/SjorsvanGool/IFCtoLBD

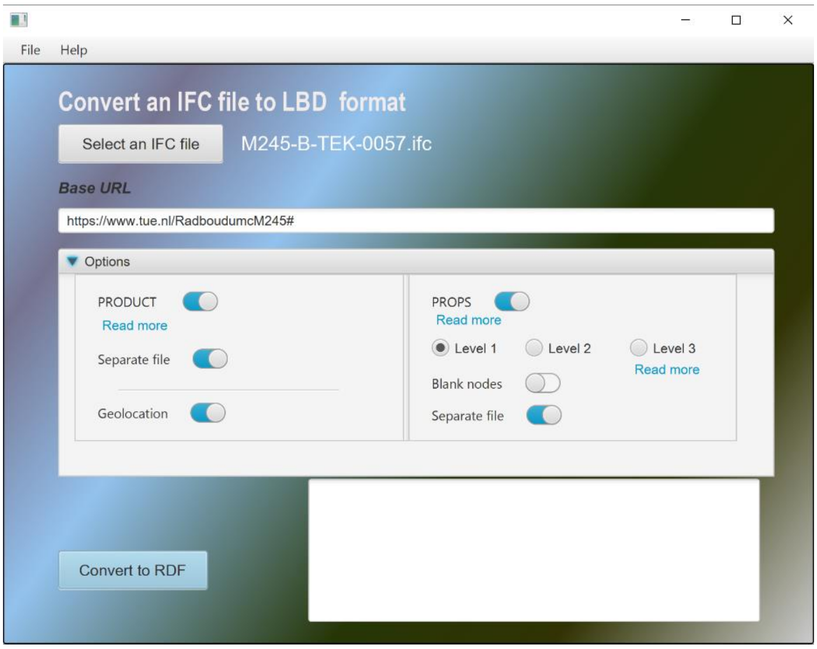


Figure 21: Settings used in the IFC to LBD converter

Still, as one of the core principles of semantic modelling is its flexibility, it would be contradictory to keep the graphs in local Turtle files. To prevent this, GraphDB has been used to make the graphs easily accessible. For now, GraphDB ran on a localhost server but this can be easily changed to online. The method of uploading and enriching a graph in GraphDB through Python code is shown in chapter 5.1.1.

4.1.3. Converting IFC to COLLADA

Converting IFC to COLLADA can be done using the IfcConvert executable[10] from IfcOpenShell. The required input values are the link to the IFC file and the name of the target file. Using the Command Prompt, the input values can be given and the executable can be started, as shown in Figure 22.

```
C:\Users\sjors\Downloads\IfcConvert-0.4.0-rc2-win64>IfcConvert M245-B-TEK-0057.ifc M245-B-TEK-0057.dae
Scanning file...
Done scanning file
Creating geometry...
Done creating geometry

Conversion took 20 seconds
```

Figure 22: Usage of the IfcConvert executable

4.2. Sensor and meteorological time-series data

The sensor measurements collected by BeSense contain both occupancy and indoor comfort data and shall be used as sensor dataflows during the development of this thesis. Apart from the sensor time-series data, meteorological data shall be integrated using two APIs of Meteoserver. Time-series data (both sensor and meteorological data) shall be divided into MongoDB collections. This chapter shall describe what formats are required and how they are acquired.

4.2.1. Appending time-series sensor data to MongoDB

Sensors transfer their data to a central point. Some use a corporate provider such as Microsoft Azure, others use an open source provider such as MongoDB. To create an integration that is

[10] http://ifcopenshell.org/ifcconvert

suitable and affordable for everyone, MongoDB has been chosen. For now, a localhost MongoDB server has been used but this can, like GraphDB, easily be scaled to an online MongoDB server.

BeSense appends their data to Microsoft Azure and shows its results on a Microsoft PowerBI dashboard. To migrate the data from the Azure environment to the MongoDB environment, a script was written that queries all datapoints from Azure and appends them into MongoDB. After appending these datapoints, the script executes a check to make sure each datapoint is copied. In case datapoints are still missing, these gaps shall be identified. The script only considers the comfort values as these sensors should always provide measurements for each hour. Motion data is less predictable as, for example, no one might use the desk for a day, leading to no datapoints for that day. Consequently, it is difficult to create a check for such motion data.

The data migration script is placed in the Python file 'Data_migration_Azure_to_MongoDB_final.py'. Its UML Activity diagram is shown in Figure 23 and a larger version can be found in Appendix A3.

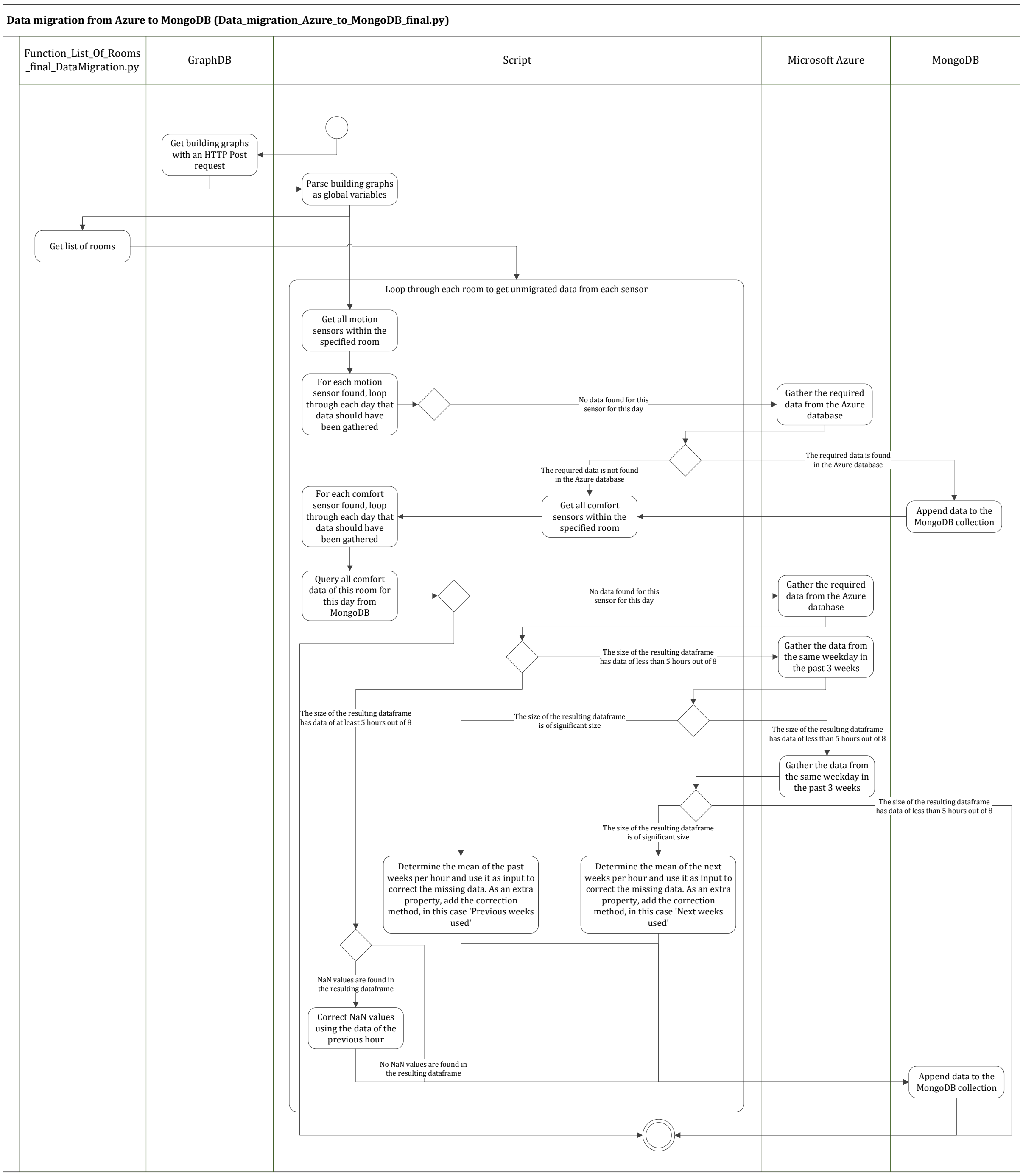


Figure 23: UML Activity diagram of the script 'Data_migration_Azure_to_MongoDB_final.py'

4.2.2. Appending meteorological time-series data to MongoDB

Meteorological data can be easily gathered through an API. In this case Meteoserver was used to gather outside air temperature, sun intensity, sun orientation, and wind speed. Gathering Meteoserver's historical data is not free of charge, but its current and predicted data is. Therefore, a periodically executed script was written that appends the current meteorological data to a MongoDB collection. Consequently, this data can in a later stage be gathered free of charge, making historical data free. Two Meteoserver sources are used of which the scripts can be found in the attached Python files 'Append_Sun_Data-final.py' and 'Append_Wind_Data-final.py', and the UML diagram in Figure 24. A larger version of the UML Activity diagram is provided in Appendix A3.

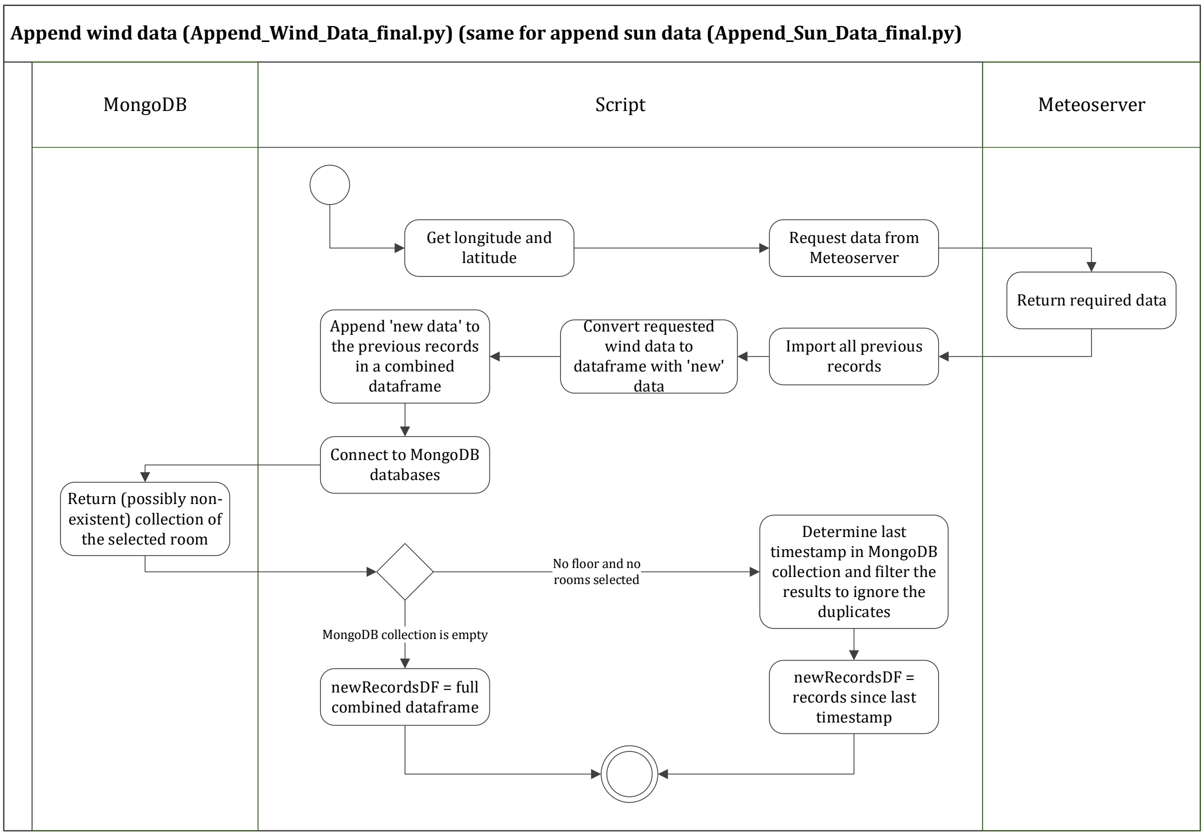


Figure 24: UML Activity diagram of the scripts 'Append_Sun_Data-final.py' and 'Append_Wind_Data-final.py'

Section 5

Data integration

After gathering and converting the required data sources, the integration should be set up, described in this section. This section shall start by elaborating on the interaction with the selected databases, GraphDB and MongoDB. The interaction entails both querying data as well as posting data. After, the LBD files shall be enriched with geometry linking datapoints, both for the gbXML format as well as the COLLADA format. For the former a dedicated ontology was written. As correct data is essential in such an integration, this section shall end with a data integrity check that verifies several common data quality issues.

5.1. Interact with databases

Both databases used in this integration must be queried from and written to. This chapter shall elaborate on the methods of doing so.

5.1.1. GraphDB

Initial upload

The initial Turtle files can be uploaded to GraphDB via either the GraphDB dashboard or via code. To fully automate the workflows, the latter option has been chosen in this thesis. Using a HTTP Post request, a full Turtle file can be uploaded to GraphDB using the code in Figure 25.

```
import requests
url = http://localhost:7200/repositories/ThesisSjorsVanGoolRadboudumcCaseStudy/rdf-graphs/BOT
payload = open("LBD files/M245-B-TEK-0057_LBD.ttl")
headers = {'content-type': 'application/x-turtle'}
r = requests.post(url, data=payload, headers=headers)
```

Figure 25: Code snippet to upload a local Turtle file to GraphDB

Post data to GraphDB

Posting individual datapoints to GraphDB works with HTTP Post requests as well. In this case the SPARQL endpoint of GraphDB, ending with '/statements', is used. To connect to this endpoint, the SPARQL Python package is used.

```
from SPARQLWrapper import SPARQLWrapper
queryString = """
    PREFIX inst: <https://www.tue.nl/RadboudumcM245#>
    PREFIX props: <https://w3id.org/props#>
    INSERT DATA { GRAPH <http://localhost:7200/repositories/ThesisSjorsVanGoolRadboudumcCaseStudy/rdf-graphs/Props>
    { """+spaceURI+""" props:hasId """+spaceId+""" } }
"""
sparql = SPARQLWrapper("http://localhost:7200/repositories/ThesisSjorsVanGoolRadboudumcCaseStudy/statements")
sparql.setQuery(queryString)
sparql.method = 'POST'
sparql.query()
```

Figure 26: Code snippet to post datapoints to GraphDB

Query data from GraphDB

Querying data from GraphDB is more complicated due to execution time requirements. Python has a very easy to use package to work with Turtle/RDF graphs: RDFLib. RDFLib can easily and quickly make queries in locally parsed graphs. However, parsing these graphs takes quite long if only a simple query is desired. For example, parsing the case study BOT and PROPS graph took 39.3 seconds (Figure 27).

```
start = time.time()
from rdflib import Graph
gBOT = Graph()
gBOT = gBOT.parse(config['GraphDBPath']['pathBOT'])
gPROPS = Graph()
```

```
gPROPS = gPROPS.parse(config['GraphDBPath']['pathProps'])
print("Execution time of parsing the BOT and PROPS graph: ", time.time()-start, "seconds")

Execution time of parsing the BOT and PROPS graph: 39.30244779586792 seconds
```

Figure 27: Locally parsing the GraphDB graphs

Parsing these graphs once would not be an issue, but the scripts of the case study are divided over several functions in varying files. Python can define variables as 'global', making it accessible to other files when it has been loaded once in the session, as shown in Figure 28.

```
global gBOT
global gPROPS
gBOT = Graph()
gBOT = gBOT.parse(config['GraphDBPath']['pathBOT'])
gPROPS = Graph()
gPROPS = gPROPS.parse(config['GraphDBPath']['pathProps'])
```

Figure 28: Parsing the graphs globally

One precondition though is that the path to the loaded global variable is imported. It must therefore be known where the file has been parsed when executing another file that uses the parsed graphs. Scripts that are called from multiple files, for example the list of rooms function, must therefore have multiple import paths, depending on where it is being called from. For example, in the code shown in Figure 29, the graphs are called from the update script.

```
def listOfRooms (building, floor, rooms):
    import UpdateResults_final as graphs
    #Append roomID's to list of rooms in case of whole building selection
    qStoreyURI = prepareQuery(
        'SELECT ?storeyURI WHERE { ?storeyURI props:category_simple ?category .}',
        initNs = { "props": 'https://w3id.org/props#'})
    for row in graphs.gPROPS.query(qStoreyURI, initBindings={'category': rdflib.Literal("Le
vels")}): #query storeyURI
        storeyURI = row.storeyURI
```

Figure 29: Calling the globally parsed graphs from the update script

Another method exists though that does not parse the whole graphs: GraphDB's SPARQL endpoint. GraphDB has a built-in functionality to execute SPARQL queries in their dashboard. The execution of these queries from Python is similar to the Post queries described earlier in this chapter. Figure 30 shows an example of a SPARQL endpoint query executed from Python.

```
from SPARQLWrapper import SPARQLWrapper, JSON
sparql = SPARQLWrapper("http://localhost:7200/repositories/ThesisSjorsVanGoolRadboudumcCase
Study") #Import SPARQL Endpoint
sparql.setQuery("""
    PREFIX inst: <https://www.tue.nl/RadboudumcM245#>
    PREFIX props: <https://w3id.org/props#>
    SELECT ?storeyURI
    WHERE{ GRAPH <http://localhost:7200/repositories/ThesisSjorsVanGoolRadboudumcCaseStudy/
rdf-graphs/Props>
        {?storeyURI props:name_simple ?level .
```

```
       FILTER (?level = '"""+level+"""')} }""") #Make sure you add the single ' around the
 multiline string break. Otherwise the outcome will be for example 02 instead of '02'. It i
s then not recognised as a string and thereby not equal to the values in the graphs, leadin
g to no outcome.
sparql.setReturnFormat(JSON)
results = sparql.query().convert()
for result in results["results"]["bindings"]:
    storeyURI = result['storeyURI']['value']
```
Figure 30: Gathering the storeyURI through a SPARQL endpoint query

Even though this query method does not require a whole graph to be parsed, it still takes about 2 seconds to execute the query. For a single query this is thereby significantly faster than the 40 seconds of fully parsing the graphs. However, these two seconds do not decrease when queries are executed multiple times in one script, in contrast to parsing the full graphs. Once the graphs are parsed, the queries take about 0.1 seconds to execute. The consideration of which method to choose thereby relies on the number of queries that can be executed once the graph has been parsed. If more than about 25 queries are conducted, it would be faster to parse the graphs in full. Due to the Python functionality of parsing graphs as a global variable, and thereby using them in multiple scripts without parsing them again, this method is more attractive to use in the scripts created in this thesis. However, it is still important to know that another method, the SPARQL endpoint, can be used as well.

5.1.2. MongoDB

Post data to MongoDB

MongoDB has a Python package 'pymongo', which eases the posting and querying process of MongoDB. Several steps must be taken to connect to the client and finally append the data to MongoDB. Figure 31 shows how these steps can be taken. This method is being applied throughout all relevant scripts created in this thesis. Essential to note is that the collection names are equal to the room ID's as defined in the graphs. This creates the direct link between the room graph data and the room time-series data, both for the sensor time-series data as well as the analysis results.

```
import pymongo
#Connect to the MongoDB client
mongodbClient = pymongo.MongoClient("mongodb://localhost:27017/")
#Connect to or create (if it doesn't exist) a MongoDB database
motionData = mongodbClient["MotionData"]
#Connect to or create (if it doesn't exist) a collection within the database
roomCollectionMotion = motionData[room]
#Convert the dataframe to append to MongoDB to a dictionary
RecordsToAppend = DataframeToAppend.to_dict('records')
#Choose the collection to append to and insert the records to append
roomCollectionMotion.insert_many(RecordsToAppend)
```
Figure 31: Process of posting data to MongoDB

Query data from MongoDB

Once the connection with the MongoDB collection has been made, data can be queried with ease. A query can be set according to the MongoDB syntax and gathered in a dictionary. Using Pandas' function 'DataFrame.from_records', the data can be appended to a Pandas DataFrame. Figure 32 shows how these steps can be taken. Similar to the posting method, this method is being applied in all scripts. Querying records from MongoDB has a very short execution time. It is thereby ideal for making near instant requests in a dashboard. As shown in Figure 32, making a significant query, in this case all motion data over the whole pilot period for room M245.02.240, only takes 0.05 seconds.

```
start = time.time()
#Initiate MongoDB
mongodbClient = pymongo.MongoClient("mongodb://localhost:27017/")
motionData = mongodbClient["MotionData"]
motionDataCollection = motionData["M245.02.240"]
#Set query according to the MongoDB syntax
motionQuery = {}
#Query the data from MongoDB according to the query
motion_dict = motionDataCollection.find(motionQuery)
#Define the columns that you would like to use from the overall JSON file
columns=['Timestamp', 'Hour', 'Value']
#Create a dataframe from the MongoDB results
roomsDF = pd.DataFrame.from_records(motion_dict, columns=columns, index=None)
print("Execution time querying all motion data for room M245.02.240:", time.time()-
start,"seconds")

Execution time querying all motion data for room M245.02.240: 0.05984783172607422 seconds
```

Figure 32: Process of querying data from MongoDB

5.1.3. Enriching LBD files with gbXML linking datapoints

In chapter 4.1 several required formats of building data have been gathered. These files are all linked through mutual identification datapoints. The most essential data source is gbXML for its geometry data. GbXML has been created for building energy analyses and thereby contains the rough geometry of surfaces and spaces. For analysis purposes, such as the mean radiant heat, this level of geometry is perfect. Considering for example a window as a simple polygon instead of their whole mesh geometry is a very efficient manner of using the geometry for analysis purposes.

5.1.3.1. Query geometry from the gbXML file of the pilot building

Currently the gbXML file is locally stored. Parsing the whole file including executing the query in Figure 33 only takes 1.7 seconds, making it suitable for direct querying from a dashboard. However, it might be that once the gbXML file is accessed remotely, parsing the file takes longer.

```
start = time.time()
#%% User input
building = "M245"
floor = "02"

#Parse gbXML file
tree = ET.parse(config['GeometryFilePath']['pathgbXML'])
root = tree.getroot() #create the root reference
ns = {'file': 'http://www.gbxml.org/schema'} #define the gbXML schema to navigate through
it later on


#%% Determine scale factor considering the size of the floor plan
```

```
campus = root.find('{http://www.gbxml.org/schema}Campus') #narrow down on the campus as all
 uses below require the data within the campus
building = campus.find('{http://www.gbxml.org/schema}Building') #narrow down on the campus
as all uses below require the data within the campus
for storey in building.findall('{http://www.gbxml.org/schema}BuildingStorey'):
    if str(storey.find('{http://www.gbxml.org/schema}Name').text) == floor:
        xCoordinates = []
        yCoordinates = []
        planarGeometry = storey.find('{http://www.gbxml.org/schema}PlanarGeometry')
        polyloop = planarGeometry.find('{http://www.gbxml.org/schema}PolyLoop')
        for cartesianPoint in polyloop.findall('{http://www.gbxml.org/schema}CartesianPoint
'):
            coordinates = []
            for coordinate in cartesianPoint.findall('{http://www.gbxml.org/schema}Coordina
te'):
                coordinates.append(float(coordinate.text))
            xCoordinates.append(coordinates[1])
            yCoordinates.append(coordinates[0])
print("Execution time for finding all x and y coordinates of the second floor:", time.time(
)-start,"seconds")

Execution time for finding all coordinates of the second floor: 1.6955184936523438 seconds
```

Figure 33: Query floor coordinates from the gbXML file

5.1.3.2. gbXML Id property

The identifications of an element within a gbXML is not the same as the URI identification in a graph. However, through other mutual datapoints between the gbXML file and the graphs, a link can be established between the two. Specifically, the bat ID property can be used as mutual datapoint. This property is embedded in the gbXML CADObjectId. Though, to ease and speed up the process of making these links during the analyses, a new datapoint is created for each relevant element in the property graph. The predicate used in the added datapoints is called 'hasId', which can be used in a later stage to quickly navigate to the correct geometry datapoints in the gbXML file.

It is important to emphasize that the data added to the graphs is not the actual geometry data but only the links between the graph elements and the gbXML elements. During the analyses the geometry data shall be gathered from the gbXML file on the spot.

To upload the initial Turtle files to GraphDB as well as enriching these graphs, a separate Python script is created. This script can be found in the attached Python file 'Graph_Preparation_Code_final.py'. The UML Activity diagram is shown in Figure 34 and a larger version can be found in Appendix A3.

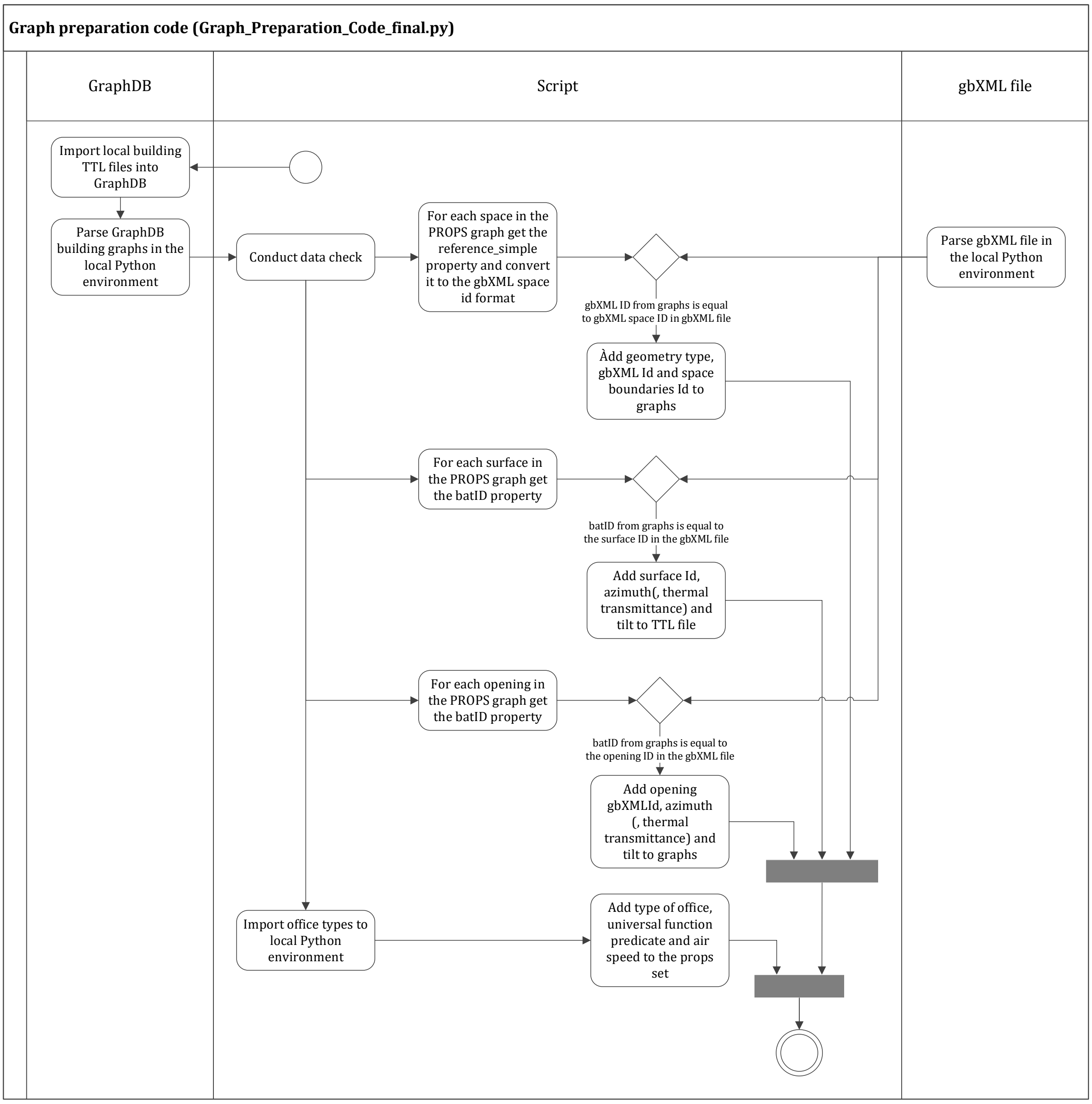


Figure 34: UML Activity diagram of the script 'Graph_Preparation_Code_final.py'

5.1.3.3. ogbxml ontology

In the current integration only the gbXML Id has been added to the property graph using the 'hasId' predicate. However, it might be that in a later stage more gbXML elements are to be added to the property graphs. For example, the gbXML file is currently locally stored, making it less accessible than the graphs in GraphDB. To ease the process of appending new gbXML datapoints in the graphs and to clarify the setup of gbXML files, a dedicated gbXML ontology has been written: Ontology for gbXML. The ontology is available on https://sjorsvangool.github.io/ogbxml/ and uses the prefix 'ogbxml'.

5.1.4. Enriching LBD files with mesh linking datapoints

As described in chapter 2.2.1.3, gbXML geometry is a simplified version of the total element geometry, containing only the data necessary for energy analyses. However, visualisation is another large application area of the geometry data of a building. As gbXML is not suitable for visualisation and a dedicated geometry format is desired according to the linked building data approach, COLLADA shall be used for building visualisation purposes. COLLADA (.dae) files contain geometry meshes of building elements. IFC files can be converted to COLLADA files using the IfcConvert tool[11], which can be converted to glTF data streams for visualisation apps.

5.1.4.1. COLLADA mesh Id property

Like gbXML, a mutual datapoint should be found to link the COLLADA element meshes to the elements in the graphs. The geometry meshes in the COLLADA file all have an id. The last characters of this id are equal to the bat ID that is given to the element in the LBD graphs. Therefore, the bat ID shall be used to make the link between an element in the PROPS graph and the COLLADA mesh elements. To efficiently navigate to these meshes from the graphs, the COLLADA mesh id's shall be added to the element properties in the PROPS graph, as is done for the gbXML geometry, using the predicate 'props:hasMeshId'. Like gbXML, the code of doing so can be found in the Python file 'Graph_Preparation_Code_final.py', its UML Activity diagram in Figure 34 and a larger version of the UML Activity diagram in Appendix A3.

5.1.4.2. Query mesh geometry from the COLLADA file of the pilot building

Like gbXML files, COLLADA files are XML-based. Querying mesh geometry data can therefore be executed in the same manner as querying gbXML files. Figure 35 demonstrates an example query including its execution time. This query returns the mesh geometry element of each surface present in the pilot building in only 1.1 seconds, making it suitable for quick analyses/visualisations. However, like gbXML, the COLLADA file is locally stored. If it would be remotely accessed it might take longer.

```
start = time.time()
#%%Parse COLLADA file in local Python environment
treeCOLLADA = ET.parse(config['GeometryFilePath']['pathCollada']) #parse the COLLADA file
rootCOLLADA = treeCOLLADA.getroot() #create the root reference
ns = {'file': 'http://www.collada.org/2005/11/COLLADASchema'} #define the COLLADA schema to
 navigate through it later on

#%% Query all surface mesh ID's
library_geometries = rootCOLLADA.find('{http://www.collada.org/2005/11/COLLADASchema}librar
y_geometries') #narrow down on the campus as all uses below require the data within the cam
pus
surfaceMeshGeometryList = []
for geometry in library_geometries.findall("{http://www.collada.org/2005/11/COLLADASchema}g
eometry"):
    if geometry is not None:
        surfaceMeshGeometryList.append(geometry)

print("Execution time of querying all surface mesh geometries in the pilot building:", time
.time()-start,"seconds")

Execution time of querying all surface mesh geometries in the pilot building:
1.119931936264038 seconds
```

Figure 35: COLLADA file query of the mesh geometry element of each surface element in the pilot building

[11] http://ifcopenshell.org/ifcconvert

Logically Figure 35 only executes one query. If the query is executed in a loop, the execution time might be significantly longer. For example, the loop in Figure 36 searches for the mesh ID of a specific surface using the bat ID as identifier. Executing this for each surface in the building takes 26.8 seconds.

```
start = time.time()
#%%Parse COLLADA file in local Python environment
treeCOLLADA = ET.parse(config['GeometryFilePath']['pathCollada']) #parse the COLLADA file
rootCOLLADA = treeCOLLADA.getroot() #create the root reference

for surfaceBatId in surfaceList:
    ns = {'file': 'http://www.collada.org/2005/11/COLLADASchema'} #define the COLLADA schem
a to navigate through it later on
    library_geometries = rootCOLLADA.find('{http://www.collada.org/2005/11/COLLADASchema}li
brary_geometries') #narrow down on the campus as all uses below require the data within the
 campus
    for geometry in library_geometries.findall("{http://www.collada.org/2005/11/COLLADASche
ma}geometry"):
        if geometry.attrib['id'][-
len(surfaceBatId):] == surfaceBatId: #link the surface in COLLADA to the surface in the gra
ph using the bat ID gathered from the PROPS graph
            surfaceMeshId = geometry.attrib['id']
print("Execution time of querying the surface mesh geometry for each bat ID in the pilot bu
ilding:", time.time()-start,"seconds")

Execution time of querying the surface mesh geometry for each bat ID in the pilot building:
26.79239559173584 seconds
```

Figure 36: Finding the mesh geometry for the bat ID of each surface in the pilot building

5.1.5. Integration diagram between the linked building data sources

To summarize the interactions that have been set up and tested in this chapter, a Neo4J-styled diagram has been established in Figure 37 for the interaction between different files after the LBD graph enrichment. A larger version of this diagram can be found in Appendix A2. Each interaction between the linked data sources is built upon one or more mutual datapoints. Providing insight in which specific datapoints are used is thereby essential.

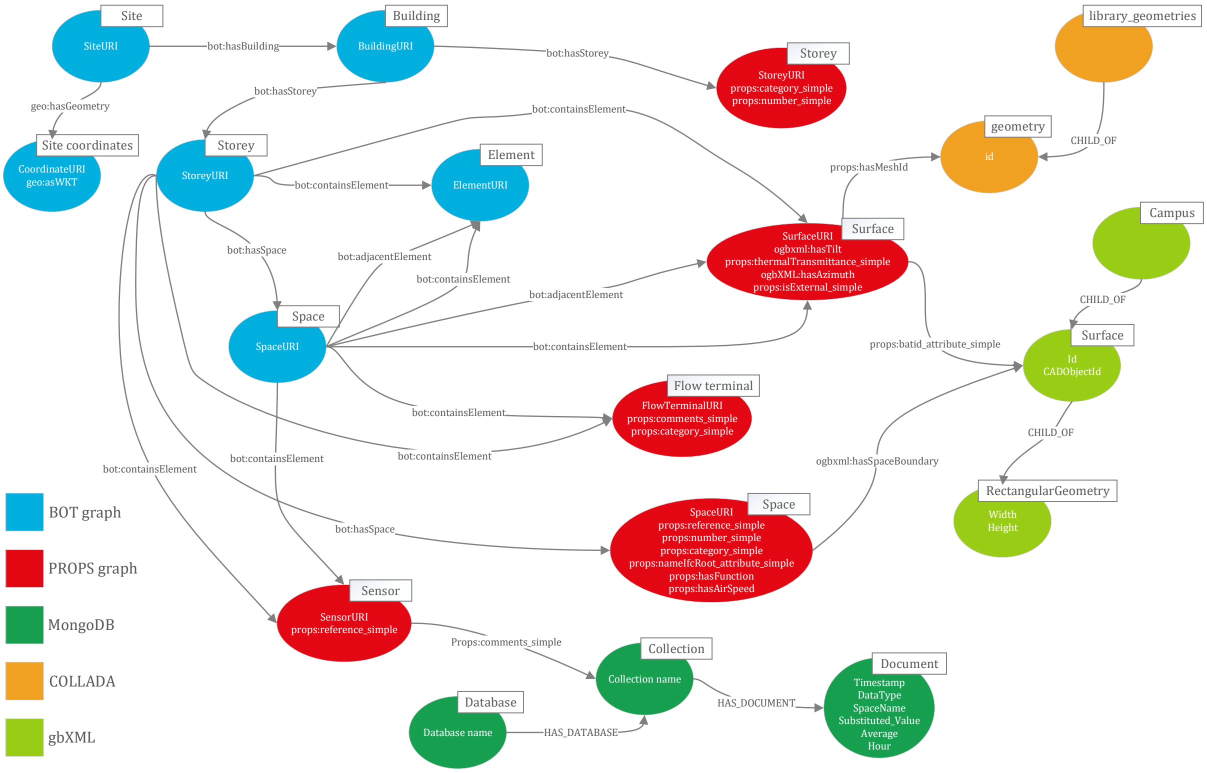


Figure 37: A Neo4J-styled diagram of the developed integration

5.2. Data integrity check

Getting the data right is the first and foremost step in conducting effective analyses. This thesis thereby focusses on correctly and efficiently integrating the building dataflows with the sensor dataflows. However, ensuring that the data is right after the integration is an essential step to take. Therefore, the following Python script has been written 'Sensor_check_code_final.py'. The UML Activity diagram can be found in Figure 38 and a larger version in Appendix 0. This script checks several common data quality issues of an integration. This chapter shall elaborate on these checks.

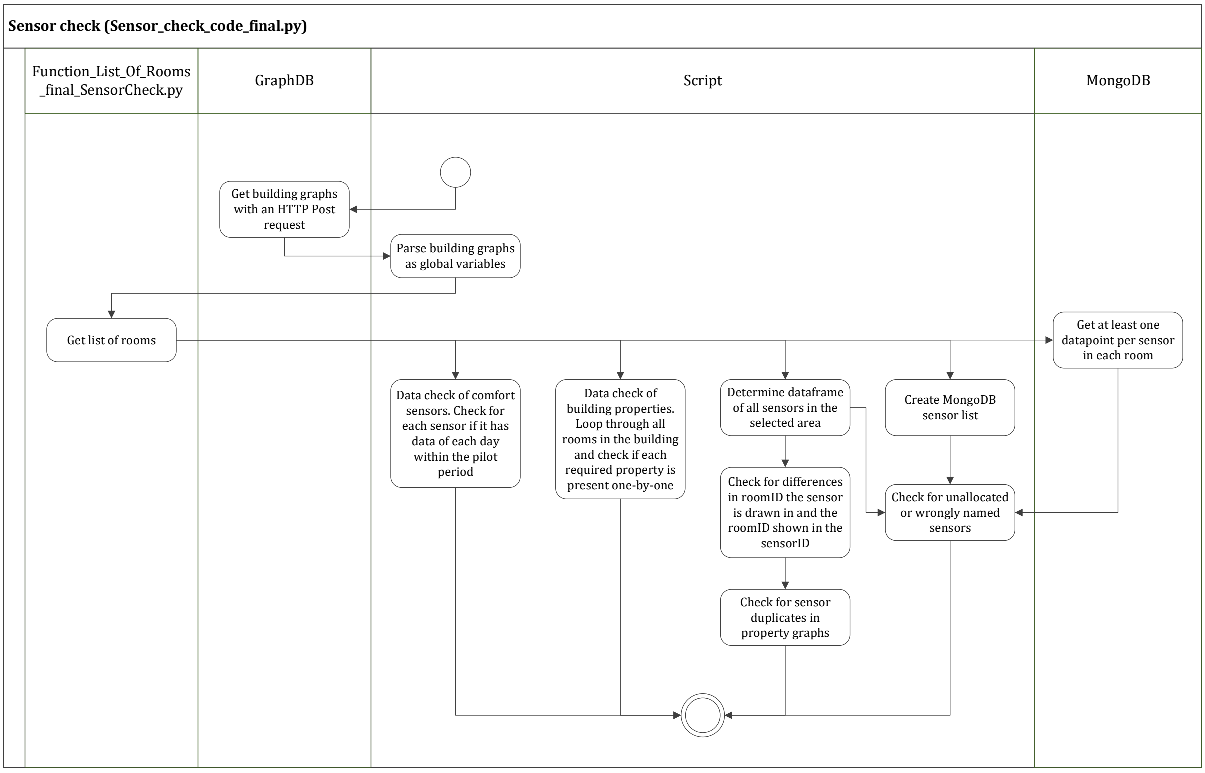


Figure 38: UML Activity diagram of the script 'Sensor_check_code_final.py'

5.2.1. Check for duplicates in the property graphs

In the proposed integration process, Revit is used to draw sensors into a space as an object. It might be that in the process a sensor is drawn twice, and that data shall in a later stage be queried and used twice as well. This would corrupt the results of any analysis so this should be identified beforehand. The developed check notifies the user if this is the case in his/her data, as shown in Figure 39.

```
Careful, the following duplicate sensor ID's have been found in the property graph:
        sensorType                         AzureID
36  BeThere sensor  M245.03.340 Vergaderruimte
86  BeThere sensor      M245.02.244 Werkplek 1
Please check if they are duplicated in the Revit file.
```

Figure 39: Duplicate check output

When checking the above error in Revit, in room M245.03.340 two identical sensors are placed on the wall, and in room M245.02.244 two identical sensors are placed under a table.

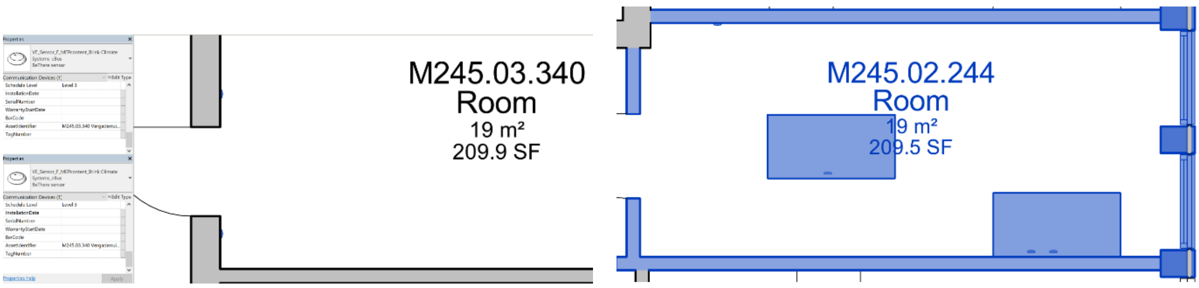


Figure 40: Checking the duplicate sensors in Revit

5.2.2. Check for differences between sensor naming in the property graphs and Azure

Another issue that might arise is that sensors are not named correctly. The link between the sensor and the building data is made based on the sensor ID. If this ID is not exactly the same in both data sources, the link is corrupt. This the most essential datapoint to be correct and must therefore be checked. The script notifies the user of eventual errors and prints the dataframe to a csv file for a better overview. Figure 41 shows the Python terminal output and Figure 42 shows the csv files with the sensors in the PROPS graph (left) and MongoDB (right). The missing data in either one of the csv's is highlighted.

```
Be careful, the following differences between the sensors in the properties graph and the sensors in
the Azure database have been found:
                      SpaceName         sensorType
108        M245.02.245 Werkplek 1    BeThere sensor
33         M245.02.245 Werkplek 5    BeThere sensor
162        M245.02.248 Werkplek 5    BeThere sensor
0      M245.02.251 Vergaderruimte    BeThere sensor
133     M245.02.251 Vergadertafel    BeThere sensor
16    M245.02.252A Vergaderruimte    BeThere sensor
145    M245.02.252A Vergadertafel    BeThere sensor
180    M245.03.340 Vergaderruimte  BeComfort sensor
89         M245.03.344 Werkplek 3    BeThere sensor
71         M245.03.350 Werkplek 2    BeThere sensor
131        M245.03.353 Werkplek 6    BeThere sensor
These differences might cause false results in the analyses. Please check these cases before moving on.
```

Figure 41: Python terminal output for the sensor naming differences

1	SpaceName	sensorType
17	M245.02.245 Vergadertafel	BeThere sensor
18	M245.02.245 Werkplek 2	BeThere sensor
19	M245.02.245 Werkplek 3	BeThere sensor
20	M245.02.245 Werkplek 4	BeThere sensor
21	M245.02.245 Werkplek 5	BeThere sensor
22	M245.02.248 Kantoor	BeComfort senso
23	M245.02.248 Werkplek 1	BeThere sensor
24	M245.02.248 Werkplek 2	BeThere sensor
25	M245.02.248 Werkplek 3	BeThere sensor
26	M245.02.248 Werkplek 4	BeThere sensor
34	M245.02.251 Vergaderruimte	BeThere sensor
35	M245.02.251 Werkplek 1	BeThere sensor
36	M245.02.251 Werkplek 2	BeThere sensor
39	M245.02.252A Vergaderruimte	BeThere sensor
40	M245.02.252A Werkplek 1	BeThere sensor
60	M245.03.344 Werkplek 1	BeThere sensor
61	M245.03.344 Werkplek 2	BeThere sensor
62	M245.03.344 Werkplek 3	BeThere sensor
78	M245.03.350 Werkplek 1	BeThere sensor
79	M245.03.350 Werkplek 2	BeThere sensor
80	M245.03.350 Werkplek 3	BeThere sensor
89	M245.03.353 Kantoor	BeComfort senso
90	M245.03.353 Werkplek 1	BeThere sensor
91	M245.03.353 Werkplek 2	BeThere sensor
92	M245.03.353 Werkplek 3	BeThere sensor
93	M245.03.353 Werkplek 4	BeThere sensor
94	M245.03.353 Werkplek 5	BeThere sensor
95	M245.03.353 Werkplek 7	BeThere sensor

1	SpaceName	sensorType
16	M245.02.245 Vergadertafel	BeThere sensor
17	M245.02.245 Werkplek 1	BeThere sensor
18	M245.02.245 Werkplek 2	BeThere sensor
19	M245.02.245 Werkplek 3	BeThere sensor
20	M245.02.245 Werkplek 4	BeThere sensor
21	M245.02.248 Kantoor	BeComfort sensor
22	M245.02.248 Werkplek 1	BeThere sensor
23	M245.02.248 Werkplek 2	BeThere sensor
24	M245.02.248 Werkplek 3	BeThere sensor
25	M245.02.248 Werkplek 4	BeThere sensor
26	M245.02.248 Werkplek 5	BeThere sensor
34	M245.02.251 Vergadertafel	BeThere sensor
35	M245.02.251 Werkplek 1	BeThere sensor
36	M245.02.251 Werkplek 2	BeThere sensor
39	M245.02.252A Vergadertafel	BeThere sensor
40	M245.02.252A Werkplek 1	BeThere sensor
60	M245.03.344 Werkplek 1	BeThere sensor
61	M245.03.344 Werkplek 2	BeThere sensor
77	M245.03.350 Werkplek 1	BeThere sensor
78	M245.03.350 Werkplek 3	BeThere sensor
87	M245.03.353 Kantoor	BeComfort sensor
88	M245.03.353 Werkplek 1	BeThere sensor
89	M245.03.353 Werkplek 2	BeThere sensor
90	M245.03.353 Werkplek 3	BeThere sensor
91	M245.03.353 Werkplek 4	BeThere sensor
92	M245.03.353 Werkplek 5	BeThere sensor
93	M245.03.353 Werkplek 6	BeThere sensor
94	M245.03.353 Werkplek 7	BeThere sensor

Figure 42: CSV files highlighting the discrepancies in sensors present in the PROPS graph and MongoDB

5.2.3. Wrong sensor allocation

```
All good, no duplicates were found in the properties graph.

Be careful, the following differences between the sensors in the properties graph and the sensors in
the Azure database have been found:
                  SpaceName      sensorType
87   M245.02.245 Werkplek 5  BeThere sensor
161  M245.02.248 Werkplek 5  BeThere sensor
25   M245.02.348 Werkplek 6  BeThere sensor
113  M245.03.348 Werkplek 6  BeThere sensor
These differences might cause false results in the analyses. Please check these cases before moving on.
```

Figure 43: Remaining issues after correcting the above differences

After correcting the above issues, the differences check still identifies some issues, as shown in Figure 43. After manually checking the remaining/new issues, it seemed that workplace 5 in room M245.02.248 was actually named ‘M245.02.245 Werkplek 5’ instead of ‘M245.02.248 Werkplek 5’. Additionally, after a manual check for the issue surrounding ‘M245.03.348 Werkplek 6’, it seems, as expected, that the floor number of this sensorID was incorrectly named in the Revit object ID. Such issues might occur more often, so a check was built in as well. In the future such issues might be overcome using the geometry of the sensor. However, for this to work the geometry of each sensor must be converted from the IFC file to the LBD files, which is not the case in the current version of the converter. Additionally, and this is more difficult to overcome, the sensor must provide its real-time exact location. This requires additional hardware which is not provided in most sensors.

To automatically check these issues, a check was developed that extracts the name of the room the sensor should be in and compares it with the room it is actually drawn in. Both a wrong room and a wrong floor will be identified in this check. This results in the output in Figure 44.

```
The following sensors seem to be placed in a different room than their sensorID indicates. Please check the
following cases:
        RoomID               SpaceName      sensorType
0  M245.03.348  M245.02.348 Werkplek 6  BeThere sensor
1  M245.02.253  M245.02.254 Werkplek 4  BeThere sensor
2  M245.02.253  M245.02.254 Werkplek 3  BeThere sensor
3  M245.02.253  M245.02.254 Werkplek 2  BeThere sensor
4  M245.02.248  M245.02.245 Werkplek 5  BeThere sensor
```

Figure 44: Python terminal output when comparing the room ID the sensor is drawn in with the room ID that the sensor ID says it is drawn in

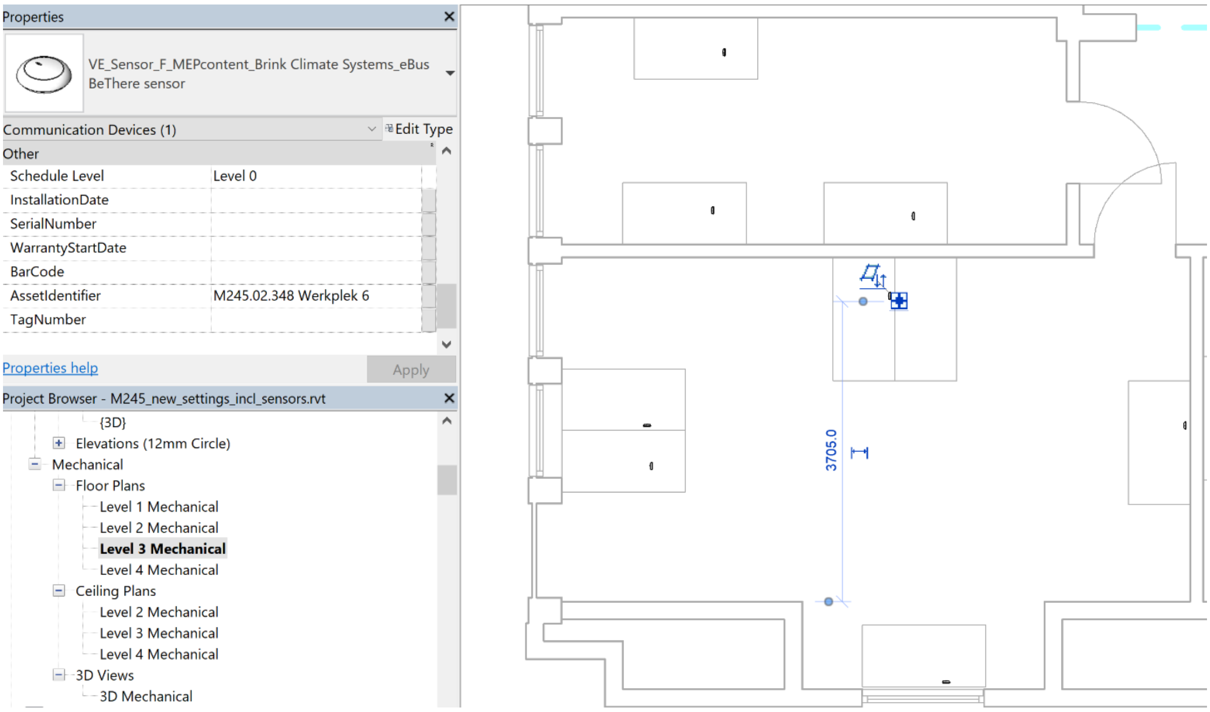


Figure 45: Checking the identified sensor ID issues in Revit

When checking the identified sensor ID issues in Revit, it seems that indeed the naming is inconsistent with the actual room the sensor is drawn in. Once these issues were corrected, some errors in room M245.02.253 still remained, as shown in Figure 46. However, it turns out that these sensors have been named incorrectly by the sensor supplier. This naming issue can therefore not be corrected by the Radboudumc, only by BeSense. For now, this issue shall therefore be ignored.

```
The following sensors seem to be placed in a different room than their sensorID indicates. Please check the
following cases:
  RoomID in property graph SensorID in property graph      sensorType
0              M245.02.253     M245.02.254 Werkplek 4  BeThere sensor
1              M245.02.253     M245.02.254 Werkplek 2  BeThere sensor
2              M245.02.253     M245.02.254 Werkplek 3  BeThere sensor
No duplicates were found in the properties graph.

All good, no differences were found between the sensor ID's in the properties graph and in the Azure database.
```

Figure 46: Remaining incorrect sensor ID naming

5.2.4. Check for missing datapoints

Apart from naming issues, datapoints can be missing in both the building graphs and the time-series sensor data. Both would result in either corrupt analysis results or errors later.

Missing datapoints in building graphs

If data is missing in the building graphs, part of the script shall not be run. When querying data in RDFLib a loop is used, shown in Figure 47. If any of the requested properties are not present, the loop continues to the next graph datapoint, leaving the script in Figure 47 untouched.

```
qBoundingSurfaces = prepareQuery(
    'SELECT ?boundingSurfaceId WHERE { ?spaceURI props:hasSpaceBoundary ?boundingS
urfaceId}',
    initNs = { "props": 'https://w3id.org/props#'})
for row in gPROPS.query(qBoundingSurfaces, initBindings={'spaceURI': rdflib.URIRef
(spaceURI)}):
    boundingSurfaceId = row.boundingSurfaceId
```

Figure 47: An example of a RDFLib query loop

To prevent this from happening, a check was built that queries all properties that are needed in the later analyses. If any of the properties are missing, the script shall identify them.

Missing sensor datapoints in comfort data

Such missing data might also be the case in the sensor time-series datapoints. It can occur that occupants pull out the plug of the comfort sensors or that the sensor (temporarily) malfunctions. Both would result in missing data that cannot be gathered afterwards. To still get viable results, a correction script has been built in the aforementioned data migration script. This correction uses two methods depending on the situation. If only three hours or less of a day are missing, the value of the previous hour shall be used. However, if more than three hours of a day are missing, the average of the same weekday of the past three weeks shall be used. To be able to identify if a value is actually measured or is corrected, including the correction method, this is added as a property to the MongoDB document. Additionally, a REST API shall be created later to be able to query the corrected datapoints.

Section 6

Executing data analyses

In section 5 the required data has been integrated with easy query and post possibilities. This integration provides numerous new possibilities in combining both building and sensor dataflows. To prove its functionality, this section shall conduct an Indoor Environmental Quality (IEQ) assessment over the measured rooms, including room efficiency, and it shall combine these results to provide actionable insights in an optimized ventilation capacity distribution. Detailed documentation is provided in the scripts and the UML activity diagrams are provided in Appendix A3.

6.1. IEQ analyses

6.1.1. Indoor Air Quality (IAQ)

Appendix A4 describes that the air quality is one of the most important criteria for a good indoor climate. The most emphasized aspect these days is the CO_2 concentration that rises due to occupation. Additionally, relative humidity is an essential element that can strongly influence the perceived air quality. Both aspects are measured in the rooms equipped with a comfort sensor, and its values are converted to classes. These classes are defined according to the Arbo criteria, as elaborated on in Appendix A4.6. Both aspects can be improved using increased ventilation. As this is an essential influence, and as the data is available due to the building dataflow integration, a ventilation class has been determined in the IAQ analysis as well. This class considers the ventilation present and compares it to the Arbo criteria. For more information on IAQ requirements, see Appendix 0.

Figure 48 summarizes the data in- and outflows of determining the IAQ. A UML Sequence diagram has been set up showing the interactions between the data sources and how the data is gathered. This diagram is shown in Figure 49 and a larger version is provided in Appendix A5.

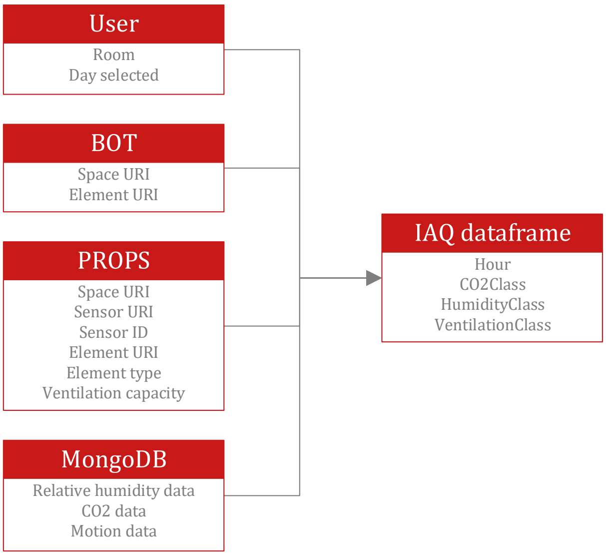


Figure 48: Summary of the data in- and outflows for determining the IAQ

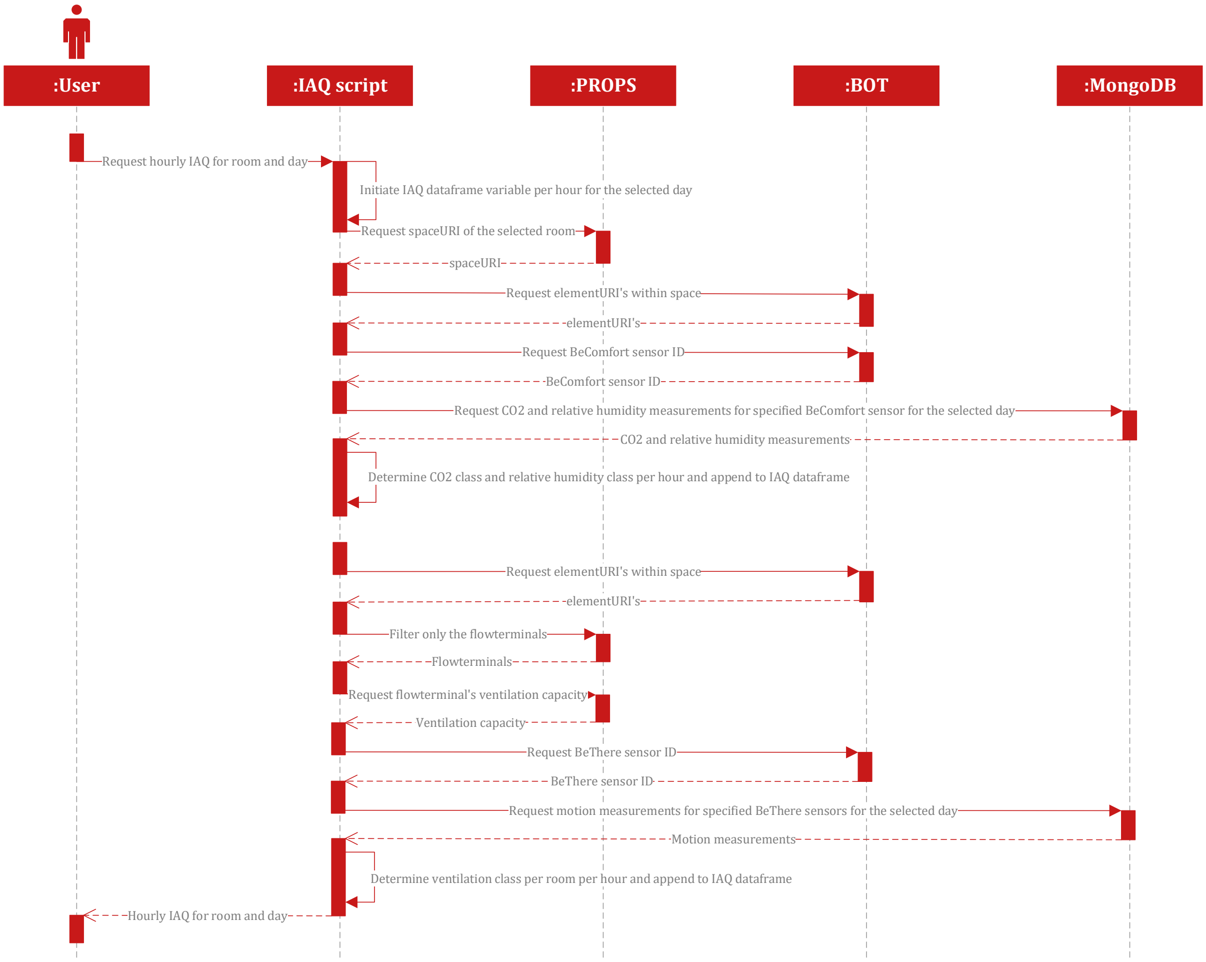


Figure 49: UML Sequence diagram of determining the IAQ

6.1.2. Thermal comfort

Thermal comfort is a more complicated analysis as it depends on more factors than simply portraying the measured sensor values in a different format. Air temperature is measured in the comfort sensor, but this does not reflect the actual thermal comfort of the occupants. In this thesis, thermal comfort has been measured using the ASHRAE 55 norm. The same calculation method has been used as in the online Berkley tool[12]. This requires air temperature, relative humidity, mean radiant temperature, air speed, metabolic rate and clothing insulation as input.

Air temperature and relative humidity can be gathered from the sensor observations, air speed and room function (for the metabolic rate) can be gathered from the PROPS graphs, and clothing insulation can be determined based on the outside temperature that day (meteorological data is being used). These can all be gathered with ease, but the mean radiant temperature is more difficult.

Figure 50 summarizes the data in- and outflows of determining the thermal comfort, including the mean radiant heat. Its UML Sequence diagram is shown in Figure 51 and a larger version is provided in Appendix A5.

[12] https://comfort.cbe.berkeley.edu/

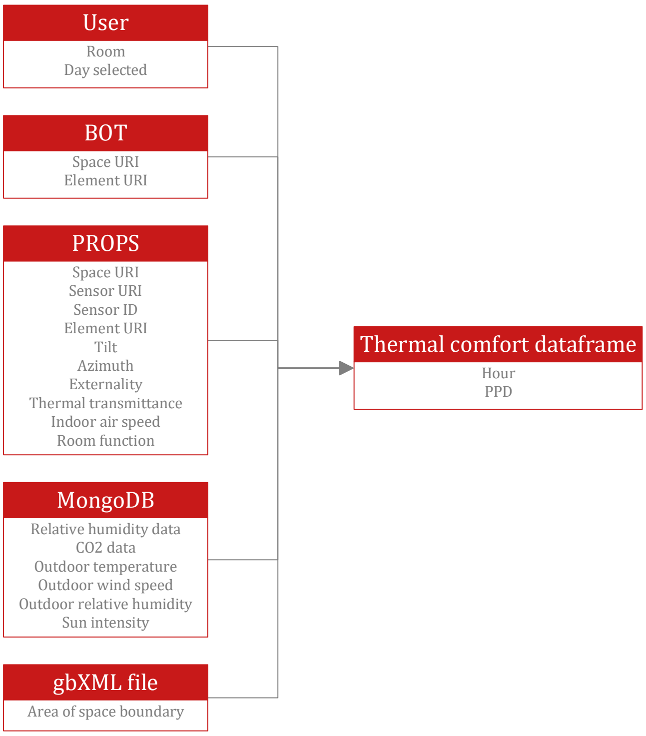


Figure 50: Summary of the data in- and outflows determining the thermal comfort

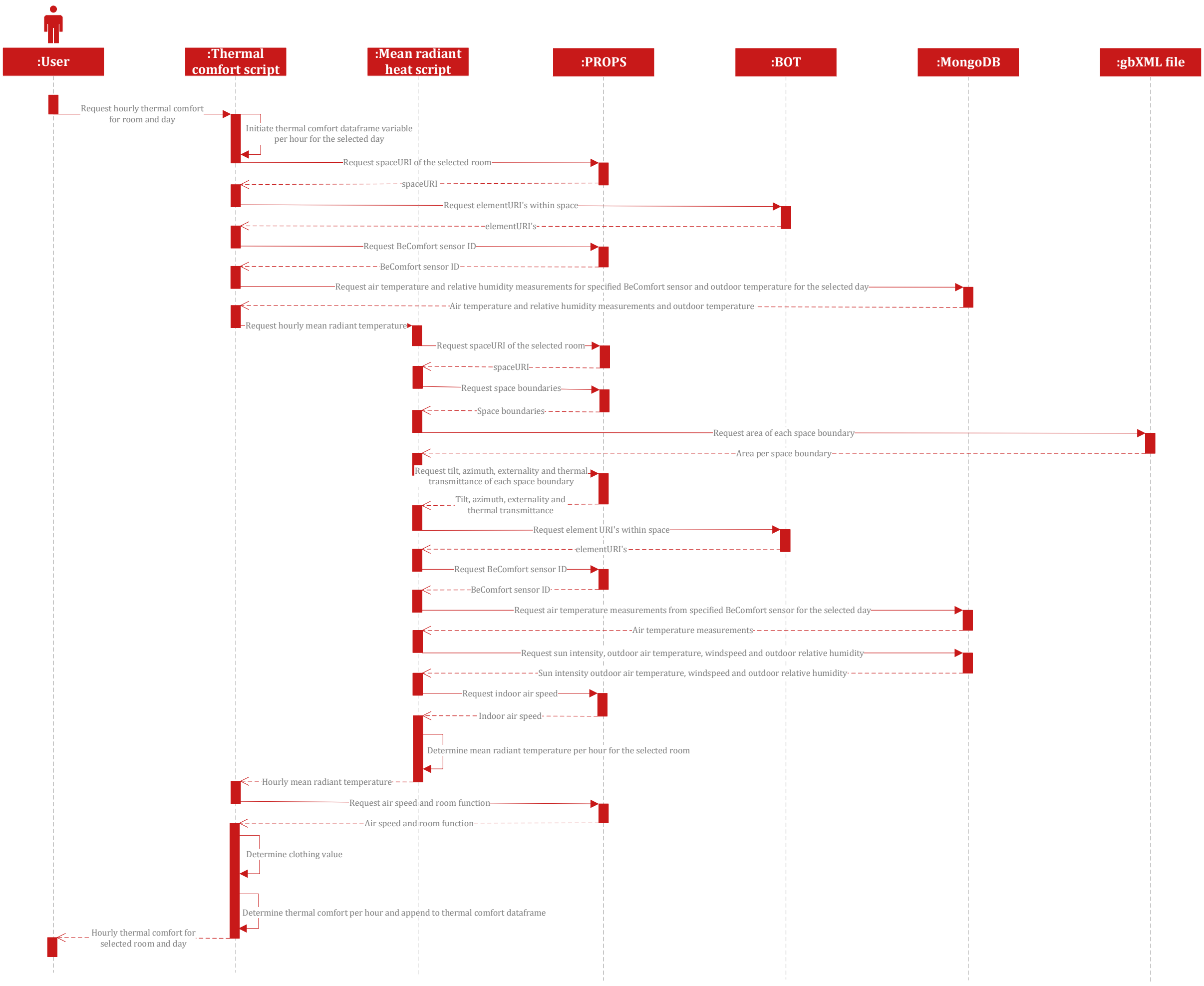


Figure 51: UML Sequence diagram of determining the thermal comfort

6.1.3. Mean radiant heat

To determine the mean radiant temperature for a space at a given time, a separate script has been created ('Function_Mean_Radiant_Heat_final.py'). In the mean radiant temperature calculation, bounding surfaces and their thermal properties are found for the selected room in the graphs, their rough geometry is determined through the gbXML file, the sun's influence based on time of the year and orientation are determined for each external surface, and the time-series data (both sensor data and several meteorological datapoints) are gathered from MongoDB. Calculating the mean radiant temperature of a room requires all integrated data sources and is thereby the perfect example of utilizing the integration created in this thesis.

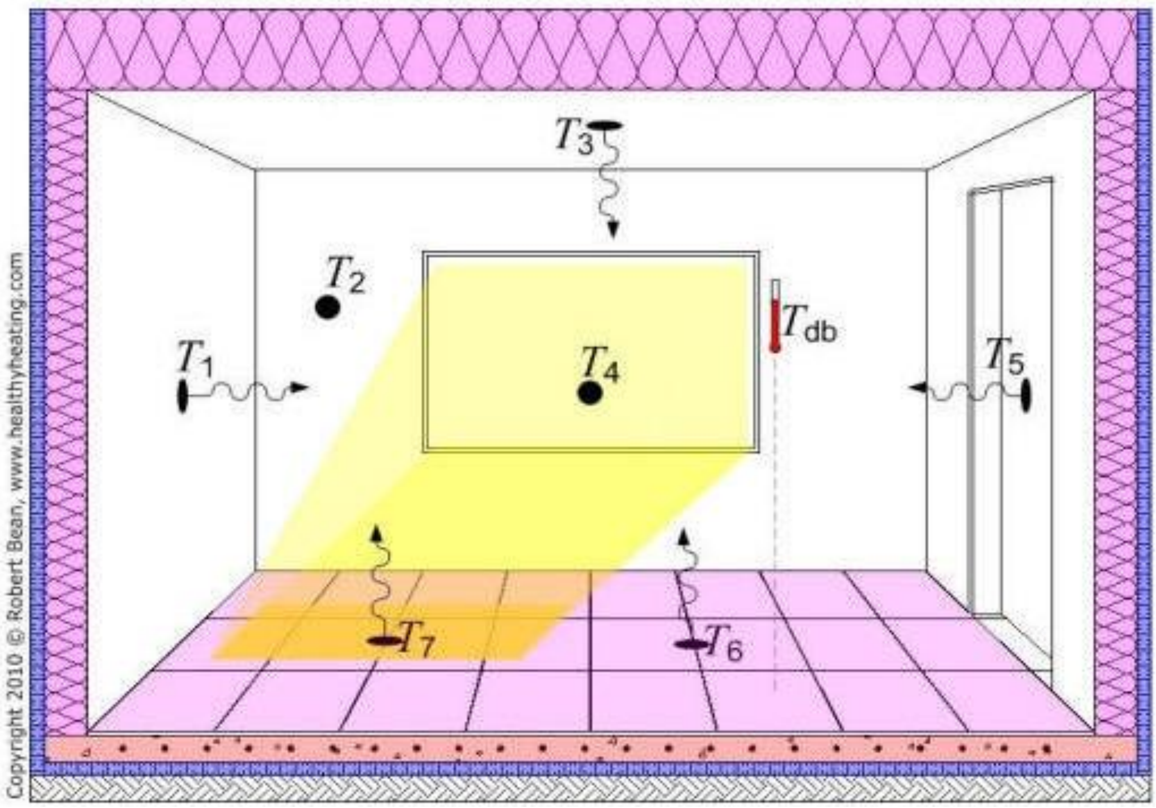


Figure 52 (Bean, 2010): Schematic of a room's mean radiant heat calculation

The mean radiant heat can be calculated using the indoor surface temperature of all bounding surfaces of a room, as shown in Figure 52. This leads to the following formula:

$$T_{mr,room} = \frac{\sum_{i=0}(T_{s,int,i} * A_{s,i})}{\sum_{i=0}(A_{s,i})} \tag{1}$$

$$\begin{aligned} \text{where } T_{mr} &= \text{mean radiant temperature of the room,} \\ T_{s,int,i} &= \text{indoor surface temperature of surface i,} \\ A_{s,i} &= \text{area of surface i} \end{aligned}$$

The full calculation including all its formula's is elaborated on in Appendix A6, as such detail is not relevant for this chapter. Using the calculation in Appendix A6, the mean radiant temperature of the room ($T_{mr,room}$) is calculated per hour, and the resulting hourly dataframe is exported from the mean radiant heat calculation script

('Function_Mean_Radiant_Heat_final.py'). This is then used as one of the inputs to determine the thermal comfort for the room, as also shown in Figure 51.

6.1.4. Room occupancy efficiency

Maintaining good IEQ is significantly easier when the occupation is very low. Additionally, efficient building exploitation remains an essential topic to consider. Thereby, to realise a pragmatic approach, room occupancy efficiency has been included in the analyses as well. The raw motion data has been brought back to two factors: occupancy percentage per hour and the maximum room occupancy considering the ventilation present. The latter is essential for determining what the desired efficiency is. Financially a high efficiency is desired, but it can be argued that the occupation percentage should not result in exceeding the Arbo ventilation norms. Therefore, the building dataflow has been combined with the sensor data to determine what, considering the present ventilation, would be the maximum occupancy percentage. This can then be compared to the actual occupancy.

Figure 53 summarizes the data in- and outflows of determining room occupancy efficiency. Its UML Sequence diagram is shown in Figure 54 and a larger version is provided in Appendix A5.

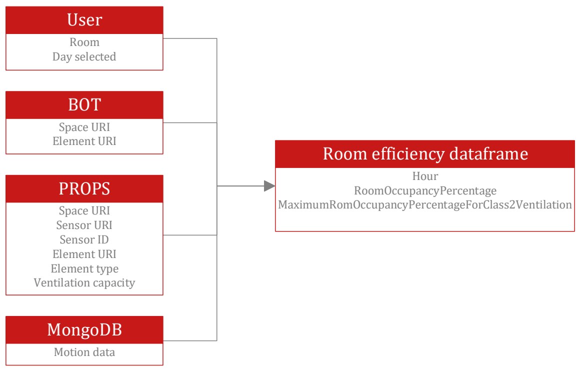


Figure 53: Summary of the data in- and outflows determining the room occupancy efficiency

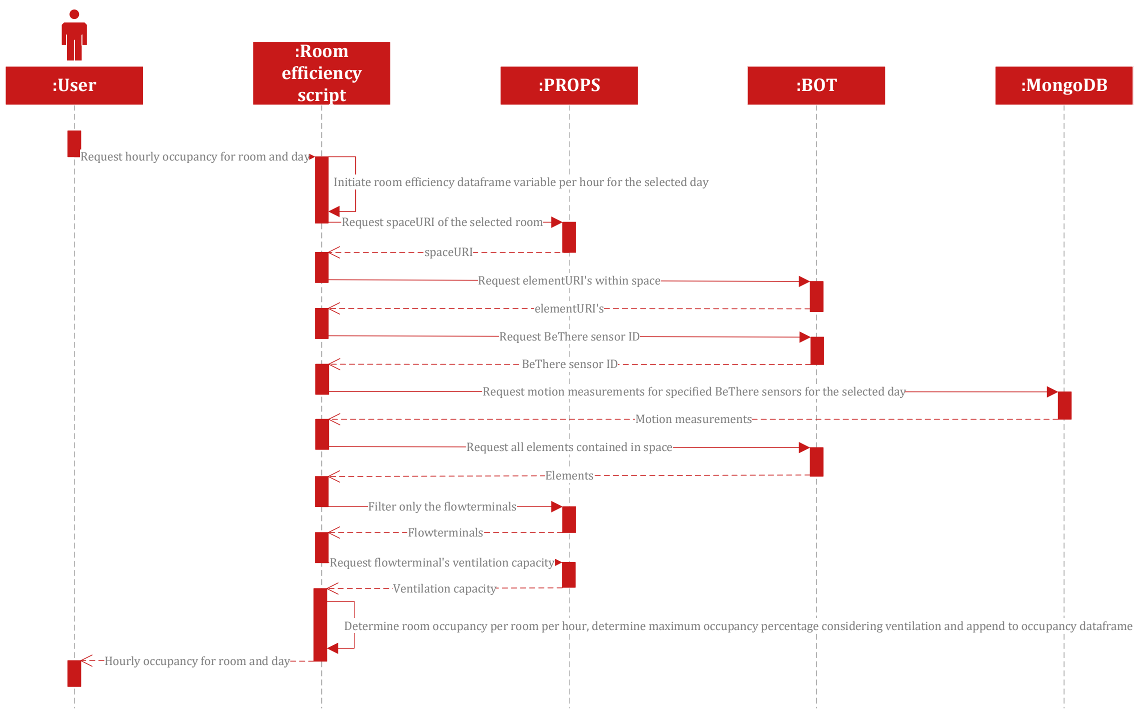


Figure 54: UML Sequence diagram of determining the room occupancy efficiency

6.1.5. Update script

Finally, an update script was written to execute the analyses. This script first determines the last timestamp entry of the overview results for the selected room in MongoDB, then it executes the analyses from that point onwards and appends the new results to MongoDB. The overview results are divided over three frequencies: room, day, hour. The room frequency shows the average results for the specific room, the day frequency shows the average results for a specific day and a specific room, and the hour results show it per hour on a specific day for a specific room. An hour is chosen as interval because the comfort sensors conduct their measurements once every hour. The day and room frequency results are simply an aggregated version of the hourly results. The update script is the only script related to the analyses that should be run. All analysis scripts are linked to the update script and do therefore not have to be touched. From the dashboard, elaborated on in section Section 7, the overview results can be queried. In the code more details can be found on the functionality of the update script. In Appendix 0 a UML Activity diagram can be found describing the workflow of the script.

6.2. Combining results for optimized actionable insights

Apart from more detailed IEQ analyses, the data integration brings more possibilities. Building managers are looking for actionable insights instead of only insights in the current conditions of a building. Using the integrated data, optimization scripts can be run. Examples of these optimization scripts are advice on redeveloping a floor for better KPI's (such as occupancy efficiency or specific IEQ aspects) or optimizing the ventilation distribution throughout a building. As the latter is a rising issue in the building stock of the Radboudumc, this optimization has been created as an example.

6.2.1. Optimized long-term ventilation capacity distribution

Existing buildings in the Radboudumc building stock often have labs in them, taking up a significant amount of the building's ventilation capacity. Additionally, it might occur that certain ventilation capacities of labs may not be merged with others, taking up extra space. If it turns out that the CO_2 concentration is too high, applying extra ventilation shall often not be possible. For example, the assessed Radboudumc building has no space on the roof for an extra air handling unit. However, by redistributing the current ventilation capacity, a lot of the issues can be reduced. Once the building and sensor data are well integrated, simple optimization algorithms can provide actionable insights.

To demonstrate this functionality, an optimization algorithm has been created that optimally redistributes the ventilation capacities considering the occupancy of the past specified time period, for example from the 1st of November 2019 until the 31st of January 2020. It considers both the number of persons present in the rooms between a selected period and the total ventilation capacity in the selected floors. This way the limited ventilation distribution can make as much impact as possible without high tech ventilation systems. Important to note is therefore that this newly created distribution is for the long term to prevent continuous adjustments.

Optimization function

First, using the motion data as input, the occupancy in persons has been determined per room per hour for the pilot period (from the 1st of November 2019 to the 31st of January 2020). After, the ventilation capacity of the selected room is evaluated against the occupancy for each hour. As using each hour for each room in the optimization function would lead to an infinite execution time, the median required ventilation is selected for each day within the selected room. These medians are appended to an IAQ evaluation dataframe, consequently only containing the rooms and the median required ventilation capacity per day. This shall be used as input for the optimization function.

The optimization function has been created using the Scipy Optimize package and has a minimization objective. It contains one objective function using a penalty, one constraint function and a standard bound. The changeable values (x_i) are contained in an array of the same length as the amount of ventilation terminals to redistribute. The following setup has been used in the optimization:

$$\begin{aligned} \min \quad & \sum_{i=0} (|(V_t - x_i)|)^2 + \sum_{i=0} (\sum_{t=0} (|(V_{mV,t} - x_i)|))^2 \\ \text{s.t.} \quad V_t & \geq \sum_{i=0} x_i \\ x_i & \geq 0 \end{aligned} \tag{1}$$

$$\begin{aligned} \text{where } V_t &= \text{total ventilation capacity of the floor,} \\ x_i &= \text{ventilation capacity of room i} \\ V_{mV,t} &= \text{median required ventilation in room i at timestamp t} \end{aligned}$$

The total ventilation capacity of the floor is simply the sum of all current ventilation terminals that are being redistributed. The median required ventilation in room i at timestamp t ($V_{mV,t}$) is the aforementioned required ventilation per room per hour. In the objective function the penalty is built up out of two sub-penalties: (1) a maximum floor capacity that should not be exceeded and (2) the required ventilation per hour that should be met. The former is the sum of a penalty determined per room. The latter is the sum of a penalty determined per hour per room and takes into consideration the aforementioned median required ventilation per hour.

The function returns an array of ventilation capacities in the same order as the input array, making it possible to merge the array with the room dataframe. As the optimization takes some time, the script shall be run periodically, and its results shall be appended to MongoDB. The dashboard shall then load the latest ventilation capacity distribution from MongoDB in less than a second.

Figure 55 shows a summary of the data in- and outflows required for the ventilation optimization script. Figure 56 shows the UML Sequence diagram showing how the data is gathered and what its interactions are. An enlarged version of the UML Sequence diagram is shown in Appendix A5.

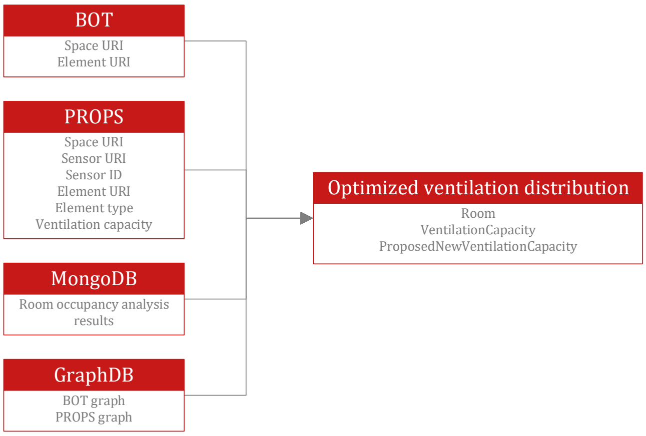


Figure 55: Summary of the data in- and outflows to conduct the ventilation capacity distribution optimization

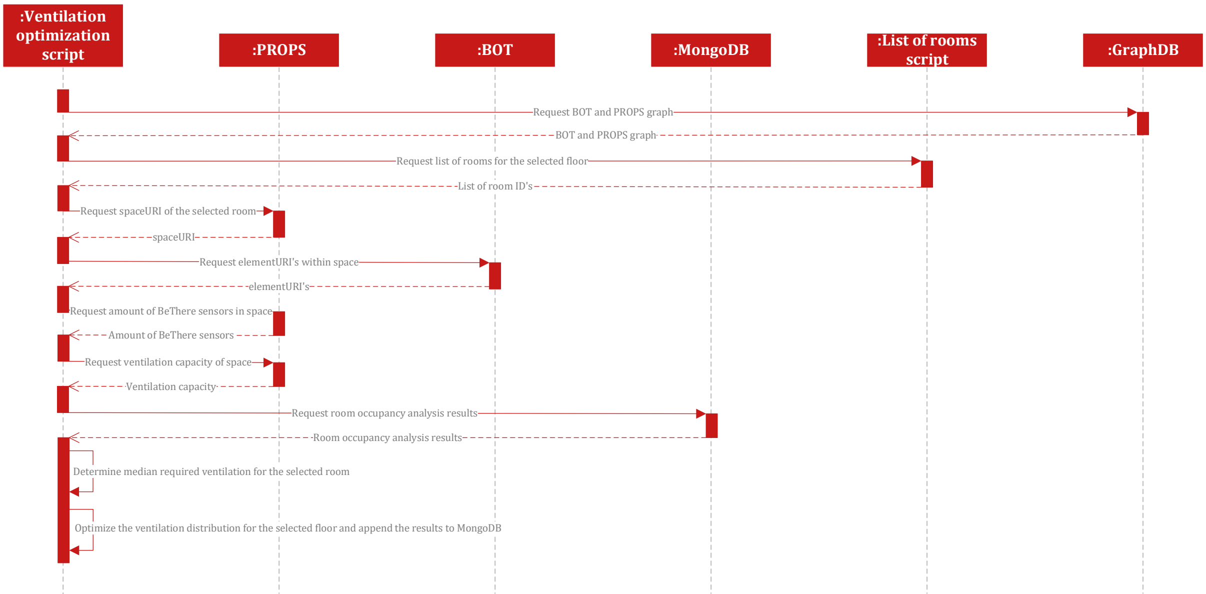


Figure 56: UML Sequence diagram of optimizing the ventilation capacity distribution

Section 7

Communicating results

One of the goals of semantic modelling is easy and flexible sharing of data. Executing all analyses on a local machine does not empower this strength. Therefore, a REST API was created which, using a HTTP Post request, exports the analyses results. However, as such a REST API returns JSON messages, the results are not quite readable. Consequently, a dashboard has been created using JavaScript's module React. This dashboard integrated the HTTP Post requests, allows easy user input and visualises the results in a clear way. Additionally, it demonstrates the easy usage of 2D geometry in creating floor plans. This section elaborates on all these developments.

7.1. Communicate analyses results via a Flask REST API

The strength of semantic modelling is in easy integration and sharing of data. Executing everything in a local Python environment does not empower this strength. Therefore, to easily communicate the sensor data and analysis results, a REST API has been created. As all preparation and analysis scripts have been created using Python, the backend has also been created using Python. Python's Flask module creates the localhost APIs and allows HTTP Post requests to function. As shown in the Python file 'Flask_REST_API_final.py' and in the accompanying UML Activity diagram in Appendix 0, several classes have been created that each request posted input, gather the required data, process the data and return JSON format records. The following classes were created:

- 'RoomAnalysisResult': querying the results of the IEQ analyses;
- 'ProblematicRooms': querying the rooms with IEQ analysis results below a certain threshold;
- 'CheckNaNValues': querying the amount of NaN values per room per floor;
- 'CheckNaNValuesSpecificRooms': querying the details of NaN values for a specific room;
- 'RequestRoomProperties': querying the area, user, function and ventilation capacity of a specific room;
- 'RequestVentilationDistributionImprovement': querying the result of a priorly executed ventilation capacity distribution optimization for a specific room. These results were appended to a MongoDB collection and queried using this API;
- 'RequestFloorCoordinates': querying the 2D room geometry for all rooms on the selected floor. These results are used to create the interactive floorplan in the dashboard.

As the APIs are all running in one script, the GraphDB graphs and gbXML file only have to be parsed once. This is especially useful for the graphs as they can be queried almost instantly once they are parsed in a local Python environment.

7.2. Interact with the REST API in an interactive dashboard

The Flask REST API described in chapter 0 sends JSON packages through HTTP Post requests. Creating a dashboard in Python, for example in Django, results in less flexibility than using JavaScript's React module. Especially considering the future ambitions of integrating a 3D viewer, using React seemed more useful than using Django. To make the HTTP post requests to the Flask backend, JavaScript's Axios module can be used. The JSON results can be processed and finally shown in the dashboard.

By creating this dashboard, the loop of gathering, interlinking, processing and communicating data has been completed. As usually the case with React dashboards, the code is distributed over several separate but interlinked JavaScript and (S)CSS files. Elaboration has been provided in the code, and the code setup is elaborated on in a UML Activity diagram found in Appendix 0. The dashboard has three main components: (1) interactive 2D floorplan, (2) room properties and (3) analysis results.

Figure 57: An overview of the dashboard with the three components highlighted

7.2.1. Interact with a 2D floor plan using coordinates from the graphs

Room ID's are often not easily interpretable and can be better communicated in a floor plan. Additionally, creating a 2D floorplan from graph data requires 2D geometry data to be used, of which the integration has only been demonstrated in the mean radiant heat up until now. To ease the interpretation of the room ID selection as well as to demonstrate the integration of 2D geometry and requesting analysis results, a 2D floorplan has been created in the dashboard by querying the coordinates of spaces from the graph data. The resulting floorplan is thereby an updated situation of the 2D space geometry as found in the linked gbXML file.

7.2.2. Room properties

If a room is selected in the floorplan by clicking on the polygon, the room ID is provided as input for the analyses and the room properties are shown. Once a change occurs in the dashboard, it will render again. The querying of properties of the selected room has been created as a function in the render loop, updating every time a new room is selected. Currently only the area, user, function and ventilation capacity have been selected as room properties, but this can easily be extended in the Python backend script 'Flask_REST_API_final.py'. The goal of displaying the room properties is to prove the integration of almost immediate SPARQL queries in a dashboard using globally parsed graphs, as elaborated on in chapter 5.1.1.

7.2.3. Analysis results

To prove the integration of building data, sensor data and meteorological data, several analyses were set up, as described in section 6. To effectively demonstrate these results, an interactive visualisation has been created in the dashboard. Depending on the required input, data can be posted to the Flask backend and results shall be returned to the dashboard. The following analyses can be queried:

- Overview of IEQ analyses results: the overview of results is simply querying the outcome of the IEQ analyses between the selected time periods for the selected frequency. Currently companies that provide a data analytics platform often show the sensor readings over time in graphs. A similar functionality is therefore desired for not the sensor readings but for the analysis results. Any room can be queried for any analysis between any timestamp range.

- Overview of problematic rooms: the problematic rooms are most interesting as something should change for these rooms. To easily query the problematic rooms, a threshold can be set and all rooms below that threshold shall be displayed. This way, the analyses results are easily utilized into insights that a building owner can tackle.
- Actionable insights in improving the ventilation capacity distribution: as elaborated on in chapter 6.2, the analyses have been taken one step further into actionable insights. Simply suggesting that the CO_2 levels are too high and thereby the ventilation capacity should be increased is not the information that a building owner can directly use. It might well be that only a limited amount of mechanical ventilation capacity is available. Linking the CO_2 levels to the ventilation capacity and extracting the possibilities is what would provide a building owner with a more useful actionable insight. Therefore, the occupancies and ventilation capacities have been analysed and a new optimized ventilation capacity distribution per floor can be queried in the dashboard.
- Continuity of sensor measurements: as described in chapter 5.2.4, due to a variety of reasons it may be that the sensors provide NaN values. It is essential for any user of the sensor readings and the conducted IEQ analyses to know if there are NaN values found in the readings. As elaborated on in chapter 5.2.4, some NaN sensor readings have been corrected when migrating the data from the Microsoft Azure database to the MongoDB database. The dashboard functionality of querying NaN values thereby returns a bar chart of all NaN readings per floor as well as how they are corrected. Additionally, a button has been created that returns a more detailed table of NaN readings for a specific room.

Section 8

Radboudumc use case results

In addition to visualizing the results in the interactive dashboard, a brief summary of the findings at the second and third floor of the M245 building of the Radboudumc shall be shown. The elaborated analyses shall be the same as the analyses described in section 6.

8.1. Continuity of sensor measurements

Sensors might provide NaN values due to connectivity issues or, most often the cause, the human component that is involved. The most probable cause of sporadic NaN values is that a person has unplugged the sensor, for example to charge their laptop. Logically, the sensor shall not conduct any measurements during that time, leading to NaN values. Occurrence of such sporadic NaN values has also been the case at the pilot area, but the cause remains unknown. A building owner that applies such sensors must however realise that there is always a risk of sensor malfunctioning, either due to hardware issues, connectivity issues or human interference. At the Radboudumc pilot the following NaN values were identified and corrected using the methods specified in the legend:

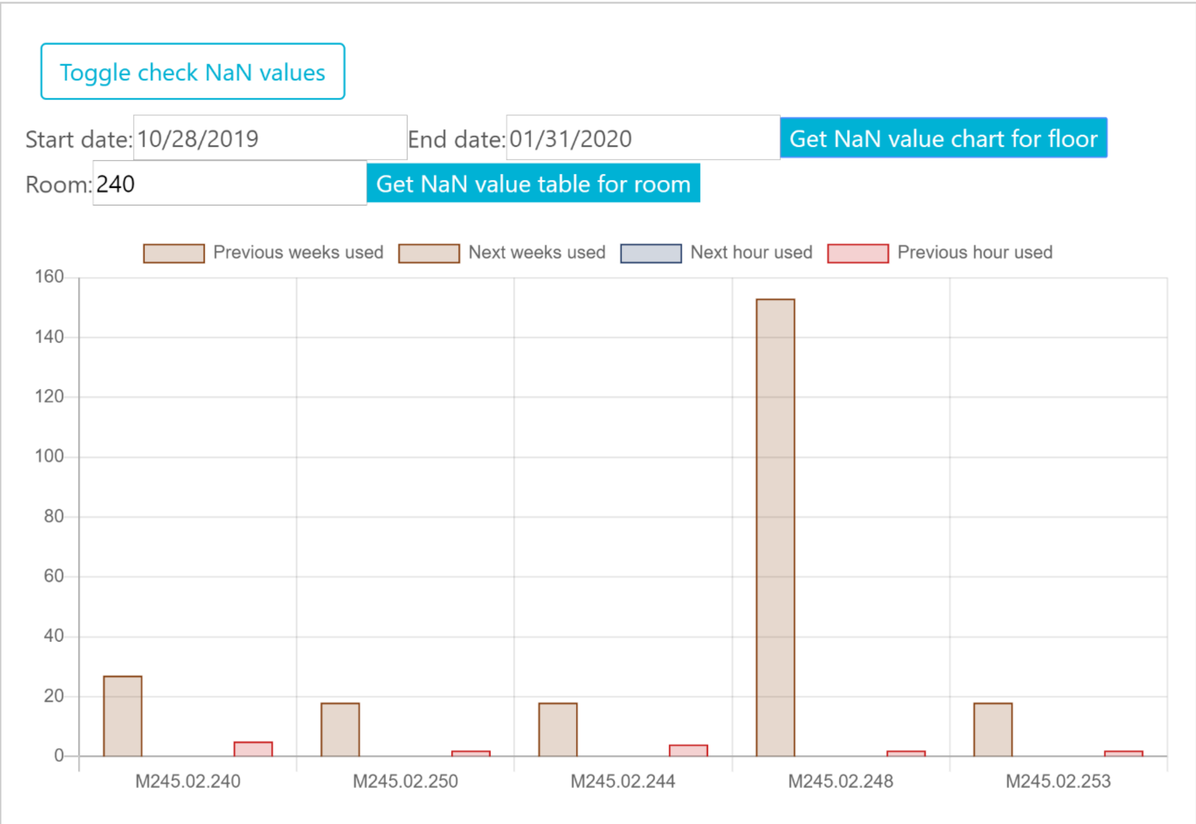


Figure 58: NaN values found for comfort data on the second floor

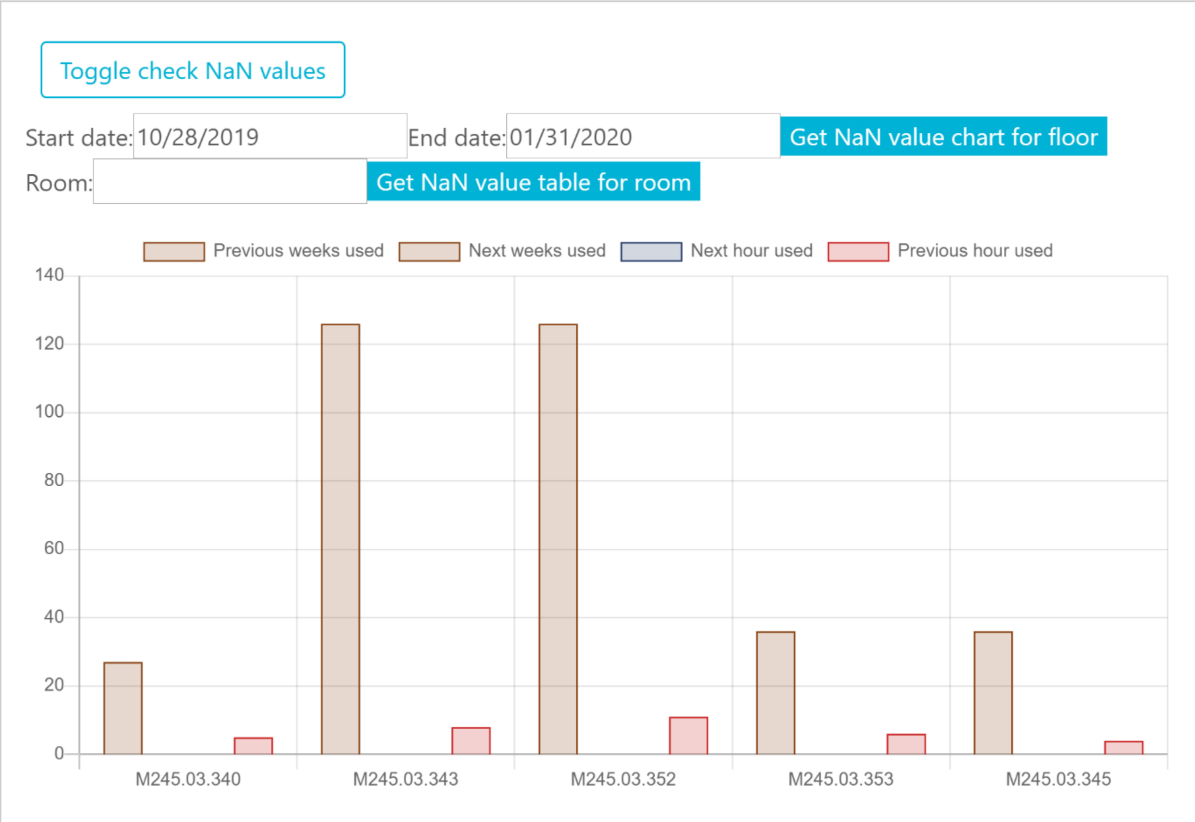


Figure 59: NaN values found for comfort data on the third floor

8.2. IEQ analyses results

To assess the state of the IEQ for both measured floors and to identify where possible adjustments should be made, the problematic rooms are identified. To do this, the analyses results of the full pilot period (1st of November 2019 until the 31st of January 2020) that are aggregated per room shall be used. The reason for using the aggregated results is that it might occur that a room temporarily has a bad IEQ. However, this does not mean that the room should be physically or behaviourally adjusted, which is the goal of identifying the problematic rooms. As shown in the code, the thresholds that identify a room as being 'problematic' are set as follows:

- Room occupancy percentage < 20%;
- CO2 class = 1;
- Humidity class = 0;
- Ventilation class = 0;
- PPD > 15.

If any of these thresholds are met, the room is identified as 'problematic'. This results in the following outcome for the second (Figure 60) and third (Figure 61) floor of building 'M245' at the Radboudumc:

Get problematic rooms

Room	RoomOccupancyPercentage
M245.02.249	7%
M245.02.249	3%
M245.02.250	9%
M245.02.245	10%
M245.02.244	12%
M245.02.248	10%
M245.02.253	19%
M245.02.251	9%
M245.02.242	6%
M245.02.252	10%
M245.02.252A	9%
M245.02.241	10%
M245.02.243	7%

Get problematic rooms

Room	PPD
M245.02.244	20
M245.02.248	21

Get problematic rooms

Room	VentilationClass
M245.02.250	0
M245.02.244	0
M245.02.253	0

Get problematic rooms

Room	CO2Class
No rooms with a CO2 class of 0 found in selected floor	

Get problematic rooms

Room	HumidityClass
No rooms with a humidity class of 0 found in selected floor	

Figure 60: Overview of problematic rooms on the second floor per category

Get problematic rooms

Room	RoomOccupancyPercentage
M245.03.342	10%
M245.03.343	17%
M245.03.349	11%
M245.03.348	12%
M245.03.352A	17%
M245.03.352	7%
M245.03.353	15%
M245.03.350	8%
M245.03.351	18%
M245.03.344	17%
M245.03.341	17%
M245.03.345	13%

Get problematic rooms

Room	CO2Class
M245.03.353	0

Get problematic rooms

Room	VentilationClass
M245.03.352	0
M245.03.353	0

Get problematic rooms

Room	PPD
M245.03.352	26

Get problematic rooms

Room	HumidityClass
No rooms with a humidity class of 0 found in selected floor	

Figure 61: Overview of problematic rooms on the third floor per category

8.3. Actionable insights in improving the ventilation capacity distribution

The ventilation capacities are not well distributed in building 'M245' of the UMC. As shown in Figure 62, ventilation pipes are often present but only a few rooms have mechanical ventilation.

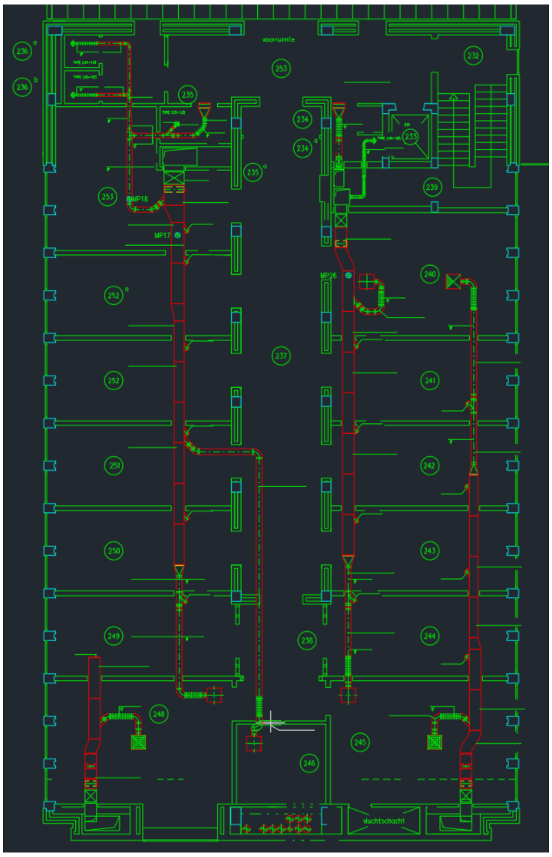


Figure 62: Ventilation distribution on the second floor of building M245

This inequal ventilation capacity distribution leads to some unnecessarily well-ventilated rooms and some badly ventilated rooms. For example, room M245.03.353 contains 7 workplaces but it does not have any mechanical ventilation. As visible in the IEQ analysis results, the CO_2 concentration is of class 0 almost all the time someone is present. During the Christmas holidays when no one was present, the CO_2 concentration was fine. However, literally all other working days have had a CO_2 class of 1 or 0. As described in Appendix 0, this means that the CO_2 concentration has been respectively between 800 and 1000ppm or above 1000ppm.

Using the optimization algorithm described in chapter 6.2.1, the following ventilation capacity distribution has been suggested for the second and third floor of the M245 office building:

Get ventilation improvements

Room	VentilationCapacity	ProposedNewVentilationCapacity
M245.02.248	225	112
M245.02.243	0	40
M245.02.241	0	75
M245.02.249	0	44
M245.02.250	0	44
M245.02.242	0	24
M245.02.251	0	42
M245.02.252A	0	25
M245.02.244	0	24
M245.02.240	220	13
M245.02.245	220	89
M245.02.252	0	39
M245.02.253	0	122

Figure 63: Proposed new ventilation capacity distribution distribution
for the second floor

Get ventilation improvements

Room	VentilationCapacity	ProposedNewVentilationCapacity
M245.03.342	140	63
M245.03.351	0	78
M245.03.350	0	59
M245.03.340	140	11
M245.03.352	0	51
M245.03.349	0	80
M245.03.353	0	189
M245.03.341	140	79
M245.03.352A	0	11
M245.03.344	140	48
M245.03.345	300	107
M245.03.343	200	123
M245.03.348	0	109

Figure 64: Proposed new ventilation capacity
for the third floor

Section 9

Conclusion and discussion of results

This section shall finalize the thesis report. First, a conclusion shall be drawn of the conducted work. After, the limitations of the conducted work shall be discussed as well as a reflection on the research questions. To place this thesis in the context of overall progression within both the scientific community and the society, the scientific and societal contribution are described. Additionally, future scientific research, future applications and possible future smart hospital initiatives are elaborated on.

9.1. Conclusion

Building data and BIM capabilities are mostly being applied in the development and construction phase of the building lifecycle, leaving the exploitation phase behind. Not surprisingly as prior to effectively utilizing building data in exploitation processes, the data must be organized and integrated in a well manner. This research aimed to take a new step in effectively integrating building data with other sources, especially the increasingly frequent sensor time-series data. It did so by stating the following main research question:

"How can building and sensor dataflows be effectively combined to provide actionable insights in improving indoor environmental quality and occupancy efficiency in an existing building stock?"

Using the created dataflow integration, various sources can be used to analyse the hourly IEQ and occupancy efficiency of an office building. With these results it has been proven that building data can complement time-series dataflows in achieving more useful results than only portraying the measured sensor values. Using building information, measured sensor values can be placed into context and actionable insights can automatically be derived instead of manually interpreting the sensor results. The integration created in this thesis provides a solution for, but is not limited to, suggesting actionable insights on improving IEQ. With the presented integration, numerous building data aspects can be easily queried and updated, easing its usage in other applications. The potential of the presented building dataflows to be used in areas such as IEQ or adaptive floor plans has not been realised yet. However, as the frequency of applying IoT devices, among others sensors, increases, continuous building monitoring and analyses are enabled. Through the integration of heterogenous data types such as demonstrated in this thesis, the various data sources can be effectively utilized. Instead of considering buildings as a final product after delivery, involved parties can consider buildings to be dynamic objects that can be continuously improved, benefitted from and learned from.

9.2. Critical evaluation of results

Even though the integration works for the IEQ analyses as this thesis intended to, there are some limitations to the integration that shall be discussed in this chapter.

Reproducibility of the conducted integration

Essential in developing an integration such as the one in this thesis is its reproducibility. In essence, this integration is reproducible for a new case study but probably errors are going to occur related to the data quality and format. When reproducing the integration, it is most important to notice its weak points: the human component involved in linking the sensors with the building data, and the content and frequency of the measured sensor values. The currently used sensors contain a few datapoints per measurement, for example for the comfort sensors 'Timestamp', 'DataType', 'SpaceName' and 'Average'. If for instance 'SpaceName' is labelled 'Spacename', the datapoints shall not be found. If careful attention is spent on these weak points, the integration should be fully reproducible.

Sample size

The case study at the Radboudumc was conducted on two floors of an office building. When applying the developed integration on a larger data set the performance shall probably suffer.

The data conversion process

Starting with the very first part of the integration, the data conversion process. The current conversion process of a Revit file to the required data formats is still manual. Ideally this would all be included in the integration chain without human inference. Additionally, as Revit is used to export the gbXML format, a non-open source application is required for the integration to work. Ideally, Revit, being a paid application, would be left out of the integration entirely.

Labelling of spaces and elements

The integration developed in this thesis links the sensors in the building data to the sensor time-series data using a mutual datapoint. This mutual datapoint is the sensor ID, which is found in the building data through the sensor property 'comments_simple'. Ideally, a more dedicated property should be used that takes in a string as well, but this property must also be converted from IFC to LBD. The current version of the IFC-LBD conversion considers a selection of IFC properties but not newly created ones. A dedicated property could therefore be created in Revit, but they shall not show up in the LBD graphs. Consequently, this thesis used 'comments_simple' as this was converted and could take in a string. Additionally, a direct check should be implemented that verifies if the mutual datapoint is correct. Currently, even though it is checked after exporting it in the data integrity check, the linkage between the sensor building data and the sensor time-series data is relatively error prone. In the future, as described in chapter 5.2.3, the geometry of a sensor can be used to automatically link the sensor to the building data and thereby removing the human component.

Also, the mean radiant heat calculation needs to know the externality of each relevant surface, making correct labelling essential. The graph property 'isExternal_simple' is used to determine it, but if this is not labelled correctly, the mean radiant heat analysis produces corrupt results. This property is automatically created by Revit and should therefore be correct. However, knowing such fragile points in an analysis is essential.

Sensor measurement continuity

Missing values may occur due to various reasons, as stated in chapter 5.2.4. One of the reasons is that most comfort sensors used a power plug, which can be unplugged. According to BeSense, it regularly occurs that occupants temporarily unplug a comfort sensor to charge one of their appliances. It is therefore highly probable that this caused some of the missing values described in chapter 5.2.4 as well. Having an accessible power plug is a weakness in using such a sensor, which should be considered when choosing the type of sensor to use. It is preferred to use a sensor that has a battery.

When migrating the data from Microsoft Azure to MongoDB, the script currently uses a loop to check the data presence of each hour for each day for each sensor. This results in a higher data certainty than only checking the last timestamp and appending all datapoints after that, but this also takes a very long time to execute. A new method should thus be found that ensures the presence of all measurements but has a shorter execution time as well.

gbXML in the loop

In the developed integration, gbXML is still in the loop for the mean radiant temperature calculation. Parts of the gbXML data are converted to the PROPS graph, but it remains in the loop. Sadly, the gbXML file is currently stored locally, as it contains information that cannot be directly appended to a building element in the graphs. The reason is that the surfaces in the building graphs are not equal to the surfaces in the gbXML file. The building graphs define surface URI's which represent a whole surface element, for example a wall from one side of the building to the other. As gbXML was created for energy analyses, the gbXML file splits up these surfaces when they are enclosing more than one room. This makes the geometry of the gbXML file ideal for indoor climate analyses, but also makes it impossible to append the individual area of a gbXML surface to a surface element in the building graph. Currently the gbXML graph stays in the loop as it works well. However, this makes storing the geometry data of the split walls in GraphDB difficult. If this is to be solved, new surface URI's can be appended to the building graphs, each representing a split surface from the gbXML file. Consequently, the surface tilt, azimuth, width, height and coordinates must be copied to the newly created surfaces.

Geometry data

3D mesh geometry is an important part of building geometry data. Therefore, the 3D geometry meshes have been integrated, as described in chapter 5.1.4. But no viewer has been created yet. Also, if the viewer had been created, the graphics seem to be without colour, making the visualisation less powerful. Using IFC might be a better option for more powerful visualisations, but, as described in chapter 2.2.1.3, careful attention must be paid to what information is included in the IFC file.

Another drawback of the current geometry integration is that desk geometry is not included in the gbXML data. Motion sensors are placed under desks and, using the developed integration, the desk that the sensor is attached to can be easily identified. However, these desks cannot be automatically drawn into the floorplan yet as gbXML data does not contain geometry for desk elements. If these desks are to be automatically drawn into a floorplan, another source of geometry should be found.

Temporary properties

More types of data can easily be linked to the building graphs. An example would be malfunctioning building elements that are appended to the graph according to the level 3 OMG ontology[13]. This ontology contains a time predicate which could indicate the issue timestamp and the resolving timestamp. Once it is resolved it should then be removed from the graph and appended to a log. This functionality should not be difficult to create using the developed integration, but it has not been demonstrated in this thesis.

Similarly, this ontology property could be used to compare the as built building performance to the currently measured building performance to provide insights and indicate the necessity of redeveloping. This functionality has currently not been demonstrated in this thesis as well.

9.3. Reflection on research questions

This chapter shall first reflect on the research questions defined in the beginning of this thesis, after which it shall elaborate on the scientific and societal contribution. Additionally, it shall discuss the future research directions and industry applications.

9.3.1. Literature review questions

What building/sensor data integration frameworks are currently suggested in literature, how do they differ, and which are most promising for monitoring building efficiency and IEQ conditions?

An extensive literature review has been conducted to thoroughly understand the types of integration frameworks and their benefits and limitations. A list of requirements was set up for the framework that was to be applied in this thesis. Considering these requirements, the hybrid approach 'semantic web + relational database' was chosen.

How can building dataflows be effectively represented and stored following the chosen integration framework?

[13] https://www.projekt-scope.de/ontologies/omg/

After narrowing the scope on the chosen framework, the methods used within the framework have been researched in further detail. The one-file approach of semantic modelling was distinguished from the domain-specific approach. Clear benefits and limitations of both approaches have been stated, and the domain specific ontologies approach was favoured over the one-file approach.

How can sensor dataflows be effectively gathered, stored and linked with the building dataflows, and which technologies are most suitable for this case?
Even though semantic modelling would not be considered any longer for sensor data due to the chosen framework, a semantic link should still be made between the (non-)relational database and the building data. Several optional ontologies were researched and briefly discussed. Also, relational databases were suggested in the framework, but non-relational databases are argued to have general benefits and limitations over relational databases. Therefore, research has been conducted in the differences between the database types, which has been used as input in the methodology where specific relational and non-relational databases were objectively compared.

9.3.2. Radboudumc case study questions

What form or forms of geometry representation is most suitable for this case, and how do they differ?
In the developed integration, it was decided that geometry should be represented in two formats. One format would contain simplified and semantically rich geometry for analysis purposes, gbXML fulfilled this role. The other format would be for visualisation purposes and tessellated mesh geometry was chosen. Considering the need to link specific mesh elements to specific building elements, COLLADA was chosen as data format.

How can the Radboudumc building and sensor dataflows be transformed to the required formats?
Several data formats were distinguished and conversion methods were found. Regarding the building data, a Revit model has been considered as the initial data. The Revit exporter has been used to convert to IFC and gbXML. IFC has then been converted to Turtle files and a COLLADA file using open source converters, as described in chapter 4.1.

How can the different formats be queried and how does each format perform?
Within this thesis Python was used to conduct all IEQ analyses. Therefore, the queries would have to be executed using a Python terminal. Graphs were initially queried using the RDFLib Python package, but this package required to parse the whole graph and had no posting possibilities to remote graphs, such as graphs in GraphDB. Another method was found using the package SPARQLWrapper, which queried and posted using direct SPARQL endpoints. The benefits were that these queries do not have to parse a whole graph and could therefore execute a query in 2 seconds, and that posts could be made to a remote graph. As the graphs in this integration were remotely (localhost) stored in GraphDB, this was a necessity. However, when making numerous queries in one Python file, 2 seconds per query is a long time. As locally parsed graphs through the RDFLib package could be queried near instantly, parsing the whole graph would be faster than making numerous SPARQL queries. Additionally, Python has the functionality to define global variables, as long as the path to the global variables is imported correctly. Applying this functionality, a graph could be locally parsed only once for executing multiple codes. Also, when defining several API classes in one script, the graphs only have to be parsed for the initialization of the API.

Consequently, for querying purposes, the method of parsing the whole graph in global variables was used. For posting the SPARQLWrapper method was used, leading to a hybrid approach.

How can the chosen formats be linked to easily query the data for analytics?

For semantic modelling the domain-specific approach was chosen, mainly through the BOT, PRODUCT and PROPS graphs. These graphs can be linked easily though the URI's that are present in all graphs. The BOT graph can be used to determine the relations between building elements, the PRODUCT graph for the type of building elements and the PROPS graph for its properties.

More of a challenge is the linking of graph elements with elements in the other data formats. The graphs remain the backbone of the building data and must thus be considered as the main building data. For specific geometry details however, the gbXML file must be queried. Similarly, for specific geometry meshes, the COLLADA file must be queried. Linking the gbXML and COLLADA elements to graph elements happens through mutual datapoints. The bat ID turned out to be a suitable mutual datapoint and has therefore been used as the linking property.

How can the indoor environmental quality be analysed using the queries?

The chapters in Appendix 0 have researched the influences on specific IEQ criteria, such as thermal comfort. Using this knowledge as input, (existing) methods were developed to determine the state of each IEQ criteria at a certain point in time. The state of the criteria was converted to a value comparable over the various IEQ criteria, namely the classes within the Arbo norms. Once the goal and the methods had been determined, the analyses were coded into separate Python scripts that could be imported into other scripts. The query methods that were previously set up were reused numerous times in exactly the same way.

How can the analysis results be effectively accessed by interested parties at the Radboudumc?

Essentially the output would be a REST API that allowed a HTTP Post request to query and gather the results through a JSON file. However, a JSON file is difficult to effectively read, and consequently an extension was sought. JavaScript's React module was explored, and a dashboard was created that integrated the previously set up REST APIs. In the dashboard all the APIs came together, and the results were demonstrated in a more effective manner.

Can the created method of combining building and sensor dataflows be seamlessly replicated?

Technically yes, but, as discussed in chapter 9.2, labelling of data is essential in this integration. Therefore, if the labelling guidelines are followed and the data is complete and formatted correctly, the integration should completely function. Every script has been dynamically coded, so any size or room amount etc. shall be accepted as an input.

Using the created integration, what future directions in realising and maintaining healthy buildings should be explored?

As the developed integration is not a goal itself but rather a method of achieving other final goals, the future directions and applications are essential to thoroughly cover. The next chapters shall therefore extensively cover both the scientific and societal future directions.

9.4. Scientific contribution

The potential synergy between building and sensor dataflows has been researched, but successfully integrating sensor data with building dataflows remains a challenge (Dave et al., 2018; Tang et al., 2019). Additionally, the current research on the integration of building and sensor dataflows focused almost exclusively on the automation of transmitting sensor data to building models (Tang et al., 2019), often not enabling operational and analysis possibilities with the integrated data.

The results of this thesis demonstrate an integration that successfully combines building and sensor dataflows, and that uses the combined dataflows for analysis/improvement purposes. These results prove that building data can complement time-series dataflows in achieving more useful results than only portraying the measured sensor values. It shows that several linked building data approaches, which are thoroughly described in literature, work well in practice. Additionally, it developed the code to easily work with these various data sources. Using the developed integration, follow up research can fully focus on the next steps of effectively utilizing the synergy between numerous data sources within buildings.

9.5. Societal contribution

One of the most essential issues in building and sensor dataflow integration is convincing companies to engage in gathering the required data in the first place. Clear business cases are needed to set such projects in motion, also at the Radboudumc. If there is more commercial interest, more support for research in the scientific community might follow. Therefore, as stated in the beginning of this thesis, the societal importance shall mostly be focussed on the possibilities that such a data integration can bring forth. This thesis demonstrates that successfully utilizing the combination of dataflows is possible, that analyses can be conducted and that insights can relatively easily be gathered.

9.6. Future research and applications

9.6.1. Future scientific research

The developed integration is not a goal itself, but rather a means to achieving other goals. Apart from the research area of integrating building data, several related areas can use the results of such data integration in their future research. This chapter shall describe several areas and their future research.

Building data integration

Starting with the research area of this thesis. Building data can serve as a central data backbone to which numerous other sources are attached. This thesis only integrated various building data sources with both sensor- and meteorological time-series data. This is a great first step, but more sources can be integrated, such as appliance data, production data, (anonymized) employee data, maintenance data, malfunction/complaints data or social data. The latter shall be described in the next paragraph. Future research in the area of building data integration would be to effectively utilize more of such sources to the central graph building data backbone.

In all of these sources, including the time-series sources that were used in this thesis, temporary properties can be relevant. Thus, another future research would be to effectively utilize such temporary properties. This includes appending new properties, updating properties, removing properties and automatically appending them to a central backlog of outdated properties.

Social data

One of the future data sources named above is social data. This thesis focussed on the objective and physical aspects of a building, but how users interact with and experience a building is even more important. After all, the comfort objective from the 1980's has been transforming towards environments in which people can thrive physically but also socially and mentally (Clements-Croome, 2019). For example, cell-offices often realise great IEQ conditions which would improve the health and wellbeing of employees, but IEQ conditions provide no insight in whether the user actually likes it. Consequently, feedback on the social elements of these cell-offices is often less positive (Cordero, 2017). Without effective usage of social data, feedback on how well a building performs in these subjective areas is difficult to assess. Monitoring buildings on a social level as well may provide useful input for facility managers, designers and users themselves (Peters, 2018).

I believe the linked building data approach can form an integration solution in effectively using this data. The Radboudumc has experimented with such social data in the past using for example feedback poles. These feedback poles, located at coffee machines or elevators, contained a Likert-scale questionnaire about various topics. Another example is Davis (2016) which used user feedback on architectural preferences through the WeWork app as input for creating or redeveloping their new offices. Building forth on the data integration in this thesis, and thus using the linked building data approach, such social data could be effectively and automatically utilized.

Building maintenance

By integrating more advanced and precise analyses on building physics performance of spaces, the state of certain elements in a room can be estimated. These estimations can provide a basis for the decision-making process of replacing or upgrading certain building elements. Using the data integration developed in this thesis, several data sources can be easily used as input for such analyses.

Long-term occupancy prediction

One of the ways to leverage sensor data is to predict occupancy. Occupancy prediction is a widely researched field with different aims. Building energy use and demand is one of the most popular research aims (Shi et al., 2017; W. Wang et al., 2018, 2017; Z. Wang & Srinivasan, 2017), often related to prediction of building occupancy and occupant behaviour (Chen, Jiang, & Xie, 2018; D'Oca & Hong, 2014; Saha, Florita, Henze, & Sarkar, 2019; Salimi, Liu, & Hammad, 2019; W. Wang et al., 2017). In all these articles, occupancy is predicted in the short term for mostly energy-related anticipation purposes. But, predicting occupancy for the coming month(s), as would be relevant for building redevelopment, is not done much yet. Building forth on the data integration in this thesis, such algorithms can be developed.

Occupancy estimation using CO_2 sensors

Sensors are, for now, expensive. For the average building owner to fully equip their buildings with sensors, strong business cases must be formed and the prices or required number of sensors needs to drop. Reusing the results that sensors produce, and thereby making another type of sensor obsolete, might strongly reduce the required number of sensors. Multiple types of sensors are being used to measure and predict building occupancy. PIR (counting) sensors are one of the types but for example CO_2 is another. These are, as well as camera's and other types, often combined for more accurate prediction or sometimes even to train models. An example of training would be to use camera data to train the CO_2 model, as Yang et al. (2018) does. This might bring the opportunity to use CO_2 sensors in meeting rooms for both occupancy prediction and improvement of comfort/performance levels. If this would prove to be a reliable method, then no count sensors would be necessary in meeting rooms any longer. Of course, the required accuracy depends on the use case, but further research in this area might stimulate the diffusion of sensors throughout the existing building stock.

Parametric design

The field of parametric design aims to explore new designs by changing a few parameters. Designing thereby emphasizes on the design process instead of the final design. Parametric design requires numerous building data inputs, depending on the use case. If it is to be applied on redevelopment instead of new construction, data on the existing building must be easily integrated. The integration developed in this thesis might provide a solution for effectively gathering the required building data sources, but the parametric design algorithms must be researched in greater detail.

A lot of research is being conducted in related fields, such as Multi-Criteria Decision-Making (MCDM) and Multi-Objective Optimization (MOO). Integrating the combination of MCDM, MOO and BIM into the early design process, the strength of BIM for a constructive communication between professionals is maintained, multiple design alternatives can be created using MOO, and an argued trade-off can be made in conflicts with more than three (conflicting) objectives (Jalilzadehazhari & Johansson, 2019). However, this combination has not been explored thoroughly in literature. To the best of their knowledge, Jalilzadehahari and Johansson (2019) argue that they were the first to apply it on visual comfort, thermal comfort, energy consumption and life cycle cost, while these are regularly used in exploring such techniques. Further research is to be conducted to fully utilize the capabilities that parametric design may provide for creating next-level designs in an efficient manner.

9.6.2. Future applications

The next step to take is to elaborate on possible future applications that can serve as input for creating strong business cases. Once these business cases are formed and accepted, building and sensor data integration can fully take off. This chapter shall further elaborate on such future applications.

Preparation: Data accessibility

Effective and pragmatic data gathering is an essential first step, and frankly could be argued to be the most important prior to other advanced analytics. It is often time consuming and thereby a waste of money. Only a small part of all building-related data is gathered, organised and stored in a well manner, and is accessible for all relevant users. Lack of usable data is thereby one of the most essential issues in data-driven operations in the whole building-lifecycle (Loyola, 2018).

This thesis focussed on the integration of well collected data. However, if no proper data is available, the integration will not function. Prior to conducting future applications, a pragmatic data gathering, storing and accessibility process must be in order. Companies like Architrave use AI to digitize documents and turn them into usable data (Tasker, 2019). Such algorithms are very useful for older data, but if new data is not stored well, it shall only be a temporary solution.

Facility management

IoT in combination with easily accessible and usable building data shall transform the role of facility managers. A facility manager shall increasingly work more data-driven and its focus shall thereby shift to optimization and quick problem solving (Vennema, 2019). By removing numerous time-consuming repetitive tasks, a facility manager can focus all its energy on the more strategic tasks. Strong developments are being made in the area of real-time building model updates, namely 'digital twins' (Tang et al., 2019). These models are used for among others quick data accessibility, building maintenance and insights based on a combination of data sources. The integration developed in this thesis might result in more efficient data utilization as well as using a higher variety of sources.

New redesign process: Program of Requirements (PoR)

Using data-driven insights, the project initiation process may be adjusted. The first step is to determine if a new project is to be started in the first place, after which its new requirements/alterations from the current situation should be determined. Several companies currently execute a part of this process. For example, Basking.io (Basking.io, n.d.) provides insights in actual space usage, Vergesense (Vergesense, n.d.) analyses occupancy on desk level and Comfy (Comfy, n.d.) combines booking data, occupancy data and environmental status of rooms into accurate analytics over time. The goals of these three companies are largely similar to BeSense, with which the pilot of this thesis is conducted, but Basking.io, Vergesense and Comfy take it a step further. For instance, Vergesense additionally assigns people per day to a desk and Comfy includes elements such as unattended booked meetings in their analytics. Using the insights from such analytics, new PoRs can be created. These occupancy data based PoRs can lead to a redesign which shall probably result in a higher building efficiency over time. By effectively combining these results with the building data of its surroundings, more detailed conclusions in possible building adjustments can be drawn.

Apart from building efficiency, the effects of building characteristics can be estimated as well. Enodo for example has data throughout 20 million properties in the US, which can quantify the effects of certain characteristics on the property rent or value (Enodo Inc., 2018). Linking such objective values to specific characteristics may be extremely valuable for building owners or potential redevelopers.

New redesign process: Data-driven design

Building development is a highly complex process from a collaborative perspective. To realise sustainable building design, multidisciplinary input is necessary and multiple criteria must form a high-performing whole (Petrova, Pauwels, Svidt, & Jensen, 2018). Tools that effectively leverage a multiplicity of data sources in the early design process are rare (Bilal et al., 2016). However, if effective integration of such data sources is created and effectively applied for design purposes, a paradigm shift to data-driven design in the building sector might be achieved.

Using more social data in this data-driven design process might be valuable as well. Big data in other industries is often used to characterize the preferences and behaviour of persons or groups of persons (Loyola, 2018). As designers often seem to be wrong about the assumptions made for the behaviour of occupants (Birt & Newsham, 2009), applying such a big data approach to a higher extent in the building design process might lead to fewer false assumptions. An example is a 2015 project of the WeWork Research Team, which researched what room characteristics their customers seemed to prefer (Davis, 2016). Using a multiplicity of data sources, among others survey data and occupancy data, a better understanding of the actual space usage and motivation could be triangulated, which they then applied in their redesign process. Currently designs are often created using solely client requirements, rules of thumb and designers experience (Petrova et al., 2018). Designers can distinguish themselves more and more in the design decision-making process, by placing a higher emphasis on using the various data sources in the existing building stock. Feedback from users, such as used by WeWork or CrowdComfort (CrowdComfort, n.d.), is one of the most valuable data sources to be integrated. The data integration developed in this thesis might provide a suitable backend for utilizing this multiplicity of data sources in the (re)design process.

Parametric design

The early building design stages often require numerous design alternatives that are evaluated and adjusted in an iterative process to assist the designers in understanding the client's wishes and requirements. Creating (partly) automated design tools that assist in this time-consuming step has been a challenge that several researchers have struggled with (Loyola, 2018). On building level, companies such as Parafin (Parafin, 2019) and Archistar (Archistar, 2019) generate building shapes based on exploitation potentials. Restrictions used here are urban regulations, official building requirements and parameter inputs by clients. A lot of administrative work after determining the shape is automatically generated as well. This design process is automatically executed for all building plots in that area as well to find better suitable locations based on the KPI weight inputs. These algorithms are promising but successfully integrating these automations in the building design process remains a challenge. Especially redevelopment seems difficult compared to new construction.

Current initiatives are taken in creating generic generative design engines, most notably being Autodesk Dreamcatcher. However, as building design is extremely complex, data needed for such customized and optimized designs must be manually gathered, except for objective parameters such as daylight, energy performance, etc. Using these generative design engines for redevelopment of buildings require accurate and a multiplicity of data sources, among others user preferences and behaviour. Again, integration of varying data sources is thereby an important factor in further generative design (re)development. The integration developed in this thesis might contribute to the required data source integration, but it must be extended to fulfil all data needs of such a parametric design algorithm.

Design evaluation

Designers are under increased pressure to create high performing designs. However, as the relationship between designers and the designed buildings often deteriorates after delivery, the feedback on the designs is often not gathered (Loyola, 2018). If designers do not experience the impact that design decisions have made in the past, it is difficult for them to predict the impact of future design decisions (Davis, 2016). A method of evaluating design decisions is through post-occupancy evaluations (POEs), but these are rarely carried out by architects (Peters, 2018). A 2015 poll in the Architects' Journal found that only 3 percent of architects always performed a POE on their projects (Peters, 2018). A survey research by Hiromoto (2015) found that the leading reason for not carrying out POEs was the time and cost required for producing useful results. Therefore, if the aforementioned multiplicity of data can be specifically utilized to make POEs more efficient or (partially) automate them, evaluations could be executed more frequently and learning from past projects may be stimulated. Again, the integration demonstrated in this thesis may form a (partial) solution to utilizing the required multiplicity of data for such evaluations.

Change management

Generally, what it all comes down to, is that the focus should be on gaining the support of involved parties for both investment and usage. Initiating cutting edge technologies and finding technically feasible solutions does not drive innovation. Innovation in a traditional sector such as the real estate sector must be driven step-by-step and involved parties must be convinced for each next step. If innovation is pushed too fast and involved parties do not understand what is happening and/or what its impact shall be, resistance shall probably develop, and innovation will be hindered. Smarter environments do not only require smarter technologies, but also smarter users and professionals (International Council for Research and Innovation in Building and Construction, 2018). Technological development in the area of facility management thereby includes training and competence development of experienced facility managers in embracing these new technologies (Araszkiewicz, 2017). Apart from all future technological developments in the built environment, successful change management shall be the most important aspect in future applications.

9.6.3. Possible future smart hospital initiatives

The use case has been executed on an office building in a UMC. Even though the pilot building is an office building, the bigger picture of a smart hospital should be discussed. This chapter shall briefly elaborate on several (future) initiatives to add to the goal of a smart hospital.

Improve information accessibility

Built environment related processes are generally more complicated at a UMC than on residential or office buildings. More factors have to be considered when evaluating redevelopment possibilities/initiatives or malfunctions in the current building stock. Using several information sources these processes function fine but acquiring the necessary information can be time consuming. By improving the accessibility of the built-environment related data and by (partly) automating repeated processes, the employees can focus on strategic/improvement processes to a higher degree.

Monitor building performance

Through continuous tracking of building performance, the as-built or as-programmed state can be continuously compared to the current state using data sources such as:

- Building data (full Revit model);
- Sensor data (both comfort and motion);
- Social data (for example complaints, feedback or ratings linked to specific building areas/elements);
- Malfunctions linked to specific building areas/elements and their probable causes;
- Production data linked to specific rooms;
- Sick leave of employees;
- Financial health of a building.

Assist (strategic) redevelopment processes

Considering the higher complexity in a UMC building, (strategic) redevelopment processes are of a higher complexity as well. Additionally, the real estate needs of departments in a UMC keeps changing to a certain degree, leading to a new redevelopment need. These redevelopment projects can be assisted using, among others, the following (future) initiatives:

- Use temporary sensors to get objective insight in the usage and indoor climate conditions. The real estate departments at the UMC can buy an x number of sensors which they can temporarily utilize at the (potentially) to be redeveloped departments for a x period of time. The placement and integration of these sensors seems to be easily possible and, using pre-programmed analyses that might be required in such projects, insights can be quickly derived.
- Using parametric design algorithms, insight may be gathered in the consequences of hypothetical situations. For example, an 'ideal' UMC or the 'ideal' location for a specific goal with specific requirements may be derived based on certain KPI's. Similarly,

consequences of a certain action can be estimated, for example an adjusted location, sizes or capacities.

- Finding a location for a new redevelopment initiative can be difficult due to high requirements, complex links with other elements in the UMC and the eventual requirement of 24/7 production. The process of finding such (temporary) locations can be assisted by providing aspects like, amongst others, logistic relations, building requirements and potential sharing possibilities.
- Taking it a step further, new schematic redevelopment plans may be created using similar parametric design algorithms. The data input would be the objectively determined PoR, specific requirements, internal logistical relations, sharing possibilities of specific rooms, etc.

Facilitate the end-user in quick insight in their real estate usage

In the end the goal is a satisfied user. In the Radboudumc each department rents specific spaces for which they pay rent and for which they can request adjustments. Their real estate usage and its consequences is currently not insightful. To satisfy the users to a higher degree, easy insight in their users can be provided by combining the following data sources in an effective manner:

- Tracking appliances and beds using RFID chips and link this to the spaces where they are at that point in time using the coordinates of location tracking. This data can later be used for space performance as well;
- Suggest meeting rooms not only on their properties, availability and distance, but also on (predicted) indoor climate considering measurements and other appointments;
- Results of indoor climate and occupancy analyses;
- Results of social data analyses (coming from their employees and patients);
- Production results of their rooms;
- Possible reductions/expansions of their real estate based on what is spatially and operationally possible. Considering the current financial systems in place the savings potential can immediately be estimated. If Bouwzaken or Vastgoed & Infrastructuur need a specific area, this area is being labelled as wanted which shall adjust the regulations of disposing the rent of these spaces. This change can be considered in the possibility calculations.

A UMC is an extremely interesting place from, among others, a built environment point of view. However, maintaining a high performing UMC from, among others, a built environment point of view is extremely difficult as well. I strongly believe that technological developments are going to ease this process in the coming years. The suggestions for a smart hospital provided in this chapter are possibilities of doing so, but numerous other options might be possible. Thorough discussions with the end-user may provide more insight in the current struggles/improvement possibilities. Communicating these struggles/improvement possibilities to a party with extensive knowledge on the technical side of buildings and software development may lead to new solutions.

Section 10

References

Abbott, D. (2004). Calming the office cacophony. *The Safety & Health Practitioner*, *22*(1), 34–36.

Agarwal, S., & Rajan, K. S. (2016). Performance analysis of MongoDB versus PostGIS/PostGreSQL databases for line intersection and point containment spatial queries. *Spatial Information Research*, *24*(6), 671–677. https://doi.org/10.1007/s41324-016-0059-1

Al horr, Y., Arif, M., Katafygiotou, M., Mazroei, A., Kaushik, A., & Elsarrag, E. (2016). Impact of indoor environmental quality on occupant well-being and comfort: A review of the literature. *International Journal of Sustainable Built Environment*, *5*(1), 1–11. https://doi.org/10.1016/j.ijsbe.2016.03.006

Altomonte, S., Schiavon, S., Kent, M. G., & Brager, G. (2019). Indoor environmental quality and occupant satisfaction in green-certified buildings. *Building Research and Information*, *47*(3), 255–274. https://doi.org/10.1080/09613218.2018.1383715

Alves, M., Carreira, P., & Costa, A. A. (2017). BIMSL: A generic approach to the integration of building information models with real-time sensor data. *Automation in Construction*, *84*(September), 304–314. https://doi.org/10.1016/j.autcon.2017.09.005

Alwetaishi, M. S. (2016). Impact of building function on thermal comfort: A review paper. *American Journal of Engineering and Applied Sciences*, *9*(4), 928–945. https://doi.org/10.3844/ajeassp.2016.928.945

An, Y. (2017). *Geological Data Extension for Subway Tunnel BIM models using a Linked Data Approach*. (0886605).

Araszkiewicz, K. (2017). Digital Technologies in Facility Management - The state of Practice and Research Challenges. *Procedia Engineering*, *196*(June), 1034–1042. https://doi.org/10.1016/j.proeng.2017.08.059

Archibus. (n.d.). Space Planning Management. Retrieved from https://archibus.com/products/space-planning-management/

Archistar. (2019). Archistar Professional: Find development opportunities in minutes. Retrieved from https://archistar.ai/professional

Aries, M. B. C. (2005). Human Lighting Demands, healthy lighting in an office environment. In *Technische Universiteit Eindhoven 2005*. https://doi.org/http://dx.doi.org/10.6100/IR594257

Arye, M. (2018). SQL vs. Flux: Choosing the right query language for time-series data. Retrieved from https://blog.timescale.com/blog/sql-vs-flux-influxdb-query-language-time-series-database-290977a01a8a/?source=post_page-----ef1e89a72377----------------------

ASHRAE. (2010). ASHRAE Standard: Thermal Environmental Conditions for Human Occupancy. In *American Society of Heating, Refrigerating and Air-Conditioning Engineers, Inc.* https://doi.org/10.1007/0-387-26336-5_1680

Autrup, H., Calow, P., Dekant, W., Greim, H., Hanke, W., Janssen, C., … Viluksela, M. (2007). *Opinion on Risk Assessment on Indoor Air Quality*. Retrieved from http://ec.europa.eu/health/ph_risk/risk_en.htm

Bailes, A. (2016). What Is the Best Indoor Relative Humidity in Winter? Retrieved from https://www.energyvanguard.com/blog/what-is-the-best-indoor-relative-humidity-in-winter

Barnsteiner, F. (2015). Elasticsearch as a Time Series Data Store. Retrieved from Elastic website:

https://www.elastic.co/blog/elasticsearch-as-a-time-series-data-store

Basking.io. (n.d.). How it works. Retrieved from https://basking.io/how-it-works/

Bean, R. (2010). Mean Radiant Temperature (MRT) - Part I. Retrieved from http://www.healthyheating.com/Definitions/Mean Radiant.htm#.XjiAImhKiUk

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: AIEDAM*, *23*(1), 89–101. https://doi.org/10.1017/S0890060409000122

Berners-Lee, T. (2000). *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web*. HarperBusiness.

Bilal, M., Oyedele, L. O., Qadir, J., Munir, K., Ajayi, S. O., Akinade, O. O., … Pasha, M. (2016). Big Data in the construction industry: A review of present status, opportunities, and future trends. *Advanced Engineering Informatics*, *30*(3), 500–521. https://doi.org/10.1016/j.aei.2016.07.001

Birt, B., & Newsham, G. (2009). Post-occupancy evaluation of energy and indoor environment quality in green buildings: a review. *3rd International Conference on Smart and Sustainable Built Environments*, 1–7.

Bonduel, M. (2018). Towards a PROPS ontology. *Presentation*, (March). Retrieved from https://github.com/w3c-lbd-cg/lbd/blob/gh-pages/presentations/props/presentation_LBDcall_20180312_final.pdf

Bonduel, M., Oraskari, J., Pauwels, P., Vergauwen, M., & Klein, R. (2018). The IFC to linked building data converter - Current status. *CEUR Workshop Proceedings*, *2159*(August), 34–43.

Bonduel, M., Wagner, A., & Pauwels, P. (2019). FOG: File Ontology for Geometry formats. Retrieved from https://mathib.github.io/fog-ontology/

Bornea, M. A., Dolby, J., Kementsietsidis, A., Srinivas, K., Dantressangle, P., Udrea, O., & Bhattacharjee, B. (2013). Building an efficient RDF store over a relational database. *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 121–132. https://doi.org/10.1145/2463676.2463718

Braulio, L. D. C., Moreno, E. D., De MacEdo, D. D. J., Kreutz, D., & Dantas, M. A. R. (2018). Towards a Hybrid Storage Architecture for IoT. *Proceedings - IEEE Symposium on Computers and Communications*, (5), 470–473. https://doi.org/10.1109/ISCC.2018.8538474

Briggs, B. (2015). Tech Trends 2015. *Deloitte*, 147. Retrieved from http://www.elperiodico.com/es/noticias/redes/youtube-guru-juvenil-3453133

Building Engines. (2019). *The Tenant Experience Gap: Building Managers and Tenants Look at the Future through Different Eyes*. Retrieved from https://www.buildingengines.com/landing/the-tenant-experience-gap-report

Building, I., & Modeling, I. (2015). Semantic web technologies for implementing cost-effective and interoperable building information modeling. *The 14th International Conference on Informatics in Economy (IE 2015)*, (January 2018).

Carbonari, A., Corneli, A., Di Giuda, G., Ridolfi, L., & Villa, V. (2018). BIM-Based Decision Support System for the Mangement of Large Building Stocks. *Proceedings of the 35th International Symposium on Automation and Robotics in Construction (ISARC)*, (Isarc). https://doi.org/10.22260/isarc2018/0049

Carlos, M. (2014). *Spreadsheets for the Calculation of*. (August 2013). https://doi.org/10.13140/RG.2.1.2778.0887

Chang, C. Y., & Chen, P. K. (2005). Human response to window views and indoor plants in the workplace. *HortScience*, *40*(5), 1354–1359.

Chang, K. M., Dzeng, R. J., & Wu, Y. J. (2018). An automated IoT visualization BIM platform for decision support in facilities management. *Applied Sciences (Switzerland)*, *8*(7). https://doi.org/10.3390/app8071086

Chen, Z., Jiang, C., & Xie, L. (2018). Building occupancy estimation and detection: A review. *Energy and Buildings*, *169*, 260–270. https://doi.org/10.1016/j.enbuild.2018.03.084

Clements-Croome, D. (2019). The role of feedback in building design 1980–2018 and onwards. *Building Services Engineering Research and Technology*, *40*(1), 5–12. https://doi.org/10.1177/0143624418812982

Comfy. (n.d.). The leading global workplace platform. Retrieved from https://www.comfyapp.com/product/

Compton, M., Barnaghi, P., Bermudez, L., García-Castro, R., Corcho, O., Cox, S., … Taylor, K. (2012). The SSN ontology of the W3C semantic sensor network incubator group. *Journal of Web Semantics*, *17*, 25–32. https://doi.org/10.1016/j.websem.2012.05.003

Condair. (n.d.). My humidity is low - what should I do? Retrieved from https://www.condair-group.com/humidity-health-wellbeing/my-humidity-is-low-what-should-i-do

Cordero, C. (2017). *Smart and Sustainable Offices (SSO): Showcasing a holistic approach to realise the next generation offices*. *69*(548), 1–10.

Corry, E., Pauwels, P., Hu, S., Keane, M., & O'Donnell, J. (2015). A performance assessment ontology for the environmental and energy management of buildings. *Automation in Construction*, *57*, 249–259. https://doi.org/10.1016/j.autcon.2015.05.002

Creative Commons. (n.d.). BSON. Retrieved from http://bsonspec.org/

CrowdComfort. (n.d.). The Hidden Sensor Network Already in your Building. Retrieved from http://crowdcomfort.com/enterprise-facilities-management/

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*, *27*(2), 206–219. https://doi.org/10.1016/j.aei.2012.10.003

D'Oca, S., & Hong, T. (2014). A data-mining approach to discover patterns of window opening and closing behavior in offices. *Building and Environment*, *82*, 726–739. https://doi.org/10.1016/j.buildenv.2014.10.021

Dave, B., Buda, A., Nurminen, A., & Främling, K. (2018a). 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

Dave, B., Buda, A., Nurminen, A., & Främling, K. (2018b). A framework for integrating BIM and IoT through open standards. *Automation in Construction*, *95*(August 2017), 35–45. https://doi.org/10.1016/j.autcon.2018.07.022

Dave, B., Kubler, S., Främling, K., & Koskela, L. (2016). Opportunities for enhanced lean construction management using Internet of Things standards. *Automation in Construction*, *61*, 86–97. https://doi.org/10.1016/j.autcon.2015.10.009

Davis, D. (2016). Evaluating Buildings with Computation and Machine Learning. *Posthuman Frontiers*, 116–123.

Deloitte. (2018). *Data is the new gold: The future of real estate service providers*. Retrieved from https://www2.deloitte.com/global/en/pages/real-estate/articles/future-real-estate-data-new-gold.html

Deutsch, R. (2015). Leveraging data Across the Building Lifecycle. *Procedia Engineering*, *118*, 260–267. https://doi.org/10.1016/j.proeng.2015.08.425

Dix, P. (2018). Why We're Building Flux, a New Data Scripting and Query Language. Retrieved from Influxdata website: https://www.influxdata.com/blog/why-were-building-flux-a-new-data-scripting-and-query-language/

Dotson, J. D. (2018). How Temperature & Humidity are Related. Retrieved from https://sciencing.com/temperature-ampamp-humidity-related-7245642.html

DUUX. (n.d.). The ideal indoor humidity. Retrieved from https://www.duux.com/blog/2017/12/07/the-ideal-indoor-humidity/

EDNA. (2018). *Intelligent Efficiency - A Case Study of Barriers & Solutions - Smart Homes*. 1–55. Retrieved from http://edna.iea-4e.org

Enodo Inc. (2018). *Enodo: Automated Multifamily Underwriting*. Retrieved from https://www.youtube.com/watch?v=D_4GUQ5MEEE

European Comission. (2003). Indoor air pollution: new EU research reveals higher risks than previously thought. *European Comission*, (September), IP/03/1278.

Fang, L. (1998). Impact of temperature and humidity on perception of indoor air quality during immediate and longer whole-body exposures. *Indoor Air*, *8*(4), 276–284. https://doi.org/10.1111/j.1600-0668.1998.00008.x

Fanger, P. O. (1973). Assessment of man's thermal comfort in practice. *British Journal of Industrial Medicine*, *30*(4), 313–324.

Farias, T. M. De, & Nicolle, C. (2015). IfcWoD , Semantically Adapting IFC Model Relations into OWL Properties. *Proceedings of the 32nd CIB W78 Conference*.

Ferguson, H., Krisnadhi, A., & Cheatham, M. (2017). *Semantic-Graph based approach to BIM , resilience , and spatial data interoperability*. 1–27.

Flynn, D. (2015). IoT considerations — storage and database, SQL, NoSQL, historical data. Retrieved from LATTICE Research website: https://medium.com/lattice-research/iot-considerations-storage-and-database-sql-NoSQL-historical-data-1b9e4d53cb32

Framling, K., Kubler, S., & Buda, A. (2014). Universal messaging standards for the IoT from a lifecycle management perspective. *IEEE Internet of Things Journal*, *1*(4), 319–327. https://doi.org/10.1109/JIOT.2014.2332005

Frics, E. M. V., Watson, I. B. M., & Management, A. (2019). *The brain of the building RICS World Built Environment Forum Summit*. (May 2020), 2019–2021.

Frontczak, M., & Wargocki, P. (2011). Literature survey on how different factors influence human comfort in indoor environments. *Building and Environment*, Vol. 46, pp. 922–937. https://doi.org/10.1016/j.buildenv.2010.10.021

Galasiu, A. D., & Veitch, J. A. (2006). Occupant preferences and satisfaction with the luminous environment and control systems in daylit offices: a literature review. *Energy and Buildings*, *38*(7), 728–742. https://doi.org/10.1016/j.enbuild.2006.03.001

Ganisen, S., Nesan, L. J., Mohammad, I. S., Mohamed, A. hakim, & Kanniyapan, G. (2015). Facility Management Variables That Influence Sustainability of Building Facilities. *Jurnal Teknologi*, *75*(10), 27–38.

García Kerdan, I., Raslan, R., Ruyssevelt, P., & Morillón Gálvez, D. (2017). ExRET-Opt: An automated exergy/exergoeconomic simulation framework for building energy retrofit analysis and design optimisation. *Applied Energy*, *192*, 33–58. https://doi.org/10.1016/j.apenergy.2017.02.006

gbXML. (n.d.). About gbXML. Retrieved from http://www.gbxml.org/

Gerrish, T., Ruikar, K., Cook, M., Johnson, M., Phillip, M., & Lowry, C. (2017). BIM application to building energy performance visualisation and managementChallenges and potential. *Energy and Buildings*, *144*, 218–228. https://doi.org/10.1016/j.enbuild.2017.03.032

Gruber, M., Trüschel, A., & Dalenbäck, J. O. (2014). Model-based controllers for indoor climate control in office buildings - Complexity and performance evaluation. *Energy and Buildings*, *68*(PARTA), 213–222. https://doi.org/10.1016/j.enbuild.2013.09.019

Harris, S., & Seaborne, A. (2013). SPARQL 1.1 Query Language. Retrieved from https://www.w3.org/TR/sparql11-query/

Heaton, J., & Parlikad, A. K. (2019). A conceptual framework for the alignment of infrastructure assets to citizen requirements within a Smart Cities framework. *Cities*, *90*(January), 32–41. https://doi.org/10.1016/j.cities.2019.01.041

Heschong Mahone Group. (2003). Windows and Offices: A Study of Office Worker Performance and the Indoor Environment. *California Energy Commision*, *37*(4), 414–435. https://doi.org/10.1175/1520-0450(1998)037<0414:TDFBIM>2.0.CO;2

Hiromoto, J. (2015). *Architect & Design Sustainable Design Leaders: Post Occupancy Evaluation Survey Report*. 1–177.

Hsieh, D., & Perry, M. (2019). Why you should use a relational database instead of NoSQL for your IoT application. Retrieved from Timescale website: https://blog.timescale.com/blog/use-relational-database-instead-of-nosql-for-iot-application/

Hu, S., Corry, E., Curry, E., Turner, W. J. N., & O'Donnell, J. (2016). Building performance optimisation: A hybrid architecture for the integration of contextual information and time-series data. *Automation in Construction*, *70*, 51–61. https://doi.org/10.1016/j.autcon.2016.05.018

Huang, L., Zhu, Y., Ouyang, Q., & Cao, B. (2012). A study on the effects of thermal, luminous, and acoustic environments on indoor environmental comfort in offices. *10th International Conference on Healthy Buildings 2012*, *2*, 963–968. https://doi.org/10.1016/j.buildenv.2011.07.022

Hutchison, D., & Mitchell, J. C. (2009). Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*.

International Council for Research and Innovation in Building and Construction. (2018). *Research Roadmap: Intelligent and Responsive Buildings*.

Jahncke, H., Hygge, S., Halin, N., Green, A. M., & Dimberg, K. (2011). Open-plan office noise: Cognitive performance and restoration. *Journal of Environmental Psychology*, *31*(4), 373–382. https://doi.org/10.1016/j.jenvp.2011.07.002

Jalilzadehazhari, E., & Johansson, P. (2019). *Advances in Informatics and Computing in Civil and Construction Engineering*. https://doi.org/10.1007/978-3-030-00220-6

Jensen, K. L., Arens, E., & Zagreus, L. (2005). Acoustical Quality in Office Workstations , As Assessed By Occupant Surveys. *Indoor Air: 10th International Conference on Indoor Air Quality and Climate*, (March), 2401–2405.

Jia, R., Jin, B., Jin, M., Zhou, Y., Konstantakopoulos, I. C., Zou, H., … Spanos, C. J. (2018). Design Automation for Smart Building Systems. *Proceedings of the IEEE*, *106*(9), 1680–1699. https://doi.org/10.1109/JPROC.2018.2856932

Kaarlela-Tuomaala, A., Helenius, R., Keskinen, E., & Hongisto, V. (2009). Effects of acoustic environment on work in private office rooms and open-plan offices - Longitudinal study during relocation. *Ergonomics*, *52*(11), 1423–1444. https://doi.org/10.1080/00140130903154579

Kang, S., Ou, D., & Mak, C. M. (2017). The impact of indoor environmental quality on work productivity in university open-plan research offices. *Building and Environment*, *124*, 78–89. https://doi.org/10.1016/j.buildenv.2017.07.003

Kang, Y. S., Park, I. H., Rhee, J., & Lee, Y. H. (2016). MongoDB-Based Repository Design for IoT-Generated RFID/Sensor Big Data. *IEEE Sensors Journal*, *16*(2), 485–497. https://doi.org/10.1109/JSEN.2015.2483499

Kanteyan, M. S., Utami, S. S., Prasetiyo, I., & Yanti, R. J. (2018). Sound insulation design improvement for reducing noise of paper factory office. *Journal of Physics: Conference Series*, *1075*(1). https://doi.org/10.1088/1742-6596/1075/1/012060

Katafygiotou, M. C., & Serghides, D. K. (2015). Bioclimatic chart analysis in three climate zones in Cyprus. *Indoor and Built Environment*, *24*(6), 746–760. https://doi.org/10.1177/1420326X14526909

Kejriwal, S., & Mahajan, S. (2016). *Smart buildings: How IoT technology aims to add value for real estate companies*.

Khan, Z. H., Islam, M. R., & Tarefder, R. A. (2019). Determining asphalt surface temperature using weather parameters. *Journal of Traffic and Transportation Engineering (English Edition)*, *6*(6), 577–588. https://doi.org/10.1016/j.jtte.2018.04.005

Kirk, P. (2017). *What New Technologies Mean for the Future of Office Space*. 1–4.

Klepeis, N. E., Nelson, W. C., Ott, W. R., Robinson, J. P., Tsang, A. M., Switzer, P., … Engelmann, W. H. (2001). The National Human Activity Pattern Survey (NHAPS): A resource for assessing exposure to environmental pollutants. *Journal of Exposure Analysis and Environmental Epidemiology*, *11*(3), 231–252. https://doi.org/10.1038/sj.jea.7500165

Klimaat Beheer. (2018). CO2 in het binnenklimaat gevaarlijk? CO2 hoeveelheid leidt tot klachten! Retrieved from http://www.klimaatbeheer.eu/blog_type/co2/

KNMI. (2017). Atmosferische CO2 concentratie blijft stijgen. Retrieved from https://www.knmi.nl/over-het-knmi/nieuws/atmosferische-co2-concentratie-blijft-stijgen

KNMI. (2018). *Winter 2018/2019*. Retrieved from https://www.knmi.nl/nederlandnu/klimatologie/maand-en-seizoensoverzichten/2019/winter

Kosonen, R., & Tan, F. (2004). The effect of perceived indoor air quality on productivity loss. *Energy and Buildings*, *36*(10 SPEC. ISS.), 981–986. https://doi.org/10.1016/j.enbuild.2004.06.005

Kuchen, E., Fisch, M. N., Leão, M., & Leão, É. B. (2009). Indoor air quality defined by measurements and questionnaires in office buildings. *Bauphysik*, *31*(5), 313–318. https://doi.org/10.1002/bapi.200910041

Lakshman, A., & Malik, P. (2010). Cassandra - A Decentralized Structured Storage System. *ACM SIGOPS Operating Systems Review*, *44*(2), 35–40. https://doi.org/10.1145/1773912.1773922

Leavitt, N. (2010). Will NoSQL Databases Live Up to Their Promise? *Computer*, *43*(2), 12–14. https://doi.org/10.1109/mc.2010.58

Ledet Training. (2018). *How Autodesk used Generative Design to create their Toronto office*. Retrieved from https://www.youtube.com/watch?v=ZQmfgGKtlXQ

Lee, K. S., Han, K. J., & Lee, J. W. (2016). Feasibility study on parametric optimization of daylighting in building shading design. *Sustainability (Switzerland)*, *8*(12), 1–25. https://doi.org/10.3390/su8121220

Lee, Y. C., Eastman, C. M., Solihin, W., & See, R. (2016). Modularized rule-based validation of a BIM model pertaining to model views. *Automation in Construction*, *63*, 1–11. https://doi.org/10.1016/j.autcon.2015.11.006

Lefrançois, M. (2017). Planned ETSI SAREF extensions based on the W3C&OGC SOSA/SSN-compatible SEAS ontology patterns. *CEUR Workshop Proceedings*, *2063*(December 2016).

Level. (2017). Humidity and condensation. Retrieved from http://www.level.org.nz/passive-design/controlling-indoor-air-quality/humidity-and-condensation/

Lowen, A. C., Mubareka, S., Steel, J., & Palese, P. (2007). Influenza virus transmission is dependent on relative humidity and temperature. *PLoS Pathogens*, *3*(10), 1470–1476. https://doi.org/10.1371/journal.ppat.0030151

Loyola, M. (2018). Big data in building design: A review. *Journal of Information Technology in Construction*, *23*(January), 259–284.

Maula, H., Hongisto, V., Östman, L., Haapakangas, A., Koskela, H., & Hyönä, J. (2016). The effect of slightly warm temperature on work performance and comfort in open-plan offices - a laboratory study. *Indoor Air*, *26*(2), 286–297. https://doi.org/10.1111/ina.12209

Mazairac, W., & Beetz, J. (2013). BIMQL - An open query language for building information models. *Advanced Engineering Informatics*, *27*(4), 444–456. https://doi.org/10.1016/j.aei.2013.06.001

McGlinn, K., Wagner, A., Pauwels, P., Bonsma, P., Kelly, P., & O'Sullivan, D. (2019). Interlinking geospatial and building geometry with existing and developing standards on the web. *Automation in Construction*, *103*(March 2018), 235–250. https://doi.org/10.1016/j.autcon.2018.12.026

Mcglinn, K., Yuce, B., Wicaksono, H., Howell, S., & Rezgui, Y. (2017). Usability evaluation of a web-based tool for supporting holistic building energy management. *Automation in Construction*, *84*(August), 154–165. https://doi.org/10.1016/j.autcon.2017.08.033

Ministerie van Sociale Zaken en Werkgelegenheid. (n.d.). Arbowetgeving en Arboregelgeving.

Retrieved from https://www.arboportaal.nl/onderwerpen/themas/arbowet--en--regelgeving

MongoDB. (n.d.). NoSQL Databases Explained. Retrieved from https://www.mongodb.com/nosql-explained

Moniruzzaman, A. B. M., & Hossain, S. A. (2013). *NoSQL Database: New Era of Databases for Big data Analytics - Classification, Characteristics and Comparison*. (June). Retrieved from http://arxiv.org/abs/1307.0191

Navitas Capital. (2019). *Artificial Intelligence (AI) for the built world*. Retrieved from https://navitascap.com/navitas-capital-artificial-intelligence-whitepaper/

Nguyen, J. L., Schwartz, J., & Dockery, D. W. (2014). The relationship between indoor and outdoor temperature, apparent temperature, relative humidity, and absolute humidity. *Indoor Air*, *24*(1), 103–112. https://doi.org/10.1111/ina.12052

Niknam, M., & Karshenas, S. (2017). A shared ontology approach to semantic representation of BIM data. *Automation in Construction*, *80*, 22–36. https://doi.org/10.1016/j.autcon.2017.03.013

Northern Arizona University. (2013). *Thermal Comfort (Relative Humidity (RH) and Air Temperature)*. Retrieved from https://www7.nau.edu/itep/main/eeop/docs/airqlty/AkIAQ_ThermalComfort.pdf%0Http://www7.nau.edu/itep/main/eeop/docs/airqlty/AkIAQ_ThermalComfort.pdf

Ontotext. (n.d.). What is semantic technology? Retrieved from https://www.ontotext.com/knowledgehub/fundamentals/semantic-web-technology/

Ontotext. (2014). *The Truth About Triplestores*. 28. Retrieved from https://ontotext.com/wp-content/uploads/2014/07/The-Truth-About-Triplestores.pdf

Parafin. (2019). Think Like a Developer, Design Like a Million Architects. Retrieved from http://www.parafin3d.com/

Park, J. S., & Yoon, C. H. (2011). The effects of outdoor air supply rate on work performance during 8-h work period. *Indoor Air*, *21*(4), 284–290. https://doi.org/10.1111/j.1600-0668.2010.00700.x

Patacas, J., Dawood, N., Vukovic, V., & Kassem, M. (2015). BIM for facilities management: Evaluating BIM standards in asset register creation and service life planning. *Journal of Information Technology in Construction*, *20*(January), 313–331.

Pauwels, P., & Roxin, A. (2016). SimpleBIM: From full ifcOWL graphs to simplified building graphs. *EWork and EBusiness in Architecture, Engineering and Construction - Proceedings of the 11th European Conference on Product and Process Modelling, ECPPM 2016*, (November), 11–18.

Pauwels, P., Van Deursen, D., Verstraeten, R., De Roo, J., De Meyer, R., Van De Walle, R., & Van Campenhout, J. (2011). A semantic rule checking environment for building performance checking. *Automation in Construction*, *20*(5), 506–518. https://doi.org/10.1016/j.autcon.2010.11.017

Pauwels, Pieter. (2014). Supporting Decision-Making in the Building Life-Cycle Using Linked Building Data. In *Buildings* (Vol. 4). https://doi.org/10.3390/buildings4030549

Pauwels, Pieter. (2017). Product and 3D geometry ontologies at action in construction industry: from manufacturer to demolition. *LOA Seminar*.

Pauwels, Pieter, de Farias, T. M., Zhang, C., Roxin, A., Beetz, J., De Roo, J., & Nicolle, C. (2017). A performance benchmark over semantic rule checking approaches in construction industry. *Advanced Engineering Informatics*, *33*, 68–88. https://doi.org/10.1016/j.aei.2017.05.001

Pauwels, Pieter, Krijnen, T., & Beetz, J. (2015). Coping with lists in the ifcOWL ontology. *Proceedings of the 22nd EG-ICE Workshop*.

Pauwels, Pieter, Krijnen, T., Terkaj, W., & Beetz, J. (2017). Enhancing the ifcOWL ontology with an alternative representation for geometric data. *Automation in Construction*, *80*, 77–94. https://doi.org/10.1016/j.autcon.2017.03.001

Pauwels, Pieter, Poveda-Villalón, M., Sicilia, Á., & Euzenat, J. (2018). Semantic technologies and interoperability in the built environment. *Semantic Web*, *9*(6), 731–734. https://doi.org/10.3233/SW-180321

Pauwels, Pieter, & 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, Pieter, Zhang, S., & Lee, Y. C. (2017). Semantic web technologies in AEC industry: A literature overview. *Automation in Construction*, *73*, 145–165. https://doi.org/10.1016/j.autcon.2016.10.003

Peters, T. (2018). Sensor Feedback in Sustainable Design Workflows. *Data Buildings*, 92–101.

Petrova, E., Pauwels, P., Svidt, K., & Jensen, R. L. (2018). Towards data-driven sustainable design: decision support based on knowledge discovery in disparate building data. *Architectural Engineering and Design Management*, *2007*. https://doi.org/10.1080/17452007.2018.1530092

Petrullo, M., Jones, S. A., Laquidara-Carr, D., Ap, L., Lorenz, A., Buckley, B., … Barnett, S. (2016). *Report World Green Building Trends 2016: Developing Markets Accelerate*. Retrieved from http://fidic.org/sites/default/files/World Green Building Trends 2016 SmartMarket Report FINAL.pdf

Pifferi, T. (2018). How to efficiently store and query time-series data. Retrieved from Medium website: https://medium.com/@neslinesli93/how-to-efficiently-store-and-query-time-series-data-90313ff0ec20

Ploennigs, J., Ba, A., & Barry, M. (2018). Materializing the promises of cognitive IoT: How cognitive buildings are shaping the way. *IEEE Internet of Things Journal*, *5*(4), 2367–2374. https://doi.org/10.1109/JIOT.2017.2755376

Quang, T. N., He, C., Knibbs, L. D., De Dear, R., & Morawska, L. (2014). Co-optimisation of indoor environmental quality and energy consumption within urban office buildings. *Energy and Buildings*, *85*, 225–234. https://doi.org/10.1016/j.enbuild.2014.09.021

Ramo, A. C., Stagni, F., Tomassetti, L., & Mathe, Z. (n.d.). *Evaluation of NoSQL databases for DIRAC monitoring and beyond On behalf of the LHCb collaboration Motivation*. Retrieved from http://cds.cern.ch/record/2011172/files/LHCb-TALK-2015-060.pdf?lipi=urn%3Ali%3Apage%3Ad_flagship3_pulse_read%3BtDYpREugTCGxpK6J2Lgn%2Fw%3D%3D

Rasila, H., & Jylhä, T. (2015). The many faces of office noise-case contact center. *Facilities*, *33*(7–8), 454–464. https://doi.org/10.1108/F-11-2013-0085

Rasmussen, H. L., Jensen, P. A., Nielsen, S. B., & Kristiansen, A. H. (2019). Initiatives to integrate operational knowledge in design: a building client perspective. *Facilities*, *37*(11–12), 799–812. https://doi.org/10.1108/F-02-2017-0021

Rasmussen, M. H., Frausing, C. A., Hviid, C. A., & Karlshoj, J. (2018). Demo: Integrating building information modeling and sensor observations using semantic web. *CEUR Workshop Proceedings*, *2213*(1), 48–55.

Rasmussen, M. H., Lefrançois, M., Bonduel, M., Hviid, C. A., & Karlshø, J. (2018). OPM: An ontology for describing properties that evolve over time. *CEUR Workshop Proceedings*, *2159*, 23–33.

Rasmussen, M. H., Pauwels, P., Hviid, C. A., & Karlshoj, J. (2017). Proposing a Central AEC Ontology That Allows for Domain Specific Extensions. *Proceedings of the Joint Conference on Computing in Construction*, *1*, 237–244.

Saha, H., Florita, A. R., Henze, G. P., & Sarkar, S. (2019). Occupancy sensing in buildings: A review of data analytics approaches. *Energy and Buildings*, *188–189*, 278–285. https://doi.org/10.1016/j.enbuild.2019.02.030

Salimi, S., Liu, Z., & Hammad, A. (2019). Occupancy prediction model for open-plan offices using real-time location system and inhomogeneous Markov chain. *Building and Environment*, *152*(January), 1–16. https://doi.org/10.1016/j.buildenv.2019.01.052

Seddigh, A., Berntson, E., Jönsson, F., Danielson, C. B., & Westerlund, H. (2015). The effect of noise absorption variation in open-plan offices: A field study with a cross-over design. *Journal of Environmental Psychology*, *44*, 34–44. https://doi.org/10.1016/j.jenvp.2015.08.004

Serghides, D. K., Chatzinikola, C. K., & Katafygiotou, M. C. (2015). Comparative studies of the occupants' behaviour in a university building during winter and summer time. *International Journal of Sustainable Energy*, *34*(8), 528–551. https://doi.org/10.1080/14786451.2014.905578

Shi, J., Yu, N., & Yao, W. (2017). Energy Efficient Building HVAC Control Algorithm with Real-time Occupancy Prediction. *Energy Procedia*, *111*(September 2016), 267–276. https://doi.org/10.1016/j.egypro.2017.03.028

Singh, A. (2019). *The IoT Applications and Consideration for NoSQL Database*. https://doi.org/Singh, Ajit, The IoT Applications and Consideration for NoSQL Database (June 5, 2019). Available at SSRN: https://ssrn.com/abstract=3399649 or http://dx.doi.org/10.2139/ssrn.3399649

Solnichkin, A. (2018). 4 Best Time Series Databases To Watch in 2019. Retrieved from devconnected website: https://medium.com/schkn/4-best-time-series-databases-to-watch-in-2019-ef1e89a72377

SpaceIQ. (n.d.). SpaceIQ. Retrieved from https://spaceiq.com/

Tang, S., Shelden, D. R., Eastman, C. M., Pishdad-Bozorgi, P., & Gao, X. (2019). A review of building information modeling (BIM) and the internet of things (IoT) devices integration: Present status and future trends. *Automation in Construction*, *101*(February), 127–139. https://doi.org/10.1016/j.autcon.2019.01.020

Tasker, J. (2019). More intelligent computing is starting to revolutionise how businesses work. Retrieved from AI - #TechCBRE website: https://tech.cbre.com/megatrends/ai/

Terkaj, W., & Pauwels, P. (2017). A method to generate a modular ifcOWL ontology. *CEUR Workshop Proceedings*, *2050*.

Terkaj, W., Schneider, G. F., & Pauwels, P. (2017). Reusing domain ontologies in linked building data: The case of building automation and control. *CEUR Workshop Proceedings*, *2050*(March 2018).

van Ballegooijen, M., Edelbroek, J., van Woensel, R., & Zarzycka, A. (2018). *Goed binnenklimaat waarborgen middels certificeringen*.

Van Den Wymelenberg, K., & Inanici, M. (2014). A critical investigation of common lighting design metrics for predicting human visual comfort in offices with daylight. *LEUKOS - Journal of Illuminating Engineering Society of North America*, *10*(3), 145–164. https://doi.org/10.1080/15502724.2014.881720

Van Der Veen, J. S., Van Der Waaij, B., & Meijer, R. J. (2012). Sensor data storage performance: SQL or NoSQL, physical or virtual. *Proceedings - 2012 IEEE 5th International Conference on Cloud Computing, CLOUD 2012*, 431–438. https://doi.org/10.1109/CLOUD.2012.18

Varjo, J., Hongisto, V., Haapakangas, A., Maula, H., Koskela, H., & Hyönä, J. (2015). Simultaneous effects of irrelevant speech, temperature and ventilation rate on performance and satisfaction in open-plan offices. *Journal of Environmental Psychology*, *44*, 16–33. https://doi.org/10.1016/j.jenvp.2015.08.001

Velux; YouGov. (2018). *The effects of modern indoor living on health, wellbeing and productivity*. Retrieved from https://press.velux.com/download/542967/theindoorgenerationsurvey14may2018-2.pdf

Vennema, E. (2019). The Brain of the Building. *RICS World Built Environment Forum Summit*.

Vergesense. (n.d.). Optimize your Workplace with AI-Powered Utilization Analytics. Retrieved from https://vergesense.com/

Vongsingthong, S., & Smanchat, S. (2015). A Review of Data management in Internet of Things. *KKU Res Journal*, 215–240. https://doi.org/10.4018/978-1-5225-1832-7.ch017

Wagner, A. (n.d.). Geometry in a Semantic Web Context. *LDAC Summer School*.

Wagner, A., Bonduel, M., & Pauwels, P. (n.d.). OMG: Ontology for Managing Geometry. Retrieved from https://www.projekt-scope.de/ontologies/omg/

Wagner, A., Moeller, L. K., Leifgen, C., & Eller, C. (2019). Building Product Ontology. Retrieved from https://www.projekt-scope.de/ontologies/bpo/

Wagner, A., & Rüppel, U. (2019). BPO: The building product ontology for assembled products. *CEUR Workshop Proceedings*, *2389*, 106–119.

Wang, W., Chen, J., & Hong, T. (2018). Modeling occupancy distribution in large spaces with multi-feature classification algorithm. *Building and Environment*, *137*(March), 108–117. https://doi.org/10.1016/j.buildenv.2018.04.002

Wang, W., Chen, J., & Song, X. (2017). Modeling and predicting occupancy profile in office space with a Wi-Fi probe-based Dynamic Markov Time-Window Inference approach. *Building and Environment*, *124*, 130–142. https://doi.org/10.1016/j.buildenv.2017.08.003

Wang, Z., & Srinivasan, R. S. (2017). A review of artificial intelligence based building energy use prediction: Contrasting the capabilities of single and ensemble prediction models. *Renewable and Sustainable Energy Reviews*, *75*(October 2016), 796–808. https://doi.org/10.1016/j.rser.2016.10.079

Wolkoff, P. (2018). Indoor air humidity, air quality, and health – An overview. *International*

Journal of Hygiene and Environmental Health, *221*(3), 376–390. https://doi.org/10.1016/j.ijheh.2018.01.015

Wong, J. K. W., Ge, J., & He, S. X. (2018). Digitisation in facilities management: A literature review and future research directions. *Automation in Construction*, *92*(April), 312–326. https://doi.org/10.1016/j.autcon.2018.04.006

Wong, L. T., Mui, K. W., & Hui, P. S. (2008). A multivariate-logistic model for acceptance of indoor environmental quality (IEQ) in offices. *Building and Environment*, *43*(1), 1–6. https://doi.org/10.1016/j.buildenv.2007.01.001

WVOI. (2019). *Arbocatalogus Onderzoekinstellingen. versie 2.1.*

Yang, J., Pantazaras, A., Chaturvedi, K. A., Chandran, A. K., Santamouris, M., Lee, S. E., & Tham, K. W. (2018). Comparison of different occupancy counting methods for single system-single zone applications. *Energy and Buildings*, *172*, 221–234. https://doi.org/10.1016/j.enbuild.2018.04.051

Zhang, C., Beetz, J., & De Vries, B. (2018). BimSPARQL: Domain-specific functional SPARQL extensions for querying RDF building data. *Semantic Web*, *9*(6), 829–855. https://doi.org/10.3233/SW-180297

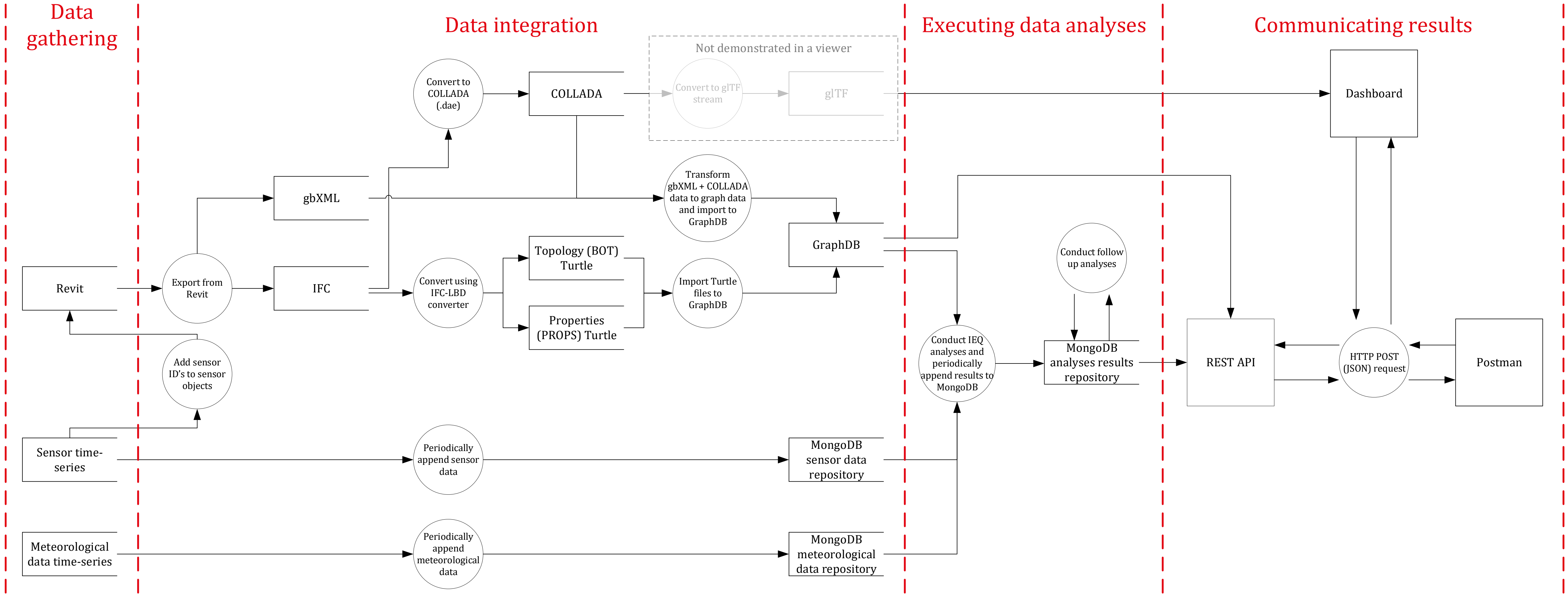
Zhao, J., Lasternas, B., Lam, K. P., Yun, R., & Loftness, V. (2014). Occupant behavior and schedule modeling for building energy simulation through office appliance power consumption data mining. *Energy and Buildings*, *82*, 341–355. https://doi.org/10.1016/j.enbuild.2014.07.033

Zhu, C., & Li, N. (2017). Study on indoor air quality evaluation index based on comfort evaluation experiment. *Procedia Engineering*, *205*, 2246–2253. https://doi.org/10.1016/j.proeng.2017.10.066

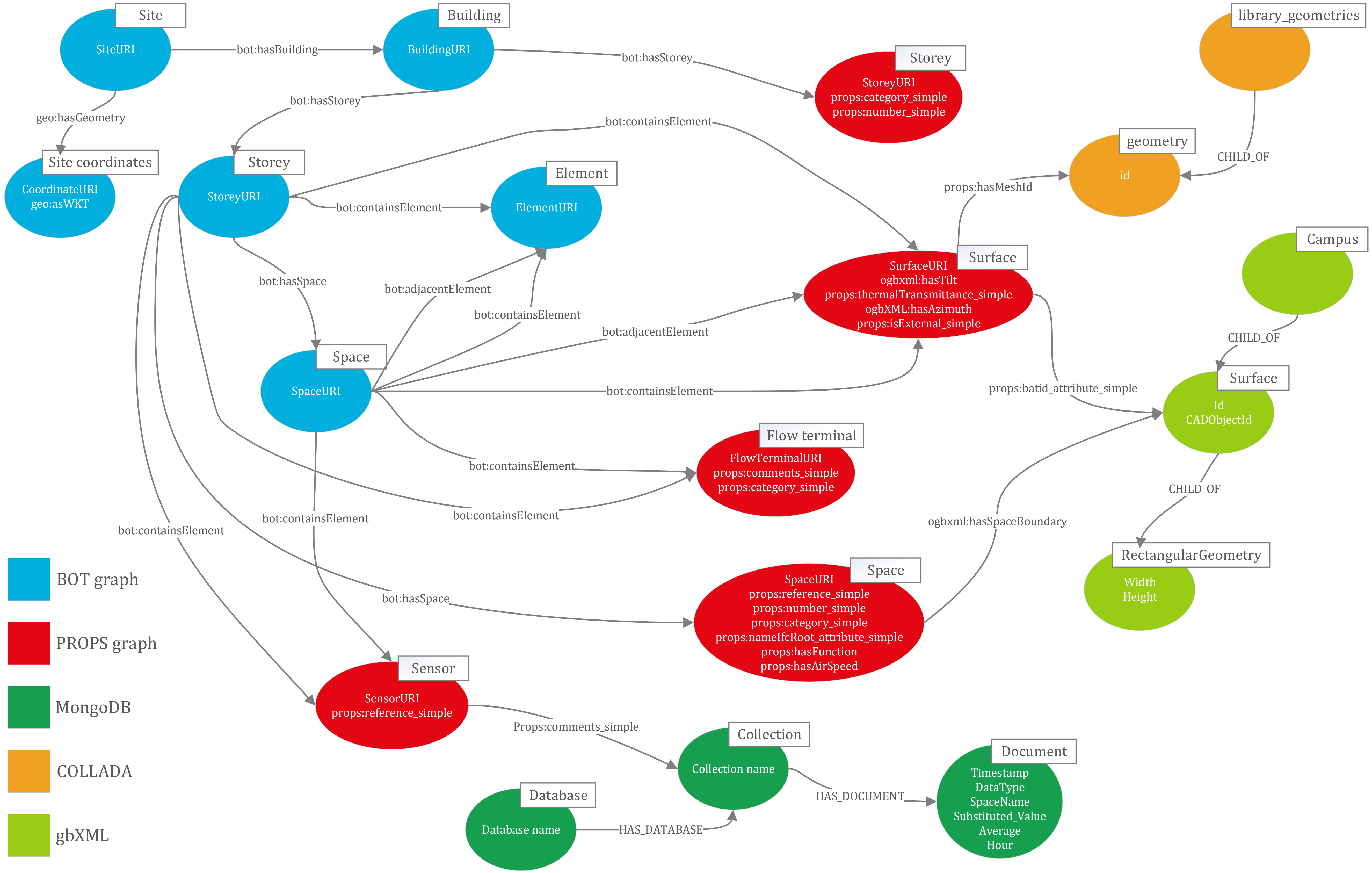
Section 11

Appendices

A1. Yourdon and Coad dataflow diagram

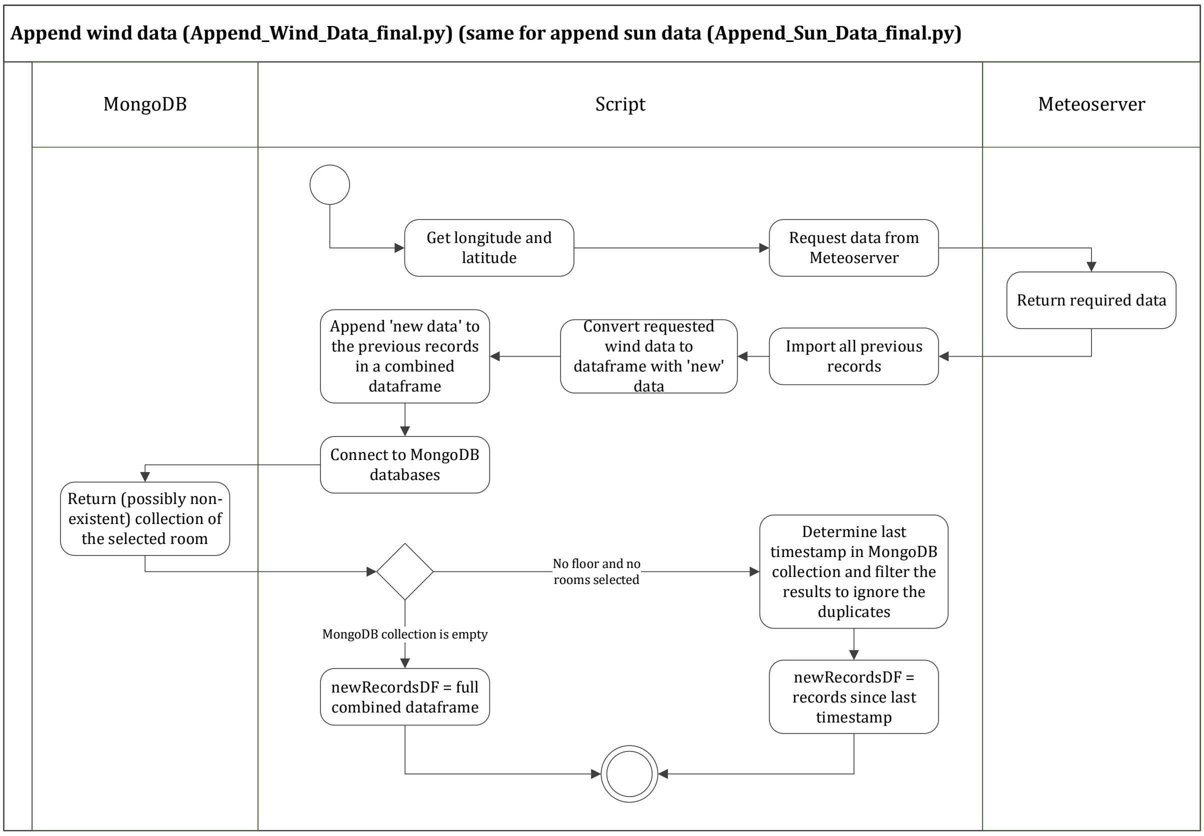


A2. Neo4J-styled diagram of the developed integration

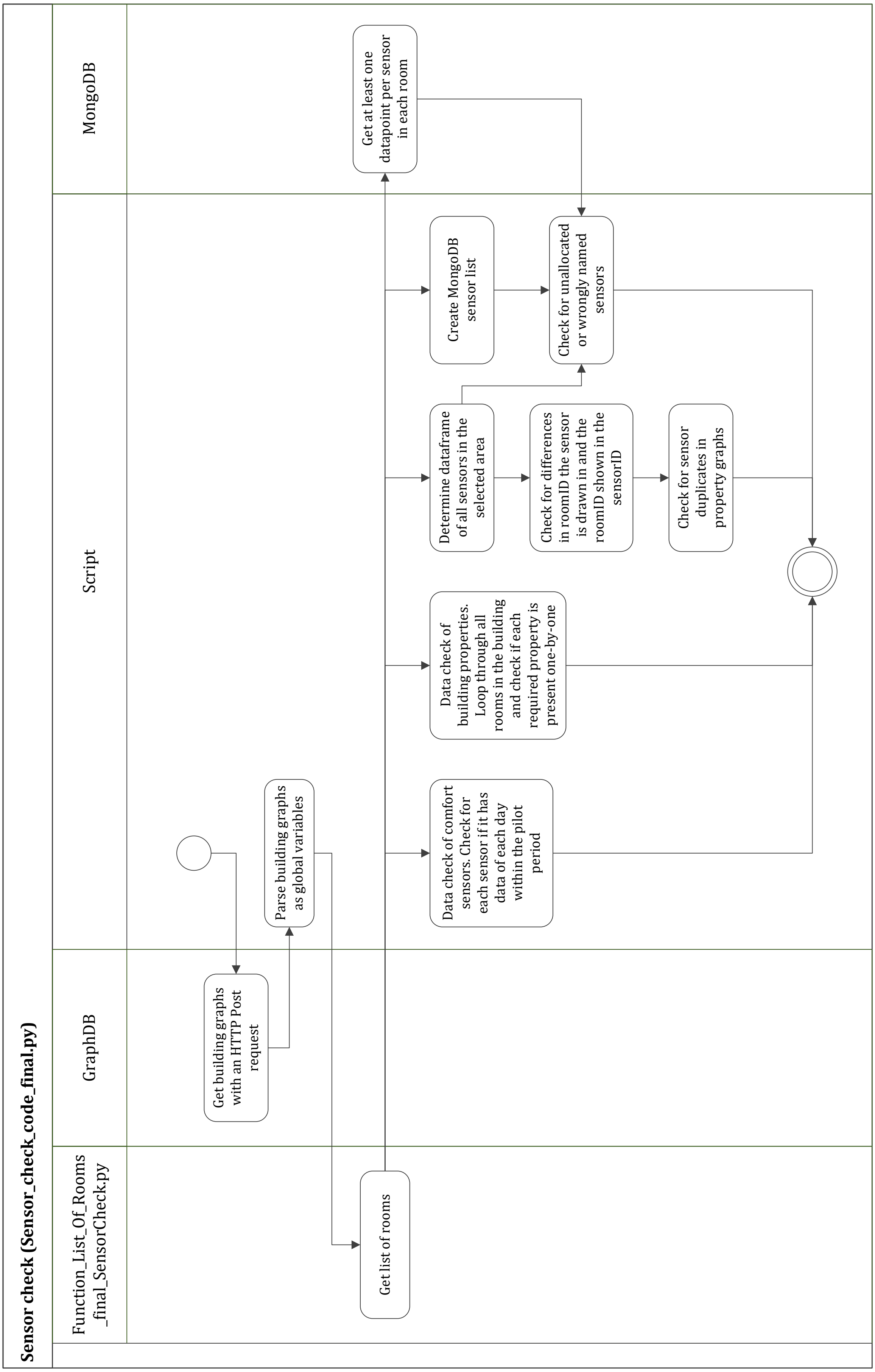


A3. UML Activity diagram of each script

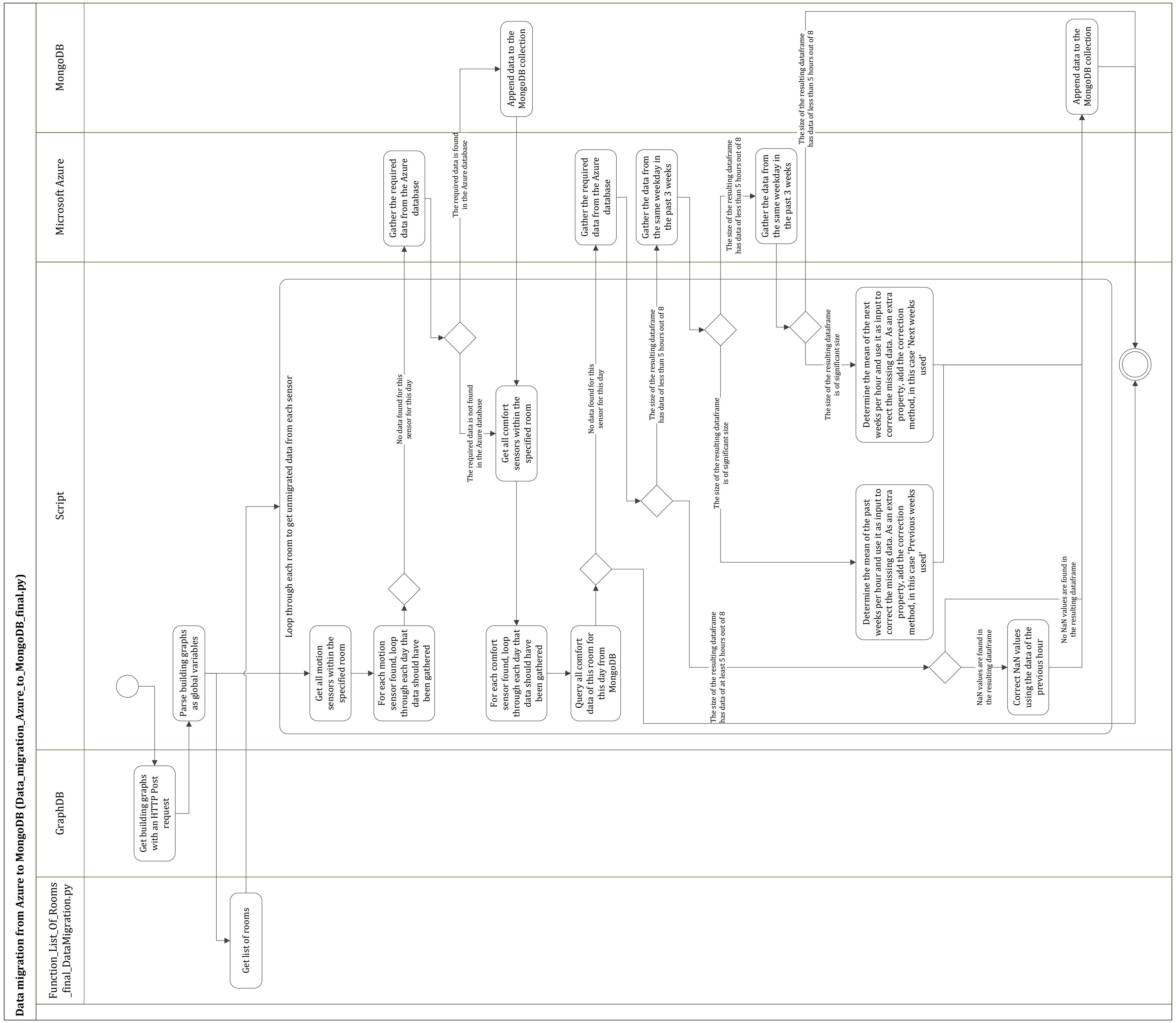
A3.1. Append wind and sun data script



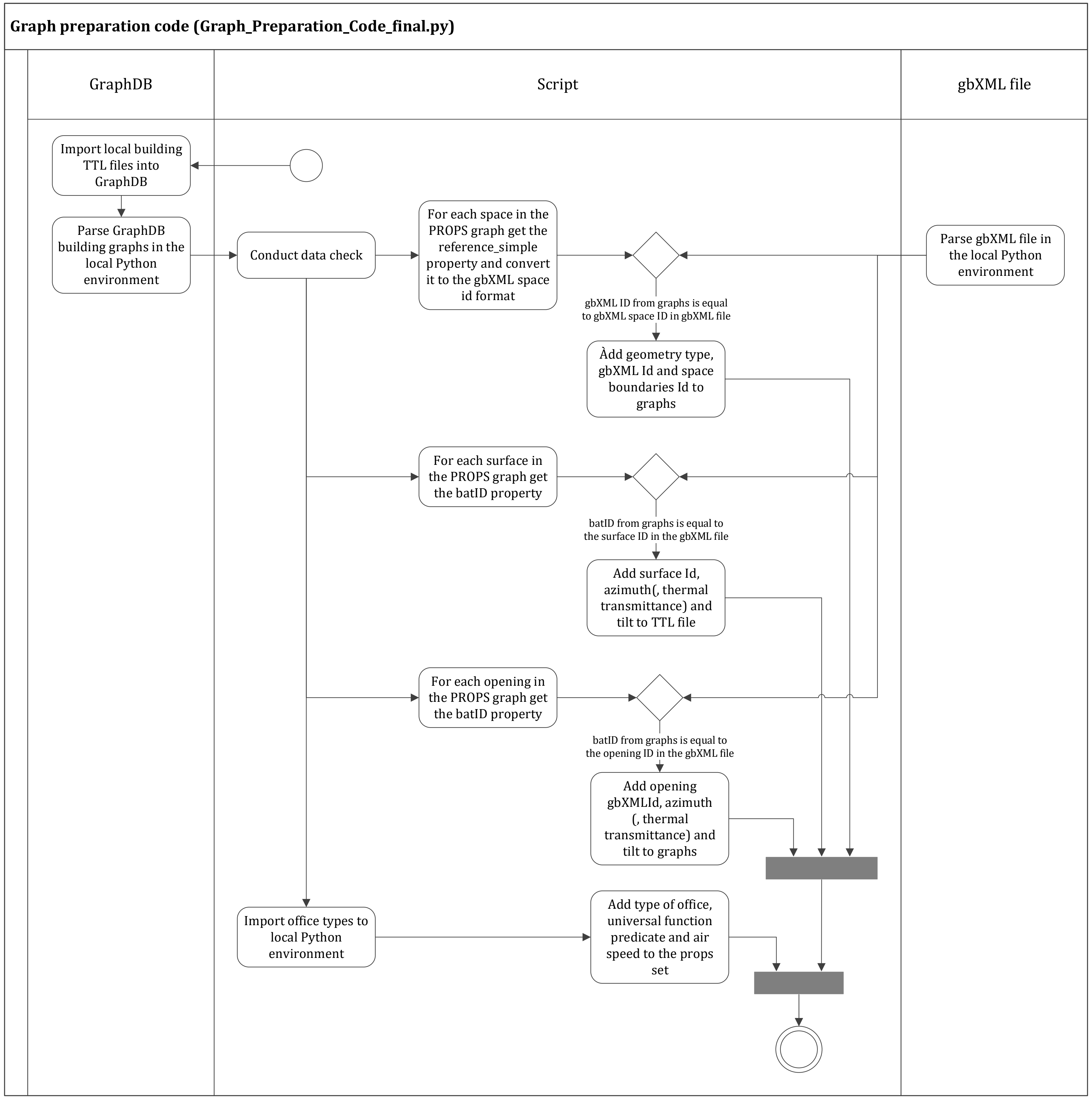
A3.2. Sensor check



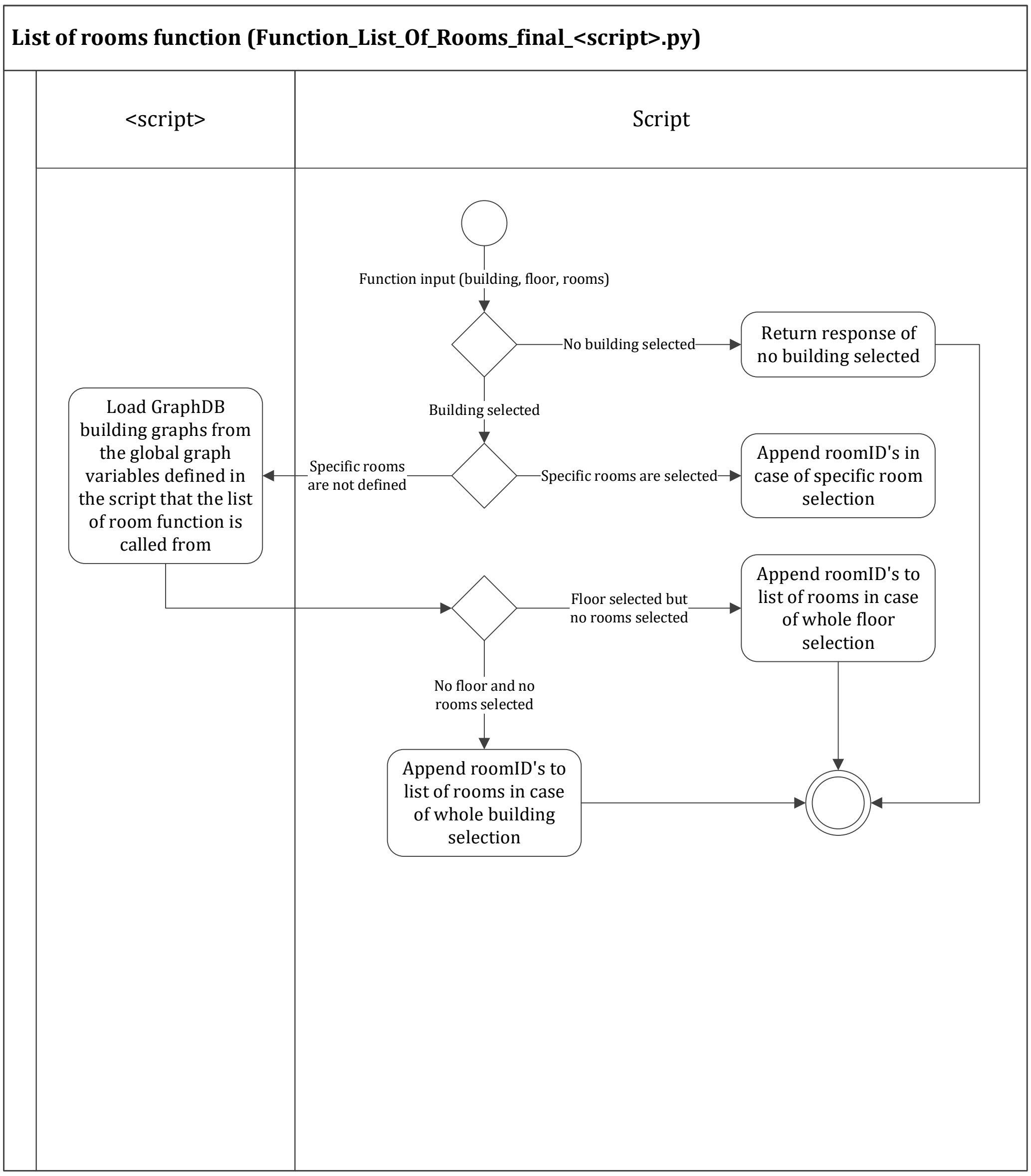
A3.2. Data migration script



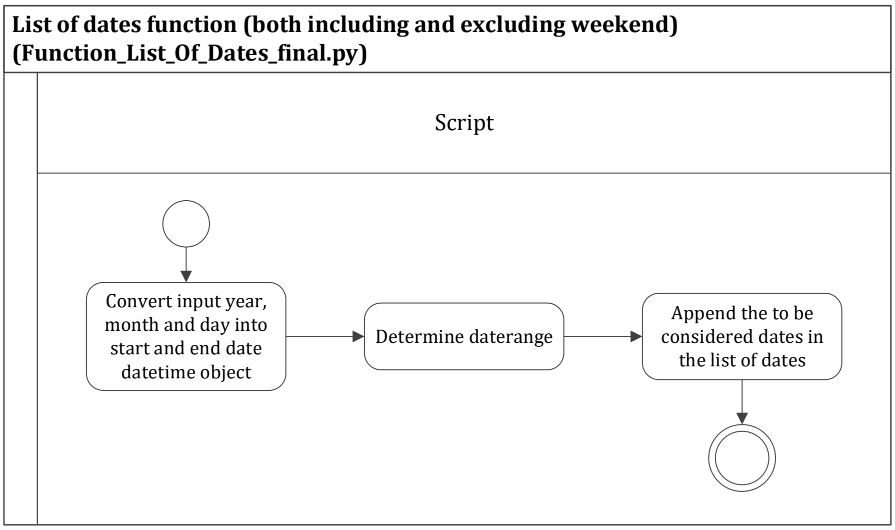
A3.3. Graph preparation script



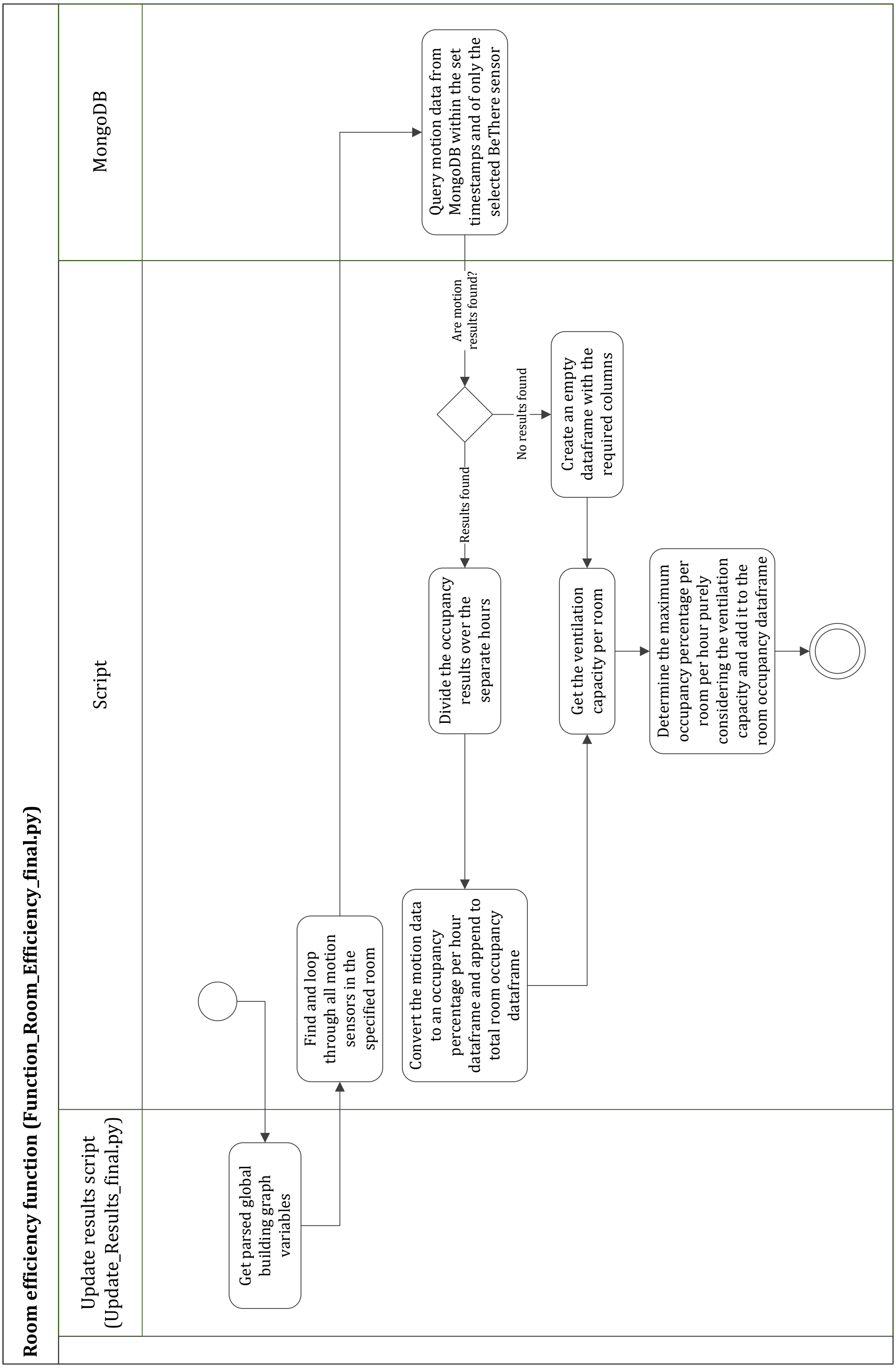
A3.5. List of rooms



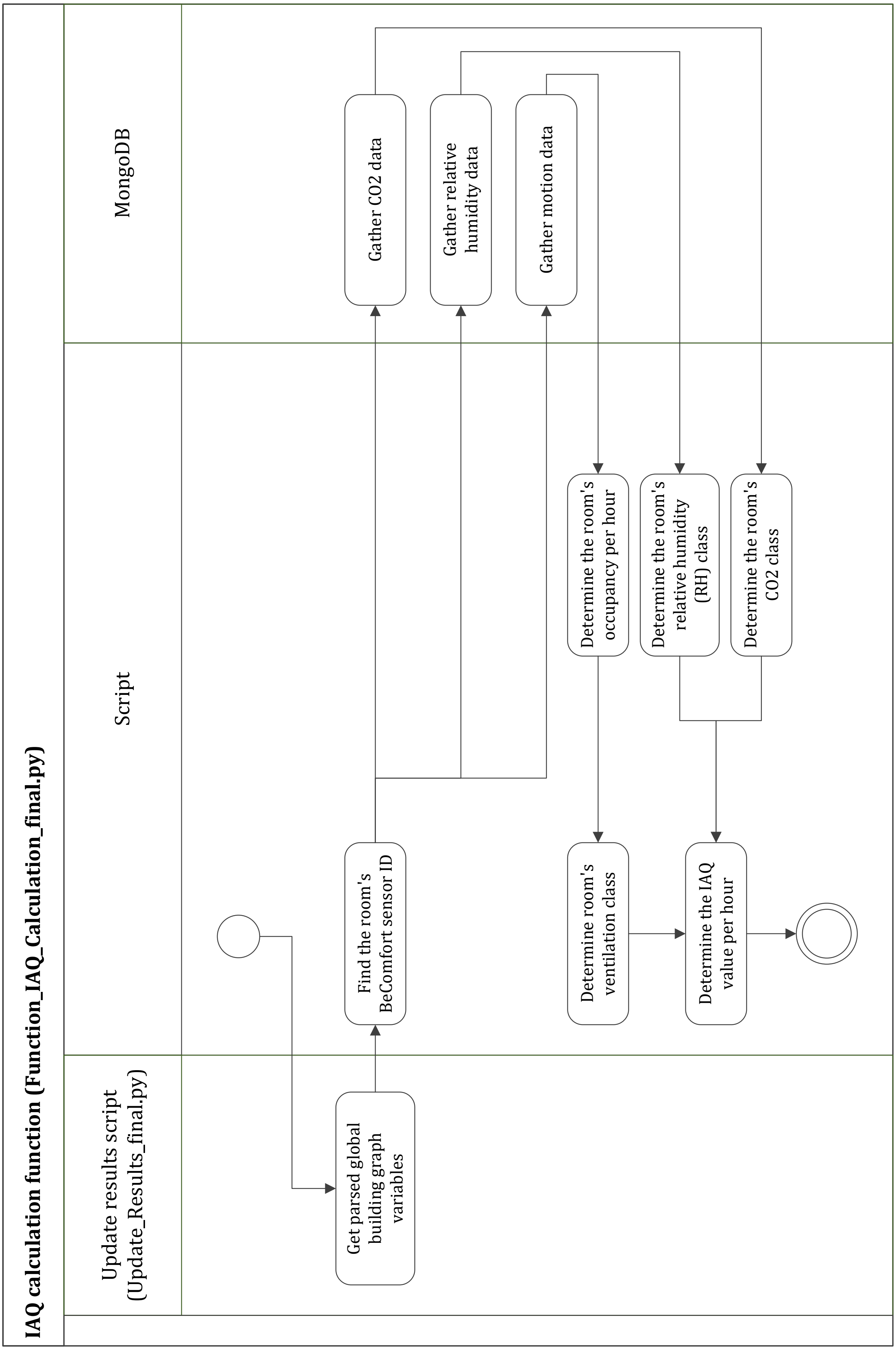
A3.6. List of dates



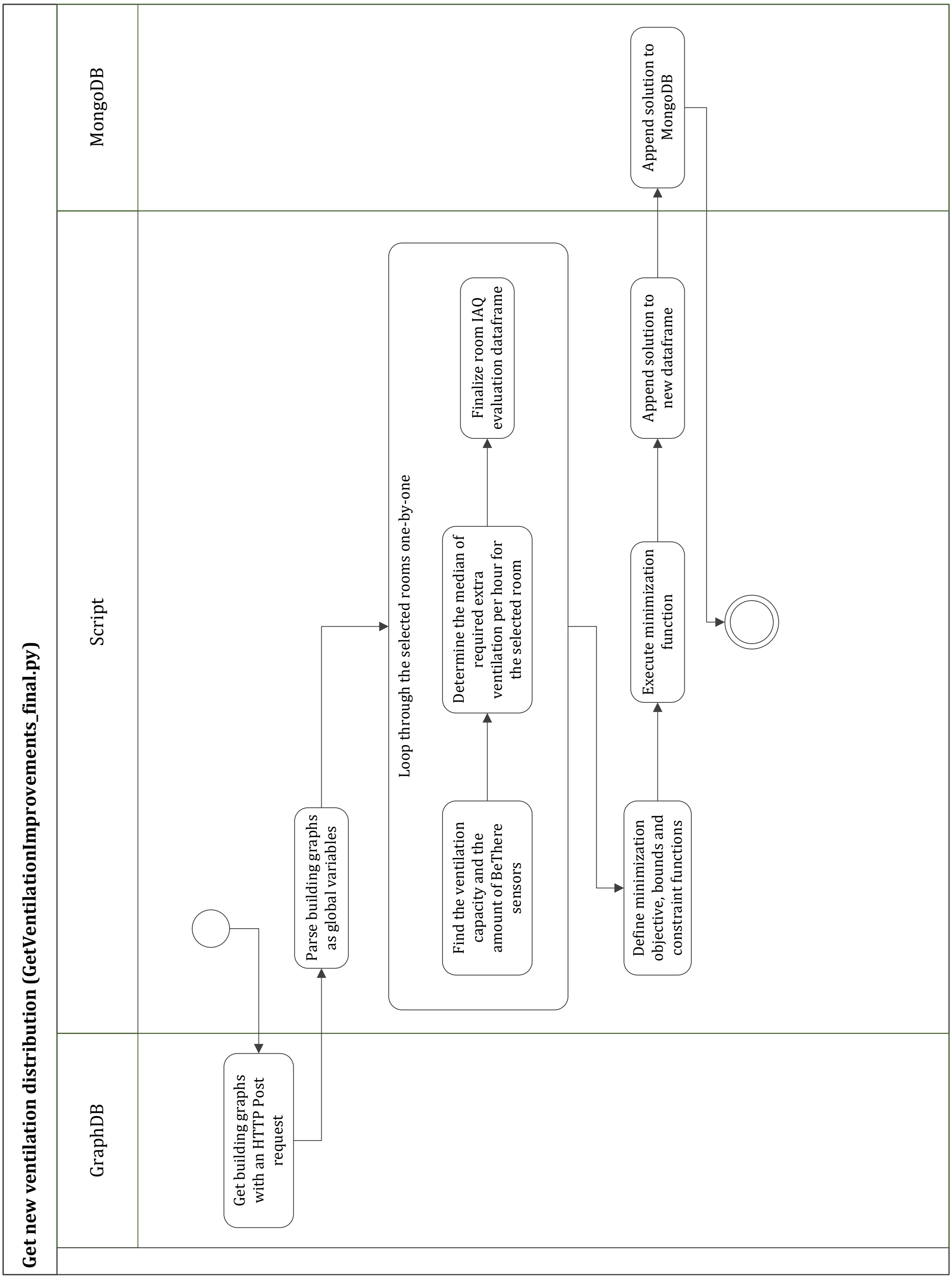
A3.7. Room efficiency



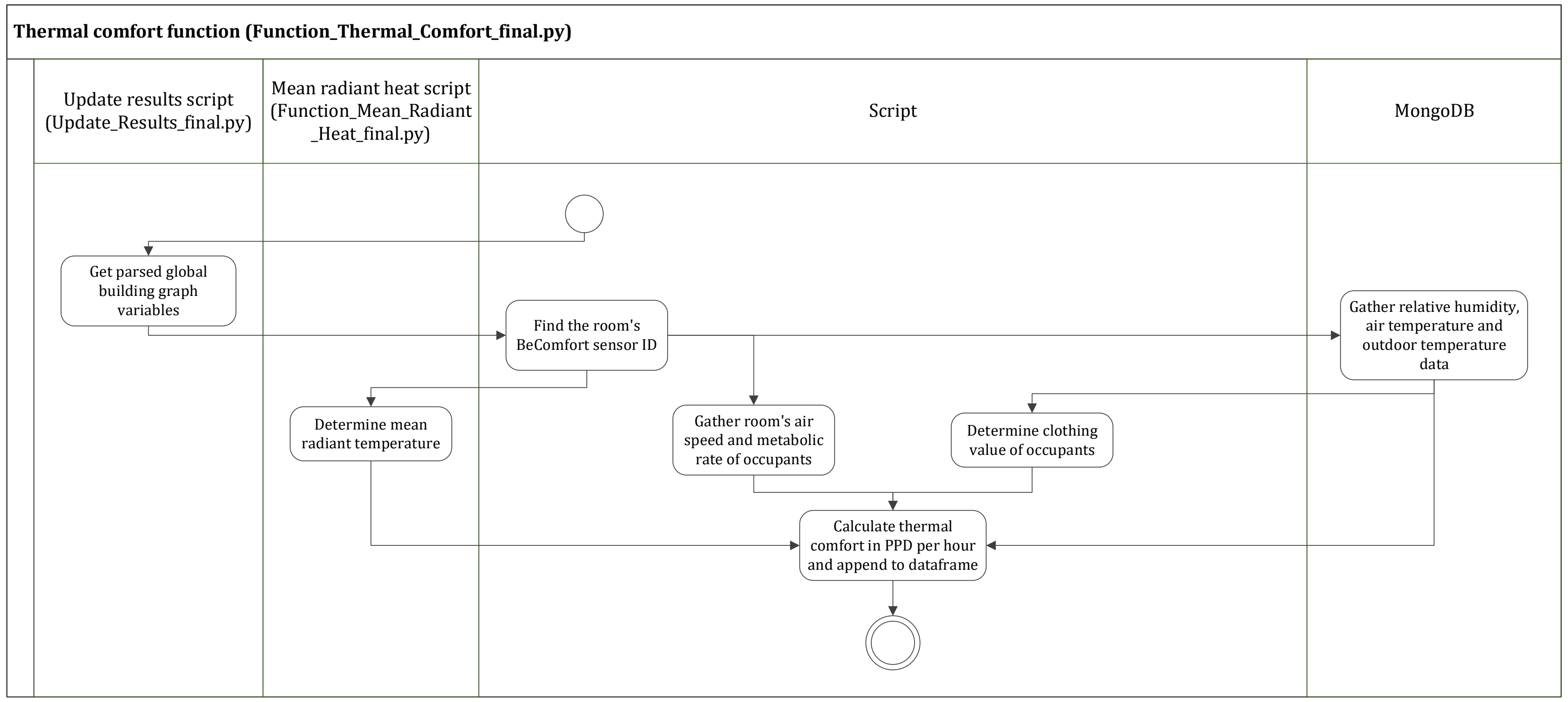
A3.8. IAQ


IAQ calculation function (Function_IAQ_Calculation_final.py)
Update results script (Update_Results_final.py)
Script
MongoDB
Get parsed global building graph variables
Find the room's BeComfort sensor ID
Gather CO2 data
Gather relative humidity data
Gather motion data
Determine the room's occupancy per hour
Determine the room's relative humidity (RH) class
Determine the room's CO2 class
Determine room's ventilation class
Determine the the IAQ value per hour

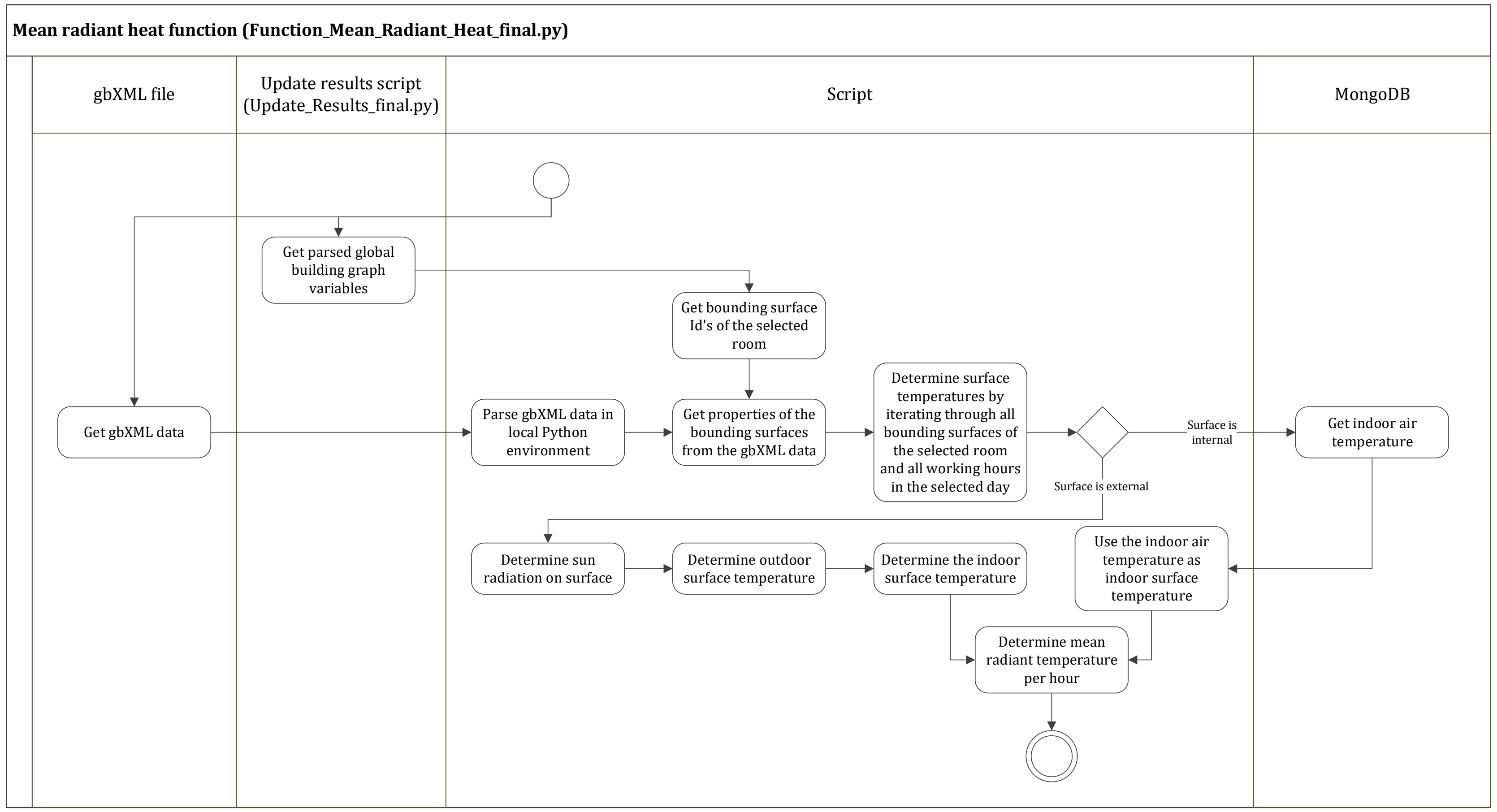
A3.9. Ventilation capacity distribution improvement



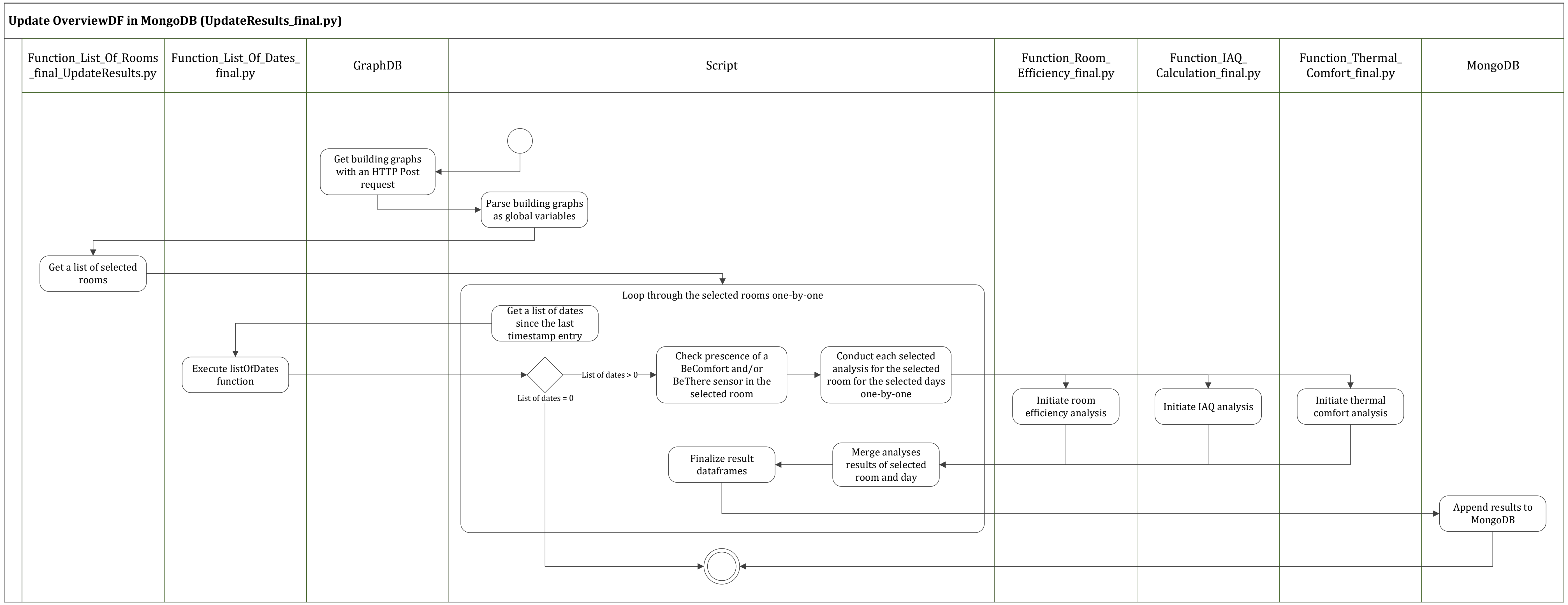
A3.9. Thermal comfort



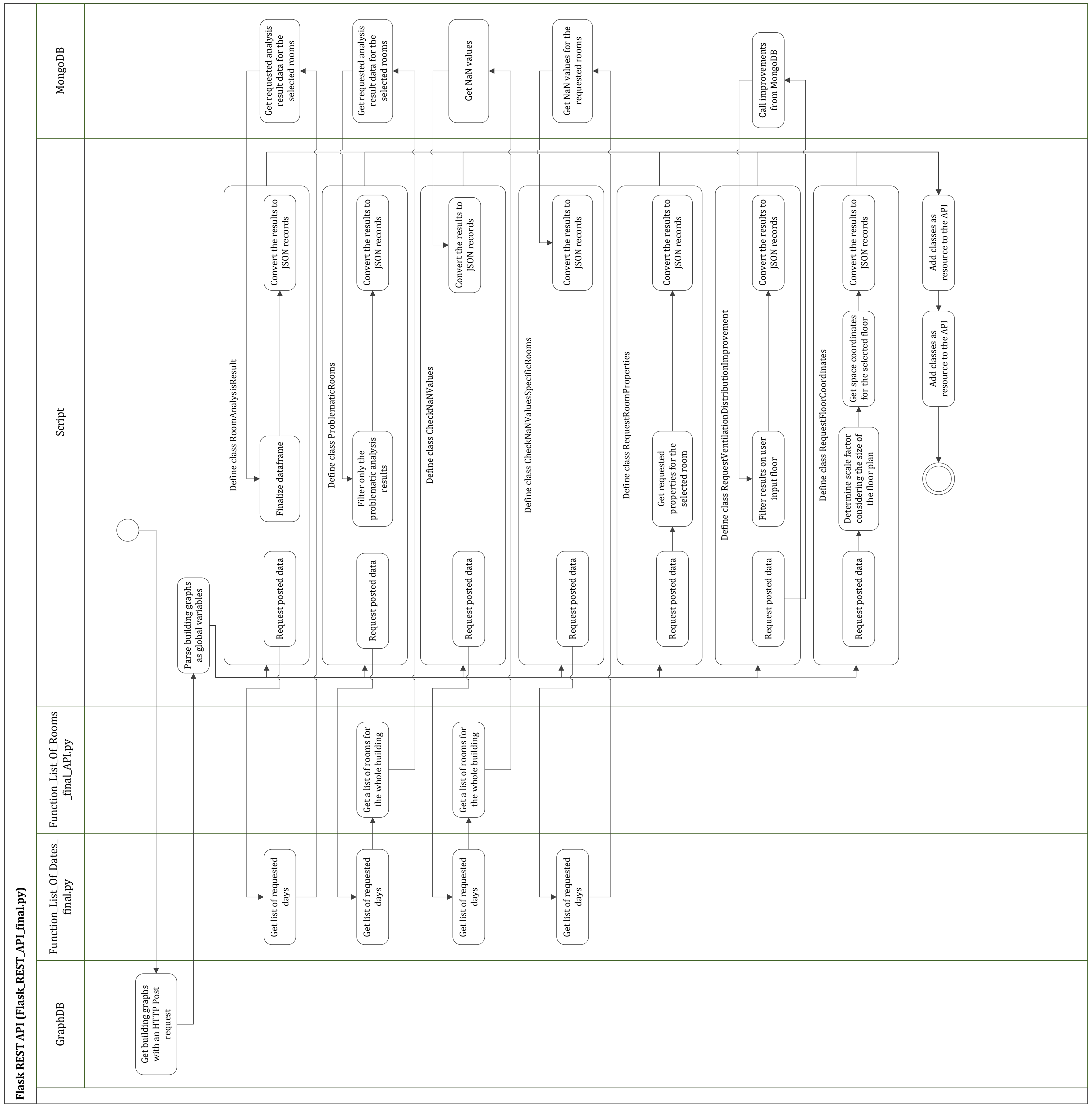
A3.10. Mean radiant heat



A3.11. Update script



A3.13. Flask REST API



A4. Indoor Environmental Quality research

Health and comfort problems in office buildings is getting increasingly prominent in both literature and the industry (Zhu & Li, 2017). Buildings are expected to meet an increased amount of health and comfortability requirements for indoor environments (Gruber, Trüschel, & Dalenbäck, 2014). As clear relations between health and comfort problems and building characteristics have been determined, a growing interest arises in occupant health and building design (Al horr et al., 2016).

One of the focus points of these indoor requirements is the Indoor Environmental Quality (IEQ). The IEQ is generally determined by four key factors: (1) Indoor Air Quality (IAQ), (2) thermal comfort, (4) visual comfort and (5) acoustic satisfaction (Al horr et al., 2016; S. Kang, Ou, & Mak, 2017; L. T. Wong, Mui, & Hui, 2008). Research has determined a clear effect of the IEQ on occupant productivity (S. Kang et al., 2017; Kosonen & Tan, 2004; Park & Yoon, 2011), comfort and health (Al horr et al., 2016).

This chapter shall investigate each of the four key IEQ factors on two aspects: (1) potential effects on occupant health and comfort, and (2) factors of influence on these factors. Additionally, it shall investigate the regulations and standards surrounding these factors, as well as methods of calculating the overall IEQ level.

A4.1. Indoor Air Quality (IAQ)

CO_2 concentrations are often perceived as the most important factor determining the IAQ (Kuchen, Fisch, Leão, & Leão, 2009; Zhu & Li, 2017). Another aspect often related to IAQ is Volatile Organic Compounds (VOC). VOC is a collective name for gas substances combined with dust and particles in the air that can become a hazard to the health of the occupant (Kuchen et al., 2009). However, as these substances are not measured by the sensors used in this thesis, VOC shall not be considered. Relative Humidity (RH) is often considered as an important factor to the IAQ as well due to its potential health effects (Fang, 1998; Wolkoff, 2018), but others do not consider it a main factor in these health effects (WVOI, 2019). Thus, relative humidity shall only be considered among the thermal comfort, not the IAQ. Consequently, this chapter shall focus on the CO_2 concentrations in relation to the IAQ.

CO_2 concentration

Potential effects on occupant health and comfort

A high CO_2 concentration is not necessarily dangerous for the health of an occupant. However, a high CO_2 concentration reduces the amount of oxygen that the body of an occupant can inhale. This may then lead to symptoms such as tiredness, headaches or concentration losses (Klimaat Beheer, 2018). Therefore, high CO_2 concentrations are generally associated with a decrease in productivity of occupants (Kosonen & Tan, 2004; Kuchen et al., 2009).

CO_2 concentrations can be precisely measured using CO_2 sensors. However, considering the numerous negative health effects of high CO_2 concentrations, literature has widely researched subjective data by collecting responses of users in different CO_2 concentrations (Kuchen et al., 2009). For example, Zhu & Li (2017) has shown that occupants felt obvious discomfort when the CO_2 concentration was 1000 parts per million (ppm). They concluded that the CO_2 concentration should stay below 700 ppm to get an excellent indoor air quality regarding CO_2.

Factors of influence

Indoor CO_2 is a metabolic product of the building occupants (Zhu & Li, 2017). Therefore, the more occupants present in a room, the quicker the CO_2 inflow rises. Two methods of reducing

CO_2 can be considered: (1) reduce the source of the CO_2 (both inside and outside of the building) or (2) increase the ventilation rate (either naturally or mechanically) (Al horr et al., 2016).

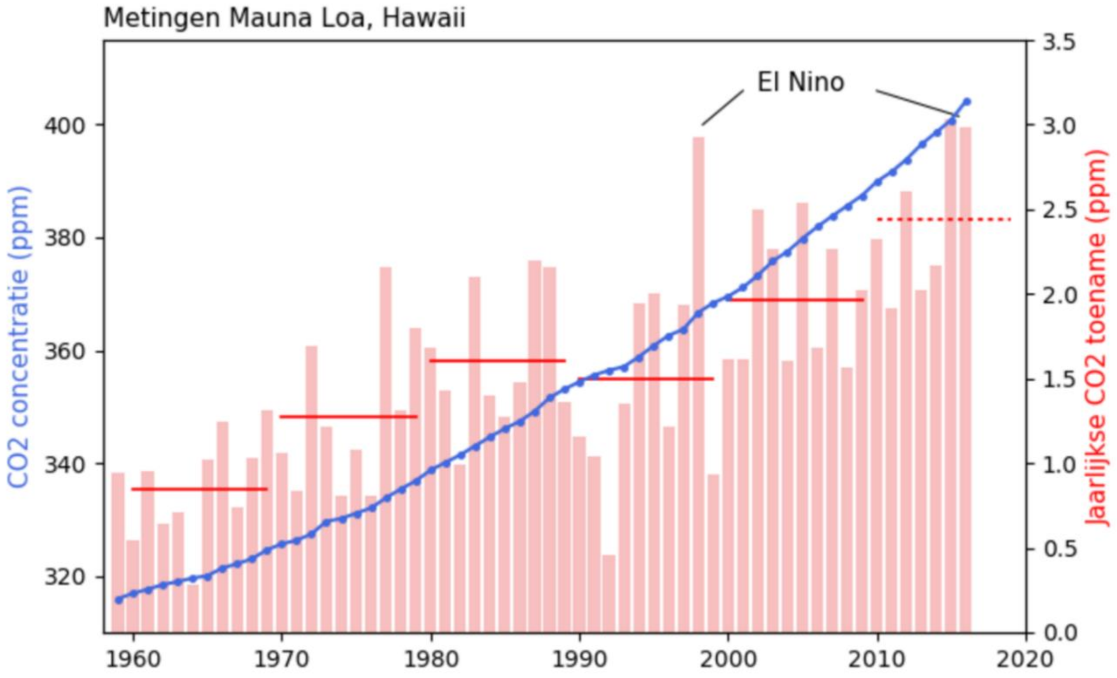


Figure 65 (KNMI, 2017): Mauna Loa Observatory atmospheric CO2 concentration measurements

The first method, reducing the sources of CO_2, regards both inside and outside. Due to the rising atmospheric CO_2 concentration, the Mauna Loa Observatory in Hawaii has measured more than 400ppm in 2015, as shown in Figure 65 (KNMI, 2017). The ambient CO_2 concentration is therefore rising, making it increasingly difficult to dilute the indoor CO_2 concentration using outside air.

As the main source of indoor CO_2 concentration is the occupants, reducing the number of occupants in a room is the most logical step in reducing the CO_2 inflow. However, logically, a tenant desires efficient exploitation, making this a less favourable step.

Therefore, the second method is most important in controlling the indoor CO_2 concentration: increase the ventilation rate. Ventilation is one of the most essential factors contributing to indoor air quality, both measured and perceived quality (Kosonen & Tan, 2004; Kuchen et al., 2009; Park & Yoon, 2011; Varjo et al., 2015). Ventilation can be both naturally as well as mechanically executed. However, in colder climates, such as the Netherlands, opening a window as quick natural ventilation is only comfortable in the summer. The winter season in the Netherlands has an average temperature of 3.4 °C (KNMI, 2018), making opening a window have a large impact on the thermal comfort (Kuchen et al., 2009). Sufficient mechanical ventilation is therefore an essential facility in buildings. Mechanical ventilation is usually designed according to key figures or according to a certain standard to achieve a healthy building label, such as LEED. These factors of influence lead to the following influence diagram shown in Figure 66.

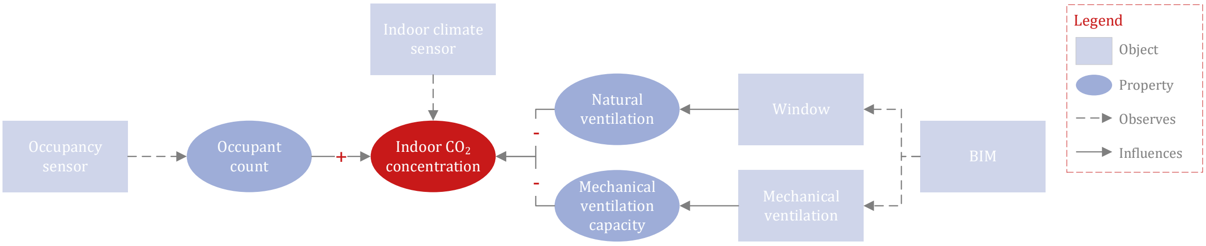


Figure 66: Influence diagram indoor CO_2 concentration

A4.2. Thermal comfort

Potential effects on occupant health and comfort

Thermal comfort is probably the most important factor affecting the IEQ, strongly affecting the indoor satisfaction as well as the productivity of an occupant (Al horr et al., 2016; S. Kang et al., 2017; Maula et al., 2016; Quang, He, Knibbs, De Dear, & Morawska, 2014). Thermal comfort is however also dependent on numerous variables and therefore difficult to maintain in the right bounds while not wasting too much energy.

Factors of influence

Two commonly used metrics to measure thermal comfort is the Predicated Mean Vote (PMV) and the Predicted Percentage of Dissatisfied (PPD) (Al horr et al., 2016). The PMV index is determined by six parameters (S. Kang et al., 2017; Katafygiotou & Serghides, 2015):

1) Physical variables (see Figure 67):
    a. air temperature;
    b. relative humidity;
    c. air velocity and
    d. mean radiant temperature.
2) Human variables:
    a. clothing insulation and
    b. metabolic rate (type of activity).

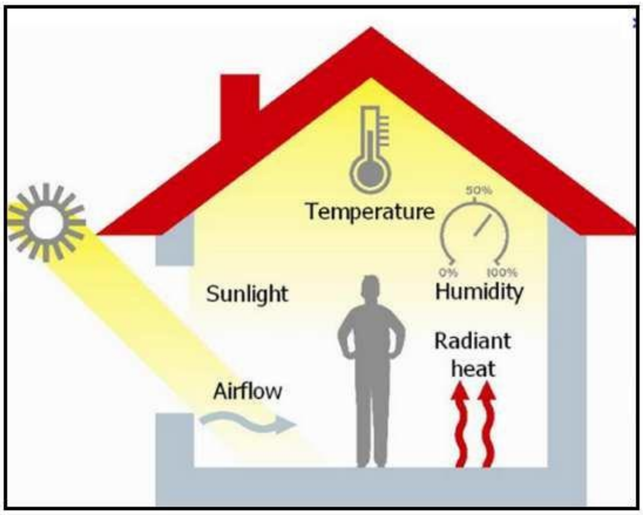


Figure 67 (Alwetaishi, 2016): Schematic of the physical influences on thermal comfort

The physical variables are in turn influenced by other variables. The first is the amount of occupants in the room, as humans react on the environment through radiation, evaporation and convection (ASHRAE, 2010; Northern Arizona University, 2013). Additionally, outdoor climate and season (Frontczak & Wargocki, 2011), as well as the building characteristics, such as thermal insulation and building tightness, determine the value of the physical variables (Autrup et al., 2007).

Consequently, thermal comfort is a complex and partly subjective factor in determining the IEQ. The factors of influence lead to the influence diagram shown in Figure 68.

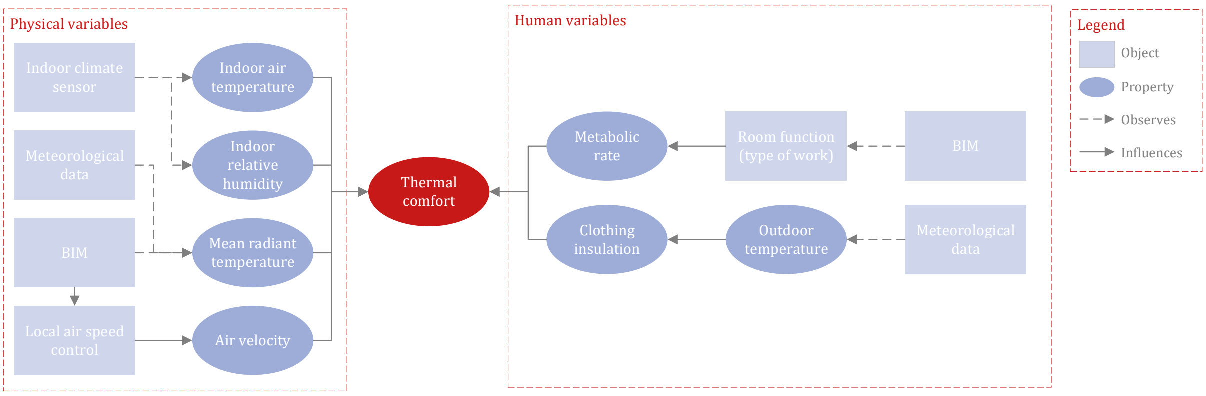


Figure 68: Influence diagram thermal comfort

A4.3. Relative humidity

Potential effects on occupant health and comfort

Humidity can be measured in either absolute humidity or relative humidity. Absolute humidity equals the amount of water in g_{water}/kg_{air} at a given pressure. The relative humidity equals the amount of water vapour in the air in relation to the maximum amount of moisture in the air at a given temperature (Nguyen, Schwartz, & Dockery, 2014). The relative humidity is expressed as a percentage with 50% being the optimum value (Lowen, Mubareka, Steel, & Palese, 2007), but 40% to 60% RH being a good range to adhere to (DUUX, n.d.).

Humidity is both of importance to thermal comfort and indoor air quality. A high relative humidity causes occupants to feel chilly in cold weather and hot in warm weather. The human body needs to cool by evaporation, which it cannot in humid warm air (Dotson, 2018). However, a low relative humidity causes dry skin and discomfort in the nose (DUUX, n.d.; Level, 2017).

Factors of influence

The relative humidity is determined by the outside humidity and any heating or cooling source that is present indoors (Northern Arizona University, 2013). The first and foremost principle is that cold air is dry air. Relative humidity depends on the temperature, so when cold air enters a building and is warmed up, the relative humidity of the entered air is decreased (Northern Arizona University, 2013). Therefore, when cold air enters a building it makes the indoor air dryer (Bailes, 2016). Cold air can enter through natural ventilation but also through any air leaks in the building. Therefore, when the relative humidity is too low, seal up any air leaks in the façade that might bring in cold air (Bailes, 2016). A source that increases the humidity is the occupant. Occupants add moisture to the air by exhaling approximately 200 millilitres of water vapour per hour while awake (Level, 2017; Northern Arizona University, 2013). Another possible source to increase the relative humidity is evaporative air conditioning, which adds moisture to the air (Condair, n.d.; Northern Arizona University, 2013). Refrigerated air conditioning however generally removes moisture when cooling the air (Northern Arizona University, 2013). These factors of influence lead to the influence diagram shown in Figure 69.

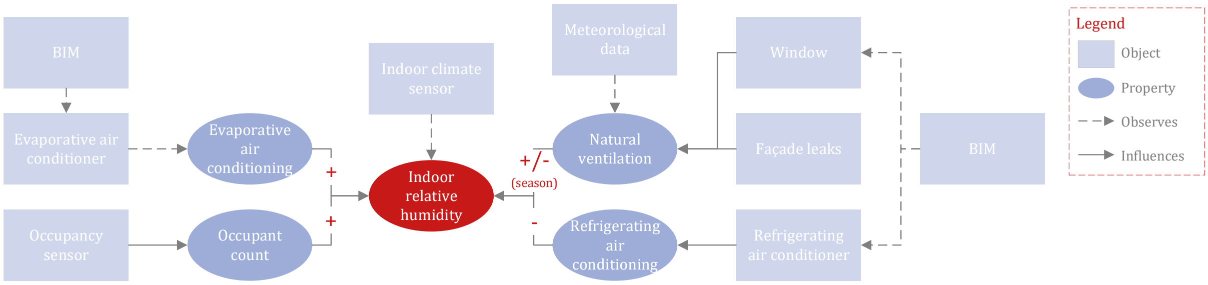


Figure 69: Influence diagram indoor relative humidity

A4.4. Visual comfort

Potential effects on occupant health and comfort

Lighting is required in every office and its intensity and type has a strong effect on occupants (Serghides, Chatzinikola, & Katafygiotou, 2015). Visual comfort entails the view and lighting quality of a workspace (Al horr et al., 2016). For example, preference for a close distance to windows seems to be well argued in literature (Aries, 2005). High visual comfort can relieve eye symptoms, reduce tiredness, decrease motivational problems, increase productivity, increase comfort and improve satisfaction (S. Kang et al., 2017). Visual comfort even impacts comfort of the occupant after work (C. Y. Chang & Chen, 2005). The visual comfort preferences do depend on the type of work being conducted. For example, people working on a computer generally prefer lower illumination levels than people not working on a computer (S. Kang et al., 2017).

Factors of influence

So visual comfort is mainly determined using two criteria: (1) view and (2) lighting. View is determined by the distance to a window and the window itself. Lighting is determined by the illumination level and the type of lighting. Generally, natural lighting (daylight) is preferred as lighting source for best human visual comfort. Occupants seem to feel uncomfortable when artificial lighting is overused instead of daylight access (Galasiu & Veitch, 2006). For daylight access, the geometry of windows, amount of glazing, etc. are of impact. To realise an optimal visual comfort, the type of work, expected daylight access, glare, window properties, artificial lighting and views should be assessed altogether while designing (Huang, Zhu, Ouyang, & Cao, 2012; Van Den Wymelenberg & Inanici, 2014).

The factors of influence lead to the following influence diagram:

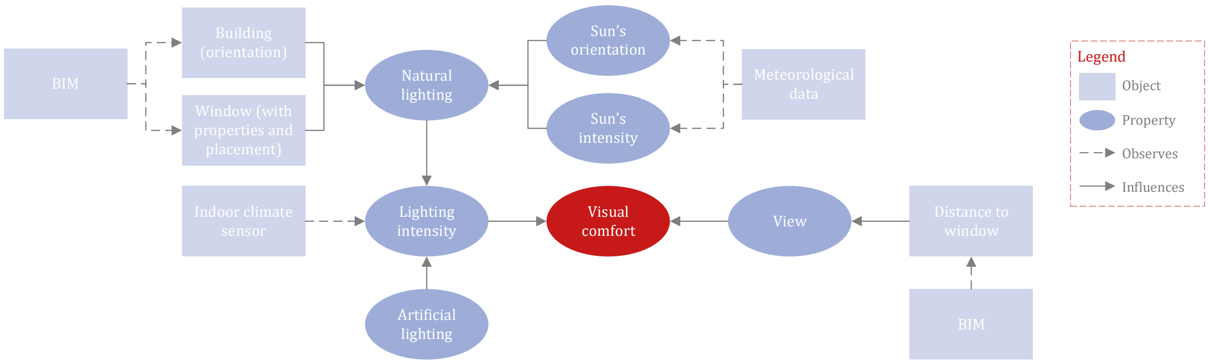


Figure 70: Influence diagram visual comfort

A4.5. Acoustic satisfaction

Potential effects on occupant health and comfort

The acoustic quality in offices is an important aspect. Acoustic quality have been noted to reduce productivity (Abbott, 2004; Al horr et al., 2016; Kanteyan, Utami, Prasetiyo, & Yanti, 2018; Seddigh, Berntson, Jönsson, Danielson, & Westerlund, 2015), cause disturbances (Jensen, Arens, & Zagreus, 2005; Seddigh et al., 2015), cause cognitive stress (Abbott, 2004; Jensen et al., 2005; Seddigh et al., 2015), and even tiredness and lack of motivation (Jahncke, Hygge, Halin, Green, & Dimberg, 2011).

Acoustic satisfaction is determined by combining both the noise level and speech privacy (Jensen et al., 2005). For instance, open-plan offices have been repeatedly criticised due to, among others, a higher noise level (Seddigh et al., 2015) and a lack of speech privacy (Kaarlela-Tuomaala, Helenius, Keskinen, & Hongisto, 2009). However, in assessing the acoustic satisfaction, the type of work is an essential element to consider. Call centres for example have a high amount of speech noise but can therefore hardly follow the conversations of their colleagues. In more silent offices, the conversations can be followed, possibly making it more distracting. Additionally, office noise does not necessarily have to be negative as is often suggested (Rasila & Jylhä, 2015), some occupants might prefer a little noise around them, others prefer to work in complete silence. It is therefore important to find a balanced noise environment instead of a completely silent one. Achieving such a balanced environment is difficult though as it is context-specific (Rasila & Jylhä, 2015). Providing occupants with a variety of possible office environments in which, among others, noise is a varying factor, might be a solution to realising balanced noise environments. A possible method is the one adopted by Autodesk while redeveloping their Toronto office. Prior to renovating their whole floor, they conducted a survey among all the employees that were going to work there indicating their work environment preferences, including noise level. After, they used generative design techniques in generating the best floor plan (Ledet Training, 2018).

Factors of influence

Consequently, the aim would be to provide occupants with several workplace options that vary in, among others, acoustic properties. As acoustic satisfaction seems to be determined by both speech privacy and noise level, these two variables should vary. The former, speech privacy, is mainly determined by the amount of occupants present. Special call rooms or concentration workplaces can be provided to provide variety in speech privacy on a floor. The latter, noise level, can be adjusted by the number of persons present, as being the source of noise, as well as room characteristics. Materials play a crucial role in determining the sound absorbing abilities in a room (Seddigh et al., 2015). Additionally, sound masking might provide a solution in eliminating acoustic irritations (Al horr et al., 2016). The factors of influence lead to the influence diagram shown in Figure 71.

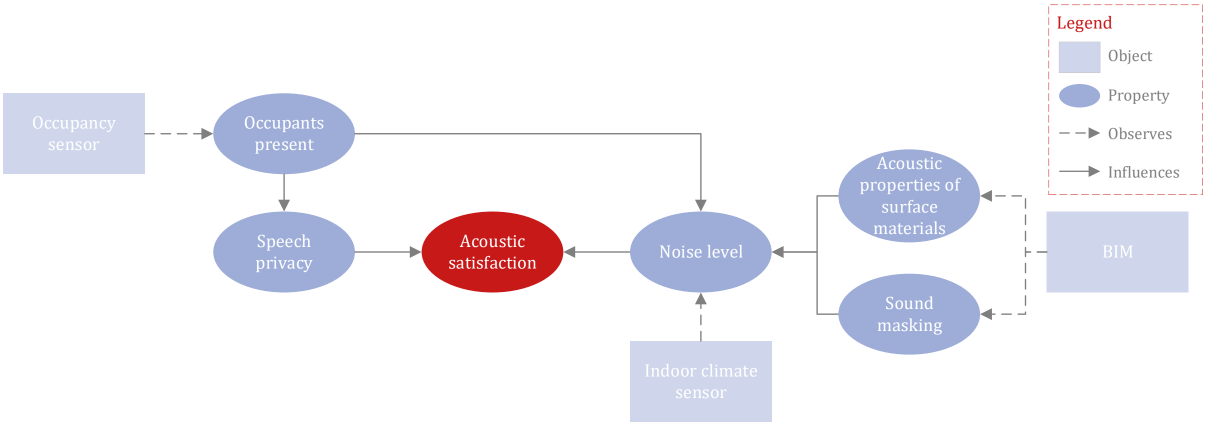


Figure 71: Influence diagram acoustic satisfaction

A4.6. Evaluation criteria for indoor environmental quality

Regulations associated with indoor environmental quality

The most notable regulations surrounding safe and healthy work environments in the Netherlands is the Arbowet. The Arbowet provides direction to adjusting measures to provide sufficient work environments (Ministerie van Sociale Zaken en Werkgelegenheid, n.d.). In their regulations NEN-norms might be used to provide a basis for arguing their required values. NEN-norms aim to standardize and normalize the regulations in numerous different contexts.

As the Arbowet is focussed on realising a safe and healthy work environment, their regulations shall be used in this thesis. In the Arbo catalogue of March 2019, they elaborated on several performance levels to aim for in work environments, among others offices. Three performance categories were distinguished: A (very good), B (good) and C (acceptable) (WVOI, 2019). The influencing factors as researched in chapter A4.1 until A4.5 shall be quantified in criteria. These criteria shall distinguish between the three performance levels as specified in the Arbo catalogue, as well as use the values from the Arbo catalogue where possible.

Standards associated with indoor environmental quality

Green building standards aim to improve building quality in both design and operation. Though, less attention was paid to post-occupancy evaluations – if a green rating actually leads to higher occupant satisfaction in practice (Altomonte, Schiavon, Kent, & Brager, 2019).

However, this is starting to change. Both LEED and BREEAM slightly reward post-occupancy evaluations (Altomonte et al., 2019). A more significant change is the introduction of the WELL standard. The WELL standard version 2 contains 112 performance measurements based on ten key aspects: air, water, nutrition, light, movement, thermal comfort, sound, materials, community and mind (van Ballegooijen, Edelbroek, van Woensel, & Zarzycka, 2018). Interestingly, the WELL standard requires a re-evaluation every three years, creating the possibility of losing the standard if the performance deteriorates. This process is schematically explained in the commissioning cycle shown in Figure 72.

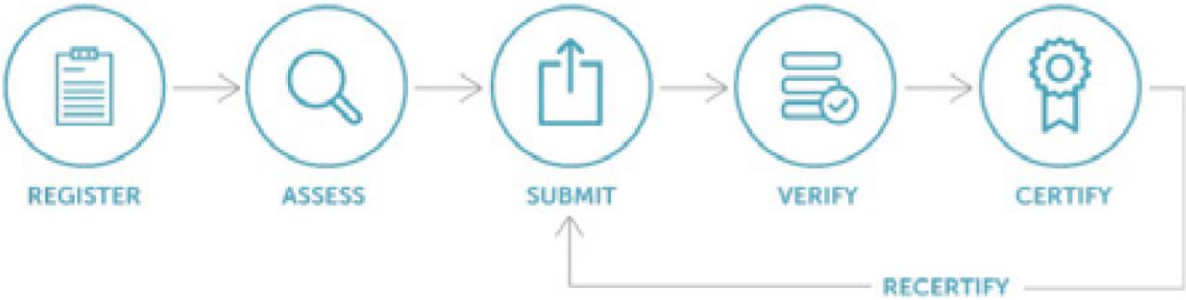


Figure 72 (van Ballegooijen et al., 2018): WELL commissioning cycle

The most notable differences between WELL and other standards, such as LEED and BREAAM, is the focus of WELL on the health and experience of the occupant instead of sustainability, and the focus on maintaining performance throughout the building lifecycle (van Ballegooijen et al., 2018). As this thesis focusses on maintaining building efficiency and user comfort during the building lifecycle, the WELL standard shall be considered instead of LEED and BREAAM.

Criteria used in this thesis

Both the Arbo catalogue and the WELL standard do not contain values on all variables determining indoor environmental quality researched in chapter A4.1 until A4.5. Therefore, some values are derived from the Arbo catalogue and some from the WELL standard. The distinguished performance levels of the Arbo catalogue are kept as a backbone for the requirements, and its values shall thus be used where possible.

Table 1 contains all criteria that shall be used in this thesis to continuously derive building improvement possibilities during the building lifecycle. The criteria categories are set up in the same order as researched in chapter A4.1 until A4.5. For each individual criteria the source (either Arbo catalogue or WELL) is provided in the category naming.

Table 1: Criteria used determining indoor environmental quality

Class **Criteria category**	**3** **(very good)**	**2** **(good)**	**1** **(acceptable)**	**0** **(not acceptable)**
Indoor Air Quality (IAQ)				
Ventilation capacity (m^3/hour/person) (Arbo)	>60	60 > 45	45 > 36	< 36
CO_2 concentration (ppm) (Arbo)	< 600	600 < 800	800 < 1000	> 1000
Relative humidity (%) (WELL)	30-50%	25-29% or 51-55%	20-24% or 56-60%	<20% or >60%
Thermal comfort (PPD) (Arbo)	<= 10	<= 10	10 < 15	> 15
Operative temperature (°C) (Arbo) (winter)	21 – 23	20 – 24	19 – 25	<18 or >25
Operative temperature (°C) (Arbo) (summer)	23.5 – 25.5	23 – 26	22 – 27	<22 or >27

Visual comfort				
Illumination level (lux) (Arbo)	750	500	500	< 500
View factor (1 to 5) (WELL)	5 or 4	3	2 or 1	0
Acoustic satisfaction				
Maximal sound level (dB(A)) (Arbo)	Depends on function (see Table 2)			

Table 2: Maximum sound level in different contexts (translated table from (WVOI, 2019))

Maximal sound level (dB(A))	**Concentration intensity**	**Communication intensity**
80	Concentration hardly possible	High
75	Low concentration (vacuum cleaning)	Significant
65	Low concentration (normal conversation	Average
55	Average (meeting room)	Low
45	Significant (reception)	Very low
35	High (office researcher)	None

The view factor is determined by the actual view an occupant has to the outside. The criteria is evaluated using the vertical- and lateral angle from the point of view of the occupant to the window dimensions, as shown in Figure 73 and Figure 74.

Prelim. View Rating	**View Angle**	
	Min - Max (degrees)	Gray Zone Range (degrees)
1	1 - 4	
1 or 2		4 - 5
2	5 - 9	
2 or 3		9 - 11
3	11 - 15	
3 or 4		15 - 20
4	20 - 40	
4 or 5		40 - 50
5	50 - 90	

Figure 73 (Heschong Mahone Group, 2003): View rating based on view angle

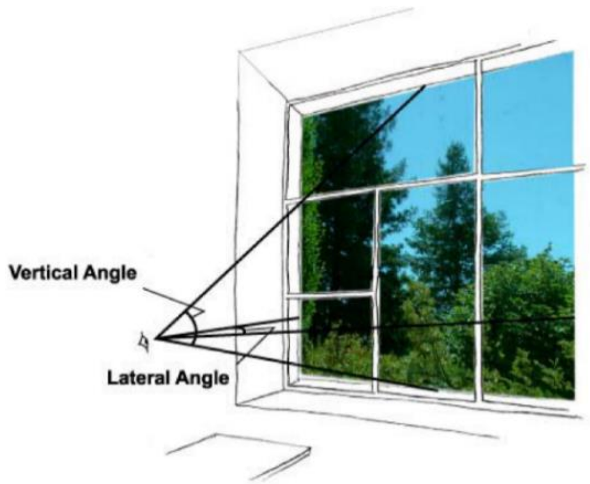


Figure 74 (Heschong Mahone Group, 2003): Vertical- and lateral view angle

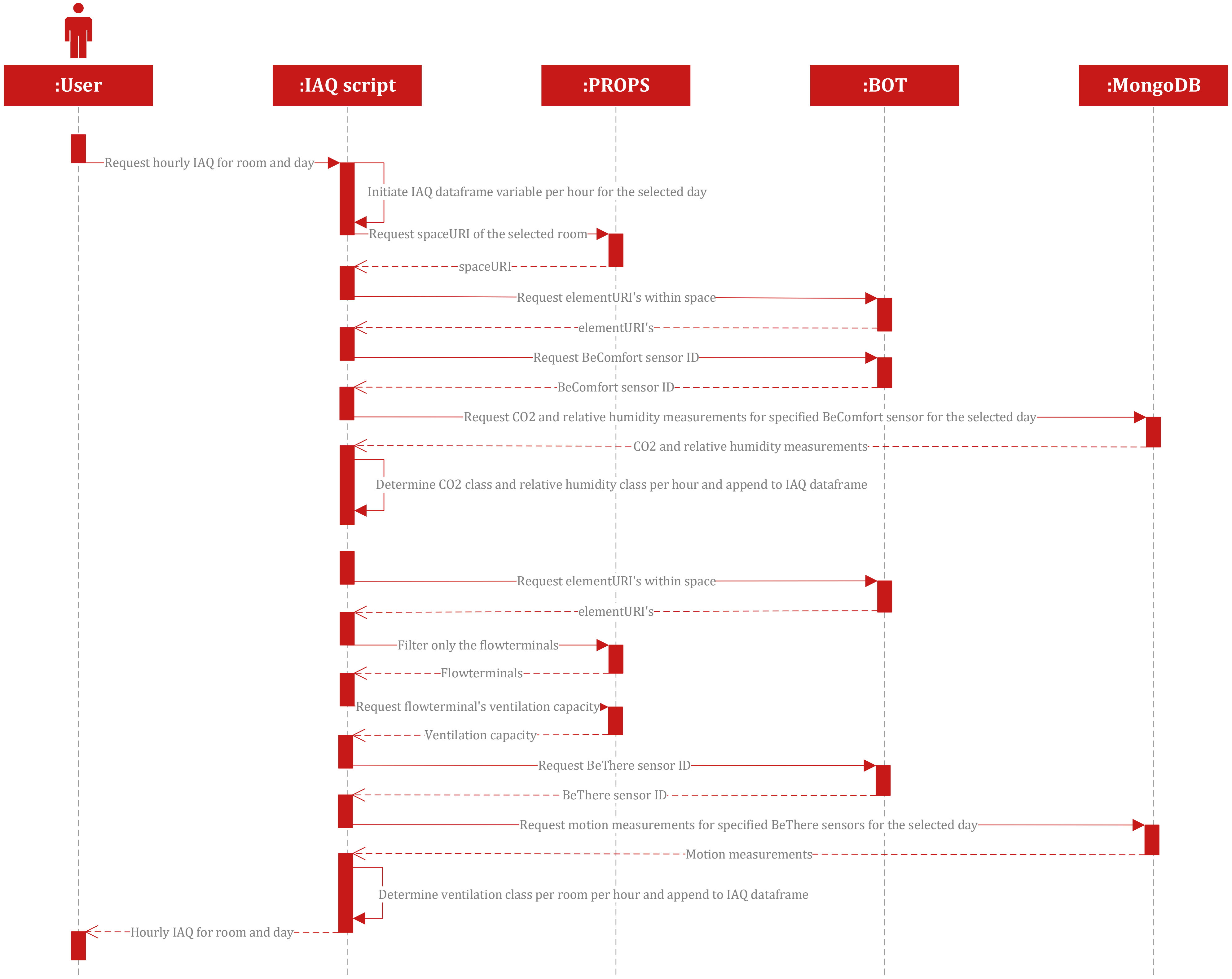
The thermal comfort calculation is more complex than the other criteria. When calculating thermal comfort, usually the model of Fanger (1973) is used. With the model of Fanger the Predicted Mean Vote (PMV) and the Predicted Percentage of Dissatisfied (PPD) are calculated. As shown in Table 1, the Arbo catalogue evaluates the thermal comfort based on the PPD. To calculate the PPD, this thesis shall make use of the spreadsheet developed by (Carlos, 2014). This calculation method follows the ASHRAE 55 guidelines using formulas that are widely supported throughout literature.

A4.7. Conclusion

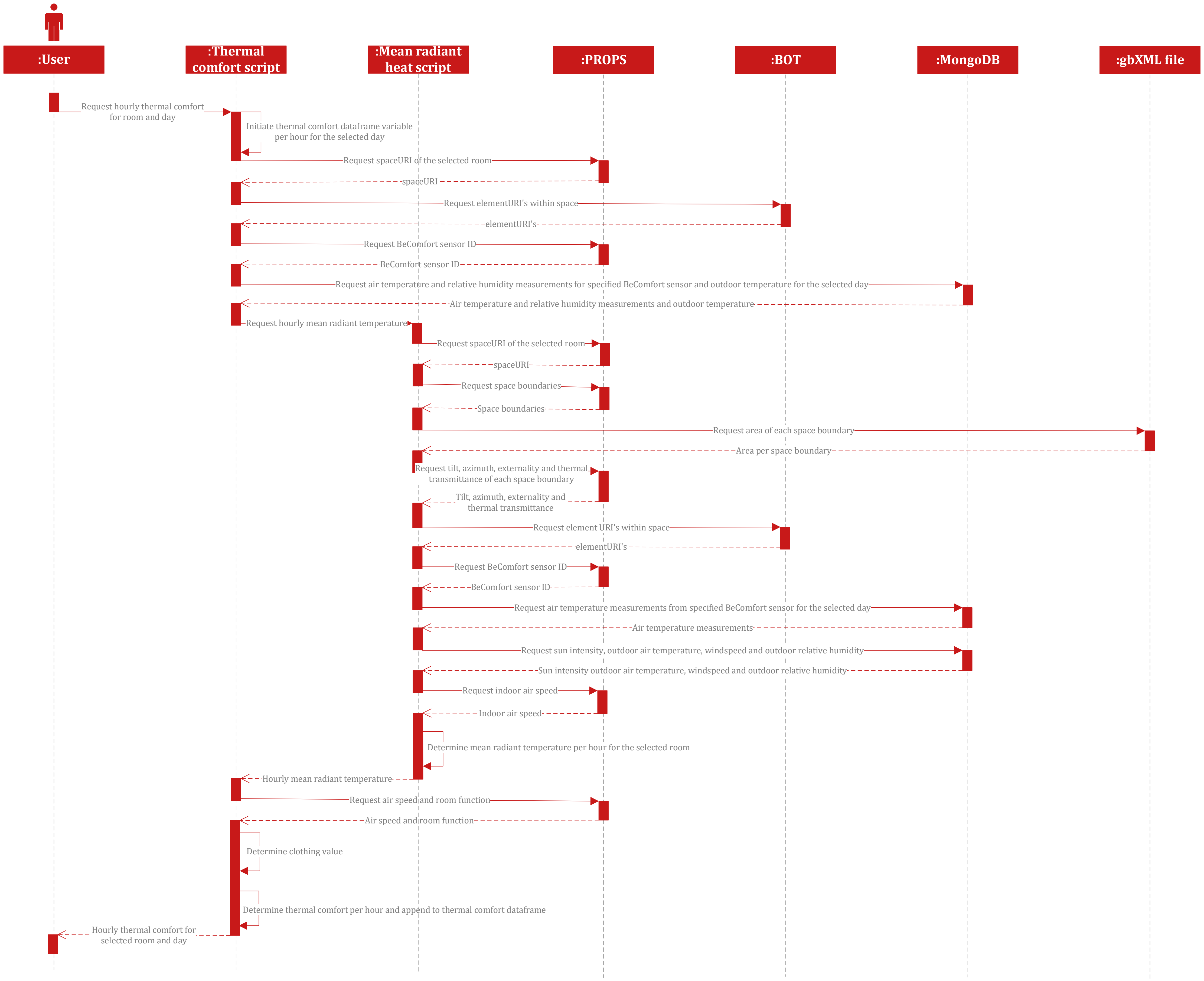
The research conducted in this appendix leads to a better understanding of the influences on the factors affecting the indoor environmental quality. Using this understanding, the accompanying criteria were searched for and quantified using the Arbowet and the WELL standard. Using these results, it is clearer what dataflows are necessary to objectively evaluate the monitored buildings on the necessity of potential improvements.

A5. UML Sequence diagrams of the IEQ analyses

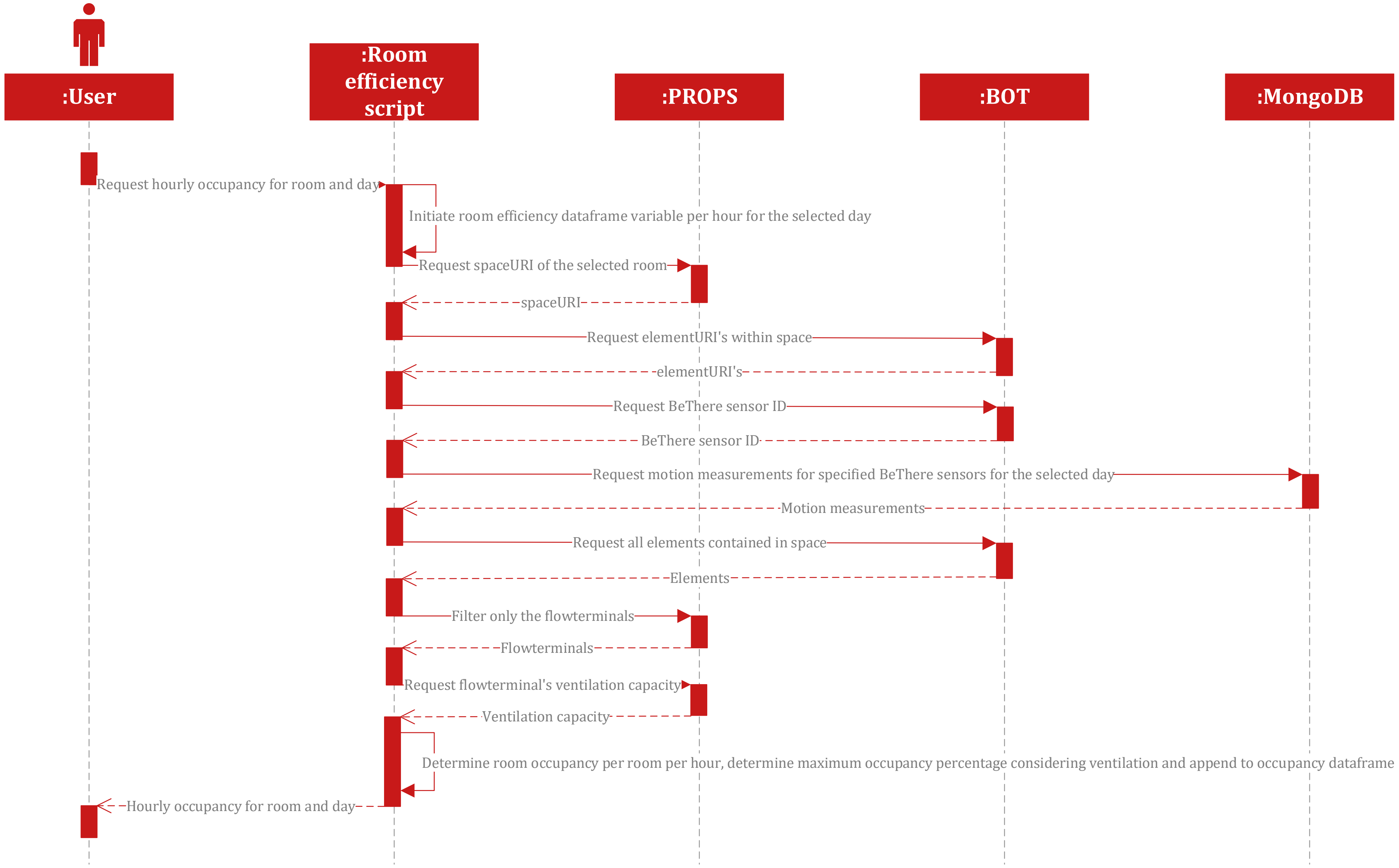
A5.1. UML Sequence diagram of determining the IAQ



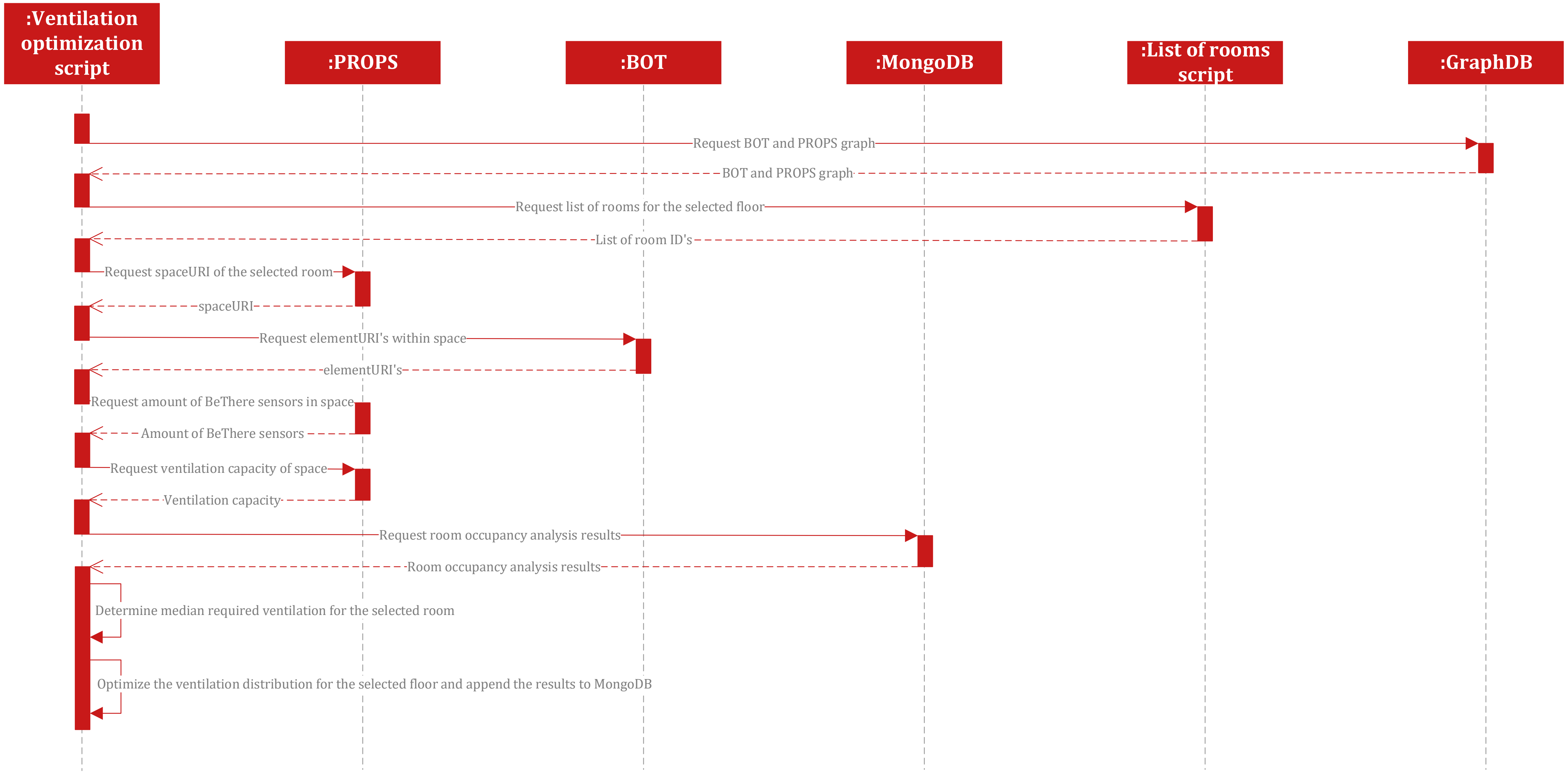
A5.2. UML Sequence diagram of determining the thermal comfort



A5.3. UML Sequence diagram of determining the room occupancy efficiency



A5.4. UML Sequence diagram of determining an optimized long-term ventilation capacity



A6. Mean radiant heat calculation

The mean radiant heat can be calculated using the indoor surface temperature of all bounding surfaces of a room, as shown in Figure 52. This leads to the following formula:

$$T_{mr,room} = \frac{\sum_{i=0} (T_{s,int,i} * A_{s,i})}{\sum_{i=0} (A_{s,i})} \tag{2}$$

$$\begin{aligned} \text{where } T_{mr} &= \text{mean radiant temperature of the room,} \\ T_{s,int,i} &= \text{indoor surface temperature of surface i,} \\ A_{s,i} &= \text{area of surface i} \end{aligned}$$

In this thesis it is assumed that the indoor surface temperatures of the internal walls are equal to the room air temperature as it gradually moves towards this temperature. Therefore, it is essential to first distinguish the internal walls from the external walls. When IFC files are converted to LBD graphs using the method described in chapter 4.1.2.2, surface elements get a Boolean property 'isExternal_simple'. Using this property an interior surface can be easily distinguished from an exterior surface.

Calculating the indoor surface temperature of the external walls is more complicated. The final equation to execute is as follows:

$$T_{s,int} = \frac{U_s * T_{s,out} + h_{air,ind} * T_{air,ind}}{U_s + h_{air,ind}} \tag{3}$$

$$\begin{aligned} \text{where } T_{s,int} &= \text{indoor surface temperature,} \\ U_s &= \text{surface thermal transmittance,} \\ T_{s,out} &= \text{outdoor surface temperature,} \\ T_{air,ind} &= \text{indoor air temperature,} \\ h_{air,ind} &= \text{indoor heat transfer coefficient} \end{aligned}$$

Prior to using the final equation, several other variables must be determined, namely the indoor heat transfer coefficient and the outdoor surface temperature. The indoor heat transfer coefficient can be determined using only the indoor air speed in the following formula:

$$h_{air,ind} = 10.45 - v_{air,ind} + 10 * v_{air,ind}^{0.5} \tag{4}$$

$$\begin{aligned} \text{where } h_{air,ind} &= \text{indoor heat transfer coefficient,} \\ v_{air,ind} &= \text{indoor air speed} \end{aligned}$$

The outdoor surface temperature is more difficult and frankly depends on a lot of variables. The calculation in this thesis is not fully correct, as the outdoor surface temperature calculation for asphalt has been used. To get this formula, a paper by Khan, Islam, & Tarefder (2019) has been used. Logically, asphalt heats up quicker than the average outdoor surface of a building, but accounting for the surface material type, colour and accompanying thermal properties is out of the scope of this thesis. Therefore, the following formula for asphalt surface temperature was used:

$T_{s,ext} = 26.081$ - 0.844*$v_{wind} + 0.479 * T_{air,out} - 0.187 * RH_{out} - 0.0173 * q_{total,s} + 0.0042254 * v_{wind} * T_{air,out} + 0.00565 * v_{wind} * RH_{out} + 0.0016 * v_{wind} * q_{total,s} + 0.00342 * T_{air,out} * RH_{out} + 0.000117 * T_{air,out} * q_{s,total} + 5.7029 * 10^{-5} * RH_{out} * q_{total,s} + 0.00425 * T^2_{air,out} + 1.9125 * 10^{-5} * q^2_{total,s}$(5)

$$\begin{aligned} \text{where } T_{s,ext} &= \text{outdoor surface temperature,} \\ v_{wind} &= \text{outdoor wind speed,} \\ T_{air,out} &= \text{outdoor air temperature,} \\ RH_{out} &= \text{outdoor relative humidity,} \\ q_{total,s} &= \text{total sun intensity on surface} \end{aligned}$$

To execute this formula, several datapoints are required from the meteorological data that has been collected in MongoDB. The sun intensity in the area must however be determined for the surface specifically. This can be done using the following formula:

$$q_{total,s} = q_{dir,s} + q_{dif,s} \tag{6}$$

$$\begin{aligned} \text{where } q_{total,s} &= \text{total sun intensity on surface,} \\ q_{dir,s} &= \text{direct sun intensity on surface,} \\ q_{dif,s} &= \text{diffuse sun intensity on surface} \\ a_s &= \text{surface azimuth} \end{aligned}$$

Both the direct and diffuse sun intensity on the specific surface is required. To calculate either of the two, several key variables are required. These can be determined using the following set of formula's:

$$\cos\theta = \sin h_{sun} * \cos t_s + \cos h_{sun} * \sin t_s * \cos(a_{sun} - a_s) \tag{7}$$

$$\begin{aligned} \text{where } \cos\theta &= \text{angle of the sun rays on the surface,} \\ h_{sun} &= \text{height of the sun in degrees,} \\ t_s &= \text{surface tilt,} \\ a_{sun} &= \text{sun azimuth,} \\ a_s &= \text{surface azimuth} \end{aligned}$$

$$a_{sun} = \frac{180 - \arcsin\left(\cos d_{sun} * \sin D_h\right.}{\cos h_{sun}} \tag{8}$$

$$\begin{aligned} \text{where } a_{sun} &= \text{sun azimuth,} \\ d_{sun} &= \text{sun declination,} \\ D_h &= \text{hour degree,} \\ h_{sun} &= \text{height of the sun in degrees} \end{aligned}$$

$$d_{sun} = 23.44 * \sin\left(360 * \frac{283 + d_y}{365}\right) \tag{9}$$

$$\begin{aligned} \text{where } d_{sun} &= \text{sun declination,} \\ d_y &= \text{day of the year} \end{aligned}$$

$$h_{sun} = \arcsin\left(\sin 52 * \sin d_{sun} - \cos 52 * \cos d_{sun} * \cos D_h\right. \tag{10}$$

where h_{sun} = height of the sun in degrees,
d_{sun} = sun declination,
D_h = hour degree

$$D_h = h * 15 \tag{11}$$

where D_h = hour degree,
h = hour of the day

Once these variables are known, the direct sun intensity can be calculated using this set of formula's:

$$q_{dir,s} = q_{dir,hor} * \cos t_s + q_{dir,ver} * \sin t_s \tag{12}$$

where $q_{dir,s}$ = direct sun intensity on surface,
$q_{dir,hor}$ = direct sun intensity on a horizontal surface,
$q_{dir,ver}$ = direct sun intensity on a vertical surface,
t_s = surface tilt

$$q_{dir,ver} = q_{sun} * \cos h_{sun} * \cos a_{sun} - a_s \tag{13}$$

where $q_{dir,ver}$ = direct sun intensity on a vertical surface,
q_{sun} = sun intensity,
h_{sun} = sun height in degrees,
a_{sun} = sun azimuth
a_s = surface azimuth

$$q_{dir,hor} = q_{sun} * \sin h_{sun} \tag{14}$$

where $q_{dir,hor}$ = direct sun intensity on a horizontal surface,
q_{sun} = sun intensity,
h_{sun} = sun height in degrees,

The diffuse sun intensity can be calculated similarly using the following formula's:

$$q_{dif,s} = q_{dif,hor} * \cos a_s + q_{dif,ver} * \sin a_s \tag{15}$$

where $q_{dif,s}$ = diffuse sun intensity on surface,
$q_{dif,hor}$ = diffuse sun intensity on a horizontal surface,
$q_{dif,ver}$ = diffuse sun intensity on a vertical surface,
a_s = surface azimuth

$$q_{dif,hor} = \frac{1}{3} * (1355 * (1 - 0.033 * \sin((d_y - \frac{93}{365}) * 360)) - q_{sun} * \sin h_{sun} \tag{16}$$

where $q_{dif,hor}$ = diffuse sun intensity on a horizontal surface,
d_y = day of the year,
q_{sun} = sun intensity,
h_{sun} = sun height in degrees

If $\cos\theta <= -0.3$:

$$q_{dif,ver} = -0.473 + 0.043 * \cos\theta * q_{dif,hor} \tag{17}$$

where $q_{dif,ver}$ = diffuse sun intensity on a vertical surface,
$q_{dif,hor}$ = diffuse sun intensity on a horizontal surface,
$\cos\theta$ = angle of the sun rays on the surface

If $\cos\theta > -0.3$:

$$q_{dif,ver} = 0.560 + 0.436 * \cos\theta - 0.35 * (\cos\theta)^2 * q_{dif,hor} \tag{18}$$

where $q_{dif,ver}$ = diffuse sun intensity on a vertical surface,
$q_{dif,hor}$ = diffuse sun intensity on a horizontal surface,
$\cos\theta$ = angle of the sun rays on the surface

Using all the elaborated formula's, the very first formula can be used to calculate the mean radiant temperature of the room ($T_{mr,room}$). This value is appended to a hourly dataframe that is exported from the mean radiant heat calculation script ('Function_Mean_Radiant_Heat_final.py'). This is then used as one of the inputs to determine the thermal comfort for the room.