TECHNICAL UNIVERSITY OF CRETE

SCHOOL OF ENVIRONMENTAL ENGENEERING

Postgraduate Thesis

Title:

“Energy Assessment of a Near Zero Energy Building (nZEB) Using Knowledge Graphs”

By Filippos Lygerakis

R.N.: 2020057530

Supervising Professor: Dionysia Kolokotsa

Supervising Committee:

1. Professor Dionysia Kolokotsa
2. Professor Theocharis Tsoutsos
3. Professor Mihalis Lazaridis

Chania 2021

Acknowledgements

I would like to express my gratitude to Professor Dionysia Kolokotsa and Dr. Nikolaos Kampelis for their very important and helpful suggestions during the preparation and progress of this research work. Their willingness to give their time has been very much appreciated and recognized.

Περίληψη

Ο κύριος στόχος αυτής της διατριβής είναι η δημιουργία ενός Γραφήματος Γνώσεως (ΓΓ) για ένα κτίριο σχεδόν μηδενικής ενεργειακής κατανάλωσης (ΜΕΚ), το οποίο θα περιλαμβάνει μια ποικιλία διαφορετικών αισθητήρων, και στη συνέχεια να εφαρμόσει ένα ερώτημα ανακάλυψης και ανάκτησης δεδομένων στο ΓΓ, προκειμένου να γίνει μια αξιολόγηση για την ενεργειακή αποδοτικότητα και την θερμική άνεση του κτιρίου. Το κτίριο ΜΕΚ που επιλέχθηκε ως περιπτωσιολογική μελέτη είναι το Leaf House, ένα κτίριο έξι διαμερισμάτων τελευταίας τεχνολογίας, που χτίστηκε από τον Όμιλο Loccioni. Η μεθοδολογία που ακολουθήθηκε περιλάμβανε πρώτα την έρευνα της τελευταίας τεχνολογίας σχετικά με τα ΓΓ σε κτίρια ΜΕΚ, τα εργαλεία μοντελοποίησης οικοδομικών πληροφοριών (ΜΟΠ), τα ΓΓ στην βιομηχανία της αρχιτεκτονικής, της μηχανικής και των κατασκευών (ΑΜΚ) και ορισμένες χρήσεις και εφαρμογές τους στη βιομηχανία δομημένου περιβάλλοντος. Στη συνέχεια, ακολουθεί η δημιουργία του ΓΓ τουLeaf House, καθώς και ένα ερώτημα ανακάλυψης και εξαγωγής δεδομένων ως παράδειγμα χρήσης του ΓΓ. Μετά από αυτό πραγματοποιείται αξιολόγηση ενεργειακής απόδοσης για το Leaf House, μαζί με μια εκτίμηση θερμικής άνεσης, με τη χρήση του Δείκτη Δυσφορίας (ΔΔ), και τέλος διεξάγωνται κάποια συμπέρασμα των ΓΓ στα κτίρια ΜΕΚ. Τα δεδομένα από τους διάφορους αισθητήρες και σημεία ρύθμισης που φιλοξενεί το κτίριο θα είναι διαθέσιμα από τη βάση δεδομένων MyLeaf, προκειμένου να τροφοδοτηθεί αυτό το ΓΓ. Το ΓΓ θα απεικονιστεί με το Brick Viewer και το Brick Studio, προκειμένου να παρουσιαστεί η σύνδεση μεταξύ των κόμβων αυτού του KG. Υπολογίστηκε ότι η ετήσια κατανάλωση ενέργειας ήταν 32,8MWh ή 69,8kWh/m^2, το 64% των οποίων προερχόταν από την κατανάλωση συστήματος HVAC και το 36% από την ηλεκτρική κατανάλωση των διαμερισμάτων. Επίσης, η συνολική καθαρή ετήσια κατανάλωση ηλεκτρικής ενέργειας το 2020 ήταν 9.801kWh και η συνολική καθαρή κανονικοποιημένη ετήσια κατανάλωση

ηλεκτρικής ενέργειας το 2020 ήταν 20,9kWh/m^2. Τα συμπεράσματα που προκύπτουν από την αξιολόγηση των αποτελεσμάτων για το ερώτημα ανακάλυψης και εξαγωγής δεδομένων, είναι μια απλή και γρήγορη διαδικασία που παρέχει στον χρήστη, που προέρχεται από οποιοδήποτε υπόβαθρο, αποτελέσματα ανάλυσης δεδομένων χωρίς τη χρήση ΜΟΠ και με πληροφορίες σχετικά με το κτίριο. Η εκτίμηση θερμικής άνεσης, η οποία εξέτασε τον Δείκτη Δυσφορίας και στα έξι διαμερίσματα, έδειξε ότι υπήρξε περίοδος το 2020,κατά τη θερινή περίοδο κυρίως, όπου κάτω από το 50% του πληθυσμού αισθάνθηκε δυσφορία. Οι μήνες που διήρκεσε η δυσφορία διέφεραν μεταξύ των διαμερισμάτων. Τα διαμερίσματα που βρίσκονται στον μεσαίο όροφο είχαν το μικρότερο χρονικό διάστημα με θερμική δυσφορία, σε σύγκριση με το ισόγειο και τον τελευταίο όροφο, οι οποίοι είχαν μεγαλύτερη χρονική περίοδο θερμικής δυσφορίας, με τον τελευταίο όροφο να έχει τη μεγαλύτερη διάρκεια. Επιπλέον, το ΓΓ σε ένα κτίριο ΜΕΚ παρέχει μια ιεραρχική αναπαράσταση των οντοτήτων του κτιρίου, η οποία χρησιμοποιείται ως βάση για πολλές ιδέες ερωτημάτων και εκμηδενίζει την ανομοιογένεια που προέρχεται από διαφορετικά μοντέλα αναπαράστασης κτιρίων, καθώς και δίνει στα δεδομένα περισσότερο νόημα και χρησιμότητα, λόγω των εισαγόμενων οντολογιών. Επίσης, είναι σημαντικό να αναφερθεί ότι το ίδιο ΓΓ διαχειρίζεται με έναν απλό τρόπο, πολλαπλές διαφορετικές βάσεις δεδομένων και αισθητήρες, σε συνδιασμό με την εύκολη πρόσβαση σε ενημερωμένες πληροφορίες που ο χρήστης μπορεί να ανακαλύψει και να εξαγάγει, σύμφωνα με τον τρέχων σκοπό. Συνεχίζοντας, το ΓΓ επιτρέπει στους χρήστες να εκτελούν ερωτήματα και να λαμβάνουν πληροφορίες σχετικά με τις οντότητες και τα δεδομένα του κτιρίου σε μια κοινή διαδικασία. Τέλος, το ΓΓ χρησιμοποιείται για σύνθετη συσχέτιση συστημάτων και ανάλυση δεδομένων χωρίς χρήση εργαλείων ΜΟΠ.

Abstract

The main goal of this thesis is to create a KG for a near Zero Energy Building (nZEB), which will include a variety of different sensors, and then apply a data discovering and retrieval query on the KG, in order to make an energy performance and thermal comfort assessment. The nZEB that was chosen as a case study is the Leaf House, a residential state of the art building of six apartments that was built by Loccioni Group. The methodology that was followed included first, the research of the state of the art about KGs in nZEB, BIM, AEC-KGs and some uses and applications of them in the Built Environment industry. Next, follows the creation of Leaf House KG, as well as a data discovering and extraction query as an example of KG's usage. After that an energy performance assessment for the Leaf House takes place, in addition to a thermal comfort assessment, using the Discomfort Index (DI), and lastly the conclusions of KGs in nZEB. The data from the different sensors and setpoints that the building is accommodating will be available by MyLeaf database, in order to feed this KG. The KG will be depicted with Brick Viewer and Brick Studio, in order to present the connection between the nodes of this KG. It was calculated that the annual energy consumption was 32.8MWh or 69.8kWh/m^2, 64% of which originating from HVAC system consumption and 36% from the apartments' consumption. What is more, the total net annual electrical energy consumption in 2020 was 9,801 kWh and total net normalized annual electrical energy consumption in 2020 was 20.9kWh/m^2. The conclusions that come up from the results' assessment for the data discovering and extraction query, are that is a simple and quick procedure that provides the user, coming from any background, with data analysis results without the use of a BIM and with information about the building. The thermal comfort assessment, which examined the Discomfort Index in all six apartments, showed that there was time in 2020, mainly in Summer time period, when under 50% of the population felt discomfort. This time period, which the discomfort lasted varied between the apartments. The apartments which are on middle storey had the shortest time period with thermal discomfort, in comparison to the bottom and top floor, which had longer time period of thermal discomfort, with top floor having the longest. What is more, the KG in a nZEB provides a hierarchical representation of the building's entities, which is used as a base for many querying ideas and it nullifies the heterogeneity coming from different building representation models, as well as it gives

to data more meaning and usefulness, due to the imported ontologies. Furthermore, it is important to be mentioned that the same KG manages in a single simple way, multiple different databases and sensors, in addition to giving easy access to up-to-date information that the user can discover and extract, according to the current agenda. Continuing, the KG allows users to perform queries and obtain information about the building's entities and data in a common procedure. At last, the KG is used for complex systems correlation and data analysis without BIM usage.

Contents

1. Introduction

1.1 Semantic Knowledge in Built Environment & AEC Industry

1.1.1 Knowledge Graphs in Built Environment

According to United Nations Environment Programme (UNEP), about 30-40% of annual energy worldwide is consumed by the building sector and what is important to note is that many residents are not satisfied with the buildings that they are tenants in, still in high performing buildings.[1]–[3]That is what escalated the demand of smart buildings, which provide the application of new technologies, like sensing technologies in a large scale, data analytics and advanced controls.[4] A smart building can be comprised of three main categories a)the components, which are the technical building equipment, the energy production equipment, as well as their sensors, b) the functions, which are enabled by the components and c) the outcomes that are produced from the building's functions.[5]

In EU, according to Energy Performance of Buildings Directive recast (EPBD recast, Directive 2010/31/EC), all new public buildings have to be nearly Zero Energy Buildings (nZEBs) by December 31, 2018 and all new buildings by December 31, 2020.[6] The nZEB is introduced as a general concept that also incorporates the autonomous buildings that are not connected to energy grids. What is more, EPBD presents nZEB as a "building that has a very good energy performance. The nearly zero energy or very low amount of energy required should be supplied to a very significant extent by energy from renewable sources, including energy from renewable sources produced on-site or nearby".[6]

It has been made clear that zero energy and zero carbon buildings have the ability to alleviate the impacts of climate change and reduce the damage all over urban and rural areas, created by the microclimate.[7] Although, ZEBs on a neighborhood level have not been researched thoroughly, due to need of a proper representation that these reshearches require, as well as the lack of specific hypotheses that these neighborhoods would follow, like the function units, the main source of renewable energy, the morphology of the neighborhood, the space occupied per resident that defers from place to place, the type of climate, the building materials and other.[7] The definition of Nearly Zero Energy Neighborhood that was given by Sornes et al. in 2014, states that "a nZEN is a cluster of residential units where the overall energy demand is low and is partly met by renewable energy self-produced within the neighborhood".[8] The physical boundary of a nZEN includes the sites of renewable energy production besides the buildings themselves.[9]

Due to the exponential growth of renewable energy resources and multi energy systems in residential areas and companies, there has been a deviation from power grids to decentralized microgrids supporting the growth of future smart grids.[10]–[12] It is noted that due to the increasing number of smart meters, there is a growth in energy related resources in microgrids and that is the reason why a sustainable and strong energy management is needed.[12]–[14] There lies the reason why smart grids (SGs) are of major significance, when referring to renewable energy resources integration in a controlled grid, in order to support the power supply computed by smart communications, sensors and measurement devices.[15] SGs are meant to be highly significant developments in real time systems, due to the fact that they include both generators and consumers activities, with their main goal being to provide electricity in a sustainable, economic and secure way.[16]While achieving these goals, SGs produce massive data about power generation, power transformation and transmission, distribution network, and electricity consumption, which are merged to create electric power big data that are used to safely and stably assist the power grid, business services and decision making procedures.[17]–[19]On the other hand, data processing analysis and knowledge mining issues emerge too, leading to the use of semantic knowledge platform usage that cover the needs of electric power data management, power consumption oriented services and SGs business development.[20] This semantic knowledge platform should aim to high-end SGs scenario applications and business services development and accomplish that by combining heterogeneous information from different sources and different SGs scenarios.[20] A way to produce a successful solution for this target, is the usage of knowledge graphs (KGs).

1.1.2 Knowledge Graphs in Architecture, Engineering & Construction (AEC) Industry and Building Information Models (BIM)

The Architecture, Engineering & Construction (AEC) industry shares a lot of fields, with different amount and kind of information shared between them, in order to achieve a common project goal. These different fields have been utilizing BIMs to manage and exchange information.

Building Information Models (BIM) are digital models of a built structure, using

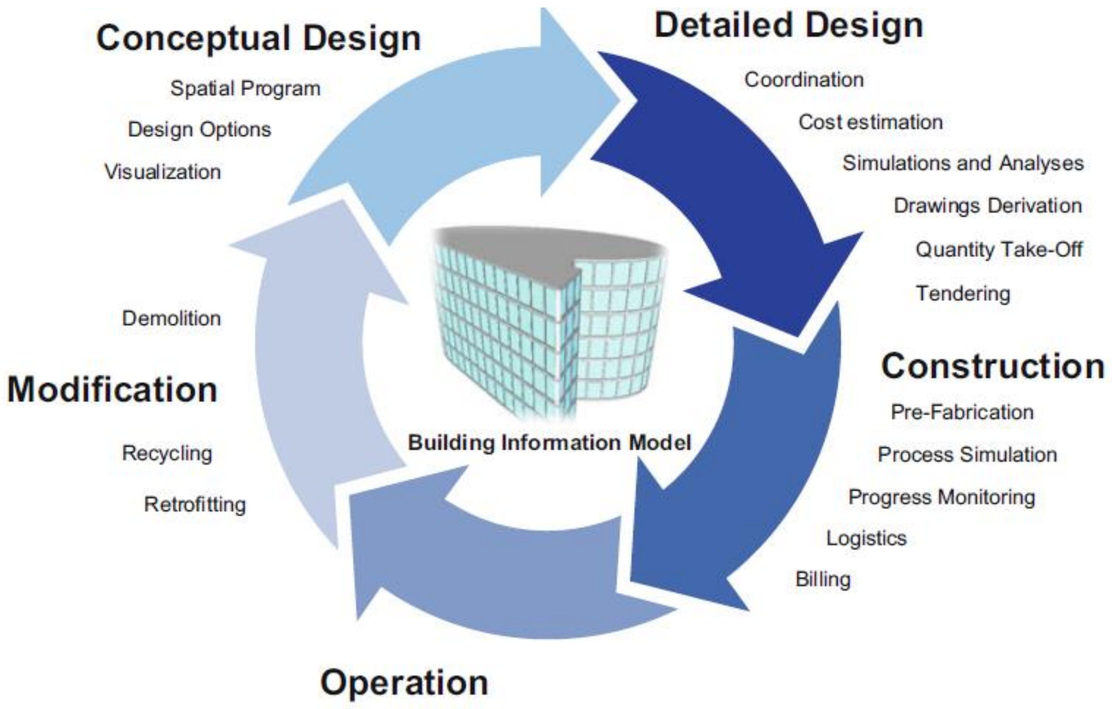


Figure 1: The Concept of BIM[24]

digital modeling and associated technologies for data collection, which was first introduced in 1970s and in the last decades has greatly influenced the AEC industry.[21]–[24]In Figure 2 the concept of BIM is being depicted with the main principle being the continuous use of digital information throughout the life cycle of a built structure. What is more, the use of BIM has made a more comfortable environment of data exchange from different fields of a project, surpassing the document-centric method that was previously used. Another use of BIM is that it can also be used to integrate domain knowledge and specific methodologies for intelligent applications based on automation.[25]Many data platforms have been created that are based on BIM systems, which helps AEC industry with accessing these more organized data.[26]Furthermore, studies have shown that BIM and the data they produce can be of greater significance when combined with semantic networks to help with the various data sources.[27] Even more, it is suggested that BIM can be used in ontology-based data management and sharing, as well as combination of different knowledge domains and reasoning, to conclude in a KG.[28]

However, even if BIM have been introduced and used for a while now, there is still a lot of data usage in document form, whereby data becomes fragmented and not easily accessed.[29], [30]Also, design changes is a common and rapid situation in AEC industry phases and keeping up with documented information only creates a confusion and an unclear goal, ending up with issues of misinformation or out-of-date assumptions resulting in errors through the process, even when using BIM workflows.[29], [31]–[33] A more structured way of data is being achieved through BIM, however the document based climate in this way of work still exist, holding back issues and errors based on that.[31]

The use of semantic web has been proposed and researched to solve document based ways of working and achieve better interaction inside the industry.[34] Ontologies are the core of semantic web design, which are characterized as formal, due to a) their ability to be read by machines, b) their explicit nature and interoperability, based on the explicit definition of the concepts and the constraints used.[35] AEC-KG is a terminology that is formed in an attempt to research ontologies, which satisfy data on the web utilizing vocabularies that already exist.[31], [36]

1.2 Knowledge Graphs (KGs)

1.2.1 Definition

Knowledge Graphs are still evolving today, yet many different attempts to provide thorough and concise definitions.[37], [38] According to a commonly used definition, a knowledge graph acquires and integrates information into an ontology and applies a reasoner to derive new knowledge.[37] This definition was given after a research was conducted, in order to come up with a working definition based on examples. As referred by Ehrlinger and Wolfram, considering that there are many diverse applications, a KG is more likely to be more similar to an abstract framework than to a mathematical structure.[37] Another approach is that a knowledge graph mainly describes real world entities and their interrelations, organized in a graph. It does that by, defining possible classes and relations of entities in a schema, while allowing for other potentially interrelating arbitrary entities with each other, covering various topical domains.[39] What is more, it is also stated recently that a knowledge graph can be viewed as a graph of data intended to accumulate and convey knowledge of the real world, whose nodes represent entities of interest and whose edges represent relations between these entities.[40]Knowledge graphs were first introduced in 1973, but it was too early to be used in a useful way, until 2012, when Google announced its KG, which was the starting point for many other companies to introduce their own.[41],[42] Many applications have been made since then and many papers have been published, all aiming to the core idea that is to represent data using graphs, even in a way to represent knowledge.[43] Graphs, in contrary to a relational model or NoSQL approaches, are more coherent and direct, using edges to represent the relations between entities and that applies for various domains.[40],[44] One more aspect of a graph is that it provides the creator with the ability to delay the definition of its schema, making it more flexible to evolve and obtain more incomplete knowledge, concluding to a continuously updated database schema or serve under an organization or a community as an ever-evolving shared form of knowledge.[43],[45]

1.2.2 Structure

Data Graphs

One of the first principles of a KG is the graph abstraction to data, making a primary data graph, which is represented by data models and processed by query languages. Modelling a graph differs in every situation, although there are some graph data models that can be adopted and customized. For example, a directed edge-labelled graph is compiled from a set of nodes and a set of directed labelled edges that connect these nodes.[46], [47] In KGs nodes stand for entities and edges for their binary relations between them. This way of modelling a graph is more appropriate when adding new sources of data. Resource Description framework (RDF) is a model based on directed edge-labelled graphs, which uses a variety of nodes, most importantly the Internationalized Resource Identifiers (IRIs), which give access to entities through Web, literals, which represent strings and other datatype values and blank nodes, which are anonymous nodes that are not assigned an identifier.[48],[49] Another type of graph is a heterogeneous graph, where each node and edge is given one type, creating a part of the graph model only from nodes and benefits from dividing nodes based on their type.[50], [51] The property graph model is another type that in contrary to directed edge-labelled graph, it is more flexible when modelling.[52]

When querying a graph, there are many languages that has been introduced, including SPARQL for RDF graphs.[53] Graph patterns are stationed at the center of a query language, which use the same models as the data graph that is being queried.[52] What is more, graph patterns also add variables as terms, which are divided into constants.[52] Next, mappings are being generated from the variables to constants in data graph in order for the graph pattern to be included in the data graph. Furthermore, due to the fact that a graph pattern exports a table of results and due to the need of relational algebra to work with these tables, more complex queries are being created.[52] Another aspect of the graph query languages is that navigational graph patterns add path expressions in queries, allowing matching arbitrary length paths between two nodes, which is expressed as a regular path and are used in graph patterns to express navigational graph patterns .[52]

Schema, Identity and Context

A KG can be identified as a data graph enhanced with representations of schema, identity, context, ontologies and rules.[40]

Schemata are used to mark structure and semantics that the KG will be based on, although it is mentioned that its definition can be delayed for after the KG's configuration.[40] One type of graph schemata is the semantic, which is used as a vocabulary for understanding terms used in a KG, while using these terms for reasoning the KG.[40] RDF Schema (RDFS) is an example of a semantic schema, which introduces subclasses, subproperties, domains and ranges for the classes and properties in an RDF graph.[54] Many more details and content about the semantics of KG terms can be given from Web Ontology Language (OWL) standard for RDF graphs.[55] In contrary of semantic schemata, validating schemata are validating existing graph data, using shapes, which are responsible for targeting set of nodes in a data graph and identifying their constraints.[56],[57] Both types of schemata are in need of a domain expert to identify definitions and constraints, although in a data graph, latent structures can be exported as an emergent schema, which uses graph as frameworks to separate quotient groups of nodes, while maintaining some structural properties of the graph.[58], [59]

In order to clarify some nodes that collide, globally unique identifiers and identity links are used to prevent naming clashes, when connected with external data and sources.[40] Long lasting persistent identifiers (PIDs) are an example of usage to uniquely identify an entity, as well as global Web identifiers, recommended by the RDF data model, using Internationalised Resource Identifiers (IRIs).[48], [60] One way to identify an entity is to connect it with a uniquely identifying information in the graph and a second way is to state that one entity has the same identity links with an external source, thus using owl:sameAs coreference, when talking in OWL standards.[40] It is practically common for data models to permit datatype values as nodes, like RDF, which uses XML Schema Datatypes (XSD).[61] Existential nodes are used to cover places that are needed to complete the connections, but are not yet identified, supported by RDF as blank nodes.[48]

When referring to context, it is meant to the truth behind the data used, which might be about individual nodes or edges or sets of edges.[40], [62]When representing context, one method is to accept it as data or another method is to alter an edge into a node, to add more context. What is more, using reification makes it possible to define edges about edges or use hierarchy representations for modelling context, as an alternative. Annotations is an automated mechanism for reasoning context, which allows mathematical definitions of a contextual domain and key operations possible within that domain that can then be applied automatically.[40]

Deductive Knowledge

It is necessary to have knowledge of the meaning of the terms that are used in order to apply entailment. This is succeeded using ontologies, which provide a formal depiction of the meaning of the terms. The most recognized ontology is the Web Ontology Language (OWL), recommended by the W3C and compatible with RDF graphs.[40], [55] In the process of interpretation, the data graph is changed to domain graph, which includes real world entities with real world connections and is involving connecting the nodes and edges of the data graph with the ones of the domain graph, thus following the same model as the data graph.[40] Linking particular patterns in the data graph with semantic conditions, which specify the interpretations that will be satisfied, concludes into the features of an ontology language.[40]

These features result in entailments and each axiom that is introduced from an ontology, brings up some conditions on the interpretation of the graph that satisfies it, which are also called graph models. One graph entails another only and only if the first is also a model of the last one or alternatively the former graph entails the latter.[40]In this context, there is not an algorithm which can decide the correct true/false answer to the question of which graph entails the other.[63] Even though it is possible to apply reasoning algorithms for ontologies that in one situation it will halt on any input ontology, there might be a risk that will end up losing some entailments, returning instead of true. Another situation is to always halt false with correct answer, only receiving input ontologies with specific features and the last situation is to only reply with correct answers for any input ontology, risking never halting on some inputs.[40]

A method that gives easy access to deductive knowledge using inference rules, is encoding if-then-style consequences, which rules are comprised of the body (if) and the head (then). They are both graph patterns and can be used to obtain entailments under ontological conditions.[40]

Description Logics (DLs), which are used to formalize the meaning of frames and semantic networks, form a family of logics, which due to the fact that semantic networks are an early version of KGs and that they have had a great impact on Web Ontology Language, are considered of great significance for the formalization of KGs.[40], [64], [65] When they were first introduced, DLs were parts of First Order Logic (FOL), which allows reasoning tasks.[66]What is more, there are differences between expressive power and computational complexity of reasoning in different DLs. Furthermore, DLs are bound to three types of elements a) individuals, b) classes and c) properties and use claims, also known as axioms about these elements.[40]There are also similarities between DLs an OWL, due to the fact that OWL was greatly influenced by the DLs.

Inductive Knowledge

In contrary to deductive knowledge, which follows specific logical consequences, inductive knowledge is based on generalized patterns from input observations, which are used to come up with new but vague predictions. An overview of popular inductive techniques are shown in Figure 1.

Analytics are based on discovering, interpreting, and communicating important patterns innate to data collections and so graph analytics are the use of analytical processes to graph data.[40] Graphs are leaning into specific types of analytics that end up in a deduction, where nodes and edges are based on the topology of the graph and are gaining their techniques from graph theory and network analytics.[67]

Machine learning, which have made a significant amount of progress in the past few years, can be used to directly refine a KG.[68] The target of KG embedding methods it to condense the graph in a continuous, low-dimensional vector space, where machine learning tasks can be embedded making it possible for embeddings to execute some low

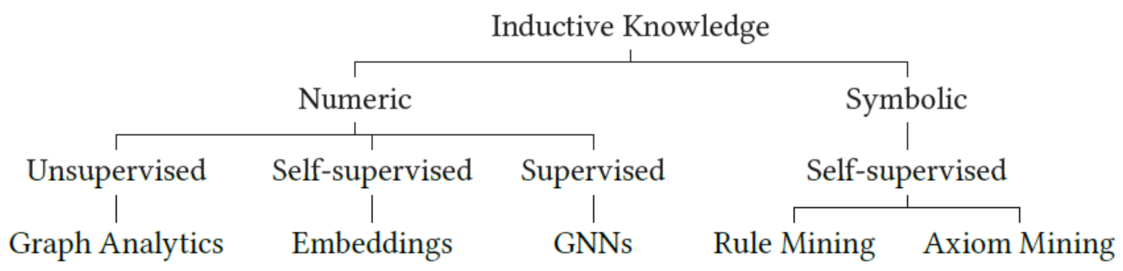


Figure 2: Conceptual overview of popular inductive techniques for knowledge[40]

level tasks around nodes and edges.[40]

Another method is to compile a custom machine learning model modified for graph structured data, with the majority of them depending on artificial neural networks.[69] A graph neural network (GNN) compiles a neural network depending on the topology of a data graph, being capable to even replace algorithms.[70], [71],[72]

A different method is to use symbolic learning to gain knowledge about hypotheses in a symbolic language that clarifies some positive and negative edges, which are automatically produced from the KG and then use these hypotheses as interpretable models, able for additionally reasoning.[40]

Creation & Enrichment

KGs are made to have a prime core, but they rely on enrichment from external sources. One method of creation and enrichment of a KG is through human editors, a costly but successful one, based on previous works.[73]–[75]Another method is extracting information from text collections, although is not an easy task to be accurate at the extraction. Otherwise, information can be extracted from markup documents, which are the foundation of the Web and which are being stripped from their markers, leaving plain text to become the source.[76], [77],[78] Structured formats, primary tables and tree-structured sources are easily found in the Web and in organizations, although the method for extracting information from them, relies on mapping the sources to a KG, which includes the creation of the mapping from the source to the KG and the use of the mapping to materialize the data as a graph.[40]

There are methods to create schemata using these sources of information, including human knowledge, concluding to the creation of ontologies, using either ontology engineering methodologies, and ontology learning.[40] Ontology engineering is about constructing and utilizing methodologies for creating ontologies, having as a main goal the least effort needed to achieve that and ontology requirements, which are needed for the clarification of the tasks of the KG, relying on its schema.[40]

Quality Assessment

Having created and enriched the KG from data sources, it is important check its quality, which is actually referring to its fitness for purpose.

The accuracy of a KG is based on how proper real life phenomena are being represented by the KG and can be further subdivided into syntactic accuracy, which checks the data based on the grammatical rules of graph model and the domain, semantic accuracy, which checks how similar are the data values to the real life phenomena and timeliness, which is how up to date is the KG.[40]

Next, when talking about the coverage in a KG, it is referring in the inclusion of all domain relevant elements, which would conclude to incomplete query results or entailments, biased models, etc. and includes completeness, which requires that there is not any missing information from datasets and representativeness, which is a dimension that assesses biases for what is included or not in the KG.[40], [79]

Following, coherency checks how well the KG complies with the semantics and constraints of schema and can be further analyzed into consistency, which checks the compliance of the KG with the logical entailment that was chosen and validity, which checks for constraint violations in the KG.[40]

Last, succinctness checks for relevant content in the KG that includes conciseness, which is about not including schema or data elements that are irrelevant to the domain, representational-conciseness, which is about how exact is the representation of the context in the KG and understandability, which is about how easily can data be understood by human users.[40], [80], [81]

Refinement

Refinement methods are used to automatically complete and correct the KG, which is known for its incompleteness, with knowledge graph completion and knowledge graph correction, respectively.[39],[82]Completion fills the missing edges that are not given neither entailed by the KG, usually with methods based on link prediction in the field of

Statistical Relational Learning.[40], [83]On the contrary to completion, correction finds and removes incorrect edges in the KG, through the methods of fact validation, which applies a possibility fact score to an edge with the help of the external sources and inconsistency repairs, which uses ontological axioms to repair inconsistencies in the KG.[40], [47]

1.3 Main Goal & Structure

The main goal of this thesis is to create a KG for the building of Leaf House, which will include many different sensors, and then apply a data discovering and retrieval query on the KG, in order to make a data analysis and a thermal comfort assessment. This will bring up some conclusions about the use of KG for a building as well as about the example of using a building KG to access data in it and assess them.

In Section 2, the state of the art about knowledge graphs in buildings is being presented and in specific presents developments with respect to BIMSO/BIMDO, BOT, OPM, ifcOWL, simpleBIM and BRICK ontologies. Next, in Section 3, the methodology that was followed is being introduced, as well as the case study's building. In the same Section, the KG creation and the data discovering and retrieval query are being presented. Following, in Section 4, the results are being analyzed and some observations are made known. What is more, in the same section, the data analysis and thermal comfort assessment take place. Lastly, in Section 5, the conclusions of this thesis are presented as well as some future work ideas.

2. State of the Art in Ontologies on Built Environment

2.1 BIM Shared Ontology (BIMSO) & BIM Design Ontology (BIMDO)

BIM Shared Ontology (BIMSO) is meant to be used by different building domains as a foundation ontology to create domain ontologies and was introduced by Mehrdad Niknama and Saeed Karshenas in 2017.[84] Its scope is focused on sharing information between different AEC-FM domains and to give answers about information connected with the elements, levels, spaces, and construction phases of a building. It uses RDF/OWL language and it is intended to be used for different building lifecycle domains in order to give access to an easy AEC-FM interface for accessing a semantic bridge of information exchange. BIMSO used UNIFORMAT II classification system to organize its elements,

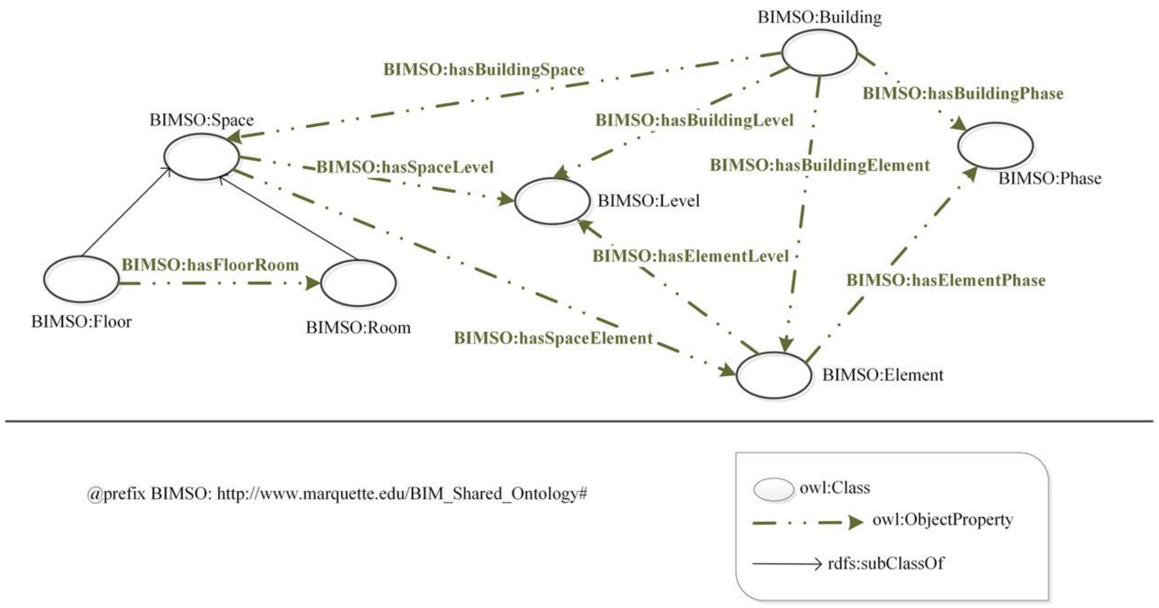


Figure 3: General view of BIMSO[84]

which consists of four main levels, level one for major group element types, level two for group element types, level three for subgroup element types and level four for individual element types.[85] The general view of BIMSO is shown in Figure 3.What is more, Uniform Resource Identifiers (URIs) are used to identify a property in an ontology and the prefix BIMSO was assigned to the URIs.[86]

BIM Design Ontology (BIMDO) is aiming to create a model that is able to design properties of building elements, like identities, sizes, and material properties and also build elements connections like intersects and hosts.[84] Like BIMSO, it is also using RDF/OWL language and targets AEC-FM domains, intending to create BIM design knowledge bases. The prefix BIMDO is given to its URIs and QUDT, an ontology for units' measurement, is added.[87] In Figure 4 shows a schematic view of a case study conducted from the same authors using BIMSO/ BIMDO and it is an example of how phases, levels, floors, rooms, and elements of the examined building are defined using the BIMSO ontology vocabulary.

They conclude to the fact that different building domains returned the same domain-specific element properties, due to the fact that they were created from the same ontology, using the same querying,.[84]

2.2 Building Topology Ontology (BOT)

Building Topology Ontology (BOT) is introduced by Rasmussen in 2016 and is a simple ontology based on the topology of a building and the physical and conceptual

objects in it as well as the connections between them.[88] In Figure 5 the simple BOT is being depicted. For this to happen, BOT sets some rules that subdivides the building into storeys and spaces. Spaces are bound by building elements and spaces can contain building elements. It is an ontology that focuses on the building as a structure and does not cover the needs of the whole AEC domain, but can be used as a central ontology to link others.[88] In Figure 6 an example of an extended BOT dataset with geographical and appliance data is depicted, combining two more ontologies to central BOT.

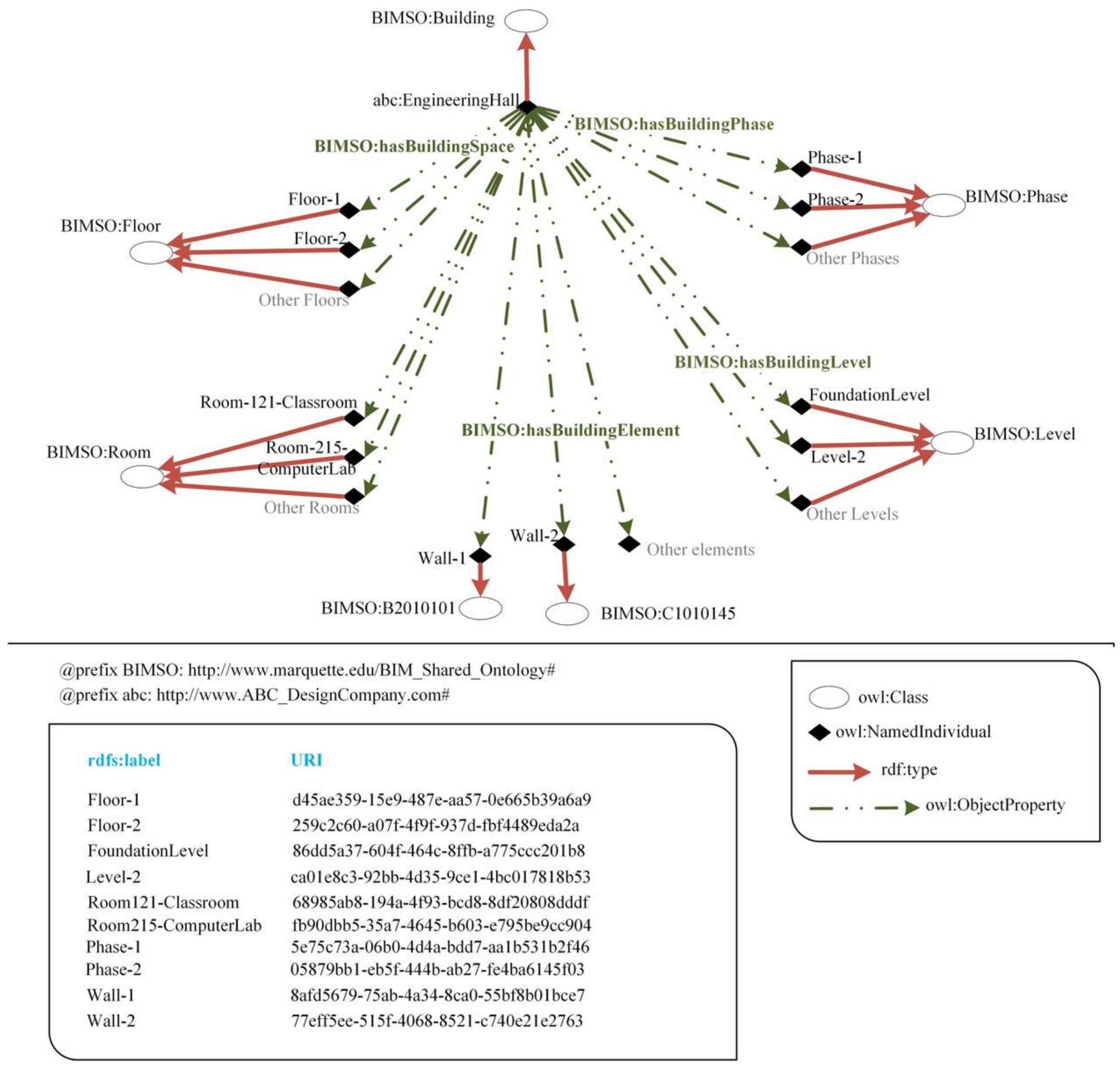
Concluding BOT is a simple base ontology for building structure that can be easily linked with other ontologies to add more information, making the procedure more customizable and malleable in different situations of the AEC industry.

2.3 Ontology for Property Management (OPM)

Ontology for Property Management (OPM) offers the vocabulary for modelling complex entities in a design environment and was proposed by Rasmussen at 2018.[89],[90]These entities are defined as complex because they might alter through time, their reliability might be on assumptions and might be based on other entities that might also alter, causing an effect on them. OPM is using SEAS, schema.org and PROV-O ontologies as extensions and can work alongside with BOT, PROPS and PRODUCT ontologies of the W3C LBD Community Group.[91]

To put OPM into test, there was a case study, which calculated the heating demand in a building through the ontology, which is also shown in Figure 6.[89] So an OPM-REST application on the AEC-KG was developed as a generic approach.

Concluding, OPM proved AEC software can utilize semantic web on construction applications and is so the answer to the question they gave in the beginning of the article: "How can we effectively store design data in a structured way, allowing interrelated data to maintain their relations intact as the project progresses, without losing the history of properties' progression?".[89]



@prefix BIMSO: http://www.marquette.edu/BIM_Shared_Ontology#
@prefix abc: http://www.ABC_DesignCompany.com#

rdfs:label	URI
Floor-1	d45ae359-15e9-487e-aa57-0e665b39a6a9
Floor-2	259c2c60-a07f-4f9f-937d-fbf4489eda2a
FoundationLevel	86dd5a37-604f-464c-8ffb-a775ccc201b8
Level-2	ca01e8c3-92bb-4d35-9ce1-4bc017818b53
Room121-Classroom	68985ab8-194a-4f93-bcd8-8df20808dddf
Room215-ComputerLab	fb90dbb5-35a7-4645-b603-e795be9cc904
Phase-1	5e75c73a-06b0-4d4a-bdd7-aa1b531b2f46
Phase-2	05879bb1-eb5f-444b-ab27-fe4ba6145f03
Wall-1	8afd5679-75ab-4a34-8ca0-55bf8b01bce7
Wall-2	77eff5ee-515f-4068-8521-c740e21e2763

Figure 4: A schematic view of part of the Engineering Hall case study knowledge base.[84]

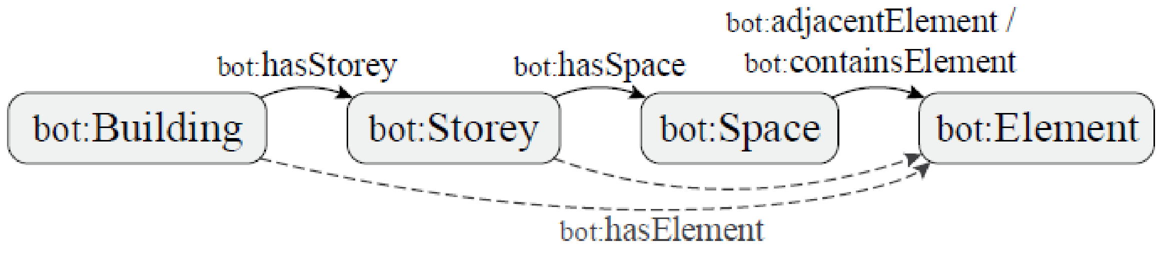


Figure 5: Simple Building Topology Ontology[88]

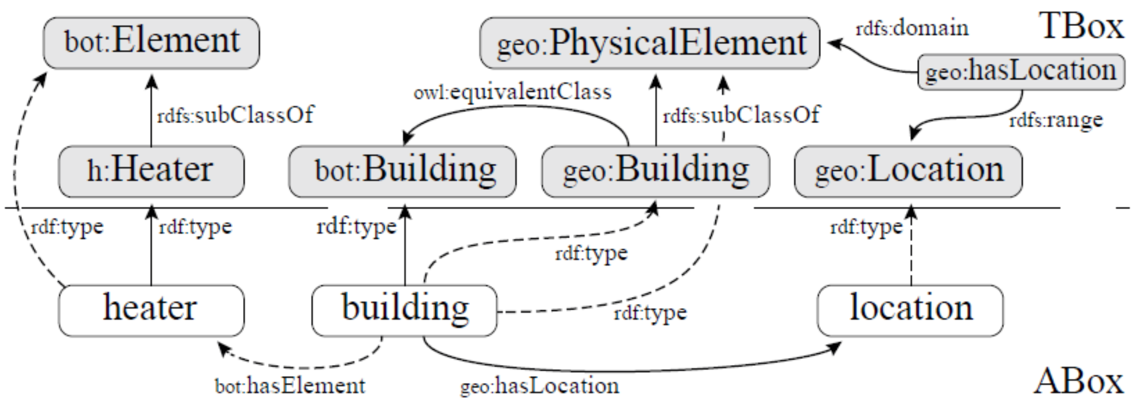


Figure 6: Extending a BOT-compliant dataset with geographical and appliance data[88]

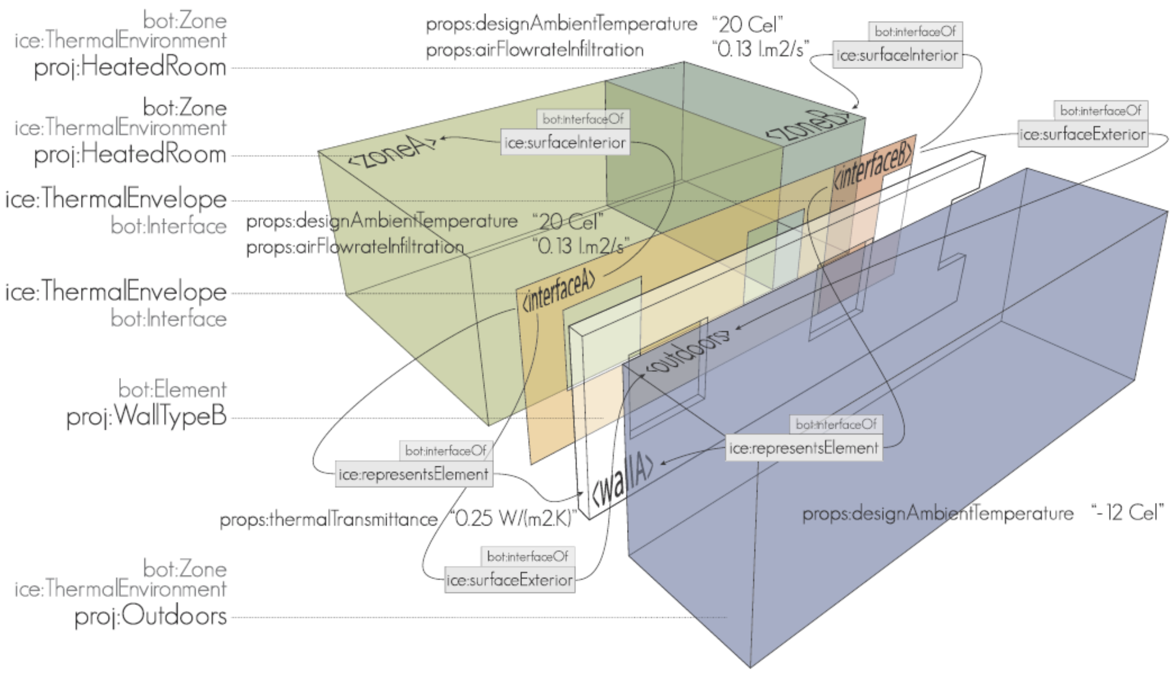


Figure 7: Visualization of the AEC-KG model for heat loss calculation case study[89]

The case study showed that OPM is a different way of working with building data and also paved a way to access and utilize BIM models as well as the data they produce, exchanging any of them between stakeholders using a the same tool.[89]

2.4 ifcOWL & simpleBIM

ifcOWL is a complex ontology language, which is a translation from IFC schema through EXPRESS data modelling language into an OWL representation and the

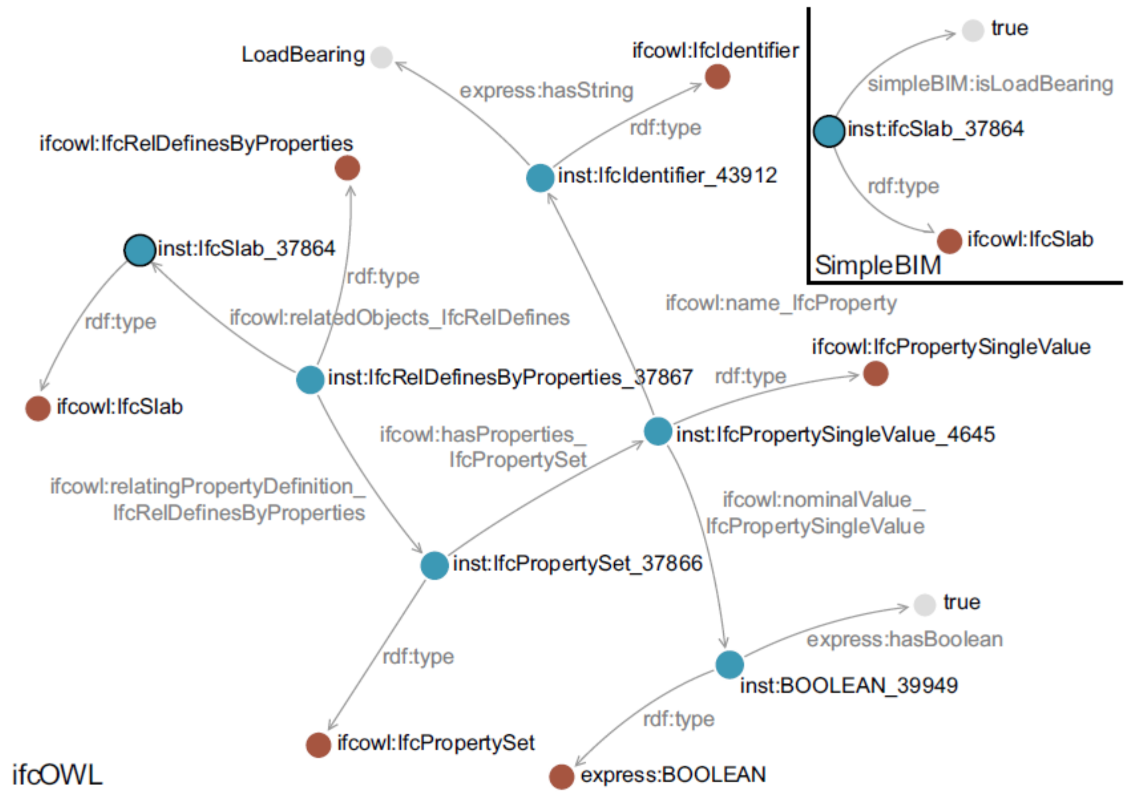


Figure 8: Visual complexity comparison of representing property assignment using ifcOWL and simpleBIM[31]

complexity is shown as a property set assigns the properties using relational nodes, two intermediate nodes are needed to insert the name and the value of the property and EXPRESS datatype is used to express literals.[31],[92]SimpleBIM is an attempt to simplify this ifcOWL as it uses the most straightforward approach to do that and in Figure 8 the difference between them is shown, as they represent the same entities. SimpleBIM also uses Turtle serialization format for RDF data models.[93]

There are three levels of complexity in ifcOWL, based on Rasmussen, who described L1 to be equal to simpleBIM, L2 to be used by BIMDO, which consider the entities as an object and so QUDT and PROVO-O can be attached and L3 to be used by OPM, which is based on SEAS and allows the property to change through time.[31], [87], [90], [94],[91]

2.5 Brick

Brick's main goal is concentrated on metadata and data points from building advancement and needs, which are based on end use applications and consists of a main ontology that establishes the core concepts and the connections between them, as well as a typology that enlarges the building's concepts.[95] Brick is a schema that answers to the problem of heterogeneity of building representation, which adds a quick and non-costly reaction to energy efficiency measures.[96]

The concept of tags is being adopted, based on Project Haystack, to add a more flexible way to annotate metadata and then these tags are altered with an ontology that boosts its concepts, creating a framework that establishes hierarchies, relationships and properties that are mandatory for building metadata.[95], [97]What is more, using an ontology provides the schema with the ability to meddle with the metadata using common tools. In Brick schema, tagset concept is introduced, which groups tags with similar properties.[96]

In Figure 9, the information concepts and the relationship to a data point is shown. Relationships are qualities that connects a point with other classes, with the major classes being the Location, the Equipment and the Measurements, also shown in Figure 10,as well as their subclasses. In Table 1, the main relationships are being shown with their respective definitions. Figure 12 shows the relationships for a subset of the example building in Figure 11 and it is understood that it represents an early visual of a KG. Brick models are making easier to represent some subsystems in buildings, as it bypasses their complex and heterogeneous character and supports the composition and hierarchies in the building.[95]In Figure 13, a comparison is shown between Brick and other similar ontologies and it is understood that Brick can represent a larger spectrum of building related information. Furthermore, Brick also stands out for its ability to access open reference implementations on existing buildings, in order to authenticate the effectiveness of the solution.[95]

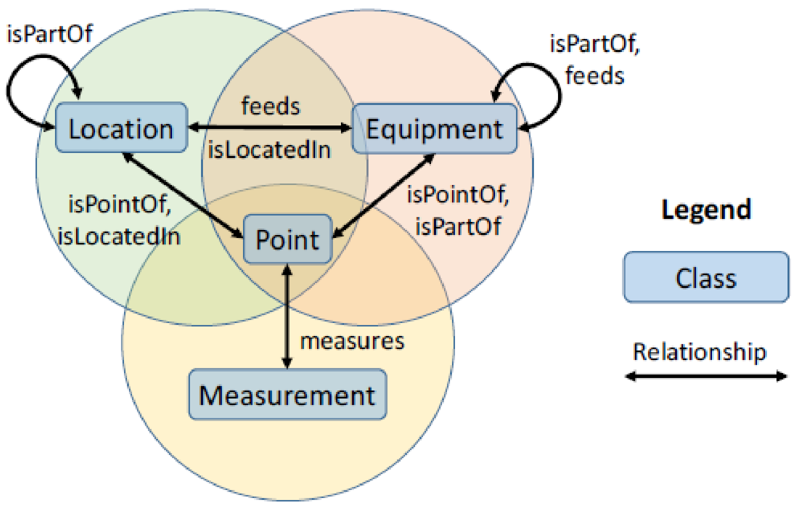
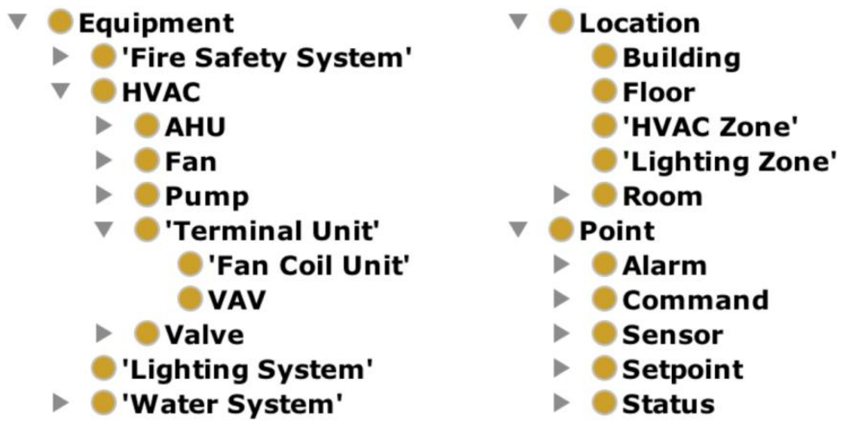


Figure 9: Information concepts in Brick and their relationship to a data point[95]



- ▼ ● Equipment
  - ▶ ● 'Fire Safety System'
  - ▼ ● HVAC
    - ▶ ● AHU
    - ▶ ● Fan
    - ▶ ● Pump
    - ▼ ● 'Terminal Unit'
      - ● 'Fan Coil Unit'
      - ● VAV
    - ▶ ● Valve
  - ● 'Lighting System'
  - ▶ ● 'Water System'
- ▼ ● Location
  - ● Building
  - ● Floor
  - ● 'HVAC Zone'
  - ● 'Lighting Zone'
  - ▶ ● Room
- ▼ ● Point
  - ▶ ● Alarm
  - ▶ ● Command
  - ▶ ● Sensor
  - ▶ ● Setpoint
  - ▶ ● Status

Figure 10: A subset of the Brick class hierarchy[95]

Relationship / Inverse	**Transitive?**	**Definition**	**Endpoints**
`contains` / `isLocatedIn`	Yes	A physically encapsulates B	Loc. / Sensor Loc. / Equip.
`controls` / `isControlledBy`	No	A determines or affects the internal state of B	Function Block / Equip.
`hasPart` / `isPartOf`	Yes	A has some component or part B (typically mechanical)	Equip. / Sensor Equip. / Equip. Loc. / Loc.
`hasPoint` / `isPointOf`	No	A is measured by or is otherwise represented by point B	Equip. / Sensor Loc. / Sensor
`feeds` / `isFedBy`	Yes	A "flows" or is connected to B	Function Block / Equip. Equip. / Equip.
`hasInput` / `isInputOf`	No	Function A has an input B	Function Block / Sensor
`hasOutput` / `isOutputOf`	No	Function A has an output B	Function Block / Sensor

Table 1: List of the Brick relationships and their definitions[95]

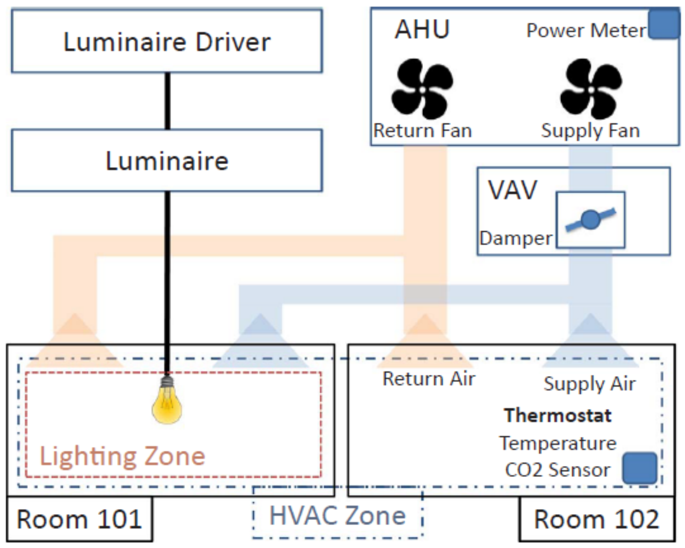


Figure 11: A simple example building that highlights the components to be modeled in a building schema[96]

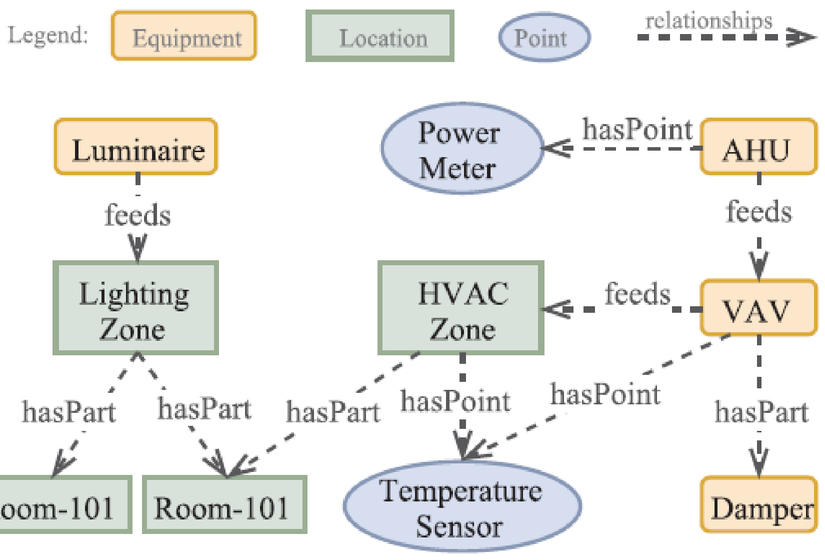
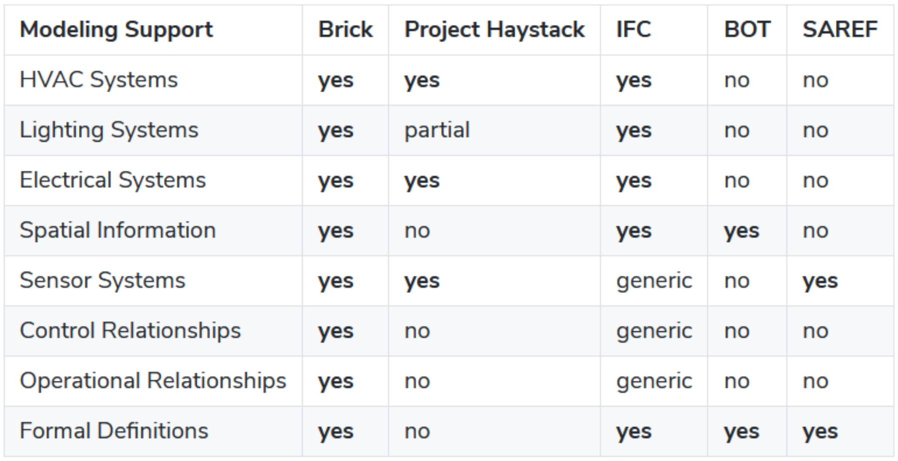


Figure 12: Brick classes and relationships for a subset of the example building in Figure 11[96]



Modeling Support	Brick	Project Haystack	IFC	BOT	SAREF
HVAC Systems	**yes**	**yes**	**yes**	no	no
Lighting Systems	**yes**	partial	**yes**	no	no
Electrical Systems	**yes**	**yes**	**yes**	no	no
Spatial Information	**yes**	no	**yes**	**yes**	no
Sensor Systems	**yes**	**yes**	generic	no	**yes**
Control Relationships	**yes**	no	generic	no	no
Operational Relationships	**yes**	no	generic	no	no
Formal Definitions	**yes**	no	**yes**	**yes**	**yes**

Figure 13: Comparison between Brick and other similar ontologies[98]

3. Methodology

The main goal of this thesis is an energy assessment in a near Zero Energy Building (nZEB) using a knowledge graph (KG). The methodology that was followed is shown in Figure 16 and consists of four main steps. First, the state of the art about KGs in nZEB, BIM, AEC-KGs and some uses and applications of them in the Built Environment industry are being researched. Then, having drawn information about the main goal, a case study is being constructed. Leaf House is selected as the case study nZEB and a knowledge graph will be built for it. The KG will be built in Python, with the assistance of Brick schema, QUDT and OWL ontologies. It will also have access in Leaf House's meters' database, which will be used to add more information about the building. The KG will represent just all six apartments and the three HVAC subsystems. Following, the results assessment takes place, where the KG is being visualized with Brick Studio and Brick Viewer, which import the .ttl file and produces a graph with the main nodes and the relations between them. What is more, in this section data are being retrieved from the KG through querying the KG in python. The data are then being used to create a data analysis based on the data that were extracted from the KG. After that, a thermal comfort assessment takes place, using the temperature and relative humidity data that were extracted from the KG, to calculate the Discomfort Index (DI). Last, conclusions and future work on the subject are being presented and discussed.

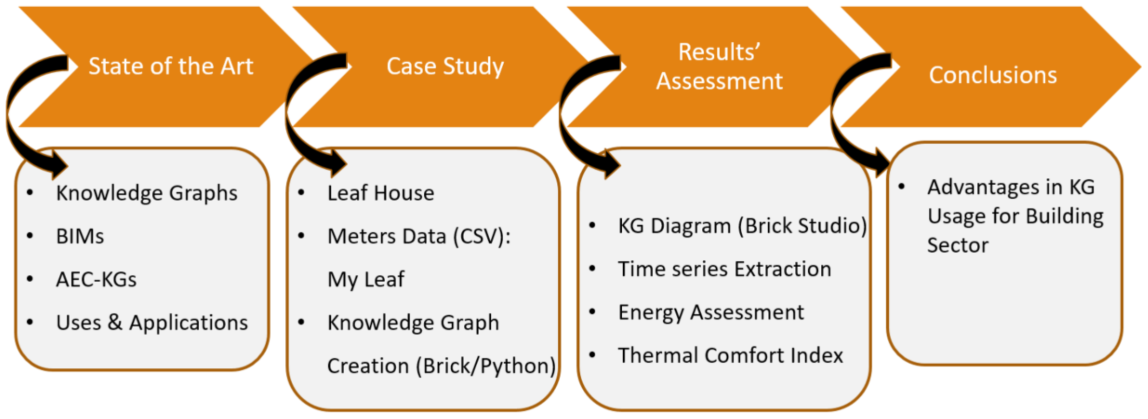


Figure 14: Methodology that was followed in current paper

3.1 Case Study: Leaf House

The building that was chosen for the case study is Leaf House, a residential nZEB that was built in 2008 by Loccioni and it is shown in Figure 17 It is located in Angeli di Rosora in Ancona, Italy (latitude 43°28′43.16 N, longitude 13°04′03.65 E, altitude 130m above sea level). It has 470m^2 area and is comprised of three stories with two a couple of flats in each story, making a total of six apartments. Leaf House has also installed a ventilated roof, solar tubes, smart monitoring and controls, building integrated photovoltaics, geothermal air preconditioning with heat pumps, solar thermal collectors, electrical storage and a user-friendly energy management system for residents.[99]

3.1.1 Electrical Energy System

The electrical energy system of Leaf House is shown in Figure 18 and is comprised of a 20 kW peak power photovoltaic plant, which is integrated into the building's roof and shades and supplies the building with renewable energy when that is possible, covering part of the energy demand.[100]The plant is made of 150 m^2 of south oriented mono crystalline modules, which provides energy both to HVAC system and to the apartments' lighting and loads. The Leaf House PV power plant is divided in three single-phase circuits, connected in order to obtain a balanced three phase circuit and what is more, each couple of the six apartments are connected to form a single phase circuit. The final three circuits are connected to form a three phase circuit.[100] The system also has access to two energy storage systems, which serve two couples of apartments and they store energy from two

Figure 15: Leaf House's Facade[99]

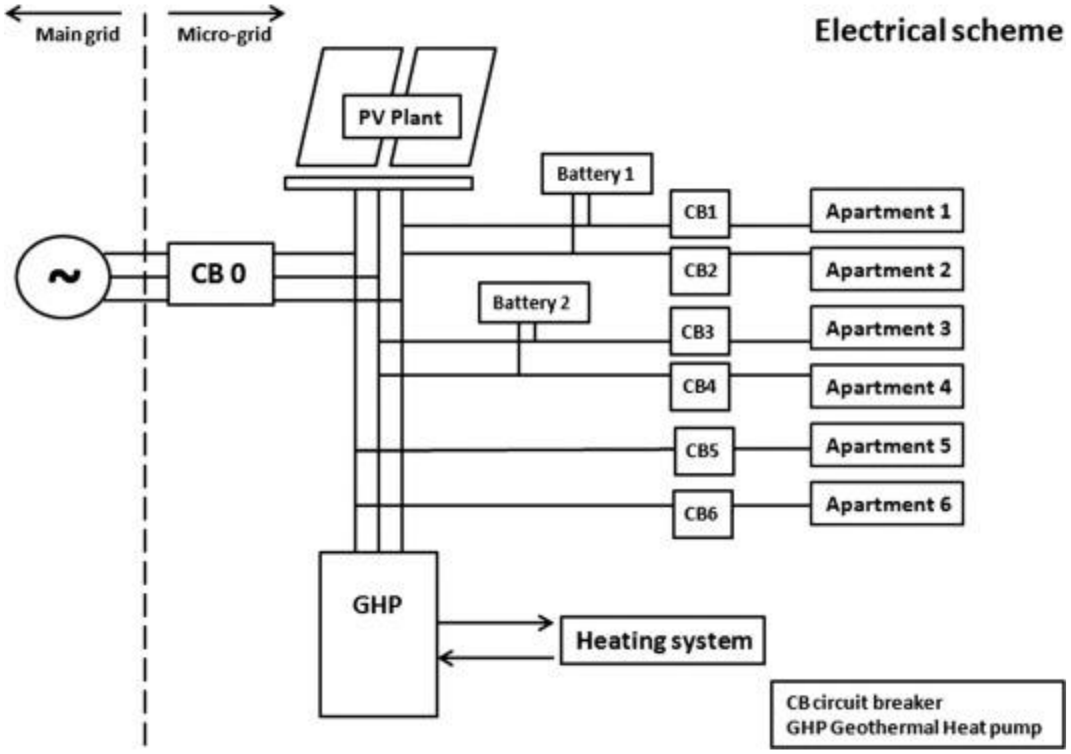


Figure 16: Leaf House's Electrical Energy System[100]

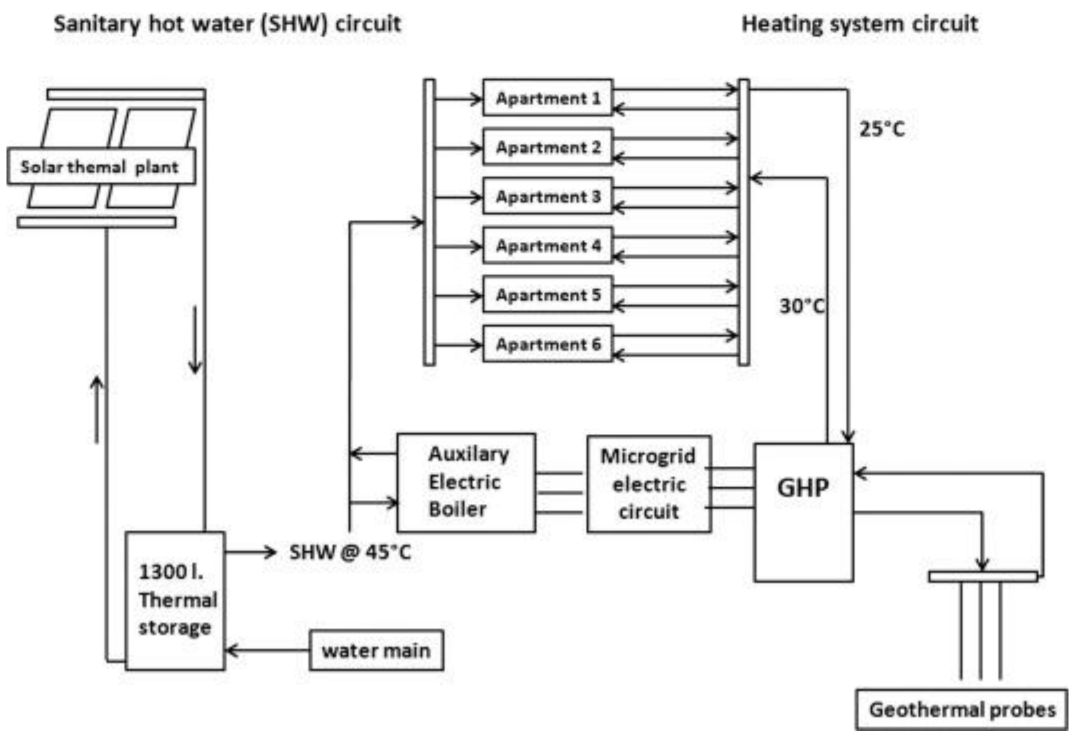


Figure 17: Leaf House's Thermal Energy System[100]

out of the three PV arrays. The storage system is comprised mainly from a 5.8 kW h Li-ion battery and its inverter based interface that manages the charge and discharge policies, a 6kWp PV array and the corresponding inverter and the energy manager that coordinates the storage, the production and the demand.[100]

3.2.2 Thermal Energy System

The building hosts a central geothermal heat pump (GHP) supplying cooling, heating and domestic hot water, as well as radiant floors, which supplies the apartments with thermal energy during cold seasons and extracting the heat during hot seasons. Domestic hot water is supplied from seven flat plane solar thermal collectors, which are

connected with a 1300l water tank acting as a thermal energy storage. When the solar panels are not being fed by enough solar energy an auxiliary electric boiler of about 15kWt supports the thermal energy production. All this can be shown in Figure 19.

3.1.3 MyLeaf Data

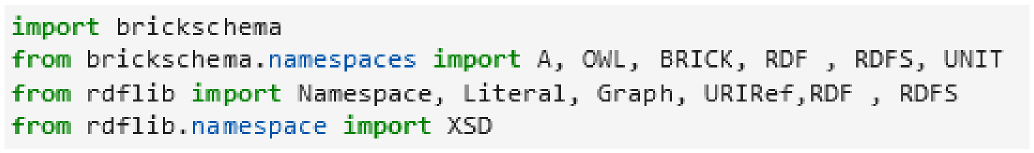
As mentioned, the building has many integrated meters, which are accessed by My Leaf platform that stores data for every building and renewable energy facility that is owned by Locioni.[101] Leaf House is one of them and there is a large number of meters and sensors placed in it. There are electric meters for all six apartments, measuring the current, voltage, energy and power of the main electrical loads and for the whole apartment. There are also electric meters for the central heating system, including meters for domestic hot water, the photovoltaic, the general distribution, the heat pumps, the storage, the humidifier as well as the general central heating system. All these parts are subdivided in L1, L2 and L3 phases, as previously mentioned, each one including two apartments, summing up to Ltot, which includes all three phases. For each phase there are meters for the current, voltage, energy and power of every meters of the central heating system. What is more, there are sensors for every apartment, measuring the air temperature, the relative humidity and the difference in indoor and outdoor temperature. Furthermore, there are setpoints for the inner temperature of different regions in the apartment. All these data are being uploaded in My Leaf dynamically and can be accessed to provide a clear view of the energy consumption of the whole building, with details for every meter in it.

3.2 Knowledge Graph Development

Imports

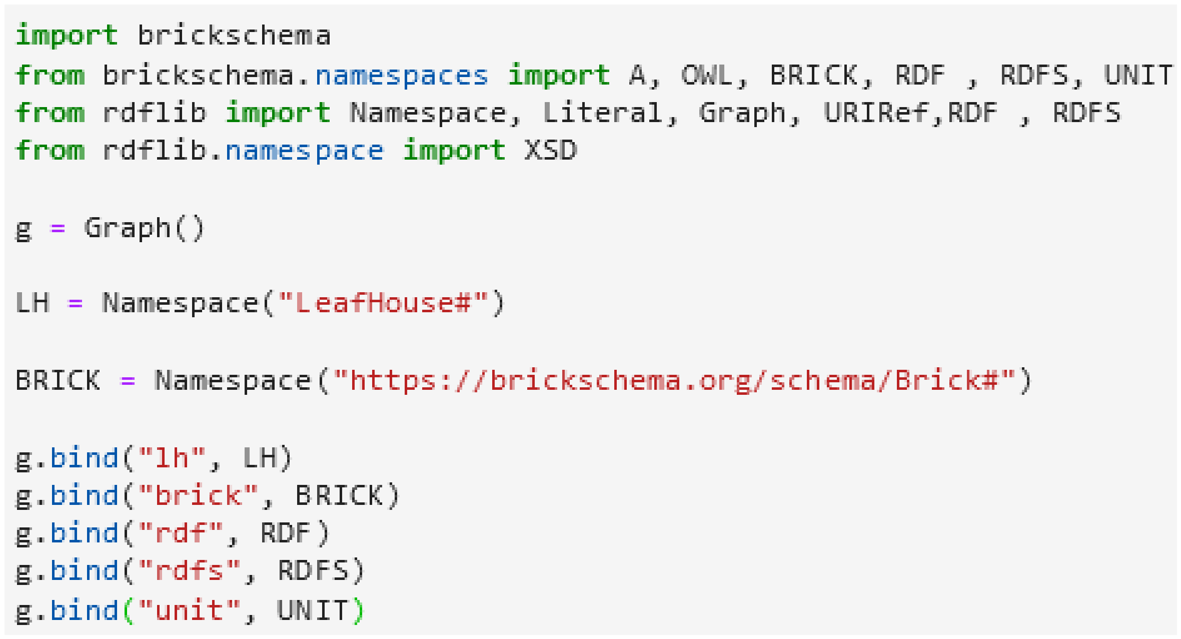
Leaf House's KG was developed using the Python programming language in Jupyter Lab, which is a user interface based on the web that uses documents and activities like Jupyter notebooks, text editors, terminals and custom components in a more flexible way.[102] Getting started with the KG, Brick schema was initially imported, as the base of the KG and after that other ontologies where added as extensions.[98] These extensions were namespaces imported from Brick schema like A, OWL, RDF and RDFS.[103]–[105] What is more, RDFlib was added, which is a RDF library and was used to import entities like Namespace, Literal, Graph, URIRef, RDF, RDFS and XSD that help with the proper establish of the RDF triples.[106] QUDT ontology was also imported, which is capable of providing information about measurement units.[87] All the necessary imports are shown in Jupyter Notebook in Figure 18.

After having imported the necessary ontologies, the graph that will hold all the triples is defined as "g". Then, the namespace "LH" is established, which purpose is to store all the entities that will be added that are connected with Leaf House. Continuing, the graph is provided with the knowledge of what the Brick namespace is in order to be allowed



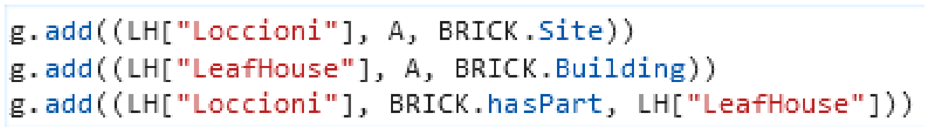
```
import brickschema
from brickschema.namespaces import A, OWL, BRICK, RDF , RDFS, UNIT
from rdflib import Namespace, Literal, Graph, URIRef,RDF , RDFS
from rdflib.namespace import XSD
```

Figure 18: Necessary Ontologies and Extensions Imported in Jupyter Notebook



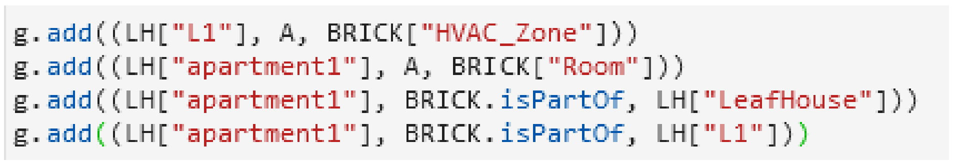
```
import brickschema
from brickschema.namespaces import A, OWL, BRICK, RDF , RDFS, UNIT
from rdflib import Namespace, Literal, Graph, URIRef,RDF , RDFS
from rdflib.namespace import XSD

g = Graph()

LH = Namespace("LeafHouse#")

BRICK = Namespace("https://brickschema.org/schema/Brick#")

g.bind("lh", LH)
g.bind("brick", BRICK)
g.bind("rdf", RDF )
g.bind("rdfs", RDFS)
g.bind("unit", UNIT)
```

Figure 19: Graph Insertion, Namespaces Establishment and Bind in Jupyter Notebook



```
g.add((LH["Loccioni"], A, BRICK.Site))
g.add((LH["LeafHouse"], A, BRICK.Building))
g.add((LH["Loccioni"], BRICK.hasPart, LH["LeafHouse"]))
```

Figure 20: Site and Building Esteblishment



```
g.add((LH["L1"], A, BRICK["HVAC_Zone"]))
g.add((LH["apartment1"], A, BRICK["Room"]))
g.add((LH["apartment1"], BRICK.isPartOf, LH["LeafHouse"]))
g.add((LH["apartment1"], BRICK.isPartOf, LH["L1"]))
```

Figure 21: Spatial Elements Definition in Jupyter Notebook

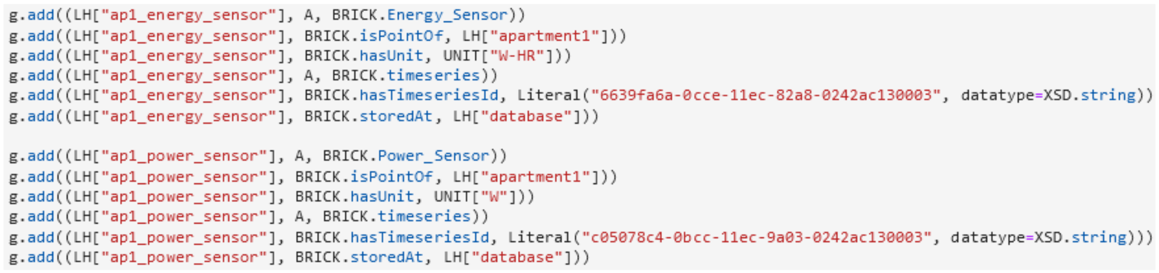
to make references about classes and relationships that are defined in Brick schema. Then, that namespace URI's are bind with prefixes. Figure 19 shows these next steps that were added in the Jupyter Notebook.

Spatial Entities

Having established the base of the graph, triples are then added to represent the building in it. First, the building and the site is defined as an RDF type “Brick:Site” and “Brick:Building” and are connected with the relation “Brick:hasPart” that makes Leaf House part of the Locioni site, which are also shown in Figure 20. Next, the rest spatial elements are defined, which are the six apartments and the three phases of the HVAC system. The HVAC system’s zones are added as “Brick:HVAC_Zone” and are named after the MyLeaf database designation L1,L2,L3 and each contain two apartments. The apartments are added as a “Brick:Room”, which are part of the building, using the relation “Brick:isPartOf”, as well as part of the HVAC zone, using the same relation. In Figure 21 the spatial elements definition for HVAC L1 phase and apartment one is presented in Jupyter Notebook. The rest phases and apartments follow the same script with different names.

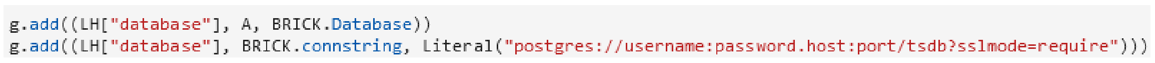
Sensors

Following the spatial elements, the building’s sensors are added, which include the energy sensors as “Brick:Energy_Sensors” and the power sensors as “Brick:Power_Sensors”, for all six apartments. The sensors are connected to the apartment with the relation “Brick:isPointOf” and units are added to them with “Brick:hasUnit” and UNIT ontology’s “UNIT[“W-HR”]”, which establishes that this sensor’s measurement units are Wh. Next, the sensor is also introduced as a timeseries, in order to be able to hold specidic data from a database. This is made possible with “Brick:timeseries” and “Brick:hasTimeseriesId”, which establish the unique id of the timeseries. UUIDs (Universally Unique Identifiers) are 128 bit numbers which are composed of 16 octets and represented as 32 base-16 characters and can be used to identify information across a computer system and the web.[107] These UUIDs are introduced into the knowledge graph with rdflib’s literal term, which are attribute values in RDF and can have different datatypes that in this situation is a string datatype. The timeseries data are stored with the relation “Brick:storedAt” in the database. The same procedure is applied for the power sensors, with the units being W. In this case study, the database that was chosen to be used is TimescaleDB, which is an open-source database invented to make SQL scalable for



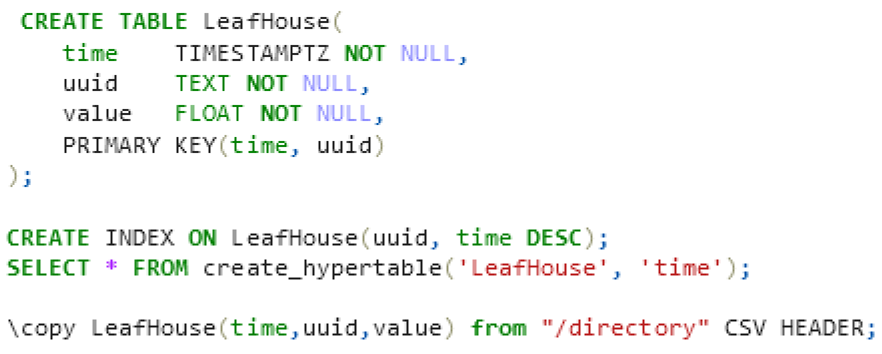
```
g.add((LH["ap1_energy_sensor"], A, BRICK.Energy_Sensor))
g.add((LH["ap1_energy_sensor"], BRICK.isPointOf, LH["apartment1"]))
g.add((LH["ap1_energy_sensor"], BRICK.hasUnit, UNIT["W-HR"]))
g.add((LH["ap1_energy_sensor"], A, BRICK.timeseries))
g.add((LH["ap1_energy_sensor"], BRICK.hasTimeseriesId, Literal("6639fa6a-0cce-11ec-82a8-0242ac130003", datatype=XSD.string))
g.add((LH["ap1_energy_sensor"], BRICK.storedAt, LH["database"]))

g.add((LH["ap1_power_sensor"], A, BRICK.Power_Sensor))
g.add((LH["ap1_power_sensor"], BRICK.isPointOf, LH["apartment1"]))
g.add((LH["ap1_power_sensor"], BRICK.hasUnit, UNIT["W"]))
g.add((LH["ap1_power_sensor"], A, BRICK.timeseries))
g.add((LH["ap1_power_sensor"], BRICK.hasTimeseriesId, Literal("c05078c4-0bcc-11ec-9a03-0242ac130003", datatype=XSD.string)))
g.add((LH["ap1_power_sensor"], BRICK.storedAt, LH["database"]))
```

Figure 22: Energy & Power Sensors' Triples for Apartment 1 in Jupyter Notebook



```
g.add((LH["database"], A, BRICK.Database))
g.add((LH["database"], BRICK.connstring, Literal("postgres://username:password.host:port/tsdb?sslmode=require")))
```

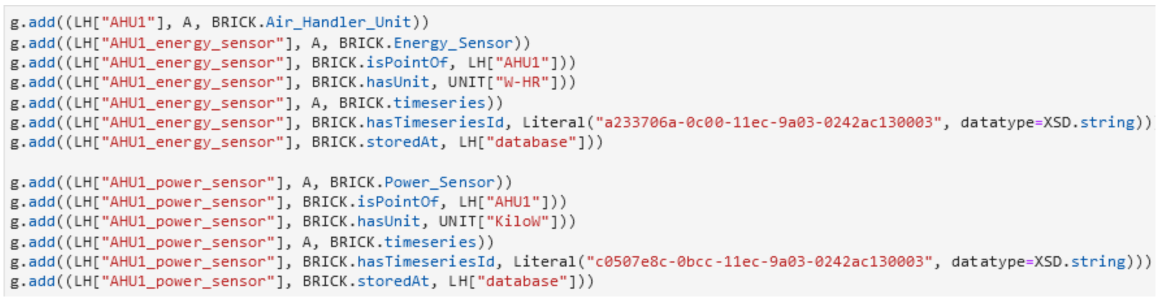
Figure 23: Database Creation and Connection Triples in Jupyter Notebook



```
 CREATE TABLE LeafHouse(
    time     TIMESTAMPTZ NOT NULL,
    uuid     TEXT NOT NULL,
    value    FLOAT NOT NULL,
    PRIMARY KEY(time, uuid)
);

CREATE INDEX ON LeafHouse(uuid, time DESC);
SELECT * FROM create_hypertable('LeafHouse', 'time');

\copy LeafHouse(time,uuid,value) from "/directory" CSV HEADER;
```

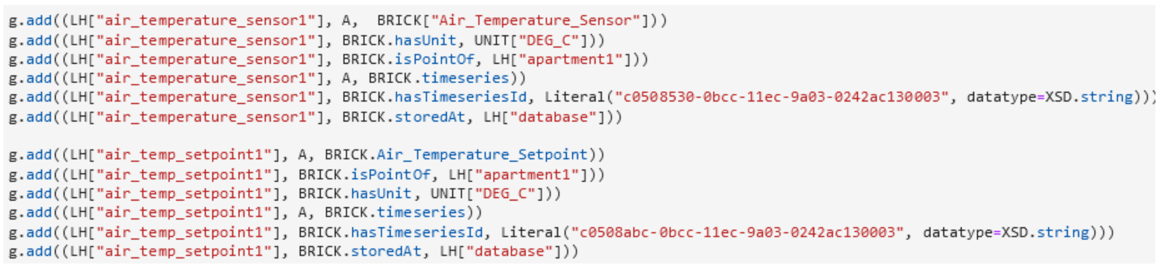
Figure 24: Table/Hypertable Creation and Data Ingestion into TimescaleDB Using PostgreSQL

.



```
g.add((LH["AHU1"], A, BRICK.Air_Handler_Unit))
g.add((LH["AHU1_energy_sensor"], A, BRICK.Energy_Sensor))
g.add((LH["AHU1_energy_sensor"], BRICK.isPointOf, LH["AHU1"]))
g.add((LH["AHU1_energy_sensor"], BRICK.hasUnit, UNIT["W-HR"]))
g.add((LH["AHU1_energy_sensor"], A, BRICK.timeseries))
g.add((LH["AHU1_energy_sensor"], BRICK.hasTimeseriesId, Literal("a233706a-0c00-11ec-9a03-0242ac130003", datatype=XSD.string))
g.add((LH["AHU1_energy_sensor"], BRICK.storedAt, LH["database"]))

g.add((LH["AHU1_power_sensor"], A, BRICK.Power_Sensor))
g.add((LH["AHU1_power_sensor"], BRICK.isPointOf, LH["AHU1"]))
g.add((LH["AHU1_power_sensor"], BRICK.hasUnit, UNIT["KiloW"]))
g.add((LH["AHU1_power_sensor"], A, BRICK.timeseries))
g.add((LH["AHU1_power_sensor"], BRICK.hasTimeseriesId, Literal("c0507e8c-0bcc-11ec-9a03-0242ac130003", datatype=XSD.string)))
g.add((LH["AHU1_power_sensor"], BRICK.storedAt, LH["database"]))
```

Figure 25: Energy & Power Sensors' Triples for Air Handling Unit 1 in Jupyter Notebook



```
g.add((LH["air_temperature_sensor1"], A,  BRICK["Air_Temperature_Sensor"]))
g.add((LH["air_temperature_sensor1"], BRICK.hasUnit, UNIT["DEG_C"]))
g.add((LH["air_temperature_sensor1"], BRICK.isPointOf, LH["apartment1"]))
g.add((LH["air_temperature_sensor1"], A, BRICK.timeseries))
g.add((LH["air_temperature_sensor1"], BRICK.hasTimeseriesId, Literal("c0508530-0bcc-11ec-9a03-0242ac130003", datatype=XSD.string))
g.add((LH["air_temperature_sensor1"], BRICK.storedAt, LH["database"]))

g.add((LH["air_temp_setpoint1"], A, BRICK.Air_Temperature_Setpoint))
g.add((LH["air_temp_setpoint1"], BRICK.isPointOf, LH["apartment1"]))
g.add((LH["air_temp_setpoint1"], BRICK.hasUnit, UNIT["DEG_C"]))
g.add((LH["air_temp_setpoint1"], A, BRICK.timeseries))
g.add((LH["air_temp_setpoint1"], BRICK.hasTimeseriesId, Literal("c0508abc-0bcc-11ec-9a03-0242ac130003", datatype=XSD.string)))
g.add((LH["air_temp_setpoint1"], BRICK.storedAt, LH["database"]))
```

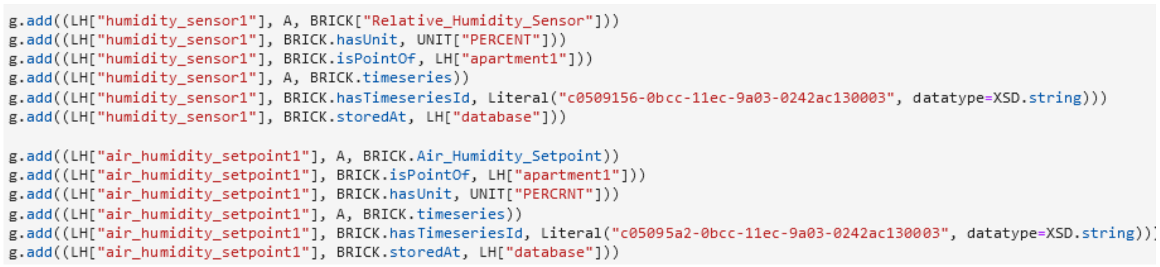
Figure 26: Air Temperature Sensor & Setpoint for Apartment 1 in Jupyter Notebook

timeseries data.[108] The database is established in the KG with “Brick:Database” and is connected to it with the relation “Brick:connstring” to a literal term that uses PostgreSQL, which is an open source object-relational database system that uses and extends the SQL language.[109] In Figure 22, the triples for the energy sensor of one apartment are shown and in Figure 23, the triples for the database creation and connection.

The sensors’ data are provided by MyLeaf webpage and are then stored into TimescaleDB. So, PostgreSQL is used to connect to the database and then the table that will contain the data is created. What is more, the time, uuid and value columns are being inserted and after that based on this table, a hypertable is created to hold the table’s data. Hypertables consist of “chunks” of tables in order to be easier to manage and to behave predictably to users familiar with standard PostgreSQL tables.[108]Then, to add the data into this hypertable, they are copied from the directory that the CSV files exist into the LeafHouse hypertable. The code for this procedure is shown in Figure 24.

Energy and power sensors also exist for the air handling units, which is added into the KG as “Brick:Air_Handler_Unit” and follows the same pattern as the apartments’ triples. In Figure 25, one of the three units’ triples are shown. In addition, there are air temperature sensors added for every apartment as “Brick:Air_Temperature_Sensor” together with air temperature setpoints as “Brick:Air_Temperature_Setpoint”. They also follow the same structure of triples, which is shown in Figure 26. The apartments are also equipped with air humidity sensors and setpoints, which are added in the KG as “Brick:Relative_Humidity_Sensor” and “Brick:Air_Humidity_Setpoint” and follow the same structure of triples as the rest sensors, as shown in Figure 27. After applying these triples for every apartment, all sensors and spatial elements have been established into the KG and it is then saved as a turtle file (.ttl).

Now that the KG is ready and in turtle file form, it can be visualized in Brick Viewer, which shows the main classes and connections between them and in Brick Studio, which shows every node and relation that was added into the KG. Brick Viewer’s graph is shown in Figure 28 and Brick Studio’s in Figure 29. In Figure 30 a closer look of the Leaf House KG, viewed in Brick Studio, is shown and it can be understood that there are three nodes, one being the Air Handling Unit 1, which is connected with “Brick:isPointOf” relation with the other two nodes, which are the energy and power sensors of the AHU1.



```
g.add((LH["humidity_sensor1"], A, BRICK["Relative_Humidity_Sensor"]))
g.add((LH["humidity_sensor1"], BRICK.hasUnit, UNIT["PERCENT"]))
g.add((LH["humidity_sensor1"], BRICK.isPointOf, LH["apartment1"]))
g.add((LH["humidity_sensor1"], A, BRICK.timeseries))
g.add((LH["humidity_sensor1"], BRICK.hasTimeseriesId, Literal("c0509156-0bcc-11ec-9a03-0242ac130003", datatype=XSD.string)))
g.add((LH["humidity_sensor1"], BRICK.storedAt, LH["database"]))

g.add((LH["air_humidity_setpoint1"], A, BRICK.Air_Humidity_Setpoint))
g.add((LH["air_humidity_setpoint1"], BRICK.isPointOf, LH["apartment1"]))
g.add((LH["air_humidity_setpoint1"], BRICK.hasUnit, UNIT["PERCRNT"]))
g.add((LH["air_humidity_setpoint1"], A, BRICK.timeseries))
g.add((LH["air_humidity_setpoint1"], BRICK.hasTimeseriesId, Literal("c05095a2-0bcc-11ec-9a03-0242ac130003", datatype=XSD.string))
g.add((LH["air_humidity_setpoint1"], BRICK.storedAt, LH["database"]))
```

Figure 27: Relative Humidity Sensor and Setpoint for Apartment 1 in Jupyter Notebook

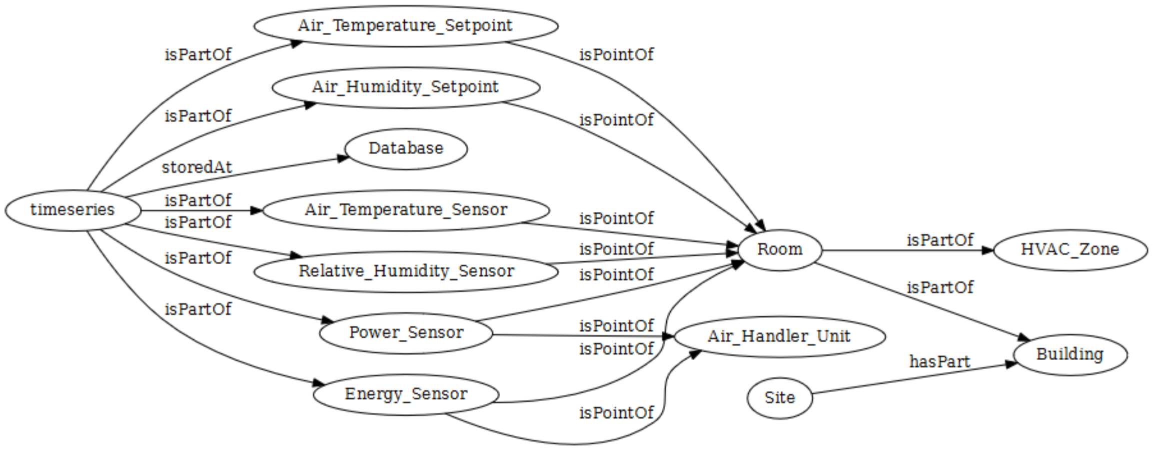


Figure 28: KG's Main Classes and Connections Viewed in Brick Viewer

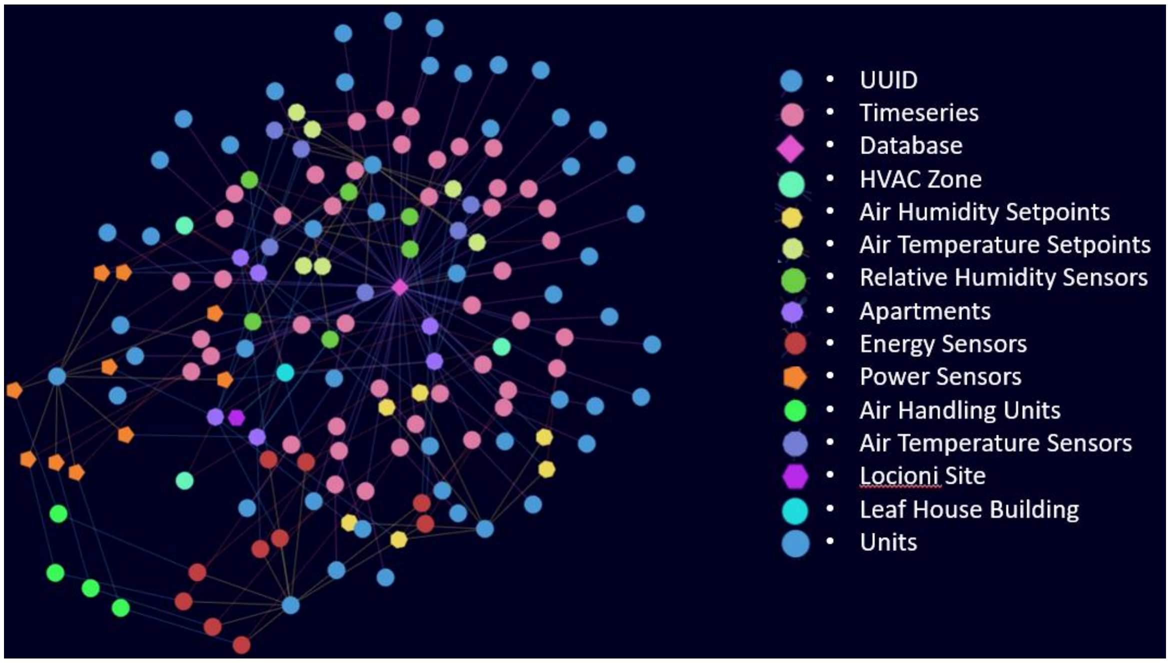


Figure 29: Leaf House KG's Entities and Relations Viewed in Brick Studio

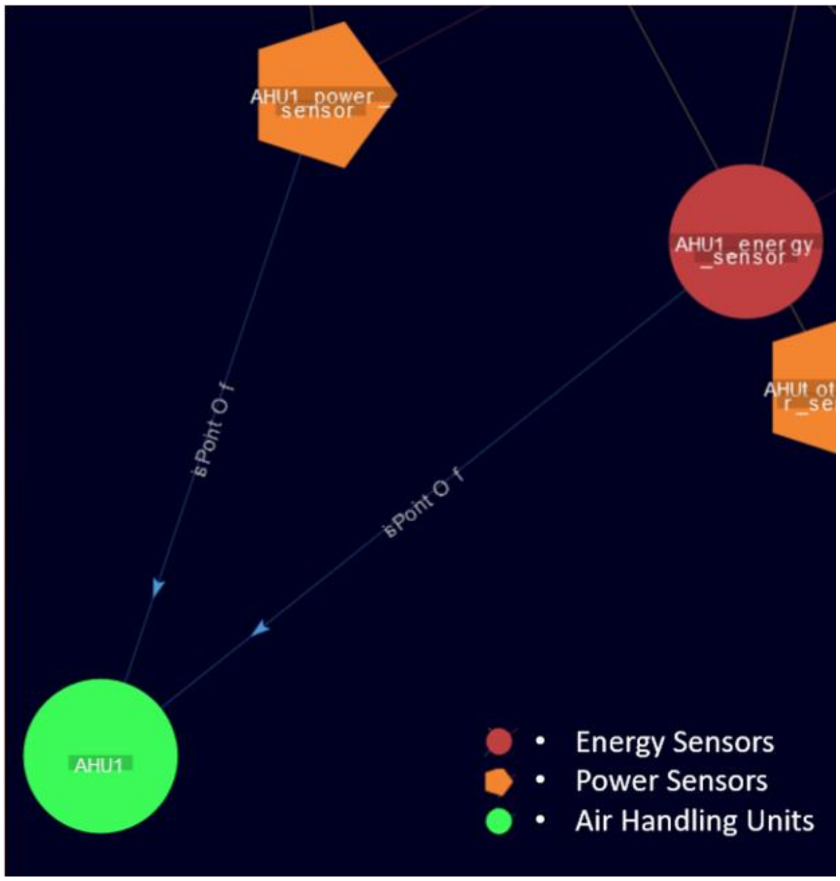
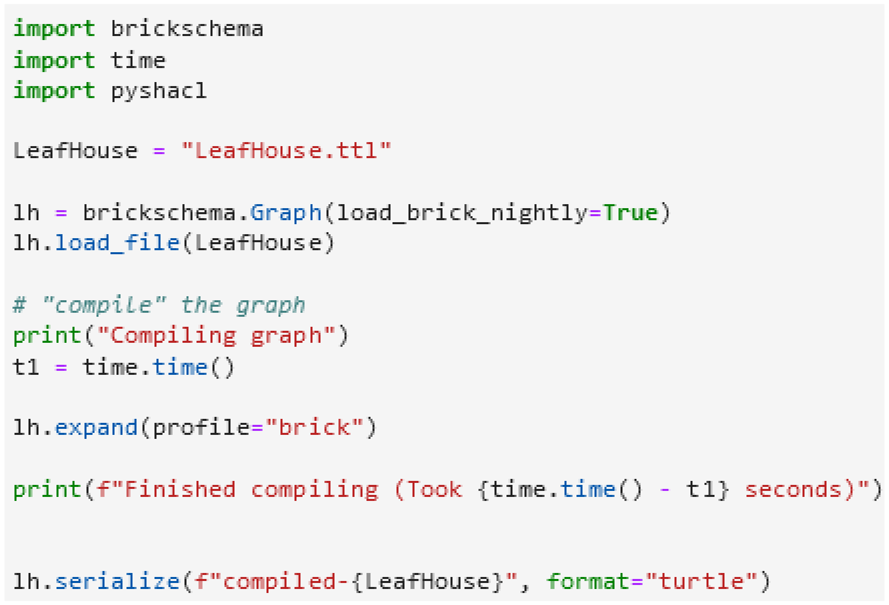
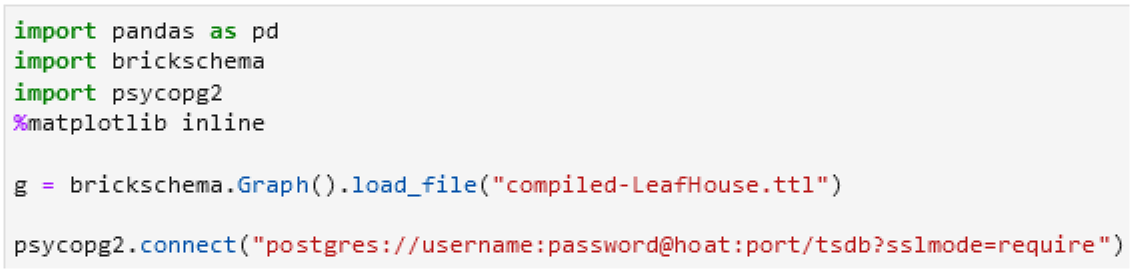


Figure 30: Air Handling Unit 1 Node Connected with Energy and Power Sensor Nodes Viewed in Brick Studio



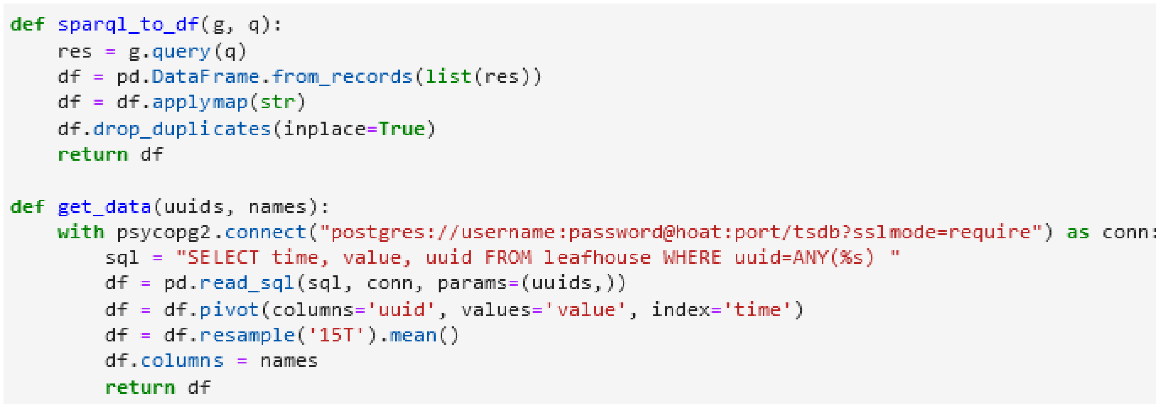
```
import brickschema
import time
import pyshacl

LeafHouse = "LeafHouse.ttl"

lh = brickschema.Graph(load_brick_nightly=True)
lh.load_file(LeafHouse)

# "compile" the graph
print("Compiling graph")
t1 = time.time()

lh.expand(profile="brick")

print(f"Finished compiling (Took {time.time() - t1} seconds)")


lh.serialize(f"compiled-{LeafHouse}", format="turtle")
```

Figure 31:Leaf House KG Compile



```
import pandas as pd
import brickschema
import psycopg2
%matplotlib inline

g = brickschema.Graph().load_file("compiled-LeafHouse.ttl")

psycopg2.connect("postgres://username:password@hoat:port/tsdb?sslmode=require")
```

Figure 32: Data Retrieval Imports, KG Load and Database Connection Establishment in Jupyter Notebook



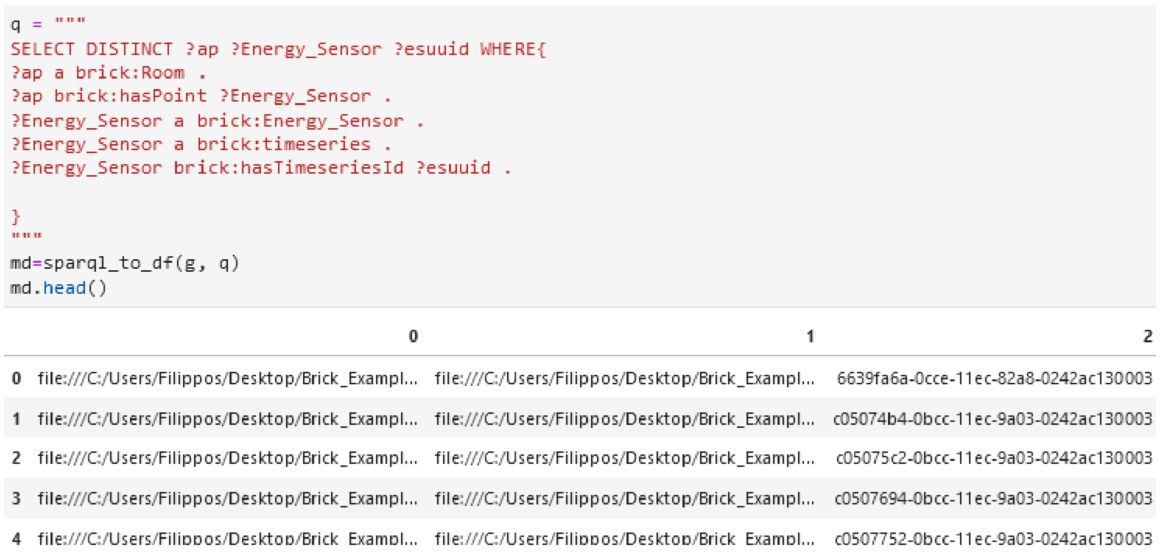
```
def sparql_to_df(g, q):
    res = g.query(q)
    df = pd.DataFrame.from_records(list(res))
    df = df.applymap(str)
    df.drop_duplicates(inplace=True)
    return df

def get_data(uuids, names):
    with psycopg2.connect("postgres://username:password@hoat:port/tsdb?sslmode=require") as conn:
        sql = "SELECT time, value, uuid FROM leafhouse WHERE uuid=ANY(%s) "
        df = pd.read_sql(sql, conn, params=(uuids,))
        df = df.pivot(columns='uuid', values='value', index='time')
        df = df.resample('15T').mean()
        df.columns = names
        return df
```

Figure 33: "sparql_to_df"and "get_data" Functions in Jupyter Notebook

3.3 KG Compiling and Timeseries Extraction

Having created the Leaf House KG, the next step is to present how it can be useful. That will be done by creating a quick and easy data retrieval query, which will use the KG graph to search and locate different parts of the building, access their stored data and present these data as well as different plots of them, all customizable to the user's intentions.

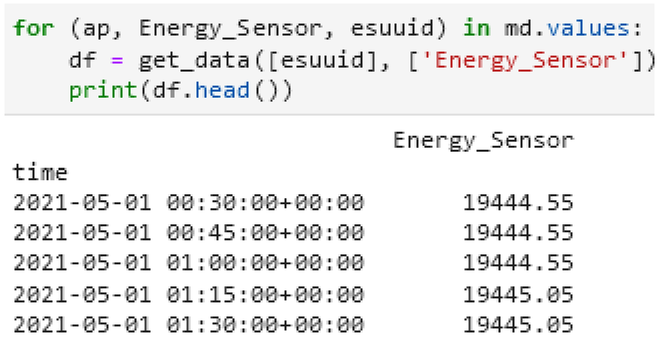
Before querying the KG, an extra procedure is added to make stronger and more precise connections between the entities, the UUIDs and the database. This procedure is presented in Figure 31 and is purpose is to compile the KG using the command "expand" and selecting its profile to be "brick", which applies "owlr+shacl+owlr" rules into the graph. For the data retrieval script, first pandas is imported, which is an open source data analysis and manipulation tool built on top of Python language, then brickschema and last psycopg2, which is a PostgreSQL database adapter for Python.[110], [111] What is more, matplotlib inline function is introduced, which renders the figures in Jupyter notebook. After that, the KG is loaded in and the connection with the database is established, as shown in Figure 32. Next, to functions are being introduced, one being "sparql_to_df", which will



```
q = """
SELECT DISTINCT ?ap ?Energy_Sensor ?esuuid WHERE{
?ap a brick:Room .
?ap brick:hasPoint ?Energy_Sensor .
?Energy_Sensor a brick:Energy_Sensor .
?Energy_Sensor a brick:timeseries .
?Energy_Sensor brick:hasTimeseriesId ?esuuid .

}
"""
md=sparql_to_df(g, q)
md.head()
```

	0	1	2
0	file:///C:/Users/Filippos/Desktop/Brick_Exampl...	file:///C:/Users/Filippos/Desktop/Brick_Exampl...	6639fa6a-0cce-11ec-82a8-0242ac130003
1	file:///C:/Users/Filippos/Desktop/Brick_Exampl...	file:///C:/Users/Filippos/Desktop/Brick_Exampl...	c05074b4-0bcc-11ec-9a03-0242ac130003
2	file:///C:/Users/Filippos/Desktop/Brick_Exampl...	file:///C:/Users/Filippos/Desktop/Brick_Exampl...	c05075c2-0bcc-11ec-9a03-0242ac130003
3	file:///C:/Users/Filippos/Desktop/Brick_Exampl...	file:///C:/Users/Filippos/Desktop/Brick_Exampl...	c0507694-0bcc-11ec-9a03-0242ac130003
4	file:///C:/Users/Filippos/Desktop/Brick_Exampl...	file:///C:/Users/Filippos/Desktop/Brick_Exampl...	c0507752-0bcc-11ec-9a03-0242ac130003

Figure 34: Apartment' Energy Sensors' Query and UUID Results for Required Entities in Jupyter Notebook



```
for (ap, Energy_Sensor, esuuid) in md.values:
    df = get_data([esuuid], ['Energy_Sensor'])
    print(df.head())
```

```
                           Energy_Sensor
time
2021-05-01 00:30:00+00:00       19444.55
2021-05-01 00:45:00+00:00       19444.55
2021-05-01 01:00:00+00:00       19444.55
2021-05-01 01:15:00+00:00       19445.05
2021-05-01 01:30:00+00:00       19445.05
```

Figure 35: Apartment's 1 Energy Sensor Data Extraction Results in Jupyter Notebook

connect the KG and query with a dataframe , make a list of the data that is required and will drop duplicate results and the other being "get_data", which connects to the database and access the hypertable that was previously introduced, retrieving in that way the data that are required and then returning a table of these data. These functions' code is shown in Figure 33.

What follows is the query to locate the UUIDs and data for the entities that the user desire. In this case study of KG there are 42 sensors, all of them connected to a different UUID and set of data. Taking the apartments' energy sensors for example, there is a query introduced in Figure 34 that establishes what is looked for in the KG and then the "sparql_to_df" function is used for the KG and the query in order to obtain the UUIDs for all apartments' energy sensors in a list, also shown in Figure 33. Next, the "get_data" function is used for the entities and UUIDs that were queried, in order to obtain a table of data, as shown in Figure 35. In this Figure, the data results are shown in two columns, one filled with the time data and the other with the energy sensor data of apartment 1.

This procedure can be done for any sensor that is connected to entities in the KG, creating an example of data discovering and retrieval in a KG. Furthermore, these data can be plotted and create a diagram, in this example of energy consumption (Wh) in apartment 1 over time. In order to further enhance the example, in section 4.1 there will be presented the diagrams of the energy and power sensors of apartment 1 and air handling unit 1 as well as the diagrams of air temperature and air humidity sensors with their respective setpoints, all over time. Then they will be discussed over their way of creation, usage and impact.

4. Results' Assessment

4.1 Discovered and Retrieved Data

Using the plot command for the data that was extracted in the previous section, the diagram in Figure 36 is created. It shows the energy consumption (kWh) in apartment 1 for the year 2020, based on the energy electric meter in Leaf House. In this diagram it can be shown that there is a gradual increase in consumption throughout the year, with a major increase in the end of summer. Next, the discovering and data retrieval procedure, as well as its plot will take place for power sensor of apartment 1, energy and power sensor of air handling unit 1 and air temperature and humidity sensors with their respective setpoints for apartment 1, all in the same time period as previously mentioned. So in Figure 37, the diagram of the first apartment's power sensor is being presented, including power consumption (W) over time. It can be understood that for the month April, the residents might be absent, due to the low power demand of the apartment 1. The max power demand seems to be approximately 3.5kW.Following, the energy and power sensor data for air handling unit 1 and for the same time period are retrieved and plotted in Figures 38 and 39. In Figure 38, the energy consumption (Wh) over time diagram is shown. There is a gradual increase throught the year that seems to create a small plateau for the months May and June and after that there is a major increase again. This happens due to the fact that these months in Italy, the climate conditions favours the building's indoor temperature and humidity, in a way that not much energy is needed to reach the setpoints. In this point, it should be reminded that the air handling unit 1 is responsible for both apartments 1 and 2.

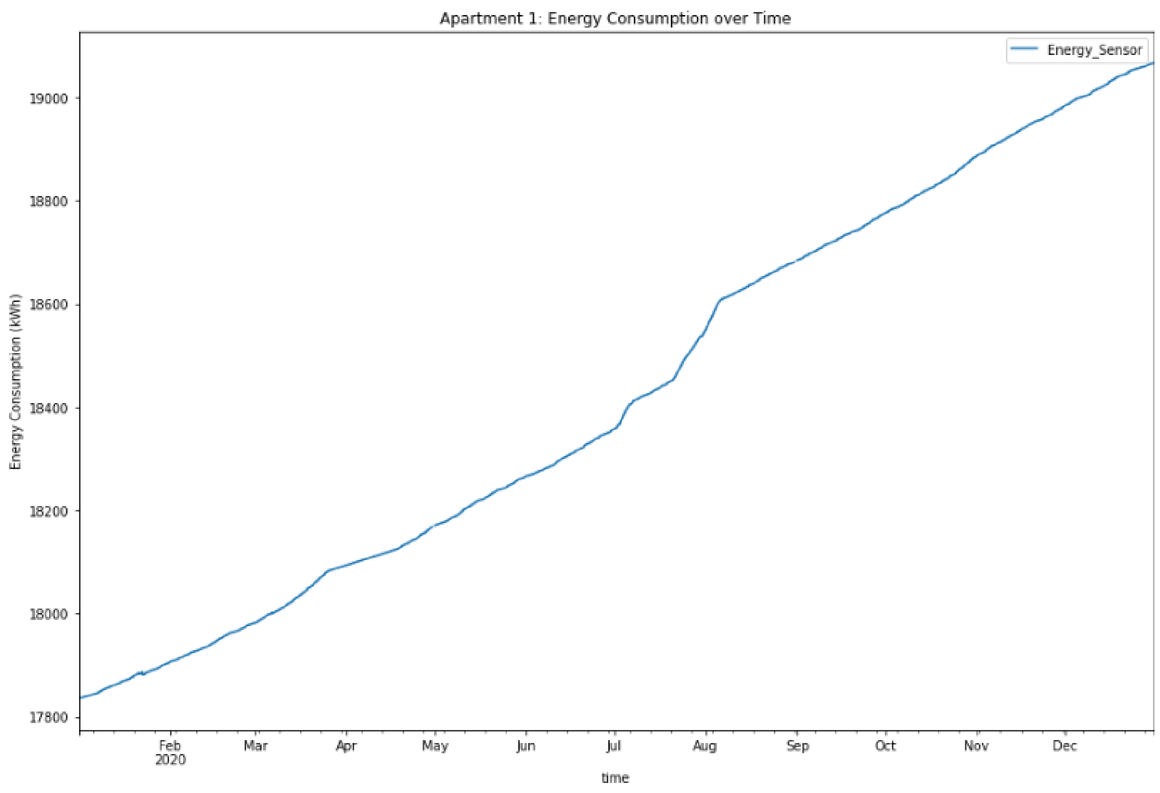


Figure 36: Annual Energy Consumption (kWh) Over Time for Apartment 1 With Data Retrieved from the KG

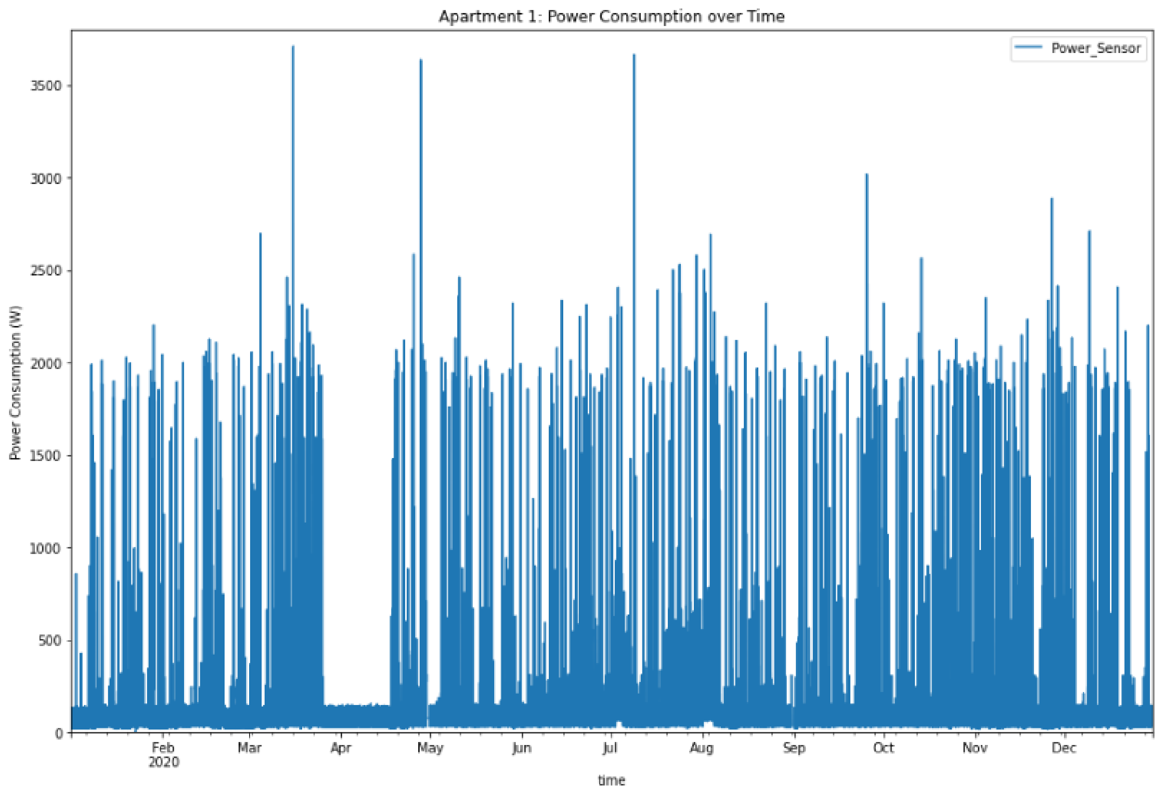


Figure 37: Annual Power Demand (W) Over Time for Apartment 1 With Data Retrieved from the KG

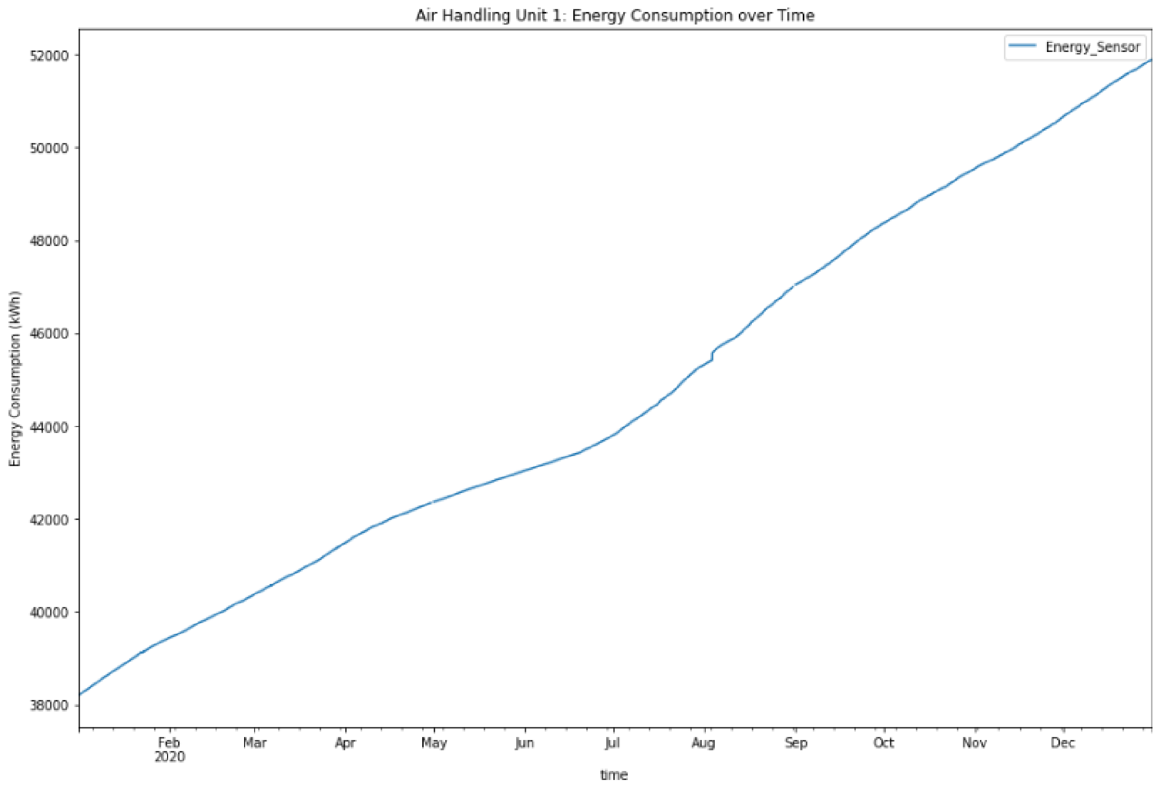


Figure 38: Annual Energy Consumption (Wh) over Time for Air Handling Unit 1 with Data Retrieved from the KG

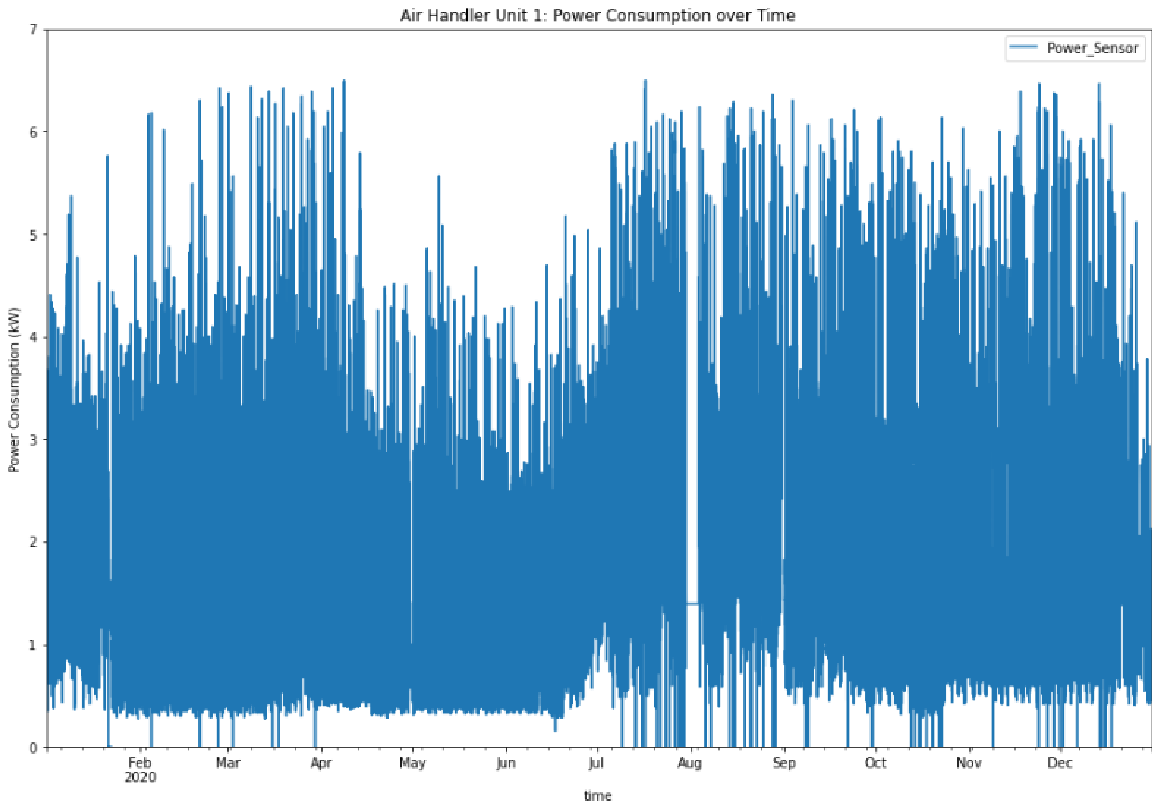


Figure 39: Annual Power Demand (kW) over Time for Air Handling Unit 1 with Data Retrieved from the KG

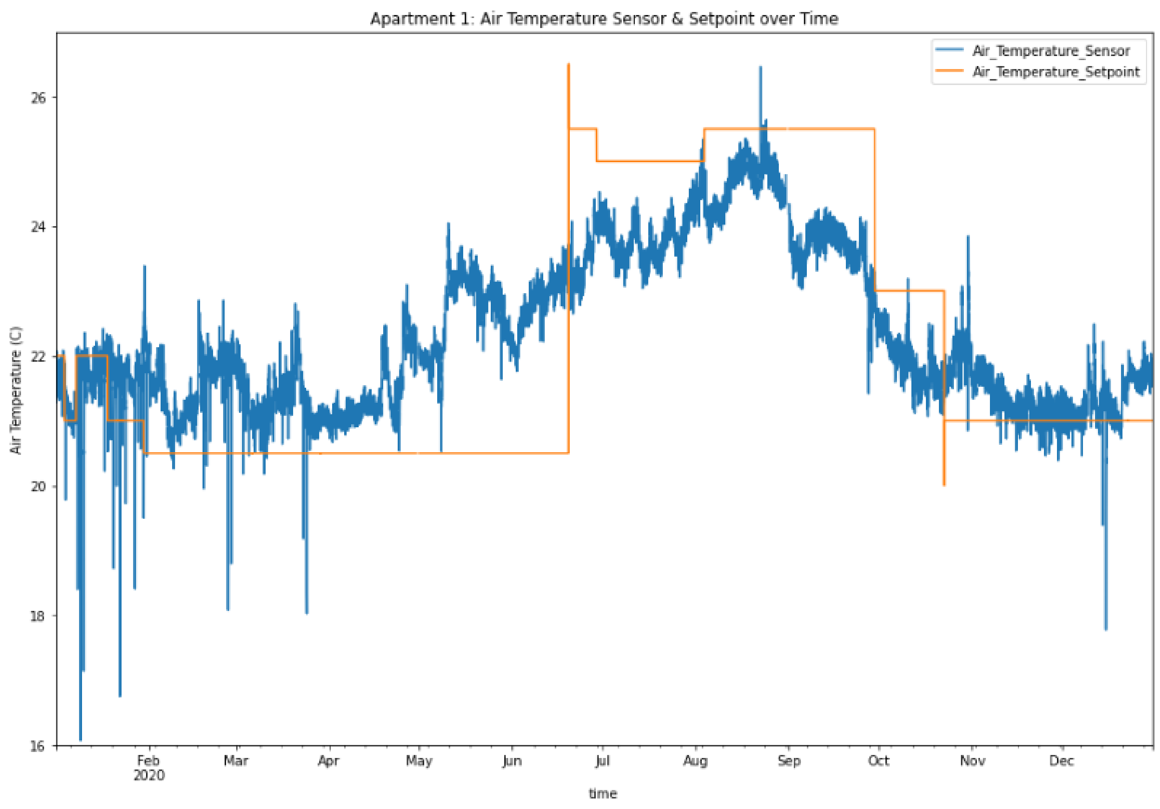


Figure 40: Annual Air Temperature (°C) Sensor and Setpoint over Time with Data Retrieved from the KG

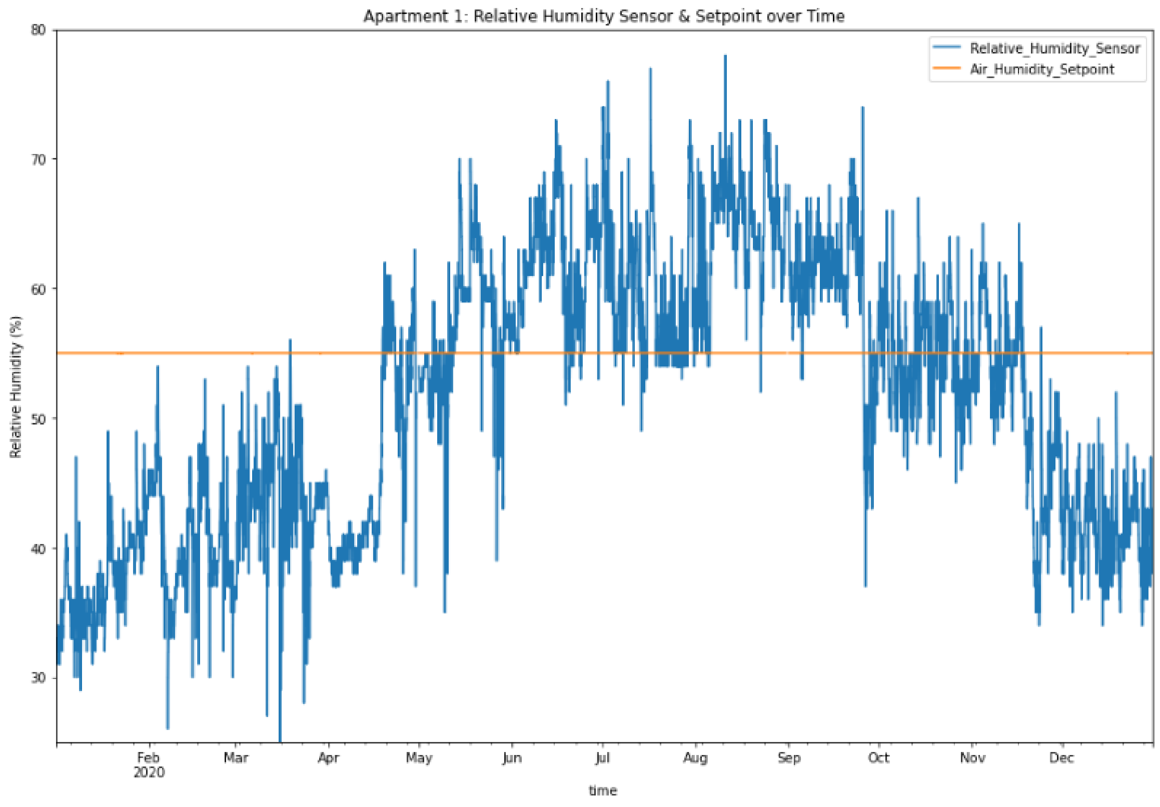


Figure 41: Annual Relative Humidity (%) Sensor and Setpoint over Time with Data Retrieved form the KG

In Figure 39, the power consumption over time diagram for the air handling unit 1 is shown and it has a max value of approximately 6.5kW. The data discovering and retrieval process is applied for the air temperature sensor and setpoint for apartment 1 and the data plot are shown in Figure 40. In this Figure, it can understood that the maximum annual indoor air temperature reaches 27^{o}C and the minimum reaches 16^{o}C . After the minimum temperature takes place the setpoint changes and brings imidiat results to the apartments temperature. Last, in Figure 41, the diagram for relative humidity sensor and setpoint over time is presented, with data retrieved with the same procedure as before. The maximum relative humidity reaches approximately 80& and a minimum of 10%. The setpoint remains the same throughout the year.

This data discovering and retrieval procedure that located the needed sensors, being the energy, power, air temperature and humidity sensors, as well as the setpoints of the last two sensors for apartment 1 and the energy and power sensor of the air handling unit 1, was an example of querying the Leaf House KG. This example was an easy extraction of the energy profile and internal room conditions for a building’s apartment, without the use of external building models and energy assessment tools. What is more, it was experienced to be an easy code to apply in a KG, in order to access a building’s data. Also, these data are plotted into diagrams, making it easier to understand their usage. The procedure is also quick and can be done by users with different backgrounds, meaning that they do not need a complex query. Furthermore, the knowledge graph has a hierarchical distribution of the buildings entities, making the data more meaningful and useful in a situation of data analysis, like this one.

4.2 Leaf House Energy Assessment

Having analyzed the data discovery and extraction procedure, these data are used to make an energy assessment of the Leaf House building for 2020. In Table 2, the monthly energy consumption for Leaf House is being presented. What is more the annual maximum power demand of the building is 16kW and the annual energy consumption is 32.8MWh or 69.8kWh/m^2. From the same table, it can also be understood that the highest energy consumption in 2020 takes place in July, August and September, which are the months of the year with the highest outdoor temperature in Italy. So, with higher outdoor temperature, there is a greater energy consumption to cover the indoor temperature setpoint, meaning greater HVAC system usage. In Figure 42, the monthly energy consumption for Leaf House in 2020 is being presented in a diagram. The same characteristics with Table 2, are being understood in Figure 42. What is more, it is also understood that January and December have high energy consumption and that is due to the fact that the climate in Italy for these months reaches the lowest outdoor temperatures, creating the need for higher indoor temperature setpoint, in order to keep the apartments warm. This is the reason, why the HVAC systems has an increase in energy consumption these months. The lowest energy consumption values are for the months April to June, which are months that the climate in Italy reaches temperatures in the same range as the indoor temperature setpoints. So the HVAC system is not required to be in great usage. The maximum energy consumption for Leaf House reaches on July at 3.9 MWh or 8.3kWh/m^2 and minimum on

	Energy Consumption (10^3kWh)	**Energy Consumption (kWh/m^2)**
January	3.2	6.8
February	2.2	4.8
March	2.6	5.6
April	2.1	4.4
May	1.7	3.6
June	2.1	4.4
July	3.9	8.3
August	3.9	8.2
September	3.3	7.0
October	2.3	4.9
November	2.4	5.2
December	3.1	6.6
Annual	32.8	69.8

Table 2: Monthly & Annual Power Demand & Energy Consumption of Leaf House for 2020

May at 1.7MWh or 3.6kWh/m^2.

In Figure 43, a pie diagram is being presented, which shows the annual energy consumption of each apartment of Leaf House. It is clear that the biggest energy consumption takes place in apartment 2 with 37% of total apartments' energy consumption. Following with 21% is apartment 6 and with 15% the apartment 3. The lowest percentage takes place in apartment 5 with only 6% of total apartments' energy consumption. The reason why apartment 5 has such a great difference might be the absence of residents in the most part of the year. On the other hand, apartment's 2 high energy consumption might be based on full occupancy and high energy consumptive habits. Apartments 1 and 2 are

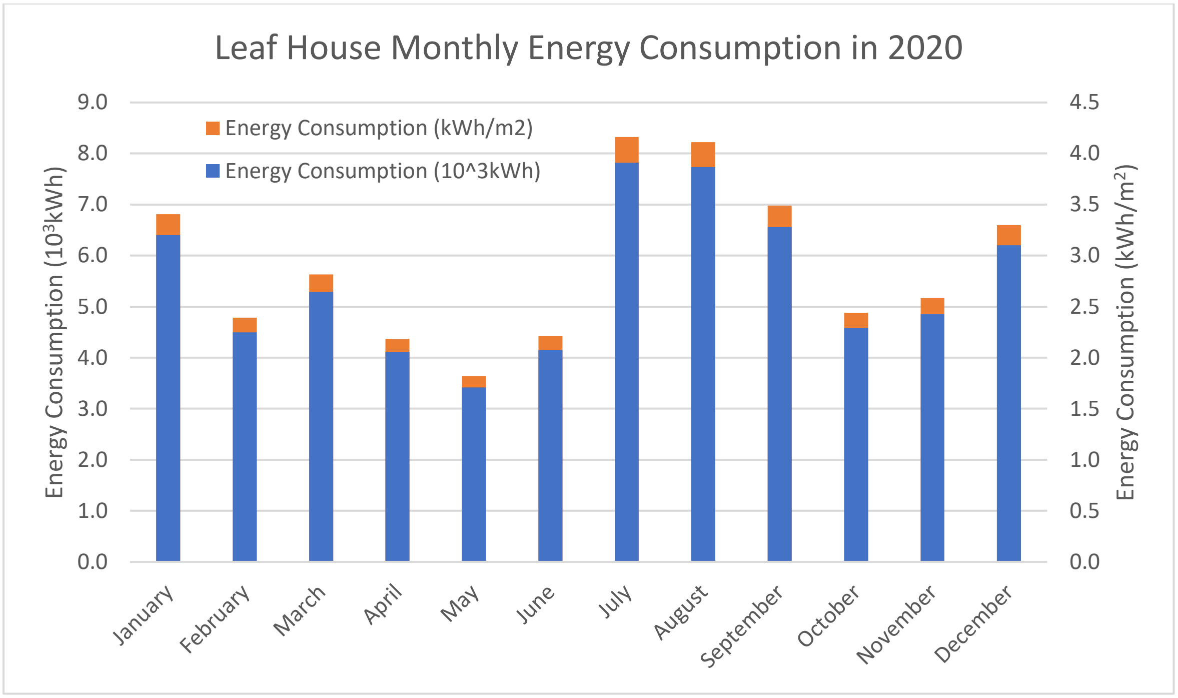


Figure 42: Monthly Energy Consumption for Leaf House in 2020

based on the ground floor of the building, 3 and 4 on the first floor and 5 and 6 on the second floor. Apartment's 6 great energy consumption can be based on the fact that is on the top floor, meaning that is exposed on outdoor climate conditions from more fronts in comparison with the other floors.

In Figure 44, a pie diagram is being shown, which includes the annual energy consumption of the apartments and HVAC system. It is understood that 64% of the Leaf House's energy consumption is due to HVAC system and 36% is due to apartments. Next, the monthly energy production of the photovoltaic panels is presented in Table 3 and in Figure 45, as a diagram. The annual PV energy production is calculated to be 23,018kWh.

Last, the total and total net annual and normalized annual electrical energy consumption have been calculated and presented in Table 4, with total annual electrical energy consumption being 32,819kWh and total net annual electrical energy consumption being 9,801 kWh and total normalized annual electrical energy consumption being69.8kWh/m^2 and total net normalized annual electrical energy consumption being 20.9kWh/m^2.

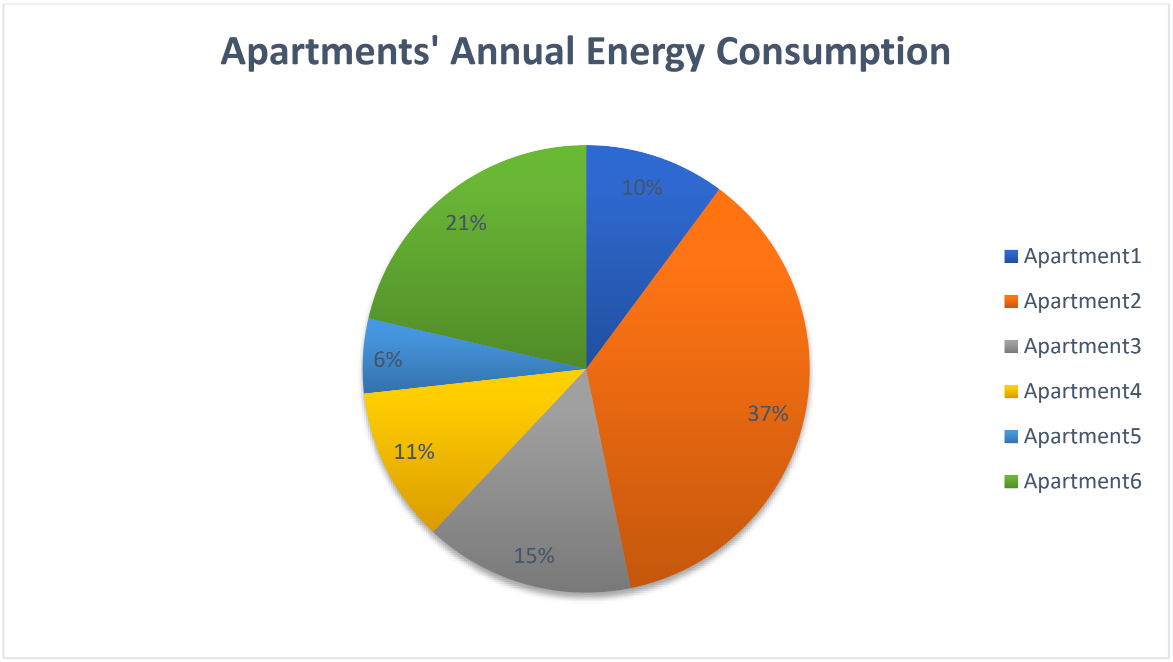


Figure 43: Apartments' Annual Energy Consumption

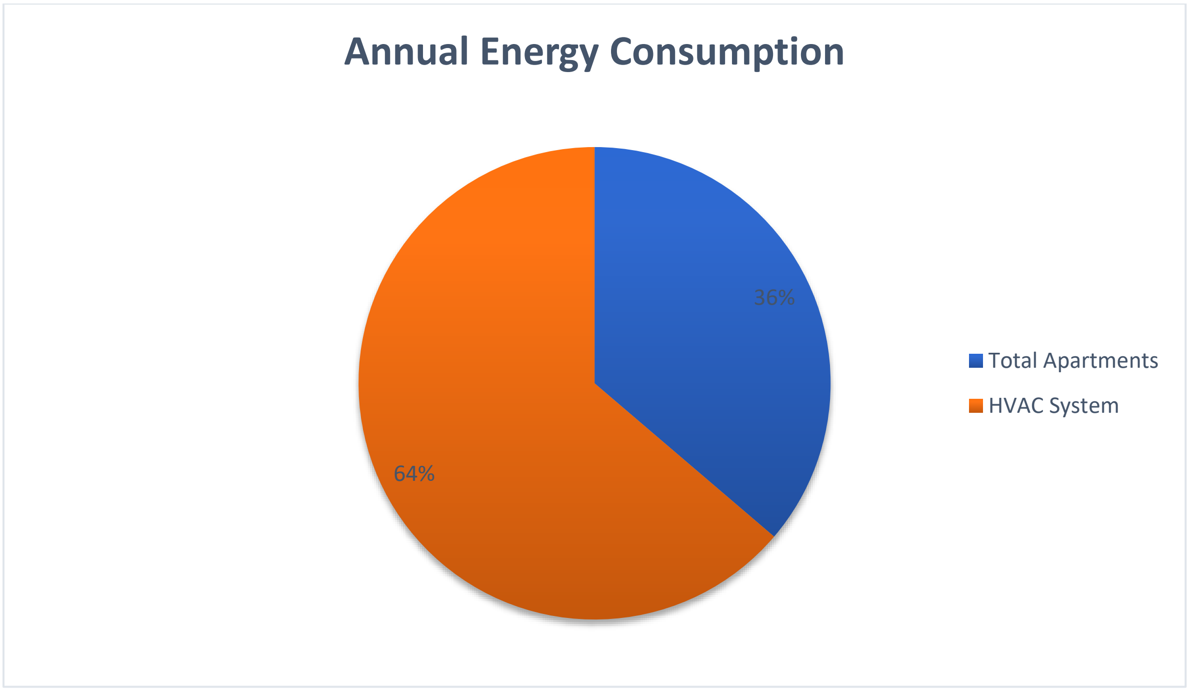


Figure 44: Annual Energy Consumption of Leaf House for 2020

	Energy Production(kWh)
January	1,312
February	1,858
March	1,732
April	2,421
May	2,447
June	2,536
July	2,784
August	2,393
September	2,080
October	1,651
November	1,002
December	8,04
Annual	23,018

Table 3: Leaf House Monthly & Annual PV Energy Production in 2020

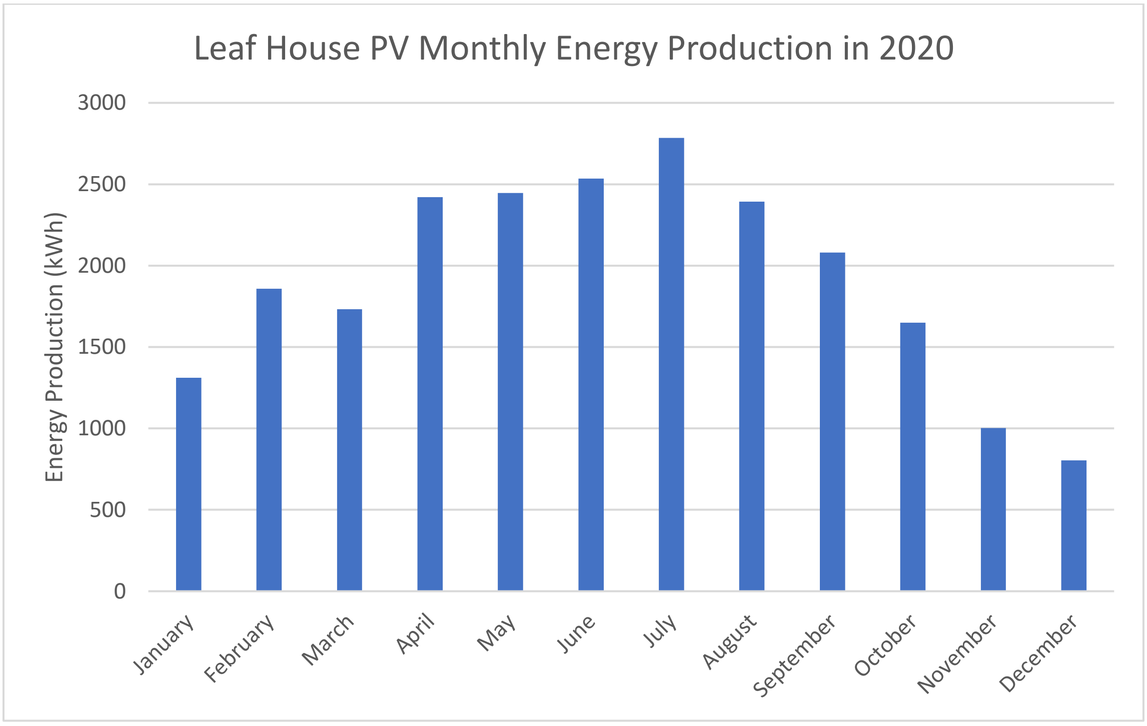


Figure 45: Leaf House Monthly PV Energy Production in 2020

	Total	**Total Net**
Annual Electrical Energy Consumption (kWh)	32,819	9,801
Normalized Annual Electrical Energy Consumption (kWh/m^2)	69.8	20.9

Table 4: Annual & Normalized Annual Energy Consumption of Leaf House in 2020

4.3 Thermal Comfort Index

Urban environment and human discomfort level have been targets of the effects of climate change and urban heat island phenomenon.[112], [113]Urban heat island phenomenon (UHI) is the phenomenon which is created due to the concentration of heat in urban areas, compared to rural areas and is of major importance, based on the fact that high UHI means high heat stress, poor air quality and high energy usage, mostly on hotter days.[114]

The human body's system is affected by the thermal environment, in obedience to the thermodynamic laws.[115] Thermal comfort is defined by ASHRAE as "the condition of mind which expresses satisfaction with the thermal environment", but the support of weather variables is equally important for the thermal comforts' judgement.[116], [117] To calculate the thermal comfort, there have been developed some thermal comfort indexes, which use weather variables, like air temperature and relative humidity.[118]One of these indexes is the Discomfort Index (DI), proposed by Thom (1959) and has been used numerous times.[119] DI is a simple index, which uses the dry bulb temperature (Tdb) and relative humidity (RH%) to calculate the human thermal comfort, as shown in equation 1. In Table 5 the discomfort index range and descriptions are presented and the information that provides is that for DI smaller than 21 there is no discomfort in the room, with DI between 21 and 24 there is discomfort to under 50% of the population, with DI between 24 and 27 over 50% of the population feels discomfort, with DI between 27 and 29 most of the population feels discomfort, with DI between 29 and 32 everyone feels stress and with DI greater than 32 there is a state of medical emergency.

$$DI = T_{db} - (0.55 - 0.005 * RH\%) * (T_{db} - 14.5) \quad (1)$$

DI Range	Description
<21	No Discomfort
21-24	Discomfort Under 50% of Population
24-27	Discomfort Over 50% of Population
27-29	Most of Population feels Discomfort
29-32	Everyone feels Stress
>32	State of Medical Emergency

Table 5: Discomfort Index (DI) Range & Description

So, with the air temperature and relative humidity data that were discovered and extracted from the Leaf House KG, the Discomfort Index was calculated for all six apartments. The DI values, calculated with the monthly average air temperature and relative humidity of the Leaf House six apartments for the year 2020, are being presented in Table 6. The same results are being presented in Figure 48 as a columns diagram. From both Table 4 and Figure 48, it can be understood that every apartment has no discomfort for the months January to April. What is more, from May to September, there is discomfort

Discomfort Index	Apartment1	Apartment2	Apartment3	Apartment4	Apartment5	Apartment6
January	19	19	19	20	19	20
February	19	20	19	20	19	20
March	19	20	20	20	19	20
April	19	20	20	19	19	21
May	21	20	21	21	21	22
June	21	21	21	21	22	23
July	21	22	21	21	22	22
August	22	22	22	20	23	23
September	22	22	21	20	22	22
October	20	21	20	20	20	21
November	19	20	19	19	19	21
December	19	19	20	19	19	20

Table 6: Monthly Average Discomfort Index (DI) for Leaf House's Six Apartments for 2020

under 50% of the population in these apartments, as the DI takes values mostly between 21 and 24. From October and for the rest of 2020 there is no discomfort in the Leaf House apartments. Apartment 1 surpasses <21 no discomfort limit on August and September, Apartment 2 does the same on July-September, Apartment 3 on August, Apartment 4 does not surpass the limit at any month, Apartment 5 from June to September and Apartment 6 from May to September. So Apartment 6 has the biggest time period that the discomfort index has values from 21-24, which means that under 50% of the population feels discomfort.

In Figure 46, the annual air temperature for all six apartments is shown for 2020. Overall, it is understood that for the majority of time, Apartment 6 has the highest temperature in comparison to the other apartments.. The maximum temperatures are reached by Apartments 5 and 6.

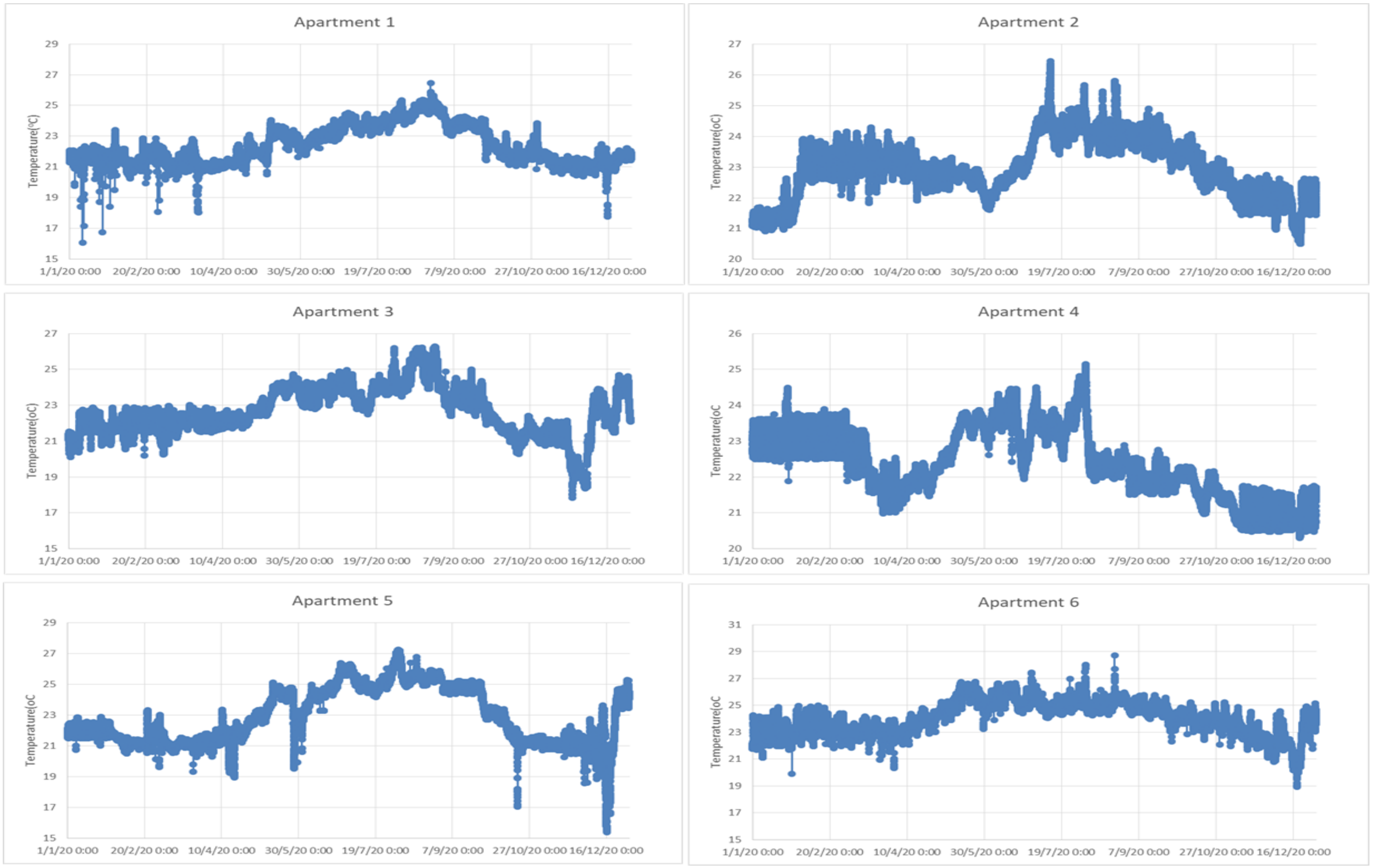


Figure 46: Annual Air Temperature for Leaf House's Apartments in 2020

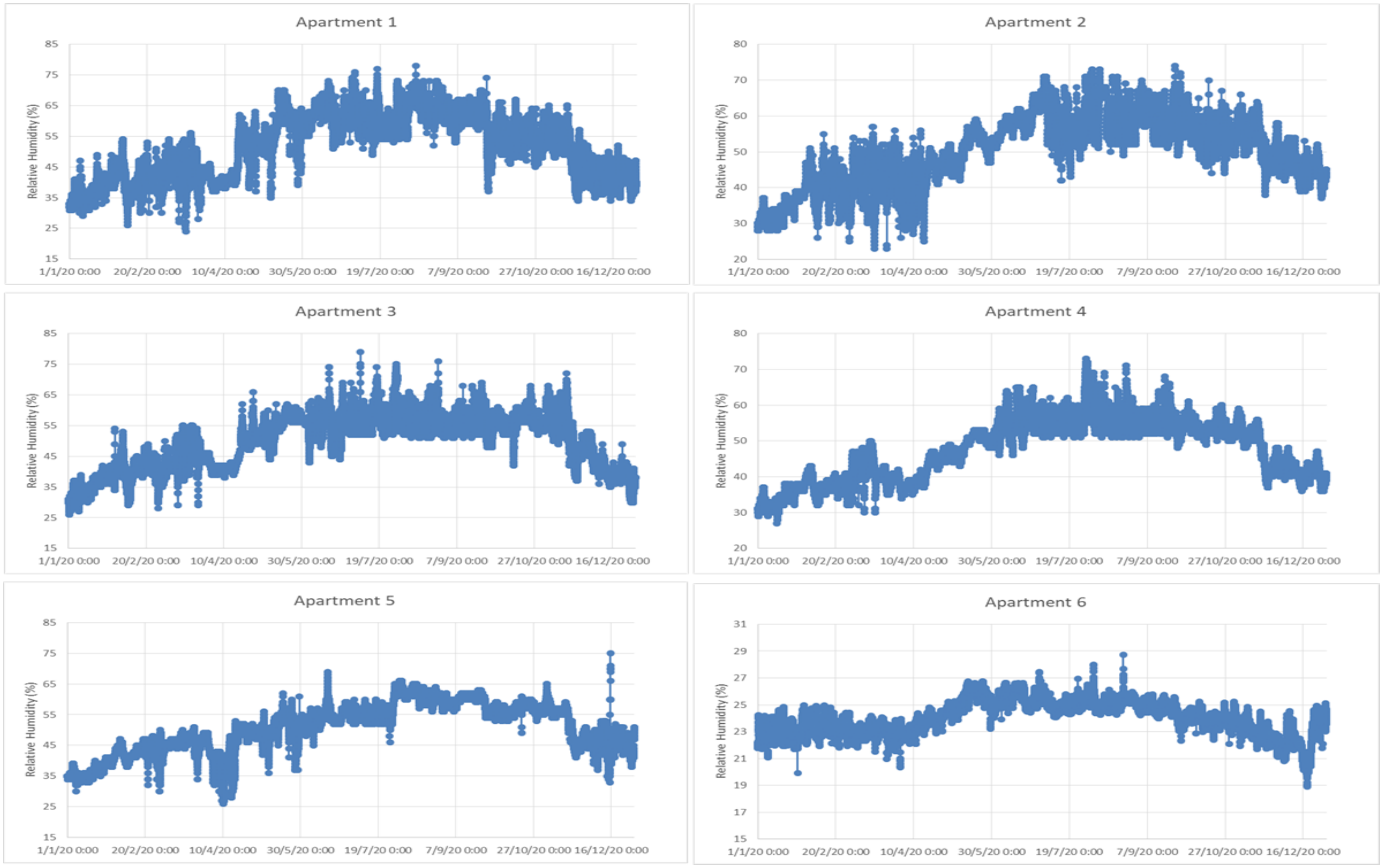


Figure 47: Annual Relative Humidity for Leaf House's Apartments in 2020

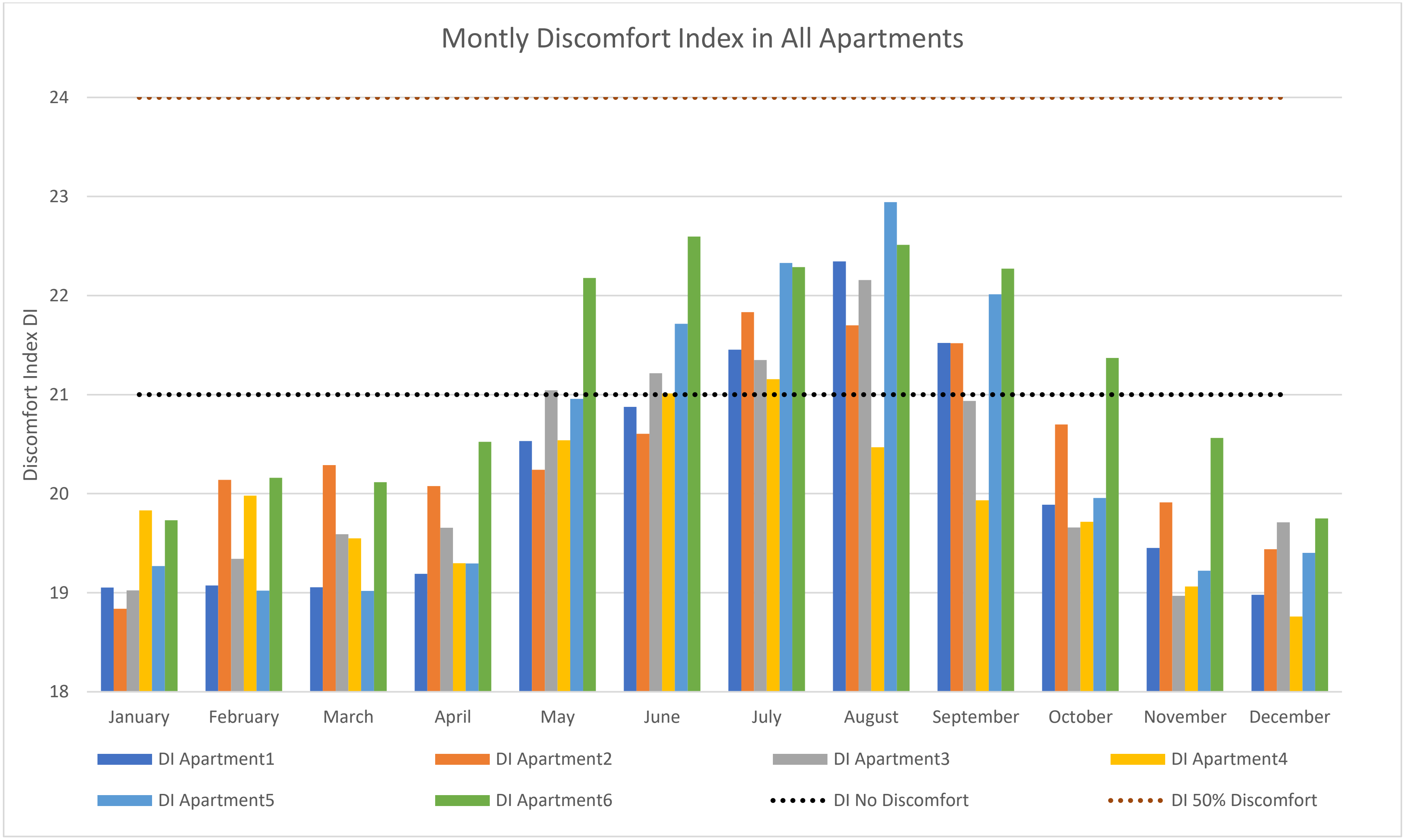


Figure 48: Monthly Average Discomfort Index (DI) for Leaf House's Six Apartments and Range of Discomfort That These Values Hit

In Figure 47, the annual relative humidity for all six apartments is presented for 2020. It is understood that Apartment 4 has the lowest relative humidity in comparison to the rest of the apartments. The maximum relative humidity is reached by Apartments 1 and 6.

The thermal discomfort index showed that there was discomfort under 50% of the population for the summer time period, in most of the apartments. Although, each apartment suffered for a different number of months. In ground floor's apartments 1 and 2, there were two and three months respectfully, where the discomfort was felt by under 50% of the population. In first floor's apartments 3 and 4, there were one and zero months respectfully, where the discomfort was felt by under 50% of the population. Last, in top floor's apartments 5 and 6, there were four and five months respectfully, where the discomfort was felt by under 50% of the population.

A more clear perspective for each apartment is shown in Figure 49, where there are six diagrams, one for each apartment, of DI over time. From these diagrams it can be understood as previous that from the beginning of 2020 till May there is no thermal discomfort in most of the apartments, following by a period of discomfort under 50% of the population from June to September and last the rest of the year continues with no thermal discomfort. However, Apartment 6 clearly has periods of thermal discomfort under 50% of the population in every month of the year. The thermal discomfort index is constantly between the ranges of no discomfort and discomfort to under 50% of the population from the beginning of 2020 till May as well as from October till the end of the year. The period in between there it is clear that there is discomfort to under 50% of the population.

In Table 7 the percentages of DI with range less than 21 and between 21 and 24 for all apartments is presented. It should be noted again that with DI less than 21, there is no thermal discomfort in the room and with DI between 21 and 24 there is thermal discomfort under 50% of the population. So it is understood that in ground floor, apartments 1 and 2, there is thermal discomfort under 50% of the population for 29% of 2020. In first floor, there is there is thermal discomfort under 50% of the population for 14% of 2020 for apartment 3 and 29% of 2020 for apartment 4. Last, for the second floor, there is there is thermal discomfort under 50% of the population for 45% of 2020 for apartment 5 and 39%

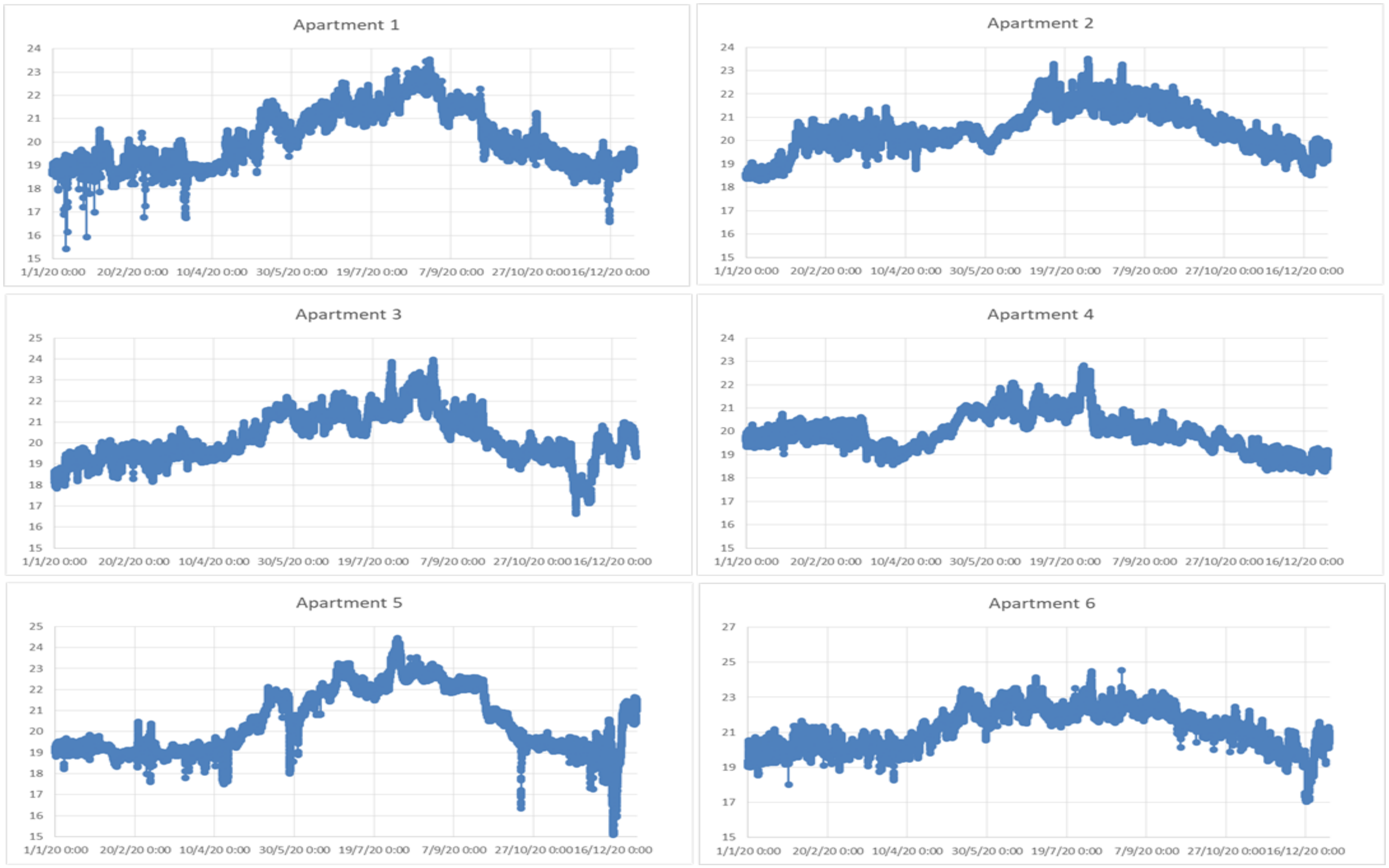


Figure 49: Discomfort Index (DI) over Time for All Apartments for 2020

of 2020 for apartment 6. It is made clear once again that the top floor, which is more exposed to the outdoor conditions, suffers the most time from thermal discomfort. Next, the ground floor suffers for a smaller time period from thermal discomfort. On the contrary to the rest of the storeys, first floor has the shortest time period with thermal discomfort, being one month total for both apartments, giving the idea that the position its holds between the other two storeys, benefits its thermal comfort conditions.

Apartment	DI <21 (%)	DI 21-24 (%)
1	71	29
2	71	29
3	86	14
4	71	29
5	55	45
6	61	39

Table 7: Percentage of Discomfort Index range in all six Apartments

5. Discussion

5.1 Conclusions

In this thesis, the main goal was to create a KG for a nZEB, which was the Leaf House building, that includes many different sensors, and then apply a data discovering and retrieval query on the KG, in order to make a data analysis assessment. This would bring up some conclusions about the use of KG for a building as well as about the example of using a building KG to access data in it and assess them.

The methodology that was followed included first, the research of the state of the art about KGs in nZEB, BIM, AEC-KGs and some uses and applications of them in the Built Environment industry, next the selection of the case study's building being the Leaf House, following the creation of KG of Leaf House, as well as a data discovering and extraction query as an example of KG's usage, continuing to the assessment of these results in a data analysis, in addition to a thermal comfort assessment using the Discomfort Index (DI) and lastly the conclusion of KGs in nZEB.

The case study that was examined, accessed the apartments and air handling units sensors' data and plotted diagrams for the year 2020. It was calculated that the annual energy consumption was 32.8MWh or 69.8kWh/m^2, 64% of which originating from

HVAC system consumption and 36% from the apartments' consumption. Apartment 2 had the highest energy consumption between the apartments, which was responsible for 37% of the total apartments' annual energy consumption. On the other hand, Apartment 5 had the lowest energy consumption between the apartments, which was responsible for only 6% of the total apartments' annual energy consumption, probably due to residents' absence. What is more the annual PV energy production was calculated to be 23,018kWh. So the total net annual electrical energy consumption in 2020 was 9,801 kWh and total net normalized annual electrical energy consumption in 2020 was 20.9kWh/m^2.

The thermal comfort assessment, which examined the Discomfort Index in all six apartments, showed that in summer time period of 2020, 50% of the population felt discomfort. The months which the discomfort lasted varied between the apartments. In ground floor's apartments 1 and 2, there were two and three months respectfully, where the discomfort was felt by under 50% of the population. In first floor's apartments 3 and 4, there were one and zero months respectfully, where the discomfort was felt by under 50% of the population. Last, in top floor's apartments 5 and 6, there were four and five months respectfully, where the discomfort was felt by under 50% of the population. So, the apartments which are on middle storey had the shortest time period with thermal discomfort, in comparison to the bottom and top floor, which had longer time period of thermal discomfort, with top floor having the longest.

The conclusions that come up from the results' assessment for the data discovering and extraction query, are that is a simple and quick procedure that provides the user, who is coming from any background, with data analysis results, without the use of a BIM, and with information about the building.

All things considered, the KG in a nZEB provides a hierarchical representation of the building's entities, which is used as a base for many querying ideas and it nullifies the heterogeneity coming from different building representation models, as well as it gives to data more meaning and usefulness, due to the imported ontologies. Furthermore, it is important to be mentioned that the same KG manages in a single simple way, multiple different databases and sensors, in addition to giving easy access to up-to-date information that the user can discover and extract, according to the current agenda. Continuing, the KG allows users to perform queries and obtain information about the building's entities and data in a common procedure. At last, the KG is used for complex systems correlation and data analysis without BIM usage.

5.2. Future Work

A suggestion of future work based on this thesis' results, could be the enrichment of the Leaf House KG with more of its real entities and information like materials, spatial measurements and renewable energy sources, in order to create a more identical representation of the building. Next, one more proposal is to dynamically link MyLeaf database and the Leaf House KG and to compare the two approaches versus an established set of performance criteria. In this context, creating a KG could be developed and assessed in terms of its advantages when applied in a more complex establishment like a near Zero Energy Neighborhood (nZEN) that consists of different buildings and renewable energy systems connected together as a microgrid. All these ideas can have as a base the information gathered in this thesis, the methodology that was followed, as well as the results that came up, in order to lead into new findings and applications of KGs in near Zero Energy Buildings or Neighborhoods.

Bibliography

[1] "United Nations Environment Programme (UNEP)." [Online]. Available: https://www.unep.org/. [Accessed: 04-Oct-2021].

[2] M. Frontczak, S. Schiavon, J. Goins, E. Arens, H. Zhang, and P. Wargocki, "Quantitative relationships between occupant satisfaction and satisfaction aspects of indoor environmental quality and building design," *Indoor Air*, vol. 22, no. 2, pp. 119–131, 2012.

[3] S. Altomonte and S. Schiavon, "Occupant satisfaction in LEED and non-LEED certified buildings," *Build. Environ.*, vol. 68, pp. 66–76, 2013.

[4] "World Green Building Trends 2016," *Smartmarket Rep.*, 2016.

[5] R. Jia *et al.*, "Design Automation for Smart Building Systems," *IEEE*, 2018.

[6] "DIRECTIVE 2010/31/EU OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 19 May 2010 on the energy performance of buildings," 2010.

[7] M. K. Nematchoua, A. Marie-Reine Nishimwe, and S. Reiter, "Towards nearly zero-energy residential neighbourhoods in the European Union: A case study," *Renew. Sustain. Energy Rev.*, vol. 135, no. November 2019, p. 110198, 2021.

[8] K. Sornes *et al.*, "ZenN Nearly Zero Energy Neighborhoods - Final report on common definition for nZEB renovation," pp. 1–89, 2014.

[9] J. Brozovsky, A. Gustavsen, and N. Gaitani, "Zero emission neighbourhoods and positive energy districts – A state-of-the-art review," *Sustain. Cities Soc.*, vol. 72, no. April, p. 103013, 2021.

[10] F. Katiraei, R. Iravani, N. Hatziargyriou, and A. Dimeas, "Microgrids Management," 2008.

[11] H. Farhangi, "The path of the smart grid," *IEEE Mag.*, 2010.

[12] S. Chun, J. Jung, X. Jin, S. Seo, and K. H. Lee, "Designing an integrated knowledge graph for smart energy services," *J. Supercomput.*, vol. 76, no. 10, pp. 8058–8085, 2020.

[13] "Semantic Web," *W3C*. [Online]. Available: https://www.w3.org/standards/semanticweb/. [Accessed: 11-Aug-2021].

[14] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web," 2001.

[15] H. Sun *et al.*, "Review of Challenges and Research Opportunities for Voltage Control in Smart Grids," *IEEE Trans. Power Syst.*, vol. 34, no. 4, pp. 2790–2801, 2019.

[16] Internatiol Electrotechnical Commision, "Smart Grids." [Online]. Available: https://www.iec.ch. [Accessed: 06-Oct-2021].

[17] L. Wen, K. Zhou, S. Yang, and L. Li, "Compression of smart meter big data: A survey," *Renew. Sustain. Energy Rev.*, vol. 91, no. January 2017, pp. 59–69, 2018.

[18] R. H. Blake and P. Mangiameli, "The effects and interactions of data quality and problem complexity on data mining," *Proc. 2008 Int. Conf. Inf. Qual. ICIQ 2008*, no. November, 2008.

[19] B. Saha and D. Srivastava, "Data quality: The other face of Big Data," *Proc. - Int. Conf.*

Data Eng., pp. 1294–1297, 2014.

[20] J. Wang, X. Wang, C. Ma, and L. Kou, “A survey on the development status and application prospects of knowledge graph in smart grids,” *IET Gener. Transm. Distrib.*, vol. 15, no. 3, pp. 383–407, 2021.

[21] W. Shou, J. Wang, X. Wang, and H. Y. Chong, “A Comparative Review of Building Information Modelling Implementation in Building and Infrastructure Industries,” *Arch. Comput. Methods Eng.*, vol. 22, no. 2, pp. 291–308, 2015.

[22] C. Y. Lee, H. Y. Chong, and X. Wang, “Streamlining Digital Modeling and Building Information Modelling (BIM) Uses for the Oil and Gas Projects,” *Arch. Comput. Methods Eng.*, vol. 25, no. 2, pp. 349–396, 2018.

[23] C. Eastman, P. Teicholz, and R. Sacks, “BIM handbook: a guide to building information modeling for owners, managers, designers, engineers and contractors,” 2011.

[24] A. Borrmann, M. König, C. Koch, and J. Beetz, *Integrating BIM in construction contracts*. 2018.

[25] Z. Pezeshki and S. A. S. Ivari, “Applications of BIM: A Brief Review and Future Outline,” *Arch. Comput. Methods Eng.*, vol. 25, no. 2, pp. 273–312, 2018.

[26] C. Boton, L. Rivest, and O. Ghnaya, “What is at the Root of Construction 4.0: a systematic review of the recent research effort,” 2020.

[27] L. Shen and D. Chua, “Application of building information modeling (BIM) and information technology (IT) for project collaboration,” 2011.

[28] Z. Z. Hu, S. Leng, J. R. Lin, S. W. Li, and Y. Q. Xiao, “Knowledge Extraction and Discovery Based on BIM: A Critical Review and Future Directions,” *Arch. Comput. Methods Eng.*, no. 0123456789, 2021.

[29] A. Deshpande, S. Azhar, and S. Amireddy, “A framework for a BIM-based knowledge management system,” *Procedia Eng.*, vol. 85, pp. 113–122, 2014.

[30] U. Isikdag, G. Aouad, J. Underwood, and S. Wu, “Building information models: a review on storage and exchange mechanisms,” *Int. Conf. IT Constr.*, pp. 135–143, 2007.

[31] M. H. Rasmussen, M. Lefrançois, P. Pauwels, C. A. Hviid, and J. Karlshøj, “Managing interrelated project information in AEC Knowledge Graphs,” *Autom. Constr.*, vol. 108, no. August, p. 102956, 2019.

[32] A. Kiviniemi, “Requirements Management Interface to Building Product Models,” 2005.

[33] H. P. Tserng and Y.-C. Lin, “Developing an activity-based knowledge management system for contractors,” 2004.

[34] R. Santos, A. A. Costa, and A. Grilo, “Bibliometric analysis and review of Building Information Modelling literature published between 2005 and 2015,” *Autom. Constr.*, vol. 80, pp. 118–136, 2017.

[35] R. Studer, V. R. Benjamins, and D. Fensel, “Knowledge Engineering: Principles and methods,” *Data Knowl. Eng.*, vol. 25, no. 1–2, pp. 161–197, 1998.

[36] B. F. Lóscio, C. Burle, and N. Calegari, “Data on the Web Best Practices,” *W3C recommendation*, 2012. [Online]. Available: https://www.w3.org/TR/2017/REC-dwbp-20170131/. [Accessed: 11-Aug-2021].

[37] L. Ehrlinger and W. Wolfram, “Towards a Definition of Knowledge Graphs,” 2016.

[38] P. Bonatti, S. Decker, A. Polleres, and V. Presutti, “Knowledge Graphs: New Directions for Knowledge Representation on the Semantic Web (Dagstuhl Seminar 18371),” *Dagstuhl Reports*, vol. 8, no. 9, pp. 29–111, 2019.

[39] Heiko Paulheim, “Knowledge Graph Refinement: A Survey of Approaches and Evaluation Methods,” *Semant. Web*, vol. 8, no. 3, pp. 489–508, 2016.

[40] C. Gutierrez and J. F. Sequeda, “Knowledge graphs,” *Commun. ACM*, vol. 64, no. 3, pp. 96–104, 2021.

[41] E. W. Schneider, “Course Modularization Applied: Thr Interface System and Its Implications For Sequence Control and Data Analysis,” 1973.

[42] A. Singhal, “Introducing the Knowledge Graph: things, not strings,” *Google Blog*, 2012.

[43] N. Noy, Y. Gao, A. Jain, A. Patterson, A. Narayanan, and J. Taylor, “Industry-scale knowledge graphs lessons and challenges,” *Queue*, vol. 17, no. 2, pp. 1–28, 2019.

[44] R. Angles and C. Gutierrez, “Survey of graph database models,” *ACM Comput. Surv.*, vol. 40, no. 1, pp. 1–39, 2008.

[45] S. Abiteboul, “Querying Semi-Structured Data,” 1997.

[46] I. Balaževic, C. Allen, and T. Hospedales, “Multi-relational poincaré graph embeddings,” *Adv. Neural Inf. Process. Syst.*, vol. 32, no. NeurIPS, pp. 1–13, 2019.

[47] A. Bordes *et al.*, “Translating Embeddings for Modeling Multi-relational Data,” 2013.

[48] R. Cyganiak, D. Wood, and M. Lanthaler, “RDF 1.1 Concepts and Abstract Syntax, W3C Recommendation,” 2014.

[49] S. Martin, Dürst; Michel, “Internationalized Resource Identifiers (IRIs),” 2005.

[50] R. Hussein, D. Yang, and P. Cudr -Mauroux, “Are meta-paths necessary? Revisiting heterogeneous graph embeddings,” *Int. Conf. Inf. Knowl. Manag. Proc.*, pp. 437–446, 2018.

[51] L. Yang, Z. Xiao, W. Jiang, Y. Wei, Y. Hu, and H. Wang, “Dynamic heterogeneous graph embedding using hierarchical attentions,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 12036 LNCS, pp. 425–432, 2020.

[52] R. Angles, M. Arenas, P. Barceló, A. Hogan, J. Reutter, and D. Vrgoč, “Foundations of modern query languages for graph databases,” *ACM Comput. Surv.*, vol. 50, no. 5, 2017.

[53] S. Harris, A. Seaborne, and E. Prud’hommeaux, “SPARQL 1.1 Query Language, W3C Recommendation,” 2013.

[54] D. Brickley and R. V. Guha, “RDF Schema 1.1, W3C Recommendation,” 2014.

[55] P. Hitzler, M. Krötzsch, B. Parsia, Patel-Schneider, P. F., and S. Rudolph, “OWL 2 Web Ontology Language Primer (Second Edition),” 2012.

[56] J. E. L. Gayo, E. Prud’hommeaux, I. Boneva, and D. Kontokostas, “Validating RDF Data. Synthesis,” 2017.

[57] H. Knublauch and D. Kontokostas, “Shapes Constraint Language (SHACL),” 2017.

[58] M. D. Pham, L. Passing, O. Erling, and P. Boncz, “Deriving an emergent relational schema from RDF data,” *WWW 2015 - Proc. 24th Int. Conf. World Wide Web*, pp. 864–874, 2015.

[59] S. Cebiric *et al.*, “Summarizing Semantic Graphs : A Survey To cite this version : HAL Id : hal-01925496 Summarizing Semantic Graphs : A Survey,” 2018.

[60] J. Hakala, “Persistent identifiers - an overview,” 2010.

[61] D. Peterson, S. Gao, A. Malhotra, C. M. Sperberg-McQueen, H. S. Thompson, and P. V. Biron, “W3C XML Schema Definition Language (XSD) 1.1 Part 2: Datatypes,” 2012.

[62] R. V. Guha, R. McCool, and R. Fikes, “Contexts for the Semantic Web,” 2004.

[63] P. Hitzler, M. Krötzsch, and S. Rudolph, *Foundations of Semantic Web Technologies*. 2010.

[64] E. E. S. Morgan-kaufmann, “A Framework for Representing Knowledge,” no. 1972, pp. 1–116, 1974.

[65] M. R. Quillian, “A notation for representing conceptual information An application to semantics and mechanical English paraphrasing,” 1963.

[66] F. Baader, I. Horrocks, C. Lutz, and U. Sattler, “An Introduction to Description Logis,” 2017.

[67] E. Estrada, “The Structure of Complex Networks: Theory and Applications,” 2011.

[68] H. Paulheim, “Knowledge graph refinement: A survey of approaches and evaluation methods,” 2017.

[69] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, “A Comprehensive Survey on Graph Neural Networks,” *IEEE Trans. Neural Networks Learn. Syst.*, vol. 32, no. 1, pp. 4–24, 2021.

[70] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, “The Graph Neural Network Model,” 2009.

[71] N. Park, A. Kan, X. L. Dong, T. Zhao, and C. Faloutsos, “MultiImport: Inferring Node Importance in a Knowledge Graph from Multiple Input Signals,” *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, pp. 503–512, 2020.

[72] N. Park, A. Kan, X. L. Dong, T. Zhao, and C. Faloutsos, “Estimating node importance in knowledge graphs using graph neural networks,” *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, pp. 596–606, 2019.

[73] Q. He, B.-C. Chen, and D. Agarwal, “Building The LinkedIn Knowledge Graph,” 2016.

[74] H. Paulheim, “How much is a triple? Estimating the cost of knowledge graph creation,”

CEUR Workshop Proc., vol. 2180, pp. 5–8, 2018.

[75] D. Vrandečić and M. Krötzsch, “Wikidata: A free collaborative knowledgebase,” *Commun. ACM*, vol. 57, no. 10, pp. 78–85, 2014.

[76] C. Lockard, X. L. Dong, A. Einolghozati, and P. Shiralkar, “CERES: Distantly supervised relation extraction from the semi structured web,” *Proc. VLDB Endow.*, vol. 11, no. 10, pp. 1084–1096, 2018.

[77] J. L. Martínez-Rodríguez, A. Hogan, and I. Lopez-Arevalo, “Information Extraction meets the Semantic Web: A Survey,” 2020.

[78] C. Lu, L. Bing, W. Lam, K. Chan, and Y. Gu, “Web Entity Detection for Semi-structured Text Data Records with Unlabeled Data,” vol. 4, no. 2, pp. 135–150, 2013.

[79] R. Baeza-Yates, “Bias on the web,” *Commun. ACM*, vol. 61, no. 6, pp. 54–61, 2018.

[80] A. Hogan, J. Umbrich, A. Harth, R. Cyganiak, A. Polleres, and S. Decker, “An empirical survey of Linked Data conformance,” *J. Web Semant.*, vol. 14, pp. 14–44, 2012.

[81] A. Zaveri, A. Rula, A. Maurino, R. Pietrobon, J. Lehmann, and S. Auer, “Quality assessment for Linked Data: A Survey,” *Semant. Web*, vol. 7, no. 1, pp. 63–93, 2016.

[82] R. West, E. Gabrilovich, K. Murphy, S. Sun, R. Gupta, and D. Lin, “Knowledge Base Completion via Search-Based Question Answering,” 2014.

[83] L. Getoor and B. Taskar, “Introduction to Statistical Relational Learning,” 2007.

[84] M. Niknam and S. Karshenas, “A shared ontology approach to semantic representation of BIM data,” *Autom. Constr.*, vol. 80, pp. 22–36, 2017.

[85] “Standard Classification for Building Elements and Related Sitework- UNIFORMAT II,” *ASTM standard*, 2015. .

[86] “Naming and Addressing: URIs,” *W3C*, 2005. .

[87] “QUDT.” [Online]. Available: http://qudt.org/.

[88] M. H. Rasmussen, “Proposing a Central AEC Ontology that Allows for Domain Specific Extensions,” no. July, pp. 237–244, 2017.

[89] M. H. Rasmussen, M. Lefrançois, P. Pauwels, C. A. Hviid, and J. Karlshøj, “Managing interrelated project information in AEC Knowledge Graphs,” *Autom. Constr.*, vol. 108, no. January, 2019.

[90] M. H. Rasmussen, M. Lefrançois, M. Bonduel, C. A. Hviid, and J. Karlshø, “OPM: An ontology for describing properties that evolve over time,” *CEUR Workshop Proc.*, vol. 2159, pp. 23–33, 2018.

[91] M. Lefrançois, “Planned ETSI SAREF extensions based on the W3C&OGC SOSA/SSNcompatible SEAS ontology patterns,” 2017.

[92] ISO10303-11, “Industrial Automation Systems and Integration - Product Data Representation and Exchange - Part 11: Description Methods: The EXPRESS Language Reference Manual,” *Stand. Int. Organ. Stand.*, 2004.

[93] E. Prud’hommeaux, G. Carothers, D. Beckett, and T. Berners-Lee, “RDF 1.1 Turtle - Terse RDF Triple Language,” 2014. [Online]. Available: http://www.w3.org/TR/2014/REC-turtle-20140225/. [Accessed: 13-Aug-2021].

[94] T. Lebo *et al.*, “PROV-O: The PROV Ontology,” 2013. [Online]. Available: https://www.w3.org/TR/prov-o/. [Accessed: 13-Aug-2021].

[95] B. Balaji *et al.*, “Brick: Towards a unified metadata schema for buildings,” *Proc. 3rd ACM Conf. Syst. Energy-Efficient Built Environ. BuildSys 2016*, pp. 41–50, 2016.

[96] B. Balaji *et al.*, “Brick: Metadata schema for portable smart building applications,” *Appl. Energy*, vol. 226, no. September 2017, pp. 1273–1292, 2018.

[97] “Project Haystack.” [Online]. Available: http://project-haystack.org/. [Accessed: 14-Aug-2021].

[98] “BrickSchema.” [Online]. Available: https://brickschema.org/. [Accessed: 15-Aug-2021].

[99] N. Kampelis *et al.*, “Evaluation of the performance gap in industrial, residential & tertiary near-Zero energy buildings,” *Energy Build.*, vol. 148, pp. 58–73, 2017.

[100] G. Comodi *et al.*, “Multi-apartment residential microgrid with electrical and thermal storage devices: Experimental analysis and simulation of energy management strategies,” *Appl. Energy*, vol. 137, pp. 854–866, 2015.

[101] “My Leaf.” [Online]. Available: https://myleaf2.loccioni.com/beta/Dashboard. [Accessed: 18-Aug-2021].

[102] “Project Jupyter.” [Online]. Available: https://jupyter.org/. [Accessed: 18-Sep-2021].

[103] “Web Ontology Language (OWL).” [Online]. Available: https://www.w3.org/OWL/. [Accessed: 19-Sep-2021].

[104] “Resource Description Framework (RDF).” [Online]. Available: https://www.w3.org/2001/sw/wiki/RDF. [Accessed: 19-Sep-2021].

[105] “RDF Vocabulary Description Language 1.0: RDF Schema (RDFS).” [Online]. Available: https://www.w3.org/2001/sw/wiki/RDFS. [Accessed: 19-Sep-2021].

[106] “rdflib 6.0.1.” [Online]. Available: https://rdflib.readthedocs.io/en/stable/index.html#. [Accessed: 19-Sep-2021].

[107] “A Universally Unique IDentifier (UUID) URN Namespace.” [Online]. Available: https://datatracker.ietf.org/doc/html/rfc4122. [Accessed: 20-Sep-2021].

[108] “TimescaleDB.” [Online]. Available: https://www.timescale.com/. [Accessed: 20-Sep-2021].

[109] “PostgreSQL.” [Online]. Available: https://www.postgresql.org/. [Accessed: 20-Sep-2021].

[110] “pandas.” [Online]. Available: https://pandas.pydata.org/. [Accessed: 21-Sep-2021].

[111] “psycopg2.” [Online]. Available: https://pypi.org/project/psycopg2/. [Accessed: 21-Sep-2021].

[112] D. L. Zhang, Y. X. Shou, R. R. Dickerson, and F. Chen, “Impact of upstream urbanization on the urban heat island effects along the Washington-Baltimore Corridor,” *J. Appl. Meteorol. Climatol.*, vol. 50, no. 10, pp. 2012–2029, 2011.

[113] A. N. Kakon, M. Nobuo, S. Kojima, and T. Yoko, “Assessment of thermal comfort in respect to building height in a high-density city in the tropics,” 2012.

[114] M. Kolokotroni and R. Giridharan, “Urban heat island intensity in London: An investigation of the impact of physical characteristics on changes in outdoor air temperature during summer,” *Sol. Energy*, vol. 82, no. 11, pp. 986–998, 2008.

[115] C. A. Njoku and M. T. Daramola, “Human Outdoor Thermal Comfort Assessment in a Tropical Region: A Case Study,” 2019.

[116] N. Djongyang, R. Tchinda, and D. Njomo, “Thermal comfort: A review paper,” *Renew. Sustain. Energy Rev.*, vol. 14, no. 9, pp. 2626–2640, 2010.

[117] American Society of Heating Refrigerating Air-Conditioning Engineers, “Thermal Environmental Conditions for Human Occupancy.,” *Third Public Rev.*, 2003.

[118] Y. Epstein and D. S. Moran, “Thermal comfort and the heat stress indices,” *Ind. Health*, vol. 44, no. 3, pp. 388–398, 2006.

[119] E. C. Thom, “The Discomfort Index,” 1959.