DATA ONTOLOGY ASSESSMENT TO SUPPORT SMART AND ONGOING COMMISSIONING OF BUILDINGS

by

Caroline F. Quinn

BSc. Computer Science, Mount Allison University, 2016

A thesis

presented to Ryerson University

in partial fulfillment of the requirements for the degree of

Master of Applied Science

in the program of

Building Science

Toronto, Ontario, Canada, 2020

AUTHOR'S DECLARATION FOR ELECTRONIC SUBMISSION OF A THESIS

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I authorize Ryerson University to lend this thesis to other institutions or individuals for the purpose of scholarly research.

I further authorize Ryerson University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

I understand that my thesis may be made electronically available to the public.

DATA ONTOLOGY ASSESSMENT TO SUPPORT SMART AND ONGOING COMMISSIONING OF BUILDINGS

Caroline Quinn

Master of Applied Science in Building Science, Ryerson University, 2020

Abstract

To achieve Canada's GHG reduction targets, building performance must be improved. Enabling buildings with Smart and Ongoing Commissioning (SOCx) applications will help to achieve peak performance in energy use and improved occupant health and comfort, at minimum cost. A comprehensive literature review highlighted the viability of Brick and Project Haystack ontologies, prompting a quantitative comparison of completeness and expressiveness using a case study with an industry ontology as the baseline for comparison. Additionally, a qualitative comparison was completed using key ontology qualities outlined in literature. A recommendation of Brick is made based on results. Brick achieved higher assessment values in completeness and expressiveness achieving 59% and 100% respectively, as compared to Haystacks 43% and 96%. Additionally, Brick exhibited five of six desirable qualities, where Haystack exhibited only three. If used by SOCx applications, the appropriate ontology permits the optimization of building performance. The recommendation of the appropriate ontology forms the basis for longer- term SOCx prototype development, which will support innovative approaches to sustainability in building operations across scale, as well as next- generation building controls and automation strategies.

Acknowledgments

I am fortunate to have a myriad of supportive people in my life and would like to acknowledge their role in strengthening my academic ability.

I would also like to thank my supervisor Jenn McArthur for providing a learning opportunity beyond expectation, and my internship manager Oskar Nilsson for encouraging my study of this important topic.

Table of Contents

List of Tables

List of Figures

1 Introduction

Climate change, and subsequently energy use reduction, has become a global focus since emerging as a concern in the late 1980s [1]. The building sector is not isolated from this phenomenon, where stakeholders involved in any phase of a buildings lifecycle must meet expectations of helping to reduce energy use. At the beginning of the operational phase of a building, HVAC systems are commissioned to ensure optimal performance and efficiency. Over time, this efficiency decreases due to changes in controls and equipment failures. These issues are typically only rectified a handful of times throughout a building's operational life by recommissioning – the process of systematically verifying and recalibrating HVAC system operations.

While recommissioning offers the opportunity to restore HVAC energy efficiency to occupants, there is a missed energy savings opportunity in the time between recommissioning activities. An ongoing commissioning process would be arduous and overly expensive to execute manually, hence Smart Building applications have been developed. These applications use data generated by building information systems, allowing for the scalable oversight required for the ongoing commissioning of a building. This functionality is provided by a suite of applications and is referred to as Smart and Ongoing commissioning (SOCx). Purpose driven applications such as equipment fault detection and energy optimization of controls, or information access driven tools such as parametric designs and Facilities Management Building Information Models (FM-BIMs) fall within the SOCx domain. SOCx is of value to facilities management professionals to test and plan maintenance projects, observe building conditions over time, retrieve specific sensor data using the visual interface of the FM-BIM, optimize controls strategies, and other interventions to improve building performance.

Information required for SOCx comes from a variety of heterogeneous data sources, including: Building Management System (BMS), Building Information Model (BIM), security, weather, third party IoT such as voice command systems (ex: Amazon Alexa), maintenance records, etc. Data sources such as BMS data cannot easily be cross referenced with a Building Information Model (BIM) for example. This disparate nature of building data is the major hurdle to the

adoption of SOCx in industry and is therefore of interest to the academic community. Normalization of data is possible though the creation of semantic data, metadata which gives meaning to, and is kept separate from, raw domain data. This semantic data allows for cross referencing and the creation of an interconnected web of datasets called Linked Data. Linked Data sets can be systematically accessed by SOCx and other applications through a query processor (ex: SPARQL, SQL). Semantic data of a specific domain, in this case building HVAC system data, is defined using a data ontology.

An ontology is a formalization of semantic data, where an information schema made up of elements such as tags, classes, or nodes and relationships are used to define the structure of semantic domain knowledge. The term "ontology" is not only theoretical; it is also used to reference an instantiated version of an ontology, referred to as Tbox and Abox components of a knowledge base respectively. For example, the Brick ontology can be used to define a residential condominium's building ontology. The former [Brick] is an ontology as previously defined and is the Tbox component; the latter [Ryerson Architectural Science building] is an instantiated ontology describing the semantic data of a building in a file and is the Abox component. An analogy: Semantic data is all the words in a dictionary that you could describe something with, an ontology (Tbox) is the sentence structure, and an instantiated ontology (Abox) is the sentence. Depending on the ontology used, source data concepts are defined with one or multiple ontology nodes or tags and relationships. While many ontologies have been proposed to define building HVAC system data in Linked Data structures two have gained traction in academia and industry: Project Haystack (henceforth referred to as "Haystack") and Brick.

The paucity of research in the comparison of Brick and Project Haystack is pertinent to industry and academia because the desire to move forward with Smart Building applications requires an ontology to give context to building data sources and make them usable. This separation between semantic contextual data and timeseries data collected at a source is necessary for the modularization of development and implementation of SOCx applications. Modularization will allow for applications to be supplied by a variety of vendors, as well as updated in order to meet the needs of system autonomy, maintenance tracking, human system interaction, reporting, *etc.* A challenge in specifying the Linked Data approach for all applications is the consideration for

both semantic (static) and timeseries (dynamic) data modalities [2]. The Linked Data approach must be able to support the high frequency and large volume of data point writes from a BMS, and complex queries for application use. The recommendation of an appropriate ontology will allow SOCx application research and development to move forward with more efficiency.

1.1 Research Objective

The objective of this research is to investigate and recommend an ontology which serves as a conduit, feeding SOCx applications with building data. The investigation evaluates two ontologies (Brick and Haystack) for their suitability to semantically represent SOCx data.

1.2 Research Questions

1. Is the Brick or Project Haystack ontology better suited to **describe** semantic building data for SOCx applications (completeness)
2. Is the Brick or Project Haystack ontology better suited to **relate** semantic building data for SOCx applications? (expressiveness)
3. How do Brick and Project Haystack compare when assessed qualitatively based on flexibility, portability, readability, extensibility, interoperability, and queryability

2 Literature Review

Research questions are framed by the context of Brick and Haystack emerging as academic and industry prevalent ontologies respectively. The literature review offers a detailed background on the role of ontology within the Linked Data approach. It additionally describes ontologies previously developed within the SOCx context. Finally, it discusses the two most relevant ontologies, Brick and Haystack, in detail.

2.1 Background

The challenge in integrating and accessing building data sources lies in their heterogeneous nature. The most common integration technique for building data sources is referred to as *Linked Data* [3, 4, 5]. This approach typically stores building data sources separately and relies on a standardized serialization of semantic data in order to cross reference data sources and access timeseries data (sensor, actuator, meter, etc.). Different serialization formats to define semantic data exist and within the building domain semantic web technologies have proven to offer a variety of effective formats [2]. Some examples of semantic web serialization formats include Turtle (.ttl), N-Triples (. nt), JSON-LD (.jsnld), RDF/XML (.rdf), among others. Each of these serialization formats rely on the use of ontologies to define semantic data using the Resource Description Framework (RDF). Alternatively, serialization of ontology semantic data can be done in a custom format that suits the ontology. Either of these approaches is referred to as an *ontology linked* approach in this research, where an ontology is a prescribed way to define semantic data, it is a set of formal modeling rules defining how semantic data can exist. An alternative Linked Data method relies on a standard naming convention to define semantic data and uses a Uniform Resource Identifier (URI) approach to a define semantic data serialization format [6], this is referred to as a *directly linked* approach.

The *directly linked* approach has been explored within the building domain, and the Construction-Operations Building information exchange protocol (COBie) [7] was developed to act as a standard naming format to facilitate the integration of Computerized Maintenance Management System data to BIM. Research aiming to leverage this format to automatically create these links and circumvent manual work had limited success due to interoperability issues with real world application [8]. Despite this limitation, there have been industry examples where

systems conforming to a parsable and interpretable naming convention enable direct linking of building data sources. For example, the taxonomy present in a data point unique identifier can indicate the space, system, and equipment a data point is related to within the BIM [6]. This directly Linked Data architecture requires no manual implementation of an ontology; instead, the integration between data sources define the resultant semantic data.

The *ontology linked* approach relying on RDF serialization requires the use of triples to represent semantic data, where triples are comprised of a *subject, predicate,* and *object* [9]. Subject and objects are typically unique identifiers/names used to cross reference semantic data to dynamic data, or contextual information as defined by the ontology. Predicate is a relationship linking the subject and object [10]. The ontology linked approach relying on a custom serialization format can use any method to translate semantic ontology data to a digital format that can be stored, transmitted, and accessed. The *directly linked* approach relying on URI requires standard machine passable syntax of both static and dynamic data identifiers in all data sources to act as the semantic serialization format. Both Linked Data approaches allow data sources to be cross referenced using a data management system, accessing dynamic data by first querying the semantic data for context.

Both Linked Data approaches effectively create a data lake that is accessible through a common data management system [11]. In the case of ontology Linked Data, where the semantic data and timeseries data are decoupled; a query processor is used as a data management system [12, 13]. In general, a query processor will receive a request from an application, query semantic data to retrieve pointers/unique identifiers to data points within timeseries data sources such as a BMS, and then query these stores for specific data values. The data management system relies on a standard serialization of semantic data in order to bridge the gap from applications to relevant data sources. If standard serialization is not used for all data sources, a query processor would need to possess logic to convert semantic definitions from one serialization format to another. Custom query methods can act as a data management system and are required when using the directly linked approach relying on custom serialization.

The direct comparison of the latest versions of Brick and Haystack, respectively academic and industry leading ontologies, has not been completed for the purpose of SOCx. Brick has been designed to be extendible and flexible while maintain a set of semantic rules to allow for consistency in ontology definition. Haystack has also been designed to allow for extensibility for semi-structured definition of ontology and is therefore more flexible. The comparison of these ontologies for the purpose of SOCx is cumbersome as both completeness (ability to represent all relevant metadata precisely including location, type, units, etc.) and expressiveness (ability to capture relationships as found between BMS systems, and as required by SOCx) need to be assessed. Previous versions of Brick and Haystack have been compared [9], and it was found the Brick was superior, however both ontologies have been updated significantly since this comparison. Project Haystack has grown from a standardized schema to an ontology, and a new comparison for the purpose of SOCx is required.

2.2 SOCx ontologies

The ontology linked approach has been widely researched because of the strength of semantic web technologies [10, 9, 12, 2]. This Section offers a systematic review of ontologies which have been developed for Linked Data purposes aligning with SOCx. This section will review data types represented in Tbox SOCx relevant ontologies and discuss proposed ontologies, highlighting the prominent ontologies relating to SOCx: Brick and Haystack, as found in the literature.

2.2.1 Ontology Review Methodology

A systematic literature review was used to become familiarized with the state of the art in data ontology for the purpose of SOCx. Publications were selected using the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) methodology [14]. This methodology requires a systematic narrowing of relevant publications given a set of input search terms and databases through four main phases: identification, screening, eligibility, and included. Figure 1 demonstrates the number of publications considered at each phase of the PRISMA filtering process.

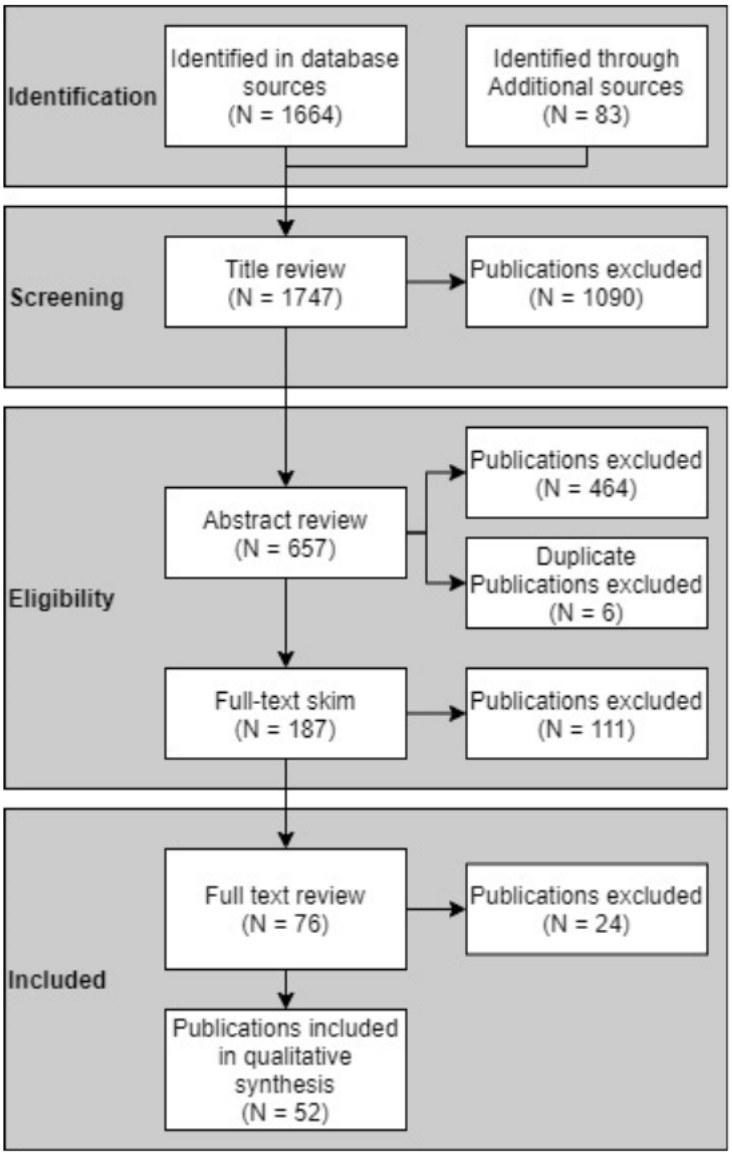


Figure 1 : PRISMA Process Flow to Discover SOCx Related Ontology Papers

To ensure that the topic was sufficiently covered, primary search terms and synonyms were used. The primary search terms identified were ontology, building, and commissioning. Synonym terms used for ontology included: framework and information model, and for commissioning included: commission, energy management, and facility management. The following logic was used to generate the complete search term: {ontology AND building AND {commission OR energy management OR facility management}} OR {framework AND building AND {commission OR energy management OR facility management}}, OR {information model AND

building AND {commission OR energy management OR facility management}}. In January 2019, the search term was entered into the six databases: ACM [15], IEEE [16], Inderscience [17], SciX [18], Scopus [19], and Taylor and Francis [20]. Several other publications found in the initial topic search were also included in a category titled “Additional” in the identification phase of the PRISMA methodology.

In the identification phase, any publication meeting the criteria of: (1) containing search terms in the title, abstract, or keywords, (2) available in English, (3) written in the last five years (since 2014), and (4) peer reviewed, was added to the list for further consideration in the next phase of the PRISMA methodology. In the screening phase, titles of publications were manually screened. If the title of a publication indicated the topic of data management but did not relate to the AECO industry, the publication was flagged as irrelevant. In the eligibility phase, abstracts of the publications were reviewed in a first pass and the full text was skimmed in a second pass. In each pass, if it was discovered that the abstract or full text did not meet the criteria of the screening phase, it was removed from the working list. Furthermore, in the eligibility phase, publications overly focused on computer science or algorithmic data calculation, discussing residential buildings, or pertaining to smart grid or smart city technology were considered irrelevant.

The IEEE and Scopus databases gave the most publications within the scope of the review, and the Inderscience database the least. In a few instances, publications appeared on both IEEE and Scopus, these were represented in IEEE, as overall this database offered a greater number of relevant papers to the research topic. Based on the PRISMA methodology, in total, 52 papers were selected and reviewed critically. Note that at the time of writing this paper, the publishing process of 2019 was incomplete, resulting in few selected publications from 2019 (Figure 2). As shown in Figure 2, the general trend has been an increase in the number of studies within the current research topic, indicating increasing recognition for the necessity of an ontology to handle the complex and heterogeneous data sources of buildings.

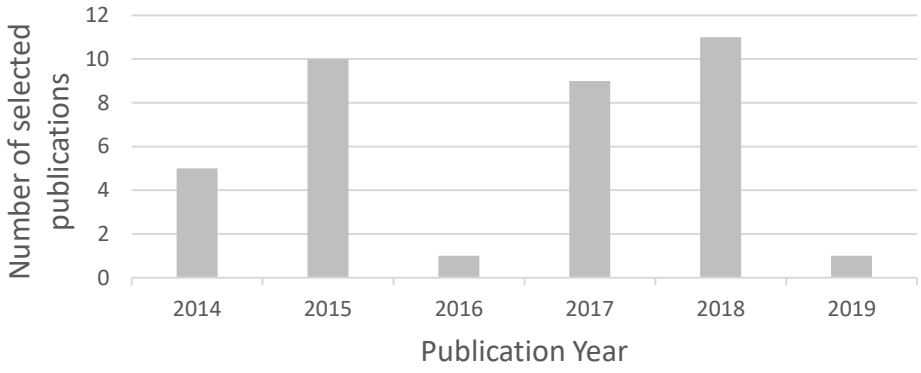


Figure 2: SOCx Related Publications Included in Literature Review by Year

2.2.2 SOCx Data Types

Building data ontologies employ data from a variety of sources in a building. These sources offer varied measures and data points that can be grouped by themes into data types. The desire to group by data types was made on the basis of existing ontologies [21, 22, 9] with more comprehensively defined data types in the literature, the current study categorized data types required for a SOCx ontology as: (1) external conditions, (2) building sensor and parameter data, (3) building actuator data, (4) energy use, (5) maintenance, (6) occupancy, (7) lighting state, (8) physical building information, (9) performance-based data, and (10) simulation-based data. Of the selected 52 papers, 31 were considered for the ontology data type analysis, as the remaining papers were reviews, comparison of existing ontologies, or ontology tools and provided inadequate information for such a comparison.

The review categorized building data as either static or dynamic and identified which type(s) were used in each ontology. Static data types describe physical building information, whereas dynamic data types involve time-series data. Ontologies with both static and dynamic data types were the most common, representing 90% of analyzed ontologies, while the remaining 10% used only static data types. No ontologies considered dynamic data only. Those ontologies using only static data types are still relevant to SOCx because they either discussed linking simulation-based data to physical building information [23, 24] or discussed how BMS elements can be mapped to physical building information [25]. Results of building data type prevalance can be see in Figure 3.

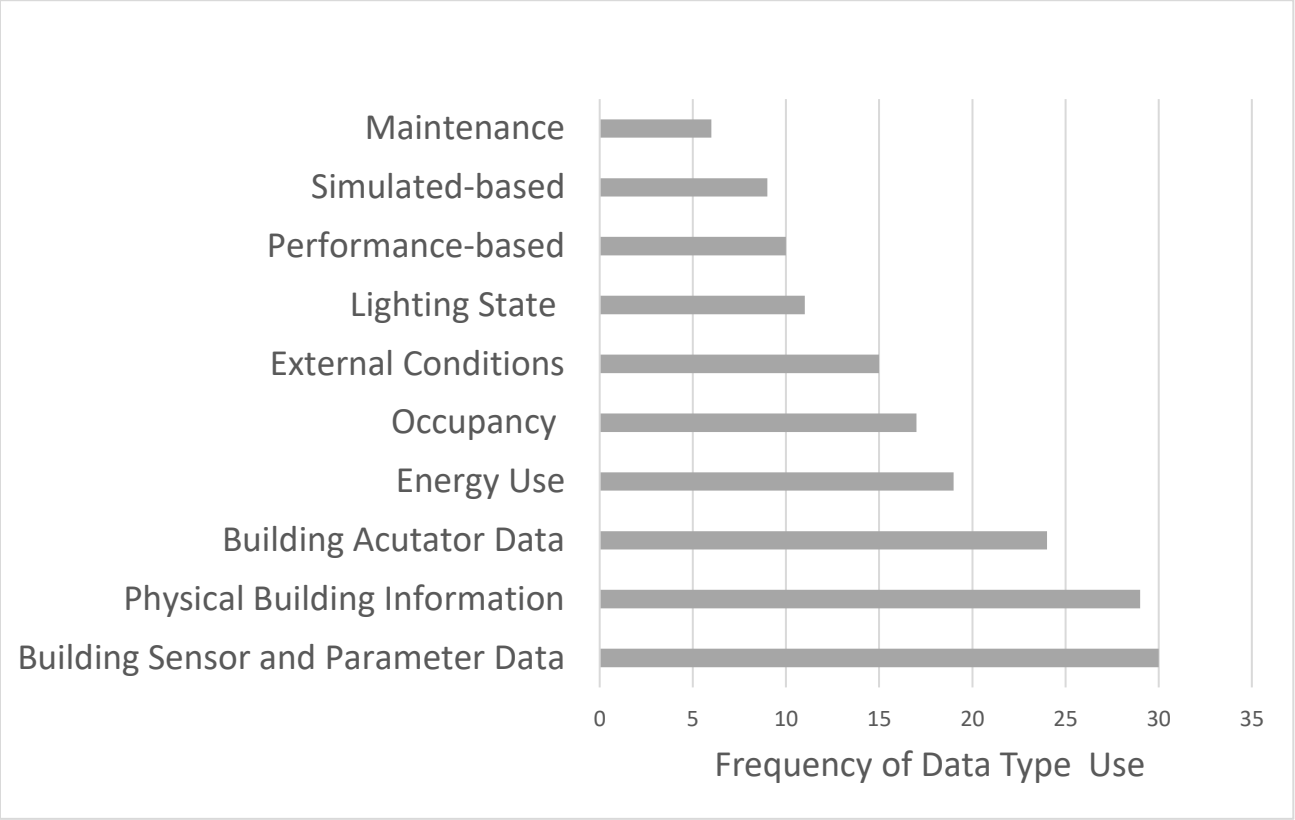


Figure 3 : SOCx Ontology Building Data Type Prevalence

Figure 3 indicates that among the building data types in building ontologies, the most common were building sensor and parameter data and physical building information. The use of BMS and integrated sensors have become more prevalent attributing the pervasive representation of building sensor and parameter data in the ontologies reviewed [26, 5, 27, 28]. The prevalent use of BMS also speaks to the frequent inclusion of building actuator data. Physical building information data has become prevalent, primarily in the construction phase of a building, due to the adoption of BIM technologies. BIM can use the IFC schema, making physical building data accessible to be included in proposed SOCx ontologies through reuse [29, 30]. Maintenance data was the least considered data type despite being highly relevant to SOCx. This is partially due to the lack of agreed methods to extract semantic knowledge from unstructured maintenance textual information such as work orders [2], suggesting that there is a research gap in this area for SOCx ontology. This review revealed that the common data types among existing ontologies included both static and dynamic data.

Physical building information is created once, other than updates as building maintenance work is completed, to represent spatial information. It is therefore a static data type. Static data types are of values as they provide context, allowing for context-aware decision making by SOCx applications. They further help to make ontologies human interpretable for the same reason. Static data types were typically defined by the IFC schema as this schema is a standard model in defining physical Architecture, Engineering, and Construction (AECO) related data. However, dynamic data types are not well suited for the IFC schema [30]. Beyond static and dynamic data types, those required for various applications should also be included. On the basis of the reviewed ontologies an SOCx ontology should cover KPIs at various spatial and temporal scales for building performance assessment, contextual information for reasoning situations, and control and optimization rules for building performance improvement. Only 32% of papers included KPI semantic data in their ontologies [2, 31, 23, 32, 33, 5, 4, 34, 28, 35].

Dynamic data types are inevitable in an SOCx ontology, as they facilitate temporal data which can be analyzed and acted on to facilitate ongoing commissioning. Data types found in BMSs were the most prevalent dynamic data types in ontologies. Among the various ontologies used for dynamic data types, the Semantic Sensor Network (SSN) ontology was frequently reused within a proposed ontology as it provides comprehensive and clear definitions for sensors and actuators in a system, applicable to a wide variety of use cases.

2.2.3 Characteristics of SOCx Ontologies

Existing ontologies were reused in 65% of ontologies reviewed. Using existing ontologies promotes a modular design fundamental to the Linked Data approach, in which elements can be added or removed depending on data domain needs. It is unlikely that wholly original ontologies will be applicable to a wide range of other buildings. A further complication of proposing a fully new ontology based on a set of proprietary building systems lies in the proprietary nature of that semantic structure defined (e.g. [5]).

Reviewed ontologies followed one of the following design methods (1) specification of a new ontology, (2) a Linked Data approach for existing ontologies, (3) a Linked Data approach for a

new ontology with existing ontologies, (4) a Linked Data approach for existing ontologies with extensions. The specification of a new ontology meant that semantic information was organized in a completely novel manner. The proposal of a Linked Data approach for existing ontologies meant that the publication reused one or more existing ontologies in a new way, either cross referencing or converting to the Linked Data approach. The proposal of a Linked Data approach for a new ontology with existing ontologies offered the same methodology as the former, however the authors had included a new ontology for an otherwise unrepresented data source. And finally, a Linked Data approach for existing ontologies with extensions specified how existing ontologies could incorporate additional semantic data in ontologies representing a specific data source. Table 1 offers a summary of ontologies in each category.

Table 1: Design Methods of Reviewed SOCx Ontologies

Design Method	Publications
(1) specification of a new ontology	[5] [9] [23] [22] [25] [36] [37] [38] [39] [40]
(2) a Linked Data approach for existing ontologies	[11] [26] [41]
(3) a Linked Data approach for a new ontology with existing ontologies	[2] [3] [4] [10] [28] [31] [32] [33] [34] [35] [42] [43] [44] [45] [46] [47] [48]
(4) a Linked Data approach for existing ontologies with extensions	[27]

The use of the SSN ontology [49] and a custom Building Automation and Control Systems (BACS) ontology [46] for the SOCx specific application of integrating IoT and BIM data sources are excellent examples of the most common design method (3). The SSN ontology is a “generic language to describe sensor assets” [50] and has been used to accomplish BMS data streaming to BIM, ultimately to provide Facilities Management with better access to BMS data [49]. SSN is used to describe BMS data with semantic tags such as “building”, “room”, and “device” and relate them to spatial building elements using the “hasPhysicalProperty”, and “sensingMethodUsed” attributes [49]. Further custom SSN tag extensions are needed to create the link between the SSN ontology and BIM. Because SSN is a broad approach rather than one developed specifically for the building domain, there is no functionality to map SSN ontology

data to the BIM as represented in open format (Industry Foundation Class, IFC). As such a mapping schema must be created to support the Linked Data approach and is therefore the reason that SSN is not the optimal ontology for SOCx. The BACS ontology is a new ontology developed by reusing existing ontologies such as SSN, Building Topology Ontology (BOT), Sensor Observation Sample and Actuator (SOSA), a fragment of the ifcOWL ontology (ifcmr), and others. Within BACS, BOT is used to describe spatial building data such as “floor”, “space” and “element”, SOSA is used to describe “Sensors” and other “FeatureofInterest” within the IoT network, and ifcmr is used to represent sensors readings as “Values” [46]. The resulting ontology is robust and covers the semantic data required to link BMS, IoT, and BIM data. The reuse of existing ontologies resolves issues of using the SSN ontology alone, as it does not need to be extended in this case, and the inclusion of ifcmr indicates that linking the BIM IFC would be possible. However, there is risk in representing semantic data differently across the relatively large number of ontologies required to represent a limited number of data sources in a set of buildings, affecting application portability.

Bajaj offers a review of ontologies in the building domain driven by their ability to respond to expected queries, and discusses which are best suited for Smart Building applications [51]. Ultimately Bajaj [51] finds that ontologies which: reuse existing ontologies, have been assessed with an ontology validator, are modular and accessible, are properly annotated, and well documented, are the most appropriate to support Smart Building applications

In contrast to reviewed ontologies using the Linked Data approach, the central data model approach was also used, this relies on integrating data sources to a common model where all data is converted into a single data format [24, 4]. Irrespective of the data integration approach, given the vast amounts of data available in existing buildings, using a query processor as a data management system is required for Smart Buildings to access ontology data. This review showed that the common language used to query ontologies for SOCx applications was SPARQL [43, 38, 4], as this query language can effectively support various relationships between structured data [9].

It is evident from this review that several gaps remain in the development of ontologies for

SOCx applications. A notable gap is the lack of a holistic ontology meeting all SOCx applications. To achieve a holistic SOCx ontology, additional research in data types to support all SOCx applications (KPI calculation, building performance improvement, FM BIM, etc.) is required. For instance, maintenance, simulation, and performance-based data are required to support SOCx applications while these data types were found in relatively few ontologies (19%, 26%, and 35%, respectively). Before all necessary data types can be included in an ontology, further research in capturing data is also required [5]. For instance, where BIM models do not exist, strategies such as 3D scanning or automated processes to convert 2D floor plans to BIM models [52] can be employed to avoid manually inputting spatial building data.

Furthermore, relationships found in ontologies are also difficult to define. For example, Borgo et al. [29] noted the difficulty in uncovering the seemingly simple relationships between IFC's "types" and "occurrences" and their ontological representation as classes and instances in OWL. The larger the number of existing ontologies to be included in a proposed ontology, the more complicated this conversion would become. Hence, further research to integrate multiple data types from various existing ontologies in a single ontology is necessary, specifically within the context of the ontology Linked Data approach [4, 3].

The most common shortcoming of the existing ontologies assessed was the lack of a case study. By not providing a case study, the scale of the ontology was difficult to interpret and there was no concrete evidence that the proposed ontology could be applied in an industry setting. Open-sourced ontologies, such as Haystack and Brick, are helpful in minimizing this issue as they allow industry professionals the opportunity to use ontologies at scale. The implementation of open-source ontologies further offers the benefits of being extendable and maintained by the domain community. However, there are challenges regarding intellectual property (IP), specifically in revealing logic supporting advanced building controls while protecting open-source ontology creators' rights, especially in commercialization cases.

Throughout proposed future research, documentation of a proposed ontology – its vocabularies, data types, specific applications, and details of reused existing ontologies will be key to the efficiency of SOCx ontology evaluation and use. Thorough documentation of case studies noting

(1) the data types; (2) a list of ontologies used, their corresponding applications, and how they were linked to each other; and (3) query languages used would be of substantial benefit to support future SOCx implementation and allow comparison between different approaches. Both Brick and Haystack ontologies offer these qualities and are competitively poised for use in industry and SOCx application research, a thorough systematic comparison is therefore of need.

2.2.4 Project Haystack

The Project Haystack non-profit corporation was formed in 2014, and acts as the steward to the open-source Haystack Ontology. As such an industry board of directors and associate members maintain and develop the ontology. Individuals on the board include smart edge hardware and software vendors: ConserveIt, Intel, J2 Innovations (by Siemens), Legrand, Siemens, and SkyFoundry. Siemens being the major industry BMS provider, though Honeywell is also indirectly involved through an associate member (Accutemp). [53]

Haystack is a tag-based semantic data representation where dictionaries of *name value* pairs are defined. Name value pairs are used to describe instances of HVAC concepts within a building, where a *value* is the definition of the concept, and the *name* is the unique string associated with the concept. These pairs are called *Defs* [54] , and are stored in portable groups called libraries, one or more libraries are used to define semantic data of a single building [55]. *Defs* are used as *Tags* on concepts to define the Abox ontology. Multiple *Defs* can be used to describe a concept during instantiation and are referred to as *Conjuncts* [54]. *Defs* can have parent child relationships, effectively defining a hierarchy of concepts, these are referred to as *associations* , an example would be the *is* tag [56]. In a Haystack Abox ontology relationships are formed between *Defs* (tags on a particular component). Typically, relationships used in a Haystack Abox ontologies are *Defs* that end with *Ref* or *Child Protos* that are *Conjuncts* defined within a *Def*. Abox Haystack relationship can be traced back to the Haystack *relationship* types of *containedBy, contains, receives,* or *supplies*. Haystack is serialized in Zinc or Trio, and RDF serializations are not yet supported [57]. Querying Haystack is done using a custom query syntax which the Haystack documentation refers to as *Filters* [58]. Figure 4 visualizes a simplified sample of the Tbox Haystack ontology to describe an AHU bypass damper command point.

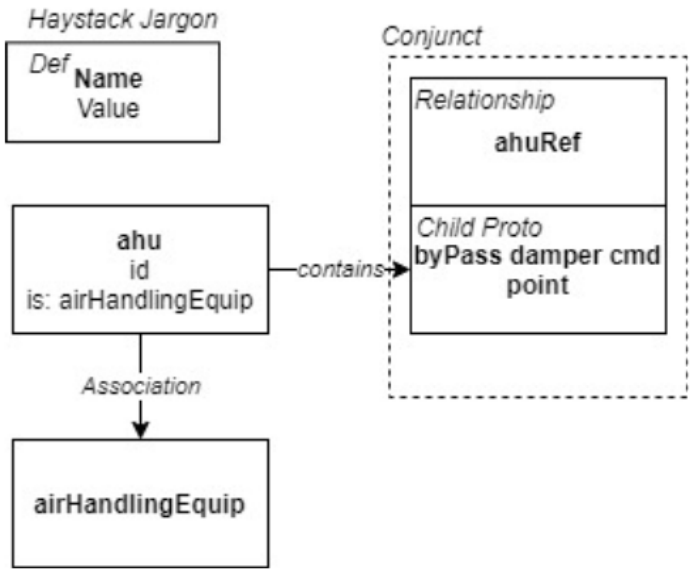


Figure 4: Abox Haystack Ontology Sample, *Italicized Text* is Schema Jargon

Earlier versions of Haystack were a standardized vocabulary (i.e. a meta model). Haystack 4 – considered in this study – offers a full ontology with hierarchy and relationships, however there is still a strong focus on the meta model. Haystack is grounded in object-oriented design, with the goal of describing the logical and physical aspects of HVAC systems, as opposed to explicitly describing semantics. The primary purpose of Haystack moving toward an ontology schema is to support semantic web technologies that were not previously supported. A critique of Haystack has been its over flexibility, leading to Abox ontologies that are not accessible to applications due to unexpected definitions of semantic data. The Haystack Tagging Ontology (HTO) proposed by Charpenay is an application of the Haystack ontology which supports semantic web technologies, and structures the use of tags while extending the vocabular included in the ontology [10]. Though Haystack 4 is described as supporting semantic web technologies, this functionality has not yet been made available to the public. HTO uses Semantic Web technologies (RDF, OWL, SPARQL) to address this gap in Haystack [10].

2.2.5 Brick

Brick was initially described in an academic journal publication in 2016 [59], further publications have since added detail to the ontology description [9]. The brick ontology is being collaborated on by researchers at Carnegie Mellon, Berkeley, University of California San

Diego, University of California Los Angeles, University of Virginia, and the University of southern Denmark. Beyond the academic community Brick is supported by Johnson Controls and some regulatory bodies such as the US Department of Energy and the European Commission [60]. There are more industry partners involved in Haystack than there are in Brick, however Brick offers far more examples of complete building representation and development documentation.

The Brick ontology was developed using a base dataset of more than 17,700 data points found in six buildings equipped with BMS from different vendors, and has been demonstrated to capture 98% of these points [10]. Brick was designed to semantically describe building HVAC systems and is grounded in descriptive logic. As such Brick described whole building concepts in its schema. The Brick ontology schema has a hierarchical design, where Classes are defined and each subsequent layer provides more detail [10]. There are four primary Classes (Equipment, Location, Measurable, Point), each of these classes is extended into increasingly detailed subclasses. For Example, the Equipment Class has a subclass called HVAC, with a further subclass AHU, etc. Additionally, there is a relationship Class that has 16 subclasses to define each of the nine bidirectional ontology relationships. Haystack tagging leaves room for ambiguity that could cripple the portability of applications [9]. Alternatively, Brick uses a more prescribed approach to defining semantic data where subclasses are comprised of complete HVAC concept in an ordered string, ex: Discharge_Air_Remperature_Sensor. Brick relies on superclasses to achieve ambiguity. A review of ontologies indicated that Brick was weak given its lack of detailed spatial data and focus on use for Smart Building applications [51]. This spatial data can be found in BIMs, and can include part numbers, equipment manufacturer information, etc. Detailed spatial data is however not always necessary for the applications pertaining to SOCx, and Brick can be cross referenced in the Linked Data architecture to consider IFC spatial data if available. The Brick schema website [60] offers instantiated ontology examples, information on writing .ttl files, and a suite of web tools to visualize and edit .ttl files. Figure 5 is reproduced from this site and gives a graphical sample of a possible brick ontology instantiation.

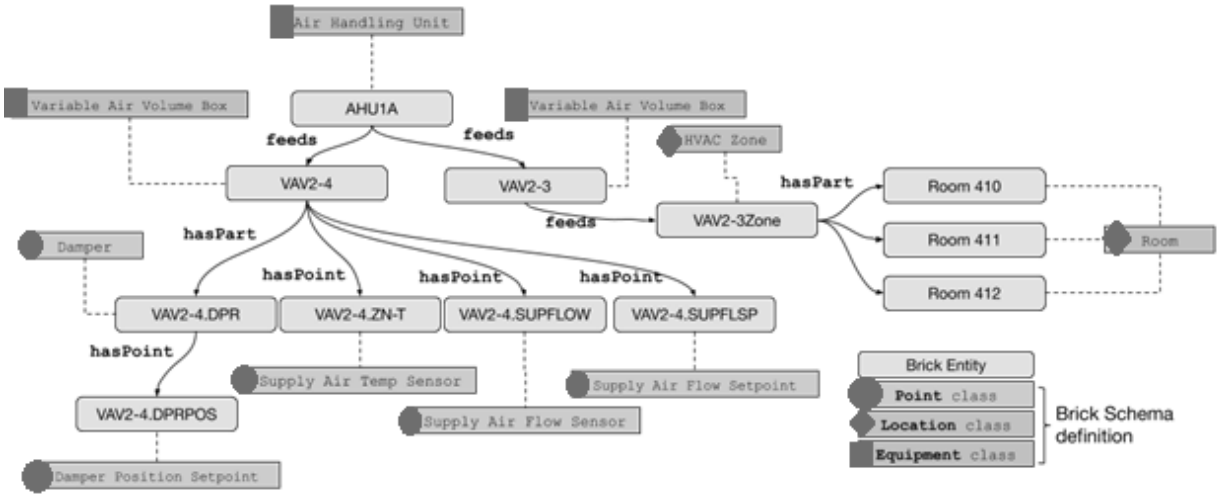


Figure 5: Abox Brick Ontology Example [61]

A query processor is key to facilitating the transfer of building data to SOCx applications in a Linked Data architecture, HodDB [13] has been designed specifically for this purpose when using Brick. The "open-source" GitHub package makes HodDB accessible for implementation, allowing application developers to have full access and test appropriately during application development. To achieve efficiency in query processing HodDB uses a decoupled database and query processor as specified by Fierro and Culler [13]. HodDB imports RDF serialized .ttl files containing building ontology data and stores it in a LevelDB, a database optimized for storing key-value pairs [62] . BTrDB [63] , optimized for the storage of timeseries data, is used to store timeseries data. In the HodDB architecture a UUID, unique identified associated to each semantic concept, allows the LevelDB and BTrDB databases to be cross referenced. Figure 6 demonstrates the specification of a UUID for a temperature sensor in a .ttl file.

```
1 @prefix bf: <https://brickschema.org/schema/1.0.1/BrickFrame#>.
2 @prefix brick: <https://brickschema.org/schema/1.0.1/Brick#>.
3 @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
4 @prefix bldg: <http://brickuniversity.edu/buildings/BuildingABC#>.
5 bldg:Temp_Sensor_1 rdf:type brick:Zone_Air_Temperature_Sensor.
6 bldg:Temp_Sensor_1 bf:uuid "1a5a7224-fa34-11e7-823a-1002b58053c7".
```

Figure 6: RDF Format Specification of a Sensor UUID in Brick Abox Ontology [13]

UUIDs are not an efficient way to access data in ontology or timeseries databases as the strings are long. For this reason, the query processor in HodDB uses several indexes to access data. Applications require that the query processor identify data points in the LevelDB containing the ontology, and then locate each of these subtrees in the BTrDB, returning the values specified in the initial SPARQL query. A sample SPARQL query can be seen in Figure 7.

```
1 #Temp Sensors (First Floor Rooms, Previous Day)
2 SELECT DISTINCT ?undefinedSensor WHERE {
3 ? undefinedSensor rdf.isType* brick:Air_Temperature_Sensor
4 ? Air_Temperature_Sensor brick:isPointOf uuid:Room_118
```

Figure 7: Sample FM-BIM Application SPARQL Query to Retrieve Room Specific Temperature Sensor form Abox Brick Ontology

The query processors would identify UUIDs of air temperature sensors described as points of a specific room indicated by the UUID. The query processor would appropriately select the semantically described temperature sensor as indicated in Figure 8. It should be noted that Brick relationships are bidirectional, and despite the ontology defining a hasPoint relationship isPointOf can be used in the query. The list of retrieved UUID's from the LevelDB would be managed by the query processor to then access the actual data values in the BTrDB timeseries database.

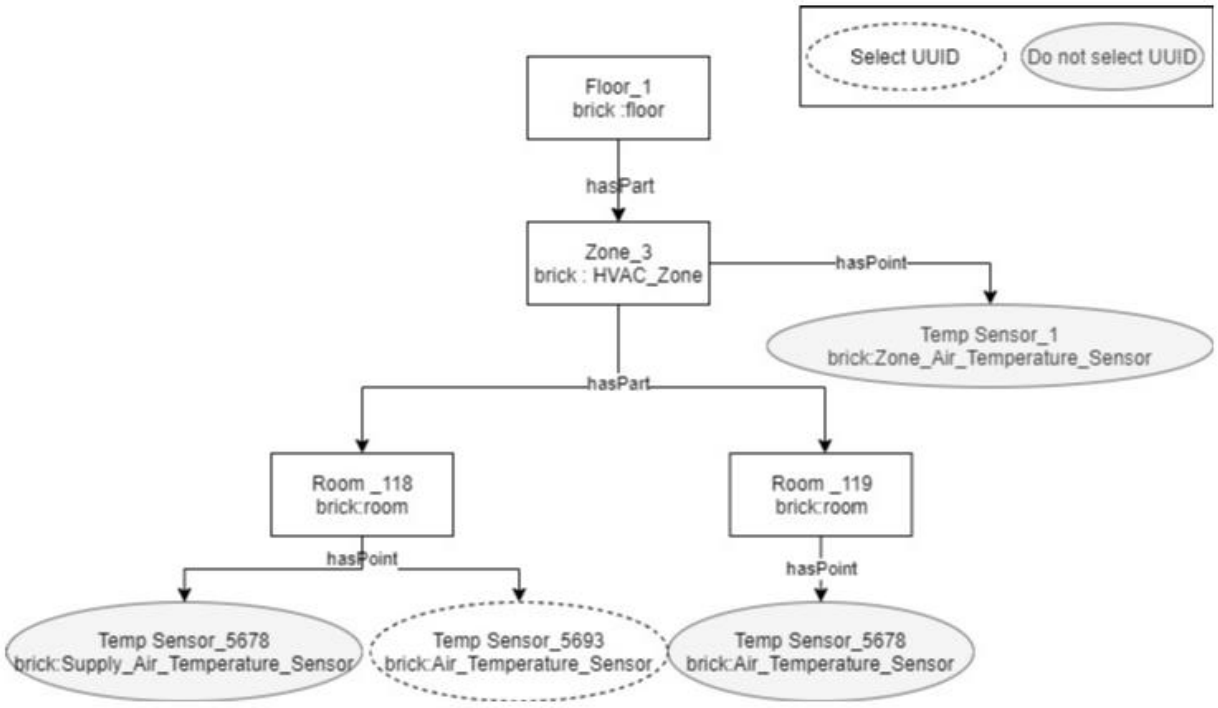


Figure 8: Sample of FM-BIM Application Retrieved Abox Brick ontology nodes in ER Diagram format

2.3 Ontology Comparison Methods

Ontology schemas are difficult to evaluate due to their declarative nature, instead they typically require evaluation through test cases yielding quantitative results [64]. Four ontology evaluation approaches have been deemed effective: (1) ontology comparison, (2) domain-specific document comparison, (3) expert review, and (4) case studies [65]. The most common of these is implementation and comparison of ontology performance using a case study [66, 9, 10, 64]. The Brick ontology was validated using the implementation and comparison method by Balaji et al. [9] comparing it against SAREF, Haystack and IFC. These four ontologies were each tested using six buildings running eight Smart Building applications; where each ontologies *completeness* was measured by data point coverage in each building, and *expressiveness* and *usability* were measured by assessing each ontologies recall of data points to meet each of the application query requirements in each building. In this comparison Brick achieved higher completeness and expressiveness scores. The Haystack ontology has been validated using a similar method by Bhattacharya et al. comparing it against IFC and SSN using three buildings,

and using a summarized list of key relationships instead of Smart Building applications [67]. The quantitative assessment done by Bhattacharya et al. assessed *completeness*, which is the coverage of data points, location, types, units, etc., and the *ability to capture key relationships (*i.e. *expressiveness).* Additionally, f*lexibility,* which is the ability of the ontology to capture uncertainty or change of building configuration, was discussed qualitatively. Bhattacharya et al. [67] found that Haystack offered the best completeness of the three assessed ontologies with 63% coverage of baseline data, as well as the best expressiveness with 77% coverage.

A qualitative assessment is also frequently used for ontology evaluation within the building domain, several *qualities* have been regarded as positive and relevant in the academic literature: flexibility, portability, readability, extensibility, interoperability, and queryability. This qualitative assessment can be carried out through document review. *Flexibility* answers the question 'can the ontology capture uncertainty in semantic data and does it use non-restrictive methods to define concept semantically?' [66]. *Portability* answers the questions 'can the same set of applications be applied across buildings (with applicable HVAC systems) using the specified ontology?' and 'is semantic data represented consistently in a machine-readable format.' [9]. *Readability* answers the question 'can domain experts and applications developers unambiguously decipher real world meaning from semantic data as presented in the ontology?' [9]. *Extensibility* answers the question 'can the ontology be customized to add new semantic concepts?' [66, 9]. *Interoperability* answers the questions 'can the ontology integrate with, and convert to, other ontologies with little to no human effort?' and 'Is the ontology serialized in an industry accepted format?' [12]. Finally, *Queryability* answers the questions 'can the Abox ontology be machine traversed and necessary information retrieved?' and 'Is there low variability in semantic relationships?' [68, 67]. The response of yes is the desired outcome to quality questions, indicating that the quality is expressed in the ontology.

2.4 Summary of Literature Review

The use of the Linked Data approach for semantic data representation offers significant benefits to SOCx applications [3, 4, 5]. An *ontology linked* approach is appreciated because it allows for the integration of multiple building data sources, facilitates portability of SOCx applications [9], and facilitates the fast and effective retrieval of building data for SOCx applications [13]. Many ontologies have been developed in the hopes of rendering building data sources more accessible SOCx applications. Two ontologies have however captured the attention of industry and academia – Brick and Haystack. This is in part due to the lack of other proposed ontologies covering all data types required by SOCx applications, and due to the lack of "open-sourced" ontologies with ample publicly available documentation and examples.

Despite the significant research to date on the topic of data ontology to support SOCx, there remains a paucity of literature regarding the comparison of the most appropriate ontology for SOCX. Specifically, when considering the most widely available, and discussed Brick and Haystack ontologies. This Thesis aims to fill this gap by presenting a qualitative and quantitative comparison of the Brick and Haystack ontologies completeness and expressiveness.

3 Methodology

A mixed methods approach is used in this research, a case study has been used for quantitative assessment, and document review for qualitative assessment. The goal of this research is to assess the ability of each *target ontology*, Brick and Haystack, to accurately represent the real world domain of SOCx – a process is referred to as *ontology validation* [64]. This is as opposed to *ontology verification* which employs tools to measure the quality of ontology instantiation [64]. Quantitative assessment includes completeness and expressiveness of the target ontologies, these measures are demonstrated in the academic literature to be valuable [9, 69, 13]. Additionally, this research qualitatively assesses the Brick and Haystack ontologies through document review. Figure 9 is a logical flowchart describing the methodology used in this research of ontology validation.

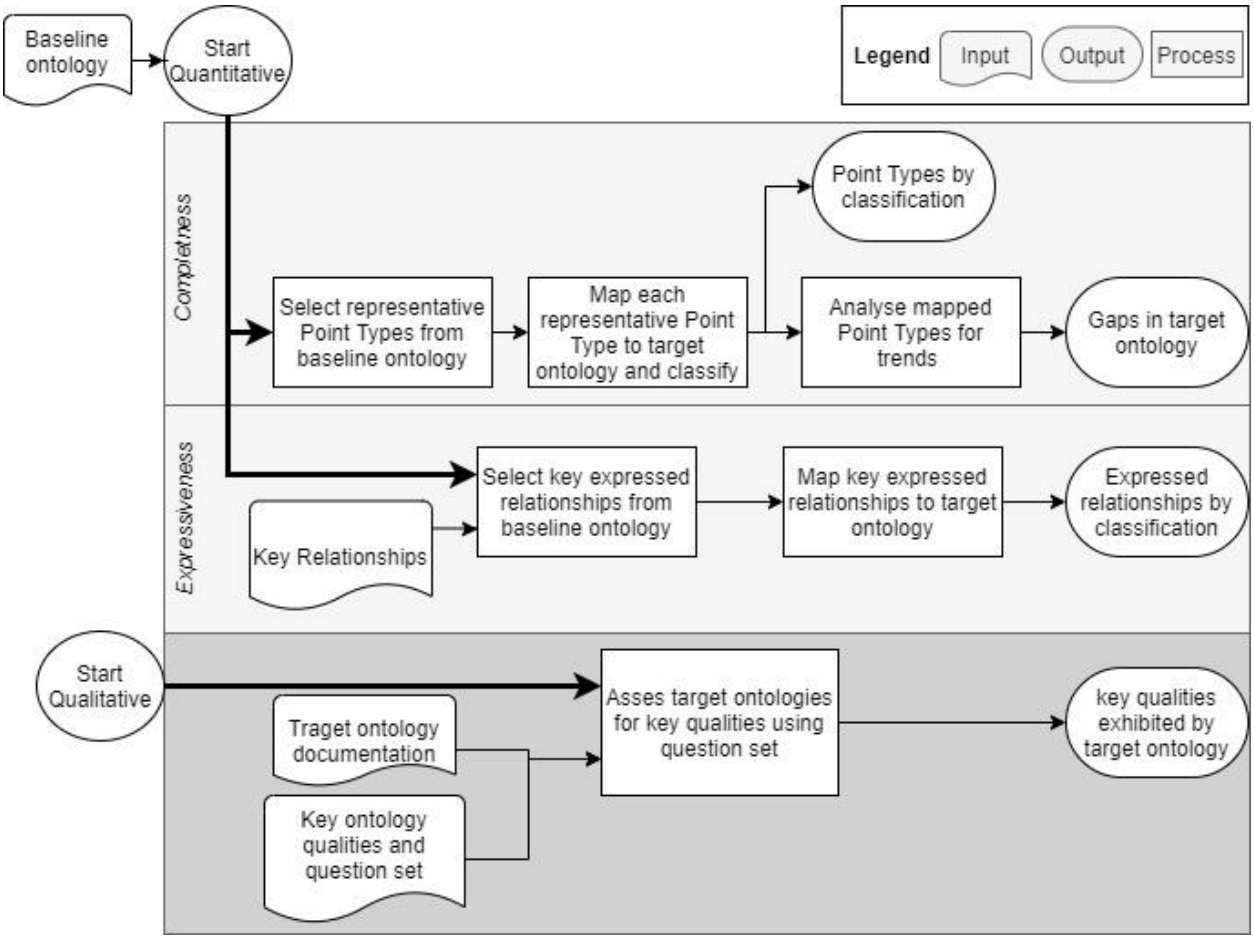


Figure 9 : Ontology Validation Method

As recommended by the literature, a case study approach has been used for ontology validation. The qualities assessed in this research are a hybrid of those previously assessed in reviewed publications, defined herein as completeness and expressiveness. The measure of an ontologies *completeness* is the breadth of domain semantic concepts represented [9, 66], while *expressiveness* is the measure of an ontologies ability to represent relationships that facilitate the retrieval of dynamic building data relevant to SOCx applications. To measure completeness and expressiveness a representative industry ontology was used as the baseline. This case study approach differs from previous research which typically use one or more buildings as a baseline. The approach of an ontology as a baseline offers the largest representation of possible building systems and data points and is therefore preferred to provide the most accurate results.

There is a large amount of uncertainty in determining all SOCx use cases, and therefore the appropriate data set for assessment of ontologies. Even very large case studies referencing many buildings as a baseline will not give a precise evaluation due to the variability in systems and unknown set of SOCx applications that will use an ontology. To manage this uncertainty a mixed method research approach is taken where both quantitative and qualitative results are explored. In the quantitative completeness and expressiveness assessment results are interpreted relative to the baseline, leading to a focus on trends in the performance of either ontology. This approach is as opposed to focusing exclusively on the quantitative results of completeness and expressiveness. Additionally, qualitative assessment of the ontologies within the Linked Data context are completed through documentation review and assessment against a predetermined set of desirable qualities.

The version of ontology used in assessment fundamentally changes results. This study is concerned with “open-sourced”, fluid technologies and therefor the most recent beta versions of Brick and Haystack are considered. The weakness of this approach lies in small body of available beta documentation, publications, and sample implementations of ontology schemas. Furthermore, the fluid state of beta versions implies changes in ontology schema by contributors over the course of study. Alternatively, using archived ontology versions would not lead to the most representative results as future implementations will not use such versions. As such, the study uses the beta Haystack Version 3.9.7 (referred to as Version 4 in marketing material) and

its documentation as published on October 24th, 2019 [70], and Brick Version 1.1.0 and its documentation as published on February 21st, 2020 [71]. A founding member of Haystack indicated on the user forum in late 2019 that they hope the release Version 4 between the end of the first quarter and beginning of the second quarter 2020 [72]. Brick Version 1.1.0 is published, and in an acceptance testing phase.

3.1 Baseline Industry Ontology

The KGS industry ontology was used as a benchmark in the case study approach of ontology validation. KGS is a commercial for-profit organization which offers customers insights on building equipment diagnostics, recommendations for energy savings, and occupant comfort condition ratings using systems already installed. Underpinning their Smart Building application is a bespoke ontology implemented for all clients. The offerings of the KGS application align with the goals of SOCx and therefore their ontology is an appropriate baseline for this case study. A weakness in using a baseline ontology lies in the assumption that the designers of the ontology have done so accurately and completely. If the baseline ontology is not representative of all building semantic data, then completeness and expressiveness will not be measured accurately. However, this flaw, if present will be applied in the assessment of both ontologies, and therefore the resultant scores will remain comparable and trend analysis will be possible.

The baseline KGS ontology is defined in a hierarchical structure, as seen in Figure 10. *Equipment Class* is the highest-level representation of systems (ex: chiller or AHU) and prevalent sub-equipment (ex: fan or pump), while *Equipment Type* represents subtypes (ex: preconditioning AHU). Equipment Types can be related within the baseline ontology using an *Allowed Equipment Association*. Semantics of data points are represented with *Point Types* which are not necessarily related to equipment concepts, these relationships can be explicitly defined in the ontology or inferred in the Smart Building application. Point Type is a subclass of *Point Class,* which is the concept of measures within a system (ex: Temperature, Flow, Position, Enable). It should be noted that the *Enable* Point Class is a command turning equipment on allowing it to run, while *Run* is a command issued to invoke equipment to run. Additionally, the Point Class *Status* is a sensor receiving Boolean on/off command signal feedback, while *Feedback*

receives numerical command signal feedback. Each Point Class has an *Engineering Unit* (ex: Celsius and liters per second) which helps to define the meaning of Point Classes and expected data values. Finally, *Measurement/Control Type* (ex: min, stage, limit) help to give standardized context to Point Types, and *Service* (ex: water, air) indicates the medium the Point Types is acting on. Ultimately, Point Type elements capture the purpose of the data point within the BMS.

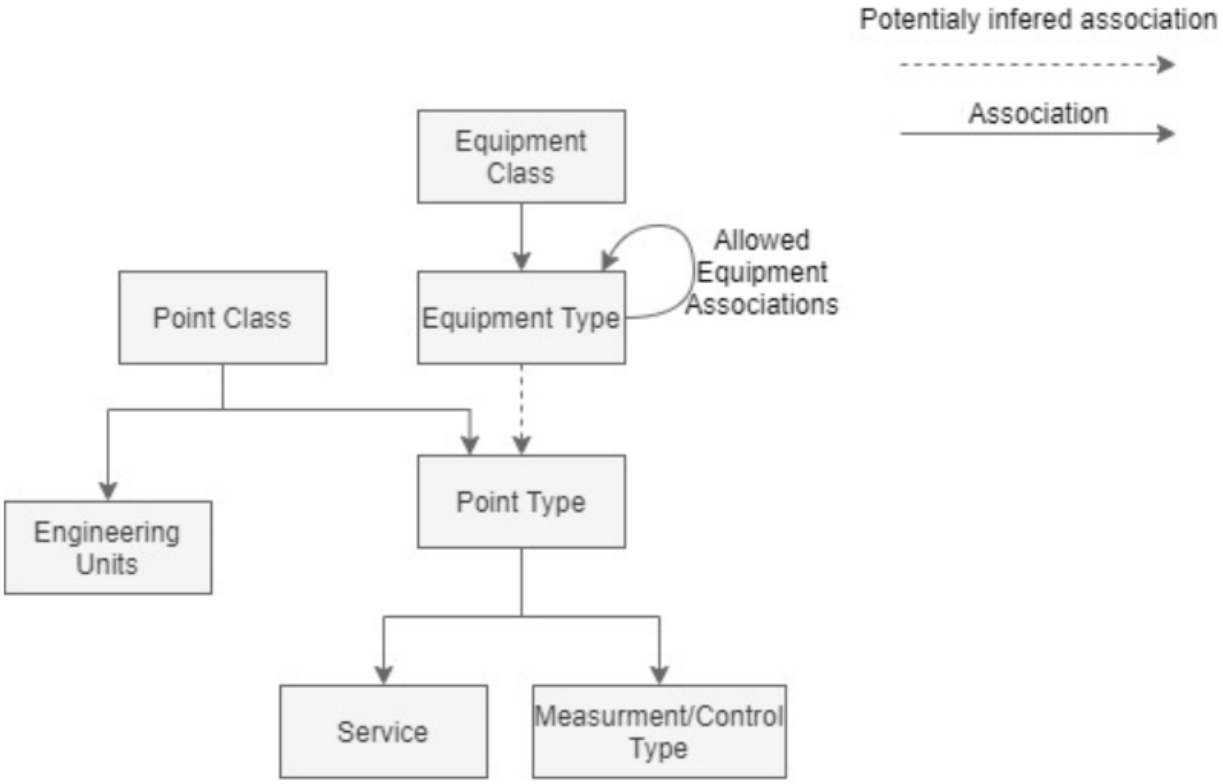


Figure 10 : KGS Baseline Ontology Tbox Schema

In the KGS ontology Point Types are the most consistent means to represent building systems. In considering a Point Type, all available related and inferred semantic data is also considered. A sample of semantic data representation in the baseline ontology centering on Point Type is shown in Figure 11 : KGS Baseline Ontology Abox SampleFigure 11. The KGS ontology has 1422 Point Types. Two steps were taken to manage the scope of *Point Types* included in the quantitative assessment. First, HVAC critical and common systems: AHU, Chiller, Boiler, Terminal Units, and Loops were selected for mapping, this reduced the number of Point Types to

997. Second, Point Types containing only redundant concepts within a system were removed from the baseline set, this left a *representative set* (non-redundant) of 440 Point Types to be used in the quantitative assessment. The representative set was selected by filtering Point Types by HVAC system and selecting those containing at least one unique *word*, where a word is a substring within the Point Type. For example, the point name *RoomAirDpTemp* was broken into the words: *Room, Air, Dp*, and *Temp*. This approach ensured that Point Types related in various Equipment Classes and Equipment Types were considered as well as all Point Classes, Services, and Measurement/Control Types within an HVAC system. Some exceptions were made where obscure Point Types only appearing in a couple of buildings were not selected as representative despite having unique words. The breakdown of selected representative Point Types by system and subsystem is shown in Table 2.

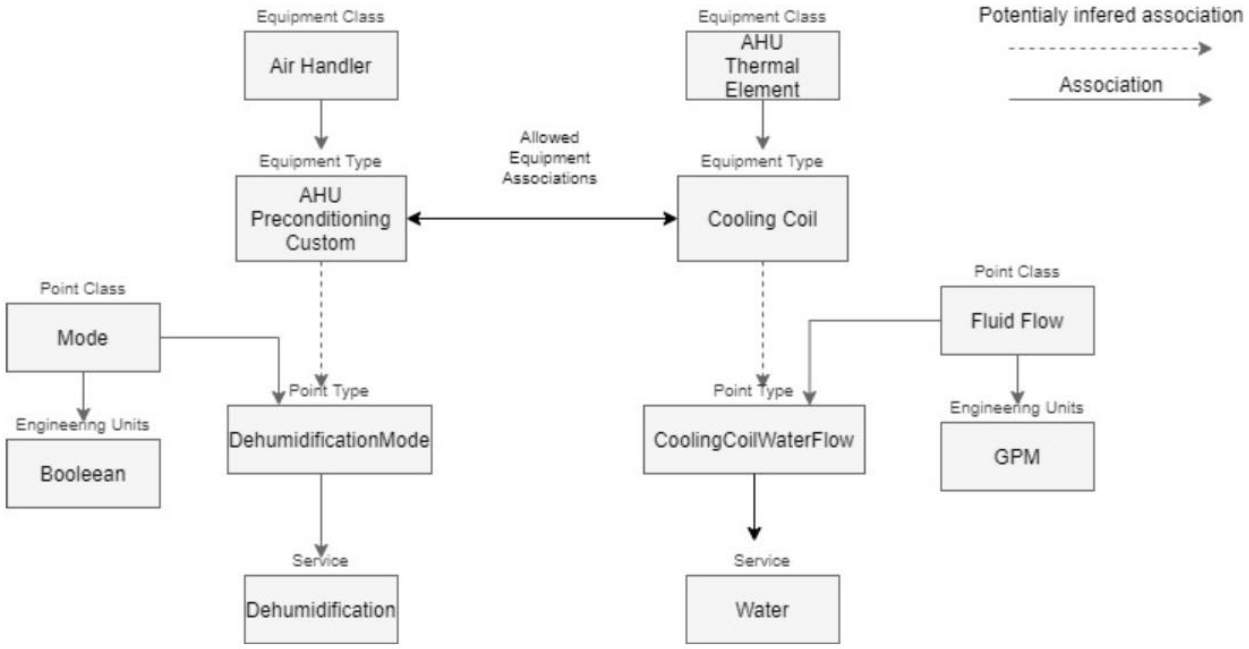


Figure 11 : KGS Baseline Ontology Abox Sample

Table 2: Representative Baseline Points Types by System

System	*Subsystem*	*Point Types*	*Representative Point Types*
AHU		**410**	**165**
	Coil	44	19
	Damper	54	17
	Enthalpy Wheel/Heat Wheel/ HR / HR Exhaust	78	29
	Fan	81	24
	Filter	5	5
	Valve	24	10
	misc. (subsystem not indicated)	124	61
Chiller		**129**	**92**
	Compressor	16	13
	Condenser	18	16
	Evaporator	16	13
	Fan	6	1
	Generator	3	3
	Heat Source	8	4
	Pump	7	1
	Valve	11	5
	misc. (subsystem not indicated)	44	36
Boiler		**41**	**31**
	Economizer / Economizer Flue / Flue	4	4
	Valve	4	2
	misc. (subsystem not indicated)	33	25
Secondary Loop (Condenser water loop)		**262**	**60**
	Heat Exchanger	48	13
	Valve	29	13
	misc. (subsystem not indicated)	185	34
Terminal Units		**155**	**92**
	Coil	10	8
	Compressor	4	4
	Damper	19	10
	Fan	18	10
	Valve	13	9
	misc. (subsystem not indicated)	91	51
Total		**997**	**440**

3.2 Completeness Measurement

Completeness was measured by assessing which of the representative Point Types could be mapped to a target ontology, this is appropriate given the baseline ontology schema as described in Section 3.1. Where the target ontology was the ontology being tested for completeness (i.e. Brick or Haystack). Completeness mapping involved mapping semantic data captured in representative Point Types to the target ontology being assessed. It should be noted that Engineering Units were not considered in this mapping as they were not easily identified for Point Types in the baseline ontology, and Brick does not yet support units. Mapping representative points was completed in the following steps, where only concepts available in the target ontology schema definition were used (i.e. no extensions added to either target ontology). The Brick mapping was serialized in a Turtle (.ttl) triple, and Haystack in tags.

1. Filter Point Types for specific system
2. Consider each Point Type name in system and map to target ontology
3. Write out semantic data representation for Point Types using ontology entities and appropriate serialization method
4. IF
    a. All semantic data maps from baseline to target ontology
        Classify as *Maps*
    b. Equipment Class and Point Class map, but one semantic gap exist in either measurement/Control Type, Service, or Equipment Type
        Classify as *Partially Maps*
    c. More than one gap in Point Class, Service, or Equipment Type or Equipment Class does not map
        Classify as *Does Not Map*

Previous studies have used the equivalent of words to map semantic data from a baseline data set to a target ontology in an effort to asses completeness [66, 9]. This approach requires little manual effort, however semantic data beyond what is represented explicitly in a word will not be evaluated. For example, the Point Type *EnthalpyWheelExhaustDischargeAirDpTemp* would map words- *Exhaus*t, *Discharge,* and *Air*- separately, however the semantic representation of this within the concept of the *Enthalpy* should be *Discharge Air*. Furthermore, words in baseline

ontology Point Types might not have exact matches in the target ontology while still having semantic mappings. For example, *HEX* in the baseline ontology maps to *Heat_Exchanger* in the Brick ontology. This study takes the approach of mapping all concepts represented by words in each Point Type manually to ensure appropriate classification.

The classification of Point Types was made on a binary basis. Each Point Type was exclusively classified as *Maps, Partially Maps*, or *Does Not Map*. In the analysis of completeness *% Point Types with Maps Classification* was calculated by dividing the number of Point Types classified as maps by the number of Point Types in the representative set. Similarly, *% Point Types with Maps or Partially Maps Classification* was calculated by summing the number of Point Types classified as *Maps* or *Partially Maps* and dividing by the total number of representative Point Types.

To better understand trends in completeness *Partially Maps* and *Does Not Map* classified Point Types was assessed for gap. Each semantic gap in a Point Type was coded by type (missing measure, equipment, medium, or concept), and the missing piece of semantic data was indicated. As such Point Types with multiple gaps contributed to the count of multiple gaps rather than a single randomly selected one. Gaps were classified as follows for trend analysis:

- **Major and Significant** – Gap classified as *Does Not Map* and affected at least 2% of Point Types in the representative set
- **Major and Insignificant** – Gap classified as *Does Not Map* and affected less than 2% of Point Types in the representative set
- **Minor and Significant** – Gap classified as *Partially Maps* and affected at least 2% of Point Types in the representative set
- **Minor and Insignificant-** Gap classified as *Partially Maps* and affected less than 2% of Point Types in the representative set

3.3 Expressiveness Measurement

Expressiveness was measured by quantifying a set of key relationships required by SOCx applications as found in the baseline ontology. Each of these key relationships were found in the baseline ontology for the set of previously identified systems (AHU, Chiller, Boiler, Terminal

Units, and Loops) on both the air and water sides. The total number of relationships assessed in target ontologies was 27.

Key relationships required by SOCx applications have been identified in previous publications [67, 9, 13, 69]. In Bhattacharya et al. [66] an evaluation of the Haystack, IFC, and SSN ontologies identified relationships supporting applications aligning with the purpose of SOCx applications including: Occupancy Modeling, Energy Apportionment, Web Displays, Model Predictive Control, Participatory Feedback, Fault Detection and Diagnosis, and non-intrusive Load Monitoring and Demand Response. The following relationships identified by Bhattacharya et al. were non-redundant and were deemed key relationships warranting assessment in the target ontologies: Sensor ↔ Location, Location ↔ Location, Equipment ↔ Location, Sensor ↔ Equipment, Equipment ↔ Equipment, Location ↔ Persons. Balaji et al. [9] used Bhattacharya et al.'s applications and built on the set of key relationships in the assessment of the Brick ontology by adding a Equipment ↔ Name relationship. The necessity for all previously mentioned relationships is supported in the development of a query processor designed for energy management applications [69], as well as one designed for fault detection, information dashboards, and gray box modellers [13].

The baseline ontology defines semantic relationships within Point Types and Allowed Equipment Associations. For example, Point Type *CoolingCoilBypassDamper* as found in the AHU system is a damper *associated to* an AHU that *controls* airflow to a cooling coil. In use, Point Type relationships are inferred by the KGS Smart Building application. However, portable SOCx applications rely on the explicit definition of relationships. All implied relationships could not easily and consistently be discovered in the baseline ontology, for this reason the key relationships were cross-referenced with the baseline ontology to guide the selection of the set of representative baseline ontology *expressed key relationships.* These expressed key relationships were found in baseline Point Types of Allowed Equipment Associations and can be seen in Table 3. Trends in expressiveness were assessed by quantitatively assessing expressed key relationships found in target ontologies classified as follows: For each expressed key relationship, check both target ontologies to see if it is present; if so, classify as *Maps* otherwise, classify as *Does not Map.*

Table 3 : Baseline Ontology Expressed Key Relationships by System and Service

	AHU		***Chiller***	***Boiler***	***Loop***	***Terminal Unit***		***Equipment Type***
	Air Side	*Water Side*	*Water Side*	*Water Side*	*Water Side*	*Air Side*	*Water Side*	
Sensor ↔ Location	N/A							Air Temp ↔ Zone
Location ↔ Location						Zone ↔ Room		N/A
Equipment ↔ Location						Terminal Unit ↔ Zone		Damper ↔ Zone
Sensor ↔ Equipment	Air RH ↔ Enthalpy Wheel	Water Temp ↔ Cooling Coil	Water Temp ↔ Condenser	HW Flow ↔ Boiler	Flow ↔ Loop	Temp Setpoint ↔ Terminal Unit	Load Fraction ↔ Terminal Unit	Command ↔ Valve
Equipment ↔ Equipment	Economizer ↔ AHU	Cooling Coil ↔ AHU	Chiller ↔ Loop	Boiler ↔ Loop	Loop ↔ AHU	AHU ↔ Terminal Unit	Loop ↔ Terminal Unit	Cooling Coil ↔ Bypass Damper
Location ↔ Persons	N/A					Room ↔ Occupants		N/A
Equipment ↔ Name	AHU ↔ Name		Condenser ↔ Name	Boiler ↔ Name	Loop ↔ Name	Terminal Unit ↔ Name		Cooling Coil ↔ Name

3.4 Qualitative Comparison

In parallel with the expressiveness and completeness assessment described above, a qualitative assessment of each ontology was performed to evaluate the functionality of the Brick and Haystack ontologies within the Linked Data context which they are designed to support. The functionality of Linked Data includes the machine interpretability and connection of additional building data ontologies. To undertake this evaluation documentation was reviewed, assessing the flexibility, portability, readability, extensibility, interoperability, and queryability of each ontology, as defined in the literature review. The outputs of this assessment were evaluated by the presence or lack of aforementioned positive qualities.

4 Results

The following results present the outcomes of the completeness, expressiveness, and qualitative assessments of the target ontologies Brick and Haystack as applied to the SOCx domain. The ontologies in question could yield different results if they are compared to a different baseline ontology. The selected baseline ontology was developed to serve a SOCx related application which includes fault detection, energy optimization, and human comfort tracking functionality and is therefore well suited to provide relevant results. Completeness was measured by quantifying the number of baseline ontology representative Point Types in key systems per classification (*Maps, Partially Maps,* or *Does Not Map*). Expressiveness was measured by quantifying the number of baseline ontology expressed key relationships per classification (*Maps* or *Does Not Map*). Finally, a defined set of desirable qualities were assessed for representation in either target ontology by reading supporting documentation and classified qualitatively as *supported* or *unsupported*.

4.1 Completeness

The Brick ontology was able to achieve a greater percent of representative Point Types classified as Maps than Haystack, with 59% and 43% achieved respectively. This trend continued in the less stringent measure where the Point Types classified as Partially Maps were also included. 77% of Brick and 69% of Haystack were classified as Maps or Partially Maps. Completeness results can be seen in Figure 12. A primary finding of completeness measurement was the mapping of the baseline ontology Point Classes and Measurement/Control Types to either target ontology. These baseline ontology features are often substrings of a Point Type and together describe its function. Mappings of Point Classes and Measurement/Control Types maintained consistency in the completeness assessment across systems assessed and are stated in respective subsections.

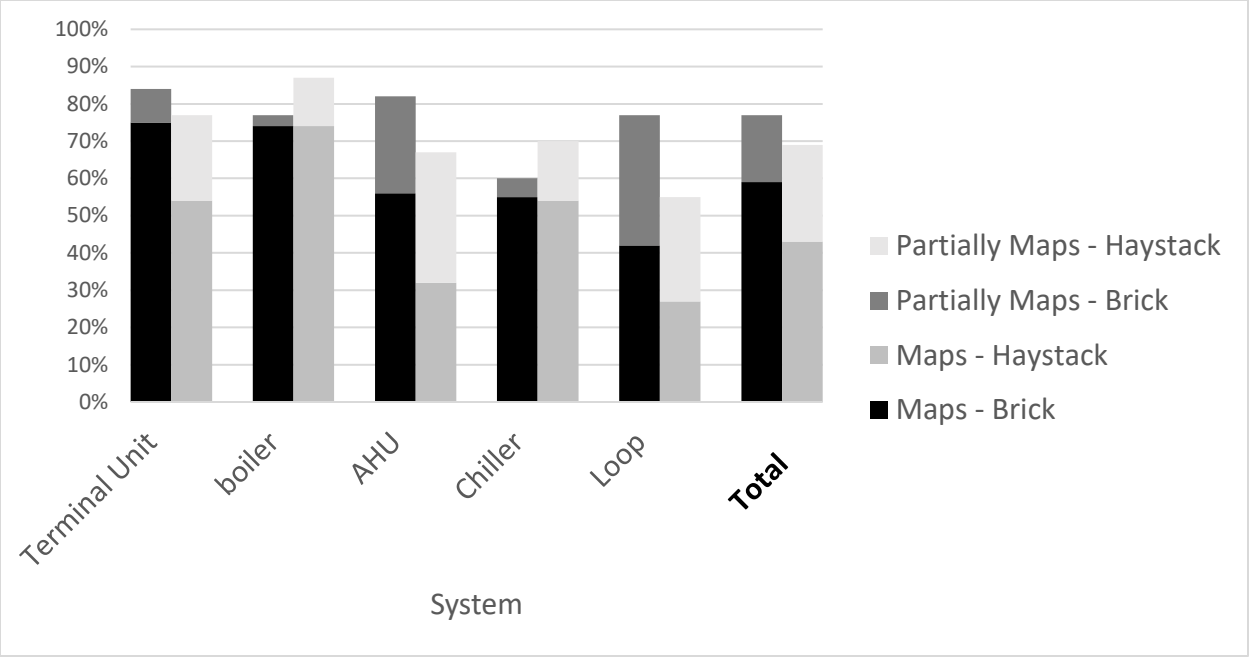


Figure 12 : Brick and Haystack Completeness Results against Baseline Representative Point Type set

4.1.1 Haystack Completeness

The Haystack ontology offered adequate semantic completeness, especially when considering Point Types classified as Maps or Partially Maps. The results for the completeness of the Haystack ontology by system type can be seen in Table 4. In the process of assessing completeness, the mapping of baseline Point Classes and Measurement/Control Types to Haystack was completed and can be seen in

Table 5. Generally, there were many semantic gaps in the Haystack ontology each affecting a small number of Point Types. There were however a handful of gaps affecting a significant number of Point Types which are described in detail below.

Some semantic data representation is not available in the Haystack ontology as required by the baseline ontology. Gaps were found as seen in Figure 13 , the Significant Gaps (those affecting 2% or more of Point Types in the baseline ontology) found in the right two quadrants of the figure are discussed below. Generally, there are many Insignificant Gaps (those affecting less than 2%) which together have a large impact on the Haystack Completeness. It should be noted that these Insignificant Gaps could be prolific if the representative Point Type set was expanded to include all Point Types in the baseline ontology.

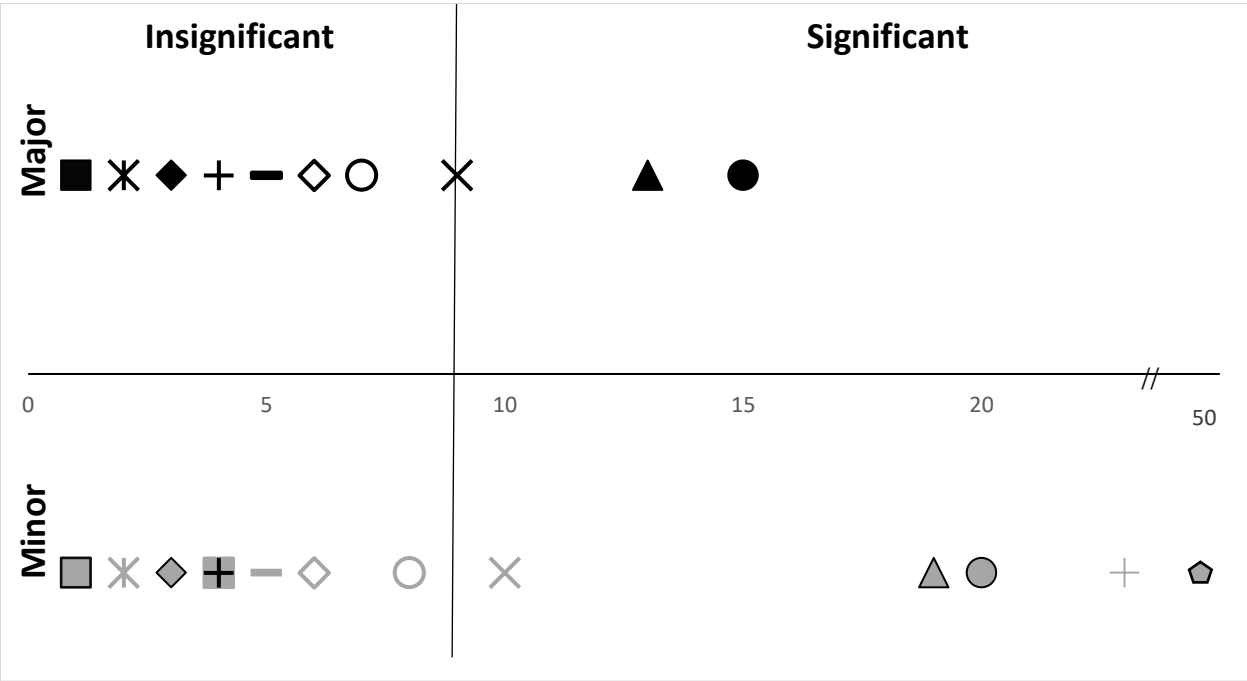
Table 4 : Haystack Completeness Results Against Baseline Representative Point Type Set

System	% Point Types with Maps Classification	% Point Types with Maps or Partially Maps Classification
AHU	32%	67%
Chiller	54%	70%
Boiler	74%	87%
Loop	27%	55%
Terminal Units	54%	77%
Total	43%	69%

Table 5 : Baseline Ontology Point Class to Haystack Point Tag Mapping

Baseline Point Class	Baseline Measurement/Control Type	Haystack Point Tag Mapping	Baseline Point Type Example
*	Min/Max/Limit	sp Tags: minVal or MaxVal	ChillerRunLoadAmpsLimit SupplyAirTempMin
	N/A		
Enable	**High/Med/Low**	cmd Tags: Enable, State	BoilerEnable
	N/A		
Run	**High/Med/Low**	cmd Tags: Run, State	BoilerRun
	N/A		
Mode	N/A	cmd Tags: Run, State	OccupancyMode
Status	**High/Med/Low**	sensor Tags: Run	RefrigerantPumpStatus
	Blank		
*****	**Feedback**	sensor pointQuantity: Baseline Point Class	ZoneSupplyFanRotationalSpeedFeedback
N/A	**Stage**	cmd Tags: Stage	BoilerStage
N/A	**Set Point**	Sp pointQuantity: Baseline Point Class	ExhaustAirFlowSetpt
*****	N/A	cmd or sensor PointQuantity: Baseline Point Class	ZoneExhaustFanSpeed
Load	N/A	sp	ZoneCompressorPartLoadFraction

* Variety of baseline ontology measures, ex: position, temperature, speed, flow, etc.



■	Conditioning Mode, Loop Overlap, Natural Ventilation, Occupancy Override , Relief, Holiday, Pre-Heat , Superheating, Setback Status, Setup Mode, Guide Vane Cooling Rate, Fire Rate, Oxygen Fraction, pH, Suction Pressure, Vibration Amplitude, Volume	■	Heat Source
✳	Tracking Mode, Subcooling, Part Run Mode, All Run Mode, How Water Loop, Boiling Temp, Conductivity, Humidity Ratio, Carbon Monoxide	✳	Clean Steam
◆	Runtime, Tracking Status, Economizer, Humidifier, Thermal Energy Storage, Heating Rate, Velocity Pressure, Lubricant Oil, Return Water	◆	Medium, Dual Temp Coil
+	Free Cooling, Generator, Supply Water	+	Economizer
−	Filter, Pre Heat/Cool Coil	−	Low
◇	Part Load, Static Pressure	◇	High, Inlet Air
○	Dual Temp Loop	○	Rest
X	Enthalpy	X	Discharge Air
▲	Alarm	△	Generic Compressor
●	Heat Recovery	●	Enthalpy Wheel
		+	Primary/Secondary
		⬟	Position

Figure 13 : Semantic Gaps of Haystack Ontology Affecting Completeness Assessment Score

Alarms (▲)

The concept of alarms did not exist in the Haystack ontology. They could not be represented as Points, and a *Tag* defining alarm had not been created in publicly available libraries. This was a Major Gap because semantic data could not be represented any other way to describe Point Types referencing alarms**.**

Heat Recovery (●)

A semantic representation for heat recovery equipment did not exist in Haystack. They could not be represented as loops or standalone HVAC equipment. This was a Major Gap because heat recovery could not be represented any other way and rendered Point Types referencing this semantic data non-mappable.

Enthalpy Wheel (●)

Haystack represents the Heat Wheel equipment with a Def, however the related definition for this equipment is that of an enthalpy wheel. The actual "enthalpy wheel" equipment Def does not exist in Haystack. Because the definition for Heat Wheel was unclear it was not possible to classify this missing equipment as a Major Gap and was therefore classified as a Minor Gap.

Enthalpy (X)

The measure of enthalpy was missing from Haystack; however, enthalpy units were represented. This was a Major Gap because semantic data could only be known if inferred from the units of a data point. This is counter to the purpose of semantic data and rendered Point Types referencing the measurement of enthalpy non-mappable.

Primary/Secondary (+)

Haystack did not have *Tags* to differentiate between primary (chilled water plant) and secondary (chilled water distribution to terminal equipment) loops. Loops could be tagged with markers including *circ* to describe secondary or *CoolingLoop* and its subtypes *closedLoop* and *openLoop* to describe primary. However, this gap extended to other Point Types which also required primary and secondary differentiation and available *Tags* were not an exact semantic mapping therefore this was a Minor Gap.

Generic Compressor (▲)

Haystack represents multiple chillers Defs each with a different compressor mechanism (ex: *chiller-rotaryScrew, chiller-reciprocal,* etc.). However, these chiller Defs are not associated to a semantic representation of a compressor, rather tightly coupled with that of the chiller, a chiller with an unknow compressor could not be represented. This is a Minor Gap because the semantic data existed but could not be used in the necessary granularity to map to the baseline ontology.

***Position* (⬟)**

Damper and valve Point Types with a position Point Class as opposed to on/off functionality could not be represented in the Haystack ontology using PointQuantity. The baseline ontology uses a generic piece of semantic data "position" that was not available in the Haystack ontology and was therefore a minor gap.

Sub-Equipment Discharge Air (X)

The baseline ontology represented Point Types for sub-equipment such as heating and cooling coils. Some Point Types for sub-equipment specified the location of the medium relative to the sub-equipment (inlet, outlet, discharge, intake, etc.). Discharge Air was a frequent measure of sub-equipment whose semantic representation could not be clearly made in Haystack. The differentiation between the equipment medium location and sub-equipment medium location could not be made. This was a Minor Gap because *Tags* could be added to represent semantic information, but the relationships between semantic information could not be accurately formed.

4.1.2 Brick Completeness

Brick was able to represent a greater number of representative Point Types from the baseline ontology than Haystack. This was true across all subsystem considering Point Types classified as Maps, excluding the Boiler system which achieved the same value. Baseline ontology Point Types were easily mapped to Brick because of its intuitive and simple ontology pattern. Relationships were key in mapping the baseline ontology and are discussed below for the mapping of Point Classes and Measurement/Control Types. The results for the completeness of the Brick ontology is shown in Table 6.

Table 6 : Brick Completeness Results Against Baseline Representative Point Type Set

System	% Point Types with Maps Classification	% Point Types with Maps or Partially Maps Classification
AHU	56%	82%
Chiller	55%	60%
Boiler	74%	77%
Loop	42%	77%
Terminal Units	75%	84%
Total	59%	77%

Point Class and Measurement/Control Type mapping from the baseline ontology to Brick can be seen in Table 7. The Brick ontology is not tag based and exhibits a hierarchical nature therefor some Point Measures did not map directly in terms of vocabulary but did in terms of semantic representation. Run and enable Point Measures are examples of this indirect mapping. The baseline ontology specifies that an *enable* Point Class is a command that allows a system to run, while a *run* Point Class is a command that causes the system to run. An application of this would be that a heat recovery system has been issued an enable command but is not actively running because it has not been issued a run command as the zone temperature setpoint has been achieved. The Command Point Classes in Brick include an Enable_Command with a subclass of Run_Enable_Command, additionally there was an On/Off Command class with a Start_Stop_Command subtype. To respect the source ontology definition the Enable_Command and Start_Stop_Command in Brick were selected for mapping to the enable and run Point Classes respectively. Some Point Classes and Measurement/Control Types did not map to a single Class in Brick and required the use of relationships. For example, the Brick Valve_Command Class did not have a position-based Subclass as the Damper_Command did (Damper_Position_Command). To specify the semantic position data for Valve_Command a Valve equipment was created in the Brick ontology for the Point Type, the Point Type was then specified as a Valve_Command and using a Controls relationship linked to the brick quantity Position. The Controls relationship has no constraints on use making this possible. To maintain consistency the concept of creating sub-equipment was also applied to damper Point Types. This

approach also facilitated the mapping of the Feedback Measurement/Control Type commonly used Point Types relating to valve and dampers.

Table 7 : Baseline Ontology Point Class to Brick Point Class Mapping

Baseline Point Class	Baseline Measurement/Control Type	Brick Point Class Mapping	Baseline Point Type Example
*	Min/Max/Limit	Min_Limit or Max_Limit	ChillerRunLoadAmpsLimit SupplyAirTempMin
	N/A		
Enable	**High/Med/Low**	Enable_Command	BoilerEnable
	N/A		
Run	**High/Med/Low**	Start_Stop_Command	BoilerRun
	N/A		
Mode	N/A	Mode_Command	OccupancyMode
Status	**High/Med/Low**	Enable_Status or Start_Stop_Status or Mode_Status or Stage_Status	RefrigerantPumpStatus
	Blank		
*****	**Feedback**	Status	ZoneSupplyFanRotationalSpeedFeedback
N/A	**Stage**	*Non mappable*	BoilerStage
N/A	**Set Point**	Setpoint	ExhaustAirFlowSetpt
*****	N/A	Command or Sensor	ZoneExhaustFanSpeed
Load	N/A	Load_Setpoint	ZoneCompressorPartLoadFraction

* Variety of baseline ontology measures, ex: position, temperature, speed, flow, etc.

In the baseline ontology, *feedback* is differentiated from *status* to indicate that the value is numeric as opposed to Boolean as found in a *run, enable*, or *mode* command. However, in mapping to the Brick ontology, both status and feedback align with the Status Class. The Status Class is broken into Subclasses and therefore each command Point Type (*run, enable, mode, etc.*) required a unique mapping to the Brick ontology. To map a Point Type with a Feedback

Measurement/Control Type, a generic Status Class was used in addition to the Measures relationship specifying the type of numeric feedback (*speed, position, etc.*) using a Brick Measurable Class. This approach was also used for Sensor and Command Point Types mapped to Brick however the relationship used in the case of Command was Controls or Regulates.

As previously described, the use of ontology relationships was required in mapping semantic data from the baseline ontology to Brick. The Brick schema definition documentation defines relationship constraints. Additionally, rules were defined in publications made by major Brick contributors [9]. Table 8 shows the combination of documentation and publication defined relationship constraints in terms of Subject and Object, where ***bold italicized text*** are constraints specified in a recent publication [9], and plain text are constraints defined in the Brick Schema definition file. Blank cells indicate no relationship constraints exist. When relationships were required for mapping those meeting domain and range requirements specified in the Brick schema definition were first referenced, if these did not meet mapping requirements relationship constraints defined in [9] were used.

Table 8 : Brick Relationship Class Constraints as Found in "Schema Definition" and "***Ontology Author Publication"***

Subject (rdfs- domain)	Predicate	Object (rdfs range)
	hasLocation	Location
Location	**IslocationOf**	
Equipment ***Setpoint***	**Regulates**	Substance
Substance	**isRegulatedBy**	Equipment ***Setpoint***
Equipment	**hasInputSubstance**	Substance
Equipment	**hasOutputSubstance**	Substance
Point ***Sensor***	**Measures**	Measurable
Measurable	**isMeasuredBy**	Point ***Sensor***
	hasTag	Tag
Tag	**isTagOf**	
Equipment ***Location***	**hasPoint**	Point
Point	**isPointOf**	***Equipment*** ***Location***
*Upstream of object, sequential process, media passed between subject and object ***Equipment***	**Feeds**	***Equipment*** ***Location***
Equipment OR ***Location***	**isFedBy**	*Downstream of subject, sequential process, media passed between the subject and object ***Equipment***
****"When a point's value is used for another point's value determination, we say that the former one controls the later one"*** [9]	**Controls**	
	isControledBy	
*Composed in part by object entity ***Equipment*** ***Location***	**hasPart**	***Equipment*** ***Location***
Equipment *Location*	**isPartOf**	*Contains subject entity *Equipment* *Location*

Some semantic data representation is not available in the Brick ontology as required by the baseline ontology. Gaps were found as seen in Figure 14, Significant Gaps (those affecting more than 2% of Point Types in the baseline ontology) are discussed below. Generally, the Brick ontology has overall fewer gaps than the Haystack ontology, including fewer Major Significant gaps. Like the Haystack Ontology however, Brick has many Insignificant Gaps representing uncommon semantic concepts that are none the less necessary in the baseline ontology.

Heat Recovery (○)

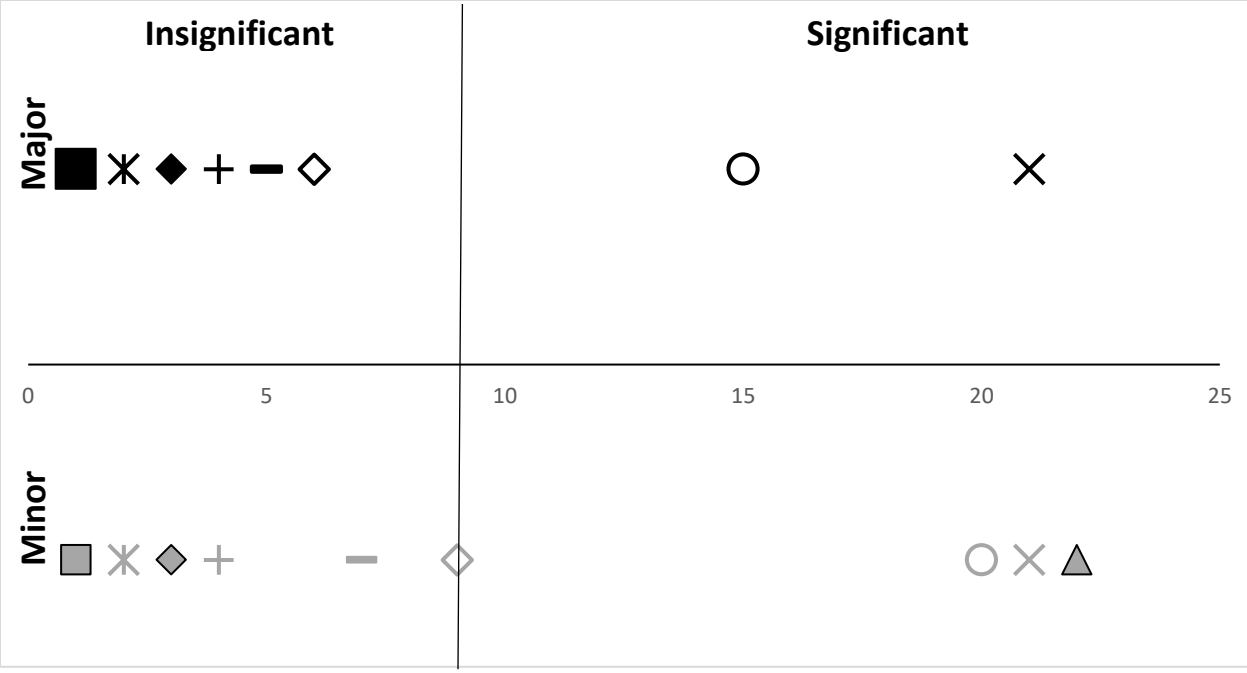
Heat Recovery equipment was missing from Brick. The ontology used the term Water System to describe loops, however there was no entity to describe heat recovery equipment as a concept. Furthermore. This was a Major Gap because the entity did not exist at all within the Brick ontology.

Refrigerant (X)

Refrigerant was missing in the Brick ontology. This medium is frequently referenced in the baseline ontology, specifically in the chiller system, both sensed and control points. This was a Major Gap because neither refrigerant nor a comparable substance was semantically represented in the ontology.

Primary/Secondary (▲)

Loops are represented by the Water_Systems_Equipment Class in the Brick ontology. This Class in the ontology only has the chilled and hot water Subclasses. There is no ability, including with the use of a Tag, to semantically differentiate primary and secondary loops in either Subclass. This gap extends to some other sub-equipment used in loops. This was a minor gap because Point Types relating to the loops could otherwise be semantically represented despite primary and secondary loops not being differentiated using a Condenser_Water Tag.



■	Fan Only, Holiday, Super Heating , Cooling Enable Outdoor Air, Setback Status, Setup Mode, Mixing Valve, Relief Damper, Fire Rate Illuminance, Oxygen Fraction, pH, Suction Pressure, Vibration Amplitude, Cooling Rate	▭	Occupancy Mode, Face Damper, Discharge Air
✳	Tracking Mode, Subcooling, Part Run/Mode , All Run, Boiling Temp, Efficiency, Humidity Ratio, Heating Rate, Clean Steam , Carbon Monoxide	✳	Volume Measure
◆	Stage Command, Tracking Status , Thermal Energy Storage Loop, Differential Pressure Measurable,	◆	Cold Deck, Hot Deck
+	Free Cooling, Generator	+	Return Air Reset
▬	Heat Source, Pre Heat/Cool Coil,	▬	Bypass Valve
◇	Radiant Terminal Unit, Process Water	◇	Inlet Air
○	Heat Recovery	○	Enthalpy Wheel
X	Refrigerant	X	None Setpoint Limit
		△	Primary/Secondary

Figure 14 : Gaps of Brick Ontology Affecting Completeness Assessment Score

Limit (✗)

Brick was not able to semantically represent Min, Max, and Limit Point Types unless they were set points or did not reference a substance. The baseline ontology had many representative data points which were Min, Max and Limit values referencing substances (inlet water, discharge air, etc.) that were not necessarily setpoints, these were internal parameters. The Brick ontology could semantically represented these Point Types as min_Limit and max_Limit , however a Controls relationship would then have to be used to indicate the substance the Point Type acts on, this is an inappropriate relationship because the Min/Max/Limit only indirectly acts on the substance through an equipment, which is not the intended use of the "controls" relationship that was created to model control logic in BMS systems. This was a Minor Gap because the semantic data could be represented in the Brick ontology, however the relationship used to add specificity of substance was not optimal.

Enthalpy Wheel (○)

The Brick ontology does have a Heat_Wheel_VFD Equipment Class, however there was not a specific concept for Enthalpy Wheel equipment. This was a Minor Gap as opposed to a Major Gap because the heat wheel concepts is defined in Brick and appears to be a superclass to what would be the subclass of enthalpy wheel.

Sub-Equipment Inlet Air (◊)

The concept of Entering or Inlet was reserved for water, there was no similar concept for air. This affected Point Types with measures specifically before an equipment. This was a Minor Gap because it did not affect whether or not the Point Type could be semantically represented, rather that the detail of the exact type of medium could not be represented.

4.1.3 Trends in Completeness

Significant Gaps were those which had the largest impacts on ontology completeness. Completeness sections of each Brick and Haystack offered detailed descriptions of Significant Gaps in semantic data representation; Figure 15 offers a visual representation of these individual and shared gaps.

In addition to shared Significant Gaps, the target ontologies shared both Major and Minor Insignificant Gaps, each impacted nine or less representative Point Types, the lists of which can be seen below. However, given that only representative Point Types from the baseline ontology were mapped it is possible that these gaps are more widespread within the baseline ontology than anticipated. Insignificant Gaps could have larger implications on completeness in the total completeness measure of either target ontology.

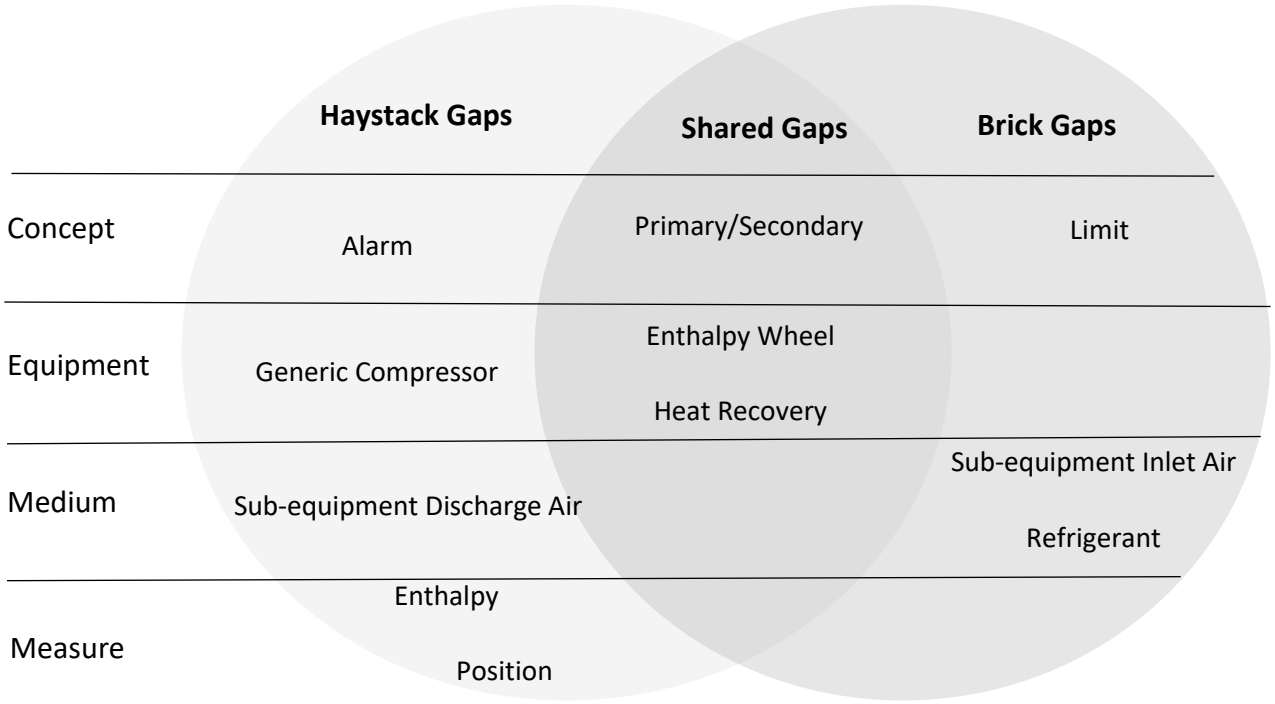


Figure 15 : Major and Minor Significant Gaps of Brick and Haystack Ontology

Shared Major Insignificant Gaps:

Missing Concepts

The following concepts are absent from both the Brick and Haystack ontologies, meaning that Point Types referencing these semantic concepts could not be represented for any equipment of systems.

- Part Run/Mode (a mode command indicating that a system run at partial capacity, e.g. run one of two fans)
- All Run (a command mode indicating that a system run at full capacity)
- Free Cooling (a command mode indicating that the AHU system does not run cooling coils)
- Holiday (a command mode indicating a system run as unoccupied)
- Setback Status
- Setup Mode (a command mode used to start up systems for the first time)
- Subcooling (a concept used in chiller controls)
- Superheating (a concept used in chiller controls)
- Tracking Mode (a command mode indicating that a system should use command to achieve the setpoint values)
- Tracking Status (the status of whether a system is using command to achieve the setpoint value)

Missing Equipment

The following equipment could not be semantically represented in either Brick or Haystack. Point Types relating to these equipment types could not be represented.

- Generator
- Thermal Energy Storage
- Pre Heat/Cool Coil

Missing Measures

The following measures are absent in both Brick and Haystack. Systems requiring commands, sensors, parameters using these measures could not be represented.

- Boiling Temp
- Cooling Rate
- Firing Rate
- Heating Rate
- Humidity Ratio
- Oxygen Fraction
- pH
- Suction Pressure
- Vibration Amplitude

Missing Substance

The following substances could not be represented as mediums which Point Types measured in Brick or Haystack.

- Carbon Monoxide

Shared Minor Insignificant Gaps

Missing Substance

The following semantic information describing a substance could not be represented in Brick or Haystack at the level of semantic detail indicated by the baseline ontology.

- Sub-equipment Discharge Air (discharge air of a sub equipment to a larger system which also acts on air and has a discharge component, this was largely an issue with air handling units)
- Sub-equipment Inlet Air (inlet air of a sub equipment to a larger system which also acts on air and has an inlet component, this was largely an issue with air handling units)

4.2 Expressiveness

As noted in the methodology, expressiveness was measured by quantifying a set of key relationships required by SOCx applications as found in the baseline ontology. The Brick ontology was found to be more expressive when target ontologies were compared using the set

of baseline ontology expressed key relationships. The Brick ontology explicitly defines relationships whereas the Haystack ontology more loosely implies a relationship between two Defs or sets of *Tags*. Respective relationships approaches follow either target ontologies pattern of defining semantic data where Brick Classes define composite HVAC concepts, and Haystack tags are used in collections to imply these concepts. Table 9 gives these detailed expressiveness results broken out by baseline system.

Table 9 : Haystack Expressiveness Results Against Expressed Baseline Key Relationships

✓ Brick ✓ Haystack	***AHU***		***Chiller***	***Boiler***	***Loop***	***Terminal Unit***		***Sub Equipment***
	Air Side	*Water Side*	*Water Side*	*Water Side*	*Water Side*	*Air Side*	*Water Side*	
Sensor ↔ Location	N/A					N/A		✓ ✓
Location ↔ Location	N/A					✓ ✓		N/A
Equipment ↔ Location	N/A					✓ ✓		✓ ✓
Sensor ↔ Equipment	✓ ✓	✓ ✓	✓ ✓	✓ ✓	✓ ✓	✓ ✓	✓ ✓	✓ ✓
Equipment ↔ Equipment	✓ ✓	✓ ✓	✓ ✓	✓ ✓	✓ ✓	✓ ✓	✓ ✓	✓ ✗
Location ↔ Persons	N/A					✓ ✓		N/A
Equipment ↔ Name	✓ ✓		✓ ✓	✓ ✓	✓ ✓	✓ ✓		✓ ✓

4.2.1 Haystack Expressiveness

Haystack relationships were able to describe almost all representative baseline ontology expressed key relationships: 96% of the 27 relationships assessed were classified as Maps. The

Haystack ontology schema relies on two bidirectional relationships (*contains/containedBy* and *receives/supplies*) to create relationships in the form of *Refs* and *Child Protos*. *Refs* are *Defs* used in Abox ontologies to create *Conjuncts* (sets of *Defs*) with the structure "Def *contains* the Def that Ref points to" relationship. *Child Protos* are *Conjuncts* defined using a "Def *contains* Child Proto" relationship. To classify a key relationship as Maps either an appropriate *Ref Def* needed to exist within a Haystack *library* (a collection of Defs), or an end point of a key assessed relationship (ex: equipment) needed to have a *Child Proto* indicated within the Def .

Haystack relationships are flexible and are defined in several vehicles within the schema. A common relationship used for Sensor ↔ Equipment relationship mapping was *equipRef,* this relationship is used in an Abox ontology to reference an equipment, which contains the sensor appropriate tag being used. Only ten other *Ref* relationships are defined in the Haystack ontology, of which *ahuRef, hotWaterPlantRef,* and *chilledWaterPlantRef* were used in mapping. In addition to *Ref* relationships *Defs* can have *Child Protos* defined, and are *contained* by a *Def. Child Protos* were used to map the Equipment ↔ Equipment air side relationship. Haystack relationships were able to bridge the gap between the air and water side of HVAC systems; however, this required the use of both *Ref* and *Child Proto* relationships which in practice would necessitate complex queries.

4.2.2 Brick Expressiveness

Brick relationships are well suited for describing representative baseline ontology expressed key relationships, 100% of the 27 relationships assessed were classified as Maps. The set of Brick relationships is clearly defined in a *Class* with constraints as described in Table 8. The Brick ontology relationships were not able to describe some Equipment ↔ Equipment relationships, specifically those relating to Loops.

Brick has nine bidirectional relationships, each with a defined inverse relationship. The explicit support of direct inverse relationships made expressiveness assessment simpler as only one use/direction of the relationship needed to be found to confirm the support of the bidirectional key relationship. For example, it was found that the Brick relationship *hasPoint* can be used to

represent the Boiler Equipment → Sensor relationship, therefore it is known that the inverse relationship *isPointOf* can be used to represent the Boiler Sensor → Equipment relationship. The most used Brick relationships in the expressiveness assessment were *hasPoint/isPointOf, hasPart/isPartOf,* and *feeds/IsFeedBy*. These relationships were used because their constraints aligned with end points in key relationships such as Equipment and Sensor. Other Brick relationships such as *measures* and *regulates* are better suited to relate more granular semantic concepts such as measurables.

A single Brick relationship could not describe some Equipment ↔ Equipment relationships, specifically those between loops and other HVAC systems. These relationships can be represented in Brick using the *feeds/isFedBy* relationship. The *feeds* relationship requires that it is used for a sequential process, and that a media is passed between the two end points. A loop is a repetitive process, and the media within the loop changes at different locations depending on the equipment the loop is interacting with; therefore, multiple relationships needed to be used to represent key relationships involving loops. For example, the Chiller↔ Loop could be represented with Chiller *feeds*→ Loop and Loop *feeds*→ Chiller. Where the first relationship is passing chilled water from the chiller to the loop, and the second is passing return warm water from the loop to the chiller.

4.3 Qualitative Analysis

The target ontology documentation indicated that the Brick ontology exhibited more desirable qualities than the Haystack ontology. The qualitative analysis assessed the target ontologies for six desirable qualities by answering qualifying questions, results can be seen in Table 10, and are discussed in respective subsections. Brick exhibited five of the six, where Haystack only exhibited three. The Brick schema represents entire HVAC systems concepts and is based in descriptive logic whereas Haystack has focuses on representing semantic data at a unit level with *Defs* that can be used in a collection (*Conjuncts*) to represent concepts. These fundamentally different ontology schema approaches trickle down to different Linked Data approaches documented on either ontology's open-source website [60, 53]

Table 10: Qualitative Results

Quality	*Qualifying Questions*	*Brick*	*Haystack*
Flexibility	Can the ontology capture uncertainty in semantic data and does it use non-restrictive methods to define concept semantically?		✓
Portability	Is semantic data represented consistently in a machine-readable format that is building agnostic?	✓	
Readability	Can domain experts and applications developers unambiguously decipher real world meaning from semantic data as presented in the ontology?	✓	
Extensibility	Can the ontology be customized to add new semantic concepts?	✓	✓
Interoperability	Can the ontology integrate with, and convert to, other ontologies using an industry accepted format with little to no human effort?	✓	
Queryability	Can the Abox ontology be machine traversed and necessary information retrieved? Is there low variability in semantic relationships?	✓	✓ *

*Haystack is only queryable through a custom *Filter* functionality, this does not meet the full "queryability" quality definition

4.3.1 Flexibility

Haystack was able to offer flexibility through the *Tag* based schema using *Defs*, which focuses on representing small units of semantic information. Alternatively, Brick has focused on representing complete HVAC concepts in *Classes* and *Subclasses*. *Tags* allow Haystack to represent uncertainty by allowing any number of *Tags* to be used for a semantic definition, including a limited number of *Tags* representing minimal semantic information. A minimal set of *Tags* could be "Temperature" and "Sensor", when the full semantic concept would also include the fact that "Air" is being *measured.* This freedom offers a non-restricted way of defining

semantic data, whereas Brick ensures concepts are prescribed with their more descriptive *Classes* composed of a set equivalent unit *Tags*. The flexibility exhibited by Haystack will allow the ontology to represent a wide variety of buildings typologies with HVAC systems that might fall outside of the norm. Flexibility will also allow for the semantic description of a subset of specific concepts within a building if the whole ontology is not desirable and beneficial to SOCx application.

Flexibility could decrease the portability of SOCx applications across buildings. For example, within Haystack the Point Tag has a non-mandatory marker pointFunction, which includes the choice of -cmd (command), sp (setpoint), or sensor (data value reading). This marker could be used to map the baseline Measurement/Control Type value of all Point Types. Alternatively, these values could be tagged with one of the Point subtypes cur-point, his-point, weather-point, or writable-point. These subtypes offer the option to define more detailed semantic data. Baseline ontology Point Types mapped to point *Tags* with defined *pointFunction* markers would be retrieved by a SOCx application using different queries than Point Types mapped to the appropriate Haystack *Point* subclass *Tags*, which do not have an assigned *pointFunction* since is not mandatory marker.

4.3.2 Portability

The Brick schemas representation of whole HVAC concepts ensures consistency across Abox ontologies facilitating SOCx application portability. Independent of building and system engineer implementing the Brick ontology, only a limited set of increasingly specific subclasses can be used to represent any HAVC system element. The same SOCx application can be used in multiple buildings using the Brick ontology because similar HVAC system components across building are guaranteed to use the same semantic definitions. Alternatively, as described in Section 4.3.1, Haystack's *Tag* based schema does not ensure consistency because it is flexible. Brick is machine readable because it is serialized in an industry accepted format (RDF). Haystack is machine readable as it is serialized in custom formats called Zinc and Trio.

4.3.3 Readability

Brick is a readable ontology because semantic data, including relationships, are explicitly defined as per its descriptive logic design. Semantic concepts are organized into a limited set of hierarchical Classes: Equipment, Point, Measurable, Relationships, and Location, setting a clear expectation of ontology use for end users. An instantiation of the Brick ontology leaves little room for user interpretation as classes represent whole semantic concepts. Alternatively Haystack's use of *Tags* leaves room for human interpretation where concepts can be represented in a variety of ways as described in Section 4.3.1. The potential for inconsistency in semantic representation within the Haystack ontology, yields an Abox ontology that is difficult to read and relate to real world meaning with confidence as an end user. As such an end user such as a Facilities Management professional could have difficulty querying the ontology to access time series building HVAC system data not directly available in the BAS.

4.3.4 Extensibility

Both ontologies are extensible, and therefore gaps found in ontology completeness can be resolved by adding semantic representations to the schema definition. The Brick ontology allows for concepts to be added by updating the "open-source" schema definition file. Concepts are added by naming the concept, placing it within the existing class hierarchy, and defining tags associated with the concept in a Python script used to build the schema definition (.ttl) file. Haystack allows their "open-sourced" ontology to be extended by defining new *Defs* directly in a Haystack library (.trio) file using the custom schema format. New Haystack *Defs* can also have relationships and be placed within the ontology hierarchy.

4.3.5 Interoperability

Brick is interoperable due to its supported serialization method using semantic web technologies described by W3C called the RDF format. The RDF format is the basis of ontology interoperability in the Linked Data approach described by W3C [73]. Other ontologies which support the RDF format can be integrated with Brick by cross-referencing common concepts, giving SOCx and other Smart applications more robust access to data .Additionally, Brick

defines tags related to each *Class* which can be used to directly convert an Abox Brick ontology to an Abox Haystack ontology. Haystack is currently in the process of building functionality to serialize to the RDF format. However, this is not yet supported; instead, the ontology is currently serialized in a custom format, which does not facilitate widespread ontology interoperability.

4.3.6 Queryablility

W3C semantic web technologies supported by Brick include a query language called SPARQL, a computer readable prescribed format for retrieving data stored in an RDF file. Additionally, Brick uses a small set of nine bidirectional relationships. This facilitates the consistent retrieval of semantic data given the limited number of relationships and therefore low complexity queries that need to be written. Haystack documentation describes a query method ("*Filter*s") that allows for the retrieval of semantic data in their custom file format, however it uses basic logic to query Abox ontologies This approach is less sophisticated than the semantic web query technology found in SPARQL supported by Brick. Furthermore, Haystack does not use a consistent set of relationships between concepts meaning that *Filter*s would need to include all possible relationship variations to ensure data recall. Without accurate data recall SOCx applications would not be able to access timeseries building data via the ontology.

5 Discussion

Overall, the results found Brick to be the superior ontology in representing, expressing, and using semantic information required by SOCx applications. Brick and Haystack ontologies used fundamentally different schemas. Haystack is based on object-oriented design while Brick is grounded in descriptive logic [74]. Differences in schemas are what positions either target ontology for optimal use in different scenarios. Despite limitations due to scope this research offers insight into the strengths and weaknesses of both Brick and Haystack, it also lays the groundwork for the selection of an ontology to support the development of SOCx applications. Further research, particularly in the area of ontology inference potential, is required to more completely understand the viability of industry usability of either target ontology.

Brick offered a more complete ontology where a greater number of semantic concepts were represented. There were fewer Significant gaps in Brick and ultimately it would require fewer custom semantics to be defined to accurately represent a building. However, both ontologies exhibited many gaps. Haystack exhibited eight Significant gaps, six being unique and included the lack of alarms and generic compressors, additionally it could not represent the measurement of enthalpy, position, and sub equipment discharge air. Three of Haystack's Significant gaps overlapped with those in Brick and included the lack of description for enthalpy wheel and heat recovery equipment, as well as the ability to differentiate between primary and secondary equipment. Brick exhibited six Significant gaps, three as previously stated and three of which were unique and included the lack of a refrigerant substance, subequipment inlet air, and limits. On a system level, loops were the most poorly represented, with the lowest completeness scores. SOCx applications will benefit from the higher completeness offered by Brick as a wider breadth of HVAC concepts can be described and therefore accessible by applications. Buildings using a more complete ontology could be compatible with a wider variety of SOCx applications, specifically those relying on clear and descriptive semantic data. While gaps in ontology schema can be filled as to not affect application effectiveness, their completion is preferred to avoid additional work in defining semantic concepts when developing a SOCx application.

Both target ontologies scored high in expressiveness, however Brick was able to express more key relationships than Haystack. Haystack used two bidirectional relationships to create *Refs* and *Child Protos* that relate various concepts but otherwise added no semantic information. Alternatively, Brick used a set of nine descriptive bidirectional relationships, each describing a different type of relationship between concepts. While Haystack relationships achieved a high score in expressiveness, they were not descriptive and as such the nature of the relationship could only be inferred. SOCx applications will benefit from the small but descriptive set of relationships used in Brick because they offer more semantic information, such as equipment sequencing, not provided in Haystack relationships. For example, Brick can differentiate a control logic command point from a command point using a *controls* or *hasPoint* relationship, where Haystack can only indicate that an equipment *contains* or *receives* a command. However, some key relationships cannot be expressed by Brick, as such a building using Haystack will offer SOCx applications a more accurate representation of interconnected HVAC systems. Brick is only able to represent a precise subset of these interconnections, capturing additional complexities that Haystack is not able to.

Ultimately it was the qualitative assessment that indicated Brick was the superior ontology to support SOCx application development. Brick exhibited more desirable qualities than Haystack, embodying five of six as opposed to three. Where the quantitative assessment indicated that Brick and Haystack were close in terms of completeness and expressiveness the qualitative assessment clearly indicated that Brick was superior. Both ontologies support extensibility as such semantic gaps in either ontology can be filled by adding customizations to the ontology schema definition. However, extensions will not be supported in ontology updates and significant testing will be required between ontology versions to ensure compatibility with SOCx applications. Both ontologies also support queryability, allowing SOCx applications to systematically access semantic data using a standardized query method. However, it should be noted that Haystack uses *Filters*, a custom querying method to retrieve semantic data, this is as opposed to the semantic web technology SPARQL that Brick uses for querying. SPARQL allows for portable queries to be written in SOCx applications that can then be used in all buildings with an Abox Brick ontology. The support of SPARQL will also allow SOCx applications to have

more easily interpreted complex queries. SOCx application engineers will be able to quickly write and understand queries, and therefore understand building data used by the application as well as on which systems and equipment the application will act.

W3C semantic web technologies also afford Brick interoperability, allowing for easier integration with other ontologies described in the RDF format, such as security systems. Multiple data sources provide SOCx applications with more robust and holistic building data. A manual mapping of semantic concepts would be necessary to integrate additional building data source ontologies with Haystack. The larger implications of Haystack's lack of interoperability lie in the power of emerging Internet of Thing (IoT) devices being used in buildings (e.g. security systems, occupancy tracking with wearable technologies, and advanced lighting systems). Assuming IoT data uses an ontology serialized to the international W3C RDF format ass Brick is, it will not be easily available to SOCx applications relying on a Haystack. This is because independent IoT ontologies will need to be manually mapped to Haystack and a query processor other than Filters will need to be used to manage the two data sources for a SOCx application. Alternatively, Brick can easily be crossed referenced with other data source ontologies serialized to RDF. Additionally, the Brick schema allows for easy conversion of an Abox Brick ontology to an Abox Haystack ontology if an application is better suited to consume semantic data in this format; the converse (Haystack to Brick conversion) is not easily possible.

Portability and readability were supported by Brick because of its explicit hierarchical schema structure, ultimately these qualities facilitate SOCx application development because concepts are represented unambiguously and consistently. Portability and readability together allow for easier development of SOCx applications that are applicable to a wide number of buildings. Because Haystack is tag based and flexible semantic definitions for the same concept can differ between Abox ontologies. A SOCx application or system engineer will likely only know one semantic representation of a concept and will therefore not be able to access data described in an unknown way. Brick is prescribed and therefore only predictable ways to define semantic concepts are used, allowing applications to use standard queries and leaving no room for interpretation or the confusion of system engineers. However, the Brick structure is restrictive

and despite facilitating direct data access (ex: querying a digital twin) it does not allow for the same flexibility that Haystack does. Such flexibility will also allow for more obscure systems to be represented by the ontology, since any combination or permutation of tags can be used to represent a semantic concept.

The specific versions of Brick and Haystack used in this study (current as of April 2020) have not been compared in any other research. However, earlier versions of Brick and Haystack have been directly compared in two of the initial presentations of the Brick ontology [9, 59]. In the first instance, only qualitative differences between the ontologies are discussed [59]. That study had similar findings as the comparison made here, namely that the Haystack *Tag* based schema offers flexibility and the lack of support for semantic web technologies is a key difference. In the second instance [9], the ontologies were quantitatively compared for completeness and, as found in this study, Brick achieved a higher score. The second comparison also highlights the lack of specific Haystack relationships as compared to Brick and the impact this has on Haystack expressiveness. Other studies have compared Haystack to alternative ontologies and found comparable results with those reported in this study [10, 66]. One study indicated that Haystack was not able to represent the location ↔ location relationships [66], this is as opposed to what has been found in the above study results. The difference in results can be attributed to a difference in Haystack version assessed. Additionally, this previous assessment of Haystack found a significantly higher completeness result, however a different methodology was used to assess completeness, using three buildings as a baseline instead of an ontology. This research is in line with the results of previous research and confirms that Brick is a superior ontology to facilitate building data access to SOCx applications despite significant changes being made to both ontologies since their previous assessment. The initiative of Haystack to move from a data schema to a full ontology was not sufficient to overcome the strength of Brick. Given this context, the new contributions made by this thesis are the findings of Brick's superiority to Haystack, despite the recent evolution of Haystack from meta-model to ontology, using a broader set of evaluation criteria than previous published studies.

The Brick ontology is best suited to serve SOCx applications designed to be “plug and play” in a variety of buildings. The restrictive nature of defining semantic data within the Brick ontology and the support of SPARQL allow for applications to be developed generically and used in multiple buildings. Additionally, descriptive relationships and a higher completeness provided by Brick indicate its appropriateness for SOCx applications with sophisticated algorithms analyzing complete HVAC systems rather than individual parts. The Haystack ontology is best suited to serve applications that require only vague semantic information and are more focused on single systems since ontology relationships are not well suited to describe the specifics of relationships. SOCx applications developed for Haystack should be able to target a wider variety of building types due to the vagueness that can be used in semantic definitions.

5.1 Limitations

The scope of assessment was the major limiting factor in this research. Using representative sets from the baseline ontology was necessary to ensure a consistent and therefore precise assessment of both completeness and expressiveness. However, this led to the potential for inaccurate results. Additionally, the baseline ontology and target ontology versions acted as limitations to the study due to their fluid nature, results found in this research are only relevant to versions assessed.

The research used an ontology baseline as opposed to single or multiple buildings to increase the number of possible HVAC systems to be included in assessment. The baseline ontology was selected over others due to its availability; typically, commercially used ontologies for SOCx application purposes are not available for academic use. The KGS ontology is not publicly available and is not an industry standard but has been used by select Smart Building system vendors. This ontology was developed for the purpose of building equipment diagnostics, recommendations for energy savings, and occupant comfort condition ratings. SOCx applications outside of this scope might require additional semantic data which has not been covered in the baseline representative set assessed in this research.

The representative Point Types set could skew the completeness results as gaps could be more pervasive if the scale of research were increased to include the full baseline ontology. Similarly, the representative expressed key relationships were equally weighted for expressiveness assessment. However, some relationships could be more important to SOCx applications. In both completeness and expressiveness assessment concepts were mapped from the baseline ontology to either target ontology. A known limitation in data science presents in this approach as common concepts could be represented by different strings/words between ontologies. Directly mapping strings was therefore not used in this research, however, there remains a possibility that common concepts were represented with unknow strings/words and misclassified as "Does Not Map". This weakness has been mitigated in the study by mapping all known synonyms for HVAC concepts, and iteratively mapping Point Types as new information and synonyms were discovered.

The study considered beta versions of both Brick and Haystack, which are in a fluid state and are updated frequently. New versions will render completeness and expressiveness assessments out of date. However, it was necessary to select the most recently available ontology schema definitions in the initial phases of research to ensure a consistent standard was being measured and results could be replicable. This limitation also impacted the qualitative assessment as supported target ontology technologies can be updated, such as adding the support of semantic web technologies to Haystack.

5.2 Future Research Needs

To improve the comparison of Brick and Haystack their usability requires further research. This includes 1) further completeness, and expressiveness assessment with a wider variety of SOCx applications (ex: occupancy tracking), 2) the assessment of additional qualities impacting ontology performance, and 3) the exploration of possible inference methodologies for brownfield buildings. These additional assessments would help to build out optimal use cases of either ontology.

Further completeness and expressiveness assessment with additional SOCx applications should be complete. This could be accomplished by using alternative baseline ontologies in use by SOCx applications where available, or by using semantic concept standards. Assessing ontologies against additional baselines would shed light on concepts and relationships necessary to functionalities not covered in the above assessment of Brick and Haystack. Other studies comparing ontologies considered qualities not covered in the above assessment, further qualities could therefore be added in the qualitative assessment. Qualities could include *depth* [75], which described the vagueness of ontology concepts, and *data modalities* [22] including more qualitative semantic representations such as comfort measures and maintenance. A number of studies have also focused on expanding expressiveness assessment beyond relationship representation to involve quantitative querability [69, 13]. General queries could be extracted from these studies to improve expressiveness assessment and reach beyond relationships presence in an important next step.

The third area of research, inference development and assessment, was initially considered for this thesis and has thus been defined in significant detail as follows. Brick and Haystack should be assessed for their ability to be inferred for brownfield buildings. It is important that ontologies can be inferred and implemented in brownfields buildings due to the long lifespan of buildings, and the need for SOCx applications in the existing building stock. Machine learning approaches to infer building semantic data that require at least some manual work (e.g. creating training data sets) have included supervised learning [76], active learning(semi-supervised) [77] [78], and transfer learning [79] [80]. Within this research context, *supervised learning* includes classification algorithms that create models by taking building data point names equivalent to the baseline Point Types as input and produce semantic data as output. Input for these models are created using a subset of the building data points, called the training set, which have correctly identified semantic information [76]. This approach is not an acceptable solution because the training set requires a large amount of manual effort to semantically tag the training set. The *active learning* approach is similar to supervised, however it relies on a smaller more strategic training data set that requires input from a Subject Matter Expert (SME) to create and represent boundaries between types of metadata [81]. The success of active learning relies on discovering

the data points representing boundary conditions to include in the training set [78]. While this approach requires less effort than supervised learning alone, the accuracy of metadata identification is highly dependent of the types of data points in the building [82] and level of control achieved with the selecting boundary data points [81]. *Transfer learning* involves using an existing complete data set (source), a building which already has semantic data, to train a classification model that will identify semantic data of a target building [79]. All of these approached require some level of manual work.

Approaches that do not require manual work include controlled perturbation [83] , and weak supervision [67] [83]. *Controlled perturbation* involves sequentially commanding HVAC systems on and off in a building and using observed timeseries data as input for an unsupervised algorithm which can identify semantic data as output [84]. This approach requires that each point in a building HVAC system be adjusted and the results measured, which is impractical given the amount of time this would affect building operation. *Weak supervision* is an emerging machine learning approach which leverages general classification data(heuristics) such as scope docs, BIM, shop drawings, etc. to generate training data. No specific data points are labeled for the purpose of training data in this method [81], rather heuristics are used for labeling.

Resembling a weak supervision approach, Bhattacharya et al. [67] developed an approach (hereon referred to as ProgSyn) that involved the basic steps of 1) unsupervised clustering taking data point names as input, 2) selecting the most relevant data point for semantic identification in heuristic format by a Subject Matter Expert (SME), and 3) updating the classification algorithm to propagate semantic information to data points within the cluster. ProgSyn is similar to weak supervision because SMEs did not directly add semantic information, rather SMEs identified heuristic information. ProgSyn has been directly benchmarked against the active learning approach Scrabble [80] in the Plaster framework [84]. In this comparison Scrabble outperforms ProgSyn. This is possible because Scrabble uses an intermediary level between input and semantic data output, where ProgSyn infers output directly from input [84]. The unsupervised clustering approach in ProgSyn has also been compared to Zodiac [77] in a clustering effectiveness study [82]. This study indicated that Zodiac clustering was superior due to its use

of Manhattan distance as opposed to Jaccard distance [82]. An additional weak learning approach [83] used equipment and data point type libraries as heuristic input, however to achieve accurate semantic inference the approach relies on all equipment being implemented with exact digital and analogue input/output which is not realistic in a brownfield building.

Ultimately further research on weak supervision should be considered because research indicates that it requires less manual effort than other machine learning approaches to semantic data inference. Weak supervision is favored over pursuing the use of timeseries data in semi-supervised approaches as timeseries data has been shown to be an unreliable inference feature [77]. Time series data could conceivably be used in inference if single string based semantic data is being inferred [76], however this is not the type of semantic data required by Brick. If timeseries data is the only available heuristic data research has been completed to aid in feature selection of HVAC timeseries data [85]. ProgSyn offers an excellent weak supervised inference approach to build on in future research by adding an intermediary layer as found in Scrabble [80] and by using Manhattan distance instead of Jaccard distance [82].

5.3 Summary and Recommendation

Brick shows the most promise in supporting SOCx applications given assessments in completeness, expressiveness and key qualities exhibited. Ultimately the version of Brick assessed is better suited to facilitate SOCx applications due to its support of semantic web technologies and schema grounded in descriptive logic. Haystack is optimal for SOCx applications used in a wider variety of buildings where only vague semantic description is available.

Ontology versions assessed are in a beta phase and highly susceptible to change, greatly impacting the results of completeness, expressiveness, and the qualitative assessment. Continuous assessment of target ontologies against a set of baseline ontologies and/or semantic models is necessary to the optimal support of SOCx ontologies. This thesis is limited by its scope of comparison. Further research is needed in ontology inference viability for brownfield buildings to overcome one of the largest hurdles in the adoption of ontology: effort to implement.

6 Conclusions

This research examines the completeness, expressiveness, and key qualities of Brick and Haystack ontologies for the purpose of representing semantic information to be used by smart and ongoing commissioning (SOCx) applications. Assessment results indicate that Brick is preferable to Haystack because of its higher completeness, expressiveness, and exhibition of more desirable qualities. Despite a recommendation of Brick, both ontologies remain in a fluid state and continuous research is needed to advance and gain pervasive industry adoption of SOCx technologies.

Brick achieved higher scores in the assessment of completeness(77%), expressiveness(100%), and key qualities, indicating that it is better suited to describe semantic building data concepts and relate them in a communicative manner to facilitate the implementation and use of SOCx applications in a Linked Data architecture. The Brick schema is grounded in descriptive logic, facilitating the clear and concise description of semantic building data. While Haystack achieves comparable scores in completeness (69%) and expressiveness (96%), it was only able to achieve three of the six key qualities, where Brick achieved five.

The Brick schema was structured in a prescribed manner that allows for readability and portability, which was not supported by Haystack. Brick also used W3C semantic web technologies, including RDF and SPARQL, which facilitate the support of other key qualities – interoperability and queryability – and lead to an overall functional Linked Data approach. The success of Brick's Linked Data approach allows for heterogeneous data normalization of building data sources required by SOCx applications, which in turn facilitate reduced energy consumption and improved building HVAC system performance.

While this research provides direction for the selection of an ontology supporting SOCx applications, it should be recognized that these ontologies are in a fluid and evolving state, requiring constant reevaluation as requirements of SOCx applications change and ontology schemas grow and develop. Beyond selecting an ontology to support SOCx applications, the largest hurdle to mass adoption will be the implementation of ontologies in brownfield buildings.

The open question of “how ontologies can be implemented for brownfield buildings with minimal manual effort while offering high quality data normalization?” persists in current discourse. This thesis has however provided a firm footing for the conversation in asserting the aptitude of Brick in facilitating SOCx applications.

References

[1] OECD, "Forword," in *Cities and Climate Change*, Paris, OECD Publishing,, 2010, pp. 3-3.

[2] E. Corry, P. Pauwels, S. Hu, M. Keane and J. O'Donnell, "A performance assessment ontology for the environmental and energy management of buildings," *Automation in Construction,* pp. 249-259, 2015.

[3] Y. Li, R. García-Castro, N. Mihindukulasooriya, J. O'Donnell and S. Vega-Sánchez, "Enhancing energy management at district and building levels via an EM-KPI ontology," *Automation in Construction,* vol. 99, pp. 152-167, 2019.

[4] S. Hu, E. Corry, M. Horrigan, C. Hoare, M. Dos Reis and J. O'Donnell, "Building performance evaluation using OpenMath and Linked Data," *Energy and Buildings,* vol. 174, pp. 484-494, 2018.

[5] C. E. Kaed, B. Leida and T. Gray, "Building management insights driven by a multi-system semantic representation approach," in *2016 IEEE 3rd World Forum on Internet of Things (WF-IoT),* 2016.

[6] M. G. Damm, "Method and system to manage complex systems knowledge". United States of America Patent 8,595,258, 2013.

[7] B. East and M. Carrasquillo-Mangual, "The COBie Guide," National Institute of Building Sciences, Washington, 2013.

[8] P. Pishdad-Bozorgia, X. Gao, C. Eastman and A. P. Selfa, "Planning and developing facility management-enabled building information model (FM-enabled BIM)," *Automation and Construction,* vol. 87, pp. 22-38, 2018.

[9] B. Balaji, A. Bhattacharya, G. Fierro, J. Gao, J. Gluck, D. Hong, A. Johansen, J. Koh, J. Ploennigs, Y. Agarwal, M. Bergés, D. Culler, R. K. Gupta, M. B. Kjærgaard, M.

Srivastava and K. Whitehouse, "Brick : Metadata schema for portable smart building applications," *Applied Energy,* vol. 226, pp. 1273-1292, 2018.

[10] V. Charpenay, S. Käbisch, D. Anicic and H. Kosch, "An Ontology Design Pattern for IoT Device Tagging System," in *International Conference on the Internet of Things (IoT)*, Seoul, 2015.

[11] P. Pauwels, E. Corry and J. O'Donnell, "Representing SimModel in the Web Ontology Language," in *International Conference on Computing in Civil and Building Engineering*, Orlando, 2014.

[12] D. Couloumb, C. E. Kaed, A. Garg, C. Healey, J. Healey and S. Sheehan, "Energy efficiency driven by a storage model and analytics on a multi-system semantic integration," in *Big Data,* Boston, 2017.

[13] G. Fierro and D. E. Culler, "Design and Analysis of a Query Processor for Brick," *ACM Transactions on Sensor Networks,* vol. 1, no. 1, 2018.

[14] D. Moher, A. Liberati, J. Tetzlaff and D. G. Altman, "Preferred reporting items for systematic reviews and meta-analyses: The PRISMA statement," *International Journal of Surgery*, vol. 8, no. 5, pp. 336-341, 2010.

[15] I. ACM, "ACM digital library," 2019. [Online]. Available: https://dl.acm.org/.

[16] I. IEEE, "IEEE Xplore digital library," 2019. [Online]. Available: https://ieeexplore.ieee.org/Xplore/home.jsp.

[17] Inderscience Enterprises Ltd., "Inderscience publishers," 2019. [Online]. Available: https://www.inderscience.com.

[18] ITCdl, "ITC digital Library," 2003. [Online]. Available: http://itc.scix.net/.

[19] Elsevier, "Scopus," 2019. [Online]. Available: https://www.elsevier.com/solutions/scopus.

[20] Informa UK Limited, "Taylor & Francis Online," 2018. [Online]. Available: https://taylorandfrancis.com/online/taylor-francis-online/.

[21] E. Corry, P. Pauwels, S. Hu, M. Keane and J. O'Donnell, "A performance assessment ontology for the environmental and energy management of buildings," *Automation in Construction*, vol. 57, pp. 249-259, 2015.

[22] A. Mahdavi and M. Taheri, "An ontology for building monitoring," *Journal of Building Performance Simulation*, vol. 10, no. 5-6, pp. 499-508, 2017.

[23] W. S. Jeong and J. W. Son, "An object-oriented physical modeling (OOPM) approach using building information modeling to support building performance simulations," 2015.

[24] P. Pauwels, S. Zhang and Y.-C. Lee, "Semantic web technologies in AEC industry: A literature overview," *Automation in Construction*, vol. 73, pp. 145-165, 2017.

[25] M. H. Rasmussen, P. Pauwels, C. A. Hviid and J. Karlshøj, "Proposing a central AEC ontology that allows for domain specific extensions," in *2017 Lean and Computing in Construction Congress*, 2017.

[26] M. Asfand-e-yar, A. Kucera and T. Pitner, "Smart buildings: Semantic web technology for building information model and building management system," 2014.

[27] A. Kučera and T. Pitner, "Semantic BMS: Allowing usage of building automation data in facility benchmarking," *Advanced Engineering Informatics*, vol. 35, pp. 69-84, 2018.

[28] H. Pruvost, O. Enae-Roscnblatt and J. Haufe, "Information integration and semantic interpretation for building energy system operation and maintenance," in *IECON 2018 - 44th Annual Conference of the IEEE Industrial Electronics Society*, 2018.

[29] S. Borgo, E. M. Sanfilippo, A. Sojic and W. Terkaj, "Towards an ontological grounding of IFC.," 2014.

[30] T. Gerrish, K. Ruikar, M. Cook, M. Johnson and M. Phillip, "Attributing in-use building performance data to an as-built building information model for lifecycle building performance management," 2015.

[31] J. Han, Y. Jeong and I. Lee, "A rule-based ontology reasoning system for context-aware building energy management," 2015.

[32] M. Kadolsky, R. Windisch and R. J. Scherer, "Knowledge management framework for monitoring systems improving building energy efficiency," in *2015 IEEE Workshop on Environmental, Energy, and Structural Monitoring Systems (EESMS) Proceedings*, 2015.

[33] S. V. Pinheiro, E. Corry and J. O'Donnell, "Requirements for a BIM-Based life-cycle performance evaluation framework to enable optimum building operation," 2015.

[34] V. Marinakis and H. Doukas, "An advanced IoT-based system for intelligent energy management in buildings," *Sensors (Basel, Switzerland)*, vol. 18, no. 2, p. 610, 2018.

[35] N. Tamani, S. Ahvar, G. Santos, B. Istasse, I. Praca, P.-E. Brun, Y. Ghamri, N. Crespi and A. Becue, "Rule-based model for smart building supervision and management," in *2018 IEEE International Conference on Services Computing (SCC)*, 2018.

[36] B. Jayan, H. Li, Y. Rezgui, J.-L. Hippolyte, B. Yuce, C. Yang and I. Petri, "An ontological approach to intelligent energy management in building," in *EG-ICE conference*, 2014.

[37] Z. U. Shamszaman, S. Lee and I. Chong, "WoO based user centric energy management system in the internet of things," 2014.

[38] D. Schachinger and W. Kastner, "Ontology-based generation of optimization problems for building energy management," in *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2017.

[39] C. E. Kaed, A. Ponnouradjane and D. Shah, "A Semantic Based Multi-Platform IoT Integration Approach from Sensors to Chatbots," in *2018 Global Internet of Things Summit (GIoTS)*, 2018.

[40] A. Mallak, A. Behravan, C. Weber, M. Fathi and R. Obermaisser, "A graph-based sensor fault detection and diagnosis for demand-controlled ventilation systems extracted from a semantic ontology," in *2018 IEEE 22nd International Conference on Intelligent Engineering Systems (INES)*, 2018.

[41] L. Bottaccioli, A. Aliberti, F. Ugliotti, E. Patti, A. Osello, E. Macii and A. Acquaviva, "Building energy modelling and monitoring by integration of IoT devices and building information models," in *2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC)*, 2017.

[42] L. Daniele, F. den Hartog and J. Roes, "Created in Close Interaction with the Industry: The Smart Appliances REFerence (SAREF) Ontology," 2015.

[43] O. H. Uribe, M. Adil, M. C. Garcia-Alegre and D. Guinea, "A context-awareness architecture for managing thermal energy in an nZEB building," in *2015 IEEE First International Smart Cities Conference (ISC2)*, 2015.

[44] S. Ahvar, G. Santos, N. Tamani, B. Istasse, I. Praça, P. Brun, Y. Ghamri and N. Crespi, "Ontology-based model for trusted critical site supervision in FUSE-IT," in *2017 20th Conference on Innovations in Clouds, Internet and Networks (ICIN)*, 2017.

[45] G. F. Schneider, P. Pauwels and S. Steiger, "Ontology-based modeling of control logic in building automation systems," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 6, pp. 3350-3360, 2017.

[46] W. Terkaj, G. F. Schneider and P. Pauwels, "Reusing Domain Ontologies in Linked Building Data: the Case of Building Automation and Control," in *8th International Workshop on Formal Ontologies meet Industry*, 2017.

[47] I. Esnaola-Gonzalez and F. J. Dìez, "IoT Integration based on Semantic Technologies for Energy Efficiency in Buildings," in *2018 Global Internet of Things Summit (GIoTS)*, 2018.

[48] S. Malakuti, J. Schmitt and T. Gamer, "From heterogeneity to uniformity in building automation systems via semantic-based engineering," in *2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2018.

[49] A. Kučera and T. Pitner, "Semantic BMS: Ontology for Analysis of Building Automation Systems Data," in *7th Doctoral Conference on Computing, Electrical and Industrial Systems (DOCEIS)*, Costa de Camperica, Portugal, 2016.

[50] H. Neuhaus and M. Compton, "The semantic sensor network ontology," in *AGILE workshop on challenges in geospatial data harmonisation*, Hannover, 2009.

[51] G. Bajaj, R. Agarwal, P. Singh, N. Georgantas and V. Issarny, "A study of existing Ontologies in the IoT-domain," *arXiv preprint*, 2017.

[52] B. Bortoluzzi, I. Efremov, C. Medina, D. Sobieraj and J. J. McArthur, "Automating the creation of building information models for existing buildings," *Automation in Construction*, vol. 105, p. 102838, 2019.

[53] "About Project Haystack," [Online]. Available: https://project-haystack.org/about. [Accessed January 2020].

[54] "Defs," 24 October 2019. [Online]. Available: https://project-haystack.dev/doc/docHaystack/Defs. [Accessed Jan 2020].

[55] "Namespaces," 24 October 2019. [Online]. Available: https://project-haystack.dev/doc/docHaystack/Namespaces. [Accessed January 2020].

[56] "Relationships," 24 October 2019. [Online]. Available: https://project-haystack.dev/doc/docHaystack/Relationships. [Accessed January 2020].

[57] "File Formats," 24 October 2019. [Online]. Available: https://project-haystack.dev/doc/docHaystack/index. [Accessed January 2020].

[58] "Filters," 24 October 2019. [Online]. Available: https://project-haystack.dev/doc/docHaystack/Filters. [Accessed January 2020].

[59] B. Balaji, A. Bhattacharya, G. Fierro, J. Gao, J. Gluc, D. Hong, A. Johansen, J. Koh, J. Ploennigs, Y. Agarwal, M. Berges, D. Culler, R. Gupta, M. B. Kjærgaard, M. Srivastava and K. Whitehouse, "Brick: Towards a Unified Metadata Schema For Buildings," in *Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, Palo Alot, 2016.

[60] "Community," 2019. [Online]. Available: https://brickschema.org/community. [Accessed January 2020].

[61] "Home," 2019. [Online]. Available: https://brickschema.org/#home. [Accessed April 2020].

[62] pwnall, "google/LevelDB," Google, 13 April 2020. [Online]. Available: https://github.com/google/leveldb. [Accessed April 2020].

[63] M. Andersen, "BTrDB: Berkeley Tree Database," [Online]. Available: http://btrdb.io/. [Accessed April 2020].

[64] D. Vrandečić, "Ontology evaluation," in *Handbook on Ontologies*, S. Staab and R. Studer, Eds., Berlin, Heidelberg, Springer Berlin Heidelberg, 2009, pp. 293-313.

[65] J. Brank, M. Grobelnik and D. Mladenic, "A survey of ontology evaluation techniques," in *Proceedings of the conference on data mining and data warehouses (SiKDD 2005)*, 2005.

[66] A. Bhattacharya, J. Ploennigs and D. Culler, "Short Paper: Analyzing metadata schemas for buildings: The good, the bad, and the ugly," in *ACM International Conference on*

Systems for Energy-Efficient Buildings, Cities, and Transportation (BuildSys), Seoul, 2015.

[67] A. A. Bhattacharya, D. Hong, D. Culler, J. Ortiz, K. Whitehouse and E. Wu, "Automated Metadata Construction To Support Portable Building Applications," in *International Conference on Systems for Energy-Efficient Buildings*, Seoul, 2015.

[68] G. Fierro, M. Pritoni, M. AbdelBaky, P. Raftery, T. Peffer, G. Thomson and D. E. Culler, "Mortar: An Open Testbed for Portable Building Analytics," in *ACM International Conference on Information and Knowledge Management (BuildSys)*, Shenzen, 2018.

[69] C. E. Kaed and M. Boujonnier, "FOrT ´ E: A Federated Ontology and Timeseries query Engine," in *iThings*, Exeter, 2017.

[70] "Downloads," 24 October 2019. [Online]. Available: https://project-haystack.dev/download. [Accessed December 2019].

[71] "Brick/Brick.ttl," 21 Febuary 2020. [Online]. Available: https://github.com/BrickSchema/Brick/blob/master/Brick.ttl. [Accessed Febuary 2020].

[72] B. Frank, "VRF support status & v4 release timeline," 13 December 2019. [Online]. Available: https://project-haystack.org/forum/topic/771. [Accessed Febuary 2020].

[73] W3C, "SEMANTIC WEB," 2015. [Online]. Available: https://www.w3.org/standards/semanticweb/. [Accessed March 2020].

[74] J. J. Bender, "Will Haystack 4 substitute Brick and Haystack 3?," Google Groups, 1 12 2019. [Online]. Available: https://groups.google.com/forum/#!topic/brickschema/Hpm8QDruJ4I. [Accessed 2020].

[75] B. Butzin, F. Golatowski and D. Timmermann, "A Survey on Information Modeling and Ontologies in Building Automation," in *43rd Annual Conference of the IEEE Industrial Electronics Society*, Beijing , 2017.

[76] J. Gao, J. Ploennigs and M. Bergés, "A Data-driven Meta-data Inference Framework for Building Automation Systems," in *ACM International Conference on Information and Knowledge Management (BuildSys)*, Seoul, 2015.

[77] B. Balaji, C. Verma, B. Narayanaswayy and Y. Agarwal, "Zodiac: Organizing Large Deployment of Sensors to Create Reusable Applications for Buildings," in *ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation(BuildSys)*, Seoul, 2015.

[78] D. Hong, H. Wang and K. Whitehouse, "Clustering-based Active Learning on Sensor Type Classification in Buildings," in *ACM International Conference on Information and Knowledge Management (CIKM)*, Melbourne, 2015.

[79] D. Hong, H. Wang, J. Ortiz and K. Whitehouse, " The Building Adapter: Towards Quickly Applying Building Analytics at Scale," in *ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, Seoul, 2015.

[80] J. Koh, B. Balaji, D. Sengupta, J. McAuley, R. Gupta and a. Y. Agarwal, "Scrabble: Transferrable Semi-Automated Semantic Metadata Normalization Using Intermediate Representation.," in *ACM International Conference on Information and Knowledge Management (BildSys)*, Shenzhen, 2018.

[81] A. Ratner, P. Varma, B. Hancock and C. Ré, "Weak Supervision: A New Programming Paradigm for Machine Learning," The Stanford AI Lab Blog, Stanford, 2019.

[82] Z. Shi, G. R. Newsham, L. Chen and H. ,. Gunay, "Evaluation of Clustering and Time Series Features for Point Type Inference in Smart Building Retrofit," in *ACM International Conference on Systems*, New York, 2019.

[83] F. Leonardi, H. Reeve, T. Wagner, Z. Xiong and J. Park, "Assisted Point Mapping to Enable Cost-effective Deployment of Intelligent Building Applications," in *International High Performance Buildings Conference*, West Lafayette, 2016.

[84] J. Koh, D. Hong, R. Gupta, K. Whitehouse, H. Wang and Y. Agarwal, "Plaster: an integration, benchmark, and development framework for metadata normalization method," in *ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation (BuildSys)*, Shenzhen, 2018.

[85] J. Gao and M. Bergés, "A large-scale evaluation of automated metadata inference approaches on sensors from air handling units," *Advanced Engineering Informatics*, vol. 27, pp. 14-30, 2018.

[86] Y. Arayici, T. Onyenobi and C. Egbu, "Building Information Modelling (BIM) for Facilities Management (FM): The Mediacity Case Study Approach," *International Journal of 3-D Information Modeling*, vol. 1, pp. 55-73, 2012.

[87] R. Volk, J. Stengel and F. Scultmann, "Building Information Modeling (BIM) for existing buildings — Literature review and future needs," *Automation in Construction*, vol. 38, pp. 109-127, 2014.

[88] P. Love, J. Matthews, I. Simpson, A. Hill and O. Olatunji, "A benefits realization management building information modeling framework for asset owners," *Automation in Construction*, vol. 37, pp. 1-10, 2014.

[89] D. Bryde, M. Broquetas and J. M. Volm, "The project benefits of building information modelling (BIM)," *International Journal of Project Managmeent*, vol. 31, no. 7, pp. 971-980, 2013.

[90] C. Preidel, S. Daum and A. Borrmann, "Data retrieval from building information models based on visual programming.," *Visualization in Engineering*, vol. 5, no. 1, pp. 1-14, 2017.

[91] G. Fierro and D. E. Culler, "Demo Abstract: HodDB - a Query Processor for Brick," in *BuildSys*, Delft, 2017.

[92] M. Kassem, G. Kelly, N. Dawood, M. Serginson and S. Lockley, "BIM in facilities management applications: a case study of a large university complex.," *Built Environment Project and Asset Managemen,* vol. 5, no. 3, pp. 261-277, 2015.

[93] M. Khaja, J. D. Seo and J. J. McArthur, "Optimizing BIM Metadata Manipulation Using Parametric Tools," in *Procedia Engineering*, Tempe, 2016.

[94] D. Quan, D. Huynh and D. Karger, "Haystack: A platform for authoring end user semantic web applications," in *International Semantic Web Conference*, Berlin, 2003.

[95] T. Kang and C. Hong, "A study on software architecture for effective BIM/GIS-based facility management data integration.," *Automation in Construction,*, pp. 25-38, 2015.

[96] M. Schmidta and C. Åhlund, "Smart buildings as Cyber-Physical Systems: Data-driven predictive control," *Renewable and Sustainable Energy Reviews*, pp. 742-756, 2018.

[97] J. Kleissl and Y. Agarwal, "Cyber-Physical Energy Systems:," in *Design Automation Conference*, Anaheim, 2010.

[98] ASHRAE, *ASHRAE Guideline 36-2018: High Performance Sequences of Operation for HVAC Systems*, Atlanta: American Society for Heating, Refrigeration and Air-Conditioning Engineers, 2018.

[99] J. McArthur, N. Shahbazi, R. Fok, C. Raghubar, B. Bortoluzzi and A. An, "Machine learning and BIM visualization for maintenance issue classification and enhanced data collection," *Advanced Engineering Informatics*, pp. 101-112, 2018.

[100] A. C. Tahir and R. Bañares-Alcántara, "A knowledge representation model for the optimisation of electricity generation mixes," *Applied Energy*, vol. 97, pp. 77-83, 2012.

[101] G. Fierro, J. Koh, Y. Agarwal, R. K. Gupta and D. E. Culler, "Beyond a House of Sticks: Formalizing Metadata Tags with Brick," in *BuildSys*, New York, 2019.

[102] B. Ramprasad, J. J. McArthur, M. Fokaefs, C. Barna, M. Damm and M. Litoiu, "Leveraging existing sensor networks as IoT devices for smart buildings," in *2018 IEEE 4th World Forum on Internet of Things (WF-IoT)*, 2018.

[103] "Associations," 24 October 2019. [Online]. Available: https://project-haystack.dev/doc/docHaystack/Associations. [Accessed January 2020].