Title

A semantics-driven framework to enable demand flexibility control applications in real buildings

Permalink

https://escholarship.org/uc/item/82g5v1f4

Authors

de Andrade Pereira, Flavia
Katsigarakis, Kyriakos
Rovas, Dimitrios
et al.

Publication Date

2025-03-01

DOI

10.1016/j.aei.2024.103049

Peer reviewed

Title

A semantics-driven framework to enable demand flexibility control applications in real buildings

Permalink

https://escholarship.org/uc/item/82g5v1f4

Authors

de Andrade Pereira, Flavia
Katsigarakis, Kyriakos
Rovas, Dimitrios
et al.

Publication Date

2025-03-01

DOI

10.1016/j.aei.2024.103049

Advanced Engineering Informatics 64 (2025) 103049

Contents lists available at ScienceDirect

Advanced Engineering Informatics

journal homepage: www.elsevier.com/locate/aei

Full length article

A semantics-driven framework to enable demand flexibility control applications in real buildings

Flavia de Andrade Pereira [a,b,c,*], Kyriakos Katsigarakis [b], Dimitrios Rovas [b], Marco Pritoni [c], Conor Shaw [a], Lazlo Paul [c], Anand Prakash [c], Susana Martin-Toral [d], Donal Finn [a], James O'Donnell [a]

[a] School of Mechanical & Materials Engineering and UCD Energy Institute, University College Dublin, Dublin, Ireland
[b] University College London, London, United Kingdom
[c] Lawrence Berkeley National Laboratory, Berkeley, United States of America
[d] CARTIF Technology Centre, Energy Division, Valladolid, Spain

ARTICLE INFO

Dataset link: https://github.com/ucl-sbde/semantics-driven_controls.git

Keywords:
Demand flexibility controls
Semantic models
Interoperability
Modularity
Brick
SAREF

ABSTRACT

Decarbonising and digitalising the energy sector requires scalable and interoperable Demand Flexibility (DF) applications. Semantic models are promising technologies for achieving these goals, but existing studies focused on DF applications exhibit limitations. These include dependence on bespoke ontologies, lack of computational methods to generate semantic models, ineffective temporal data management and absence of platforms that use these models to easily develop, configure and deploy controls in real buildings. This paper introduces a semantics-driven framework to enable DF control applications in real buildings. The framework supports the generation of semantic models that adhere to Brick and SAREF while using metadata from Building Information Models (BIM) and Building Automation Systems (BAS). The work also introduces a web platform that leverages these models and an actor and microservices architecture to streamline the development, configuration and deployment of DF controls. The paper demonstrates the framework through a case study, illustrating its ability to integrate diverse data sources, execute DF actuation in a real building, and promote modularity for easy reuse, extension, and customisation of applications. The paper also discusses the alignment between Brick and SAREF, the value of leveraging BIM data sources, and the framework's benefits over existing approaches, demonstrating a 75% reduction in effort for developing, configuring, and deploying building controls.

1. Introduction

Several countries worldwide are committed to ambitious climate goals for 2030 and 2050, driving a dual transition towards decarbonisation and digitalisation of the energy sector [1,2]. Decarbonisation efforts have been made by leveraging renewable energy sources and the electrification of heating and cooling. However, their inherent variability requires flexible consumption strategies to maintain a reliable balance between supply and demand while minimising curtailment of renewable sources [3]. Orchestrating such a balance between supply and demand requires significant digitalisation efforts [4], which include tackling interoperability challenges to facilitate data exchange between domains along the energy value chain and at their application levels [5].

Given its significant share of the global energy demand, buildings emerge as a promising sector for supporting decarbonisation efforts through Demand Flexibility (DF) applications [6,7]. DF empowers consumers to manage their demand according to local climate conditions, user needs and grid requirements [3]. One of the most impactful DF applications in buildings involves implementing supervisory control strategies to shift and shed loads for Heating, Ventilation, and Air Conditioning (HVAC) systems based on grid signals (e.g., prices or emissions) [8]. Despite the potential of this approach, most of the available research relies only on simulation, failing to capture nuances associated with real-world implementation [9]. This limitation is partly due to the lack of interoperable Building Automation Systems (BAS) in real buildings [1,10], which requires resource-intensive processes for configuring and implementing advanced supervisory control applications, including DF [11].

BAS often rely on various data inputs, formats, vendors, and communication protocols to monitor and control building systems. This heterogeneity makes it difficult for building applications, especially

---

* Corresponding author at: School of Mechanical & Materials Engineering and UCD Energy Institute, University College Dublin, Dublin, Ireland.
E-mail address: flavia.deandradepereira@ucdconnect.ie (F.d.A. Pereira).

https://doi.org/10.1016/j.aei.2024.103049
Received 1 May 2024; Received in revised form 19 October 2024; Accepted 9 December 2024

supervisory controls deployed at higher levels of the BAS hierarchy, to seamlessly integrate and exchange data with these systems to oversee and coordinate their operation [12]. Although software solutions such as gateways facilitate BAS data exchange, onboarding (configuring) new applications still predominantly rely on manual processes [13,14]. Application developers often depend on BAS natural language descriptions (metadata) of building systems and their data points to discover, map and integrate required inputs and outputs [15]. The issue is that such descriptions are usually bespoke and ambiguous and lack relationships with building topology and equipment. Consequently, developers must refer to supplementary documentation (e.g., technical drawings and manuals) or seek assistance from building operators [16]. This often leads to ad-hoc and building-specific applications, requiring repetitive and costly configuration efforts.

The industry and research community have acknowledged the potential of semantic models to support building applications and smart-grid use cases [17–19]. While the term "semantic models" is sometimes used interchangeably with ontologies, this paper specifically refers to such models as instances of ontologies that encode building domain knowledge in a machine-readable manner [17]. By representing data from specific buildings, these ontology-based models enhance data interoperability, discoverability, and mapping. The improved interoperability is achieved by enabling diverse sources to share a unified understanding of building data [14]. Meanwhile, the discoverability and mapping are facilitated by providing semantic access to heterogeneous data, allowing applications to execute uniform queries across the models from different buildings in order to discover and map their data points [20]. This enables applications to abstract building-specific intricacies, such as ad-hoc naming conventions, automating their configuration and ultimately improving their portability [21,22].

The use of semantic models has been extensively investigated for analytic purposes in various studies [23–28], yet their application in control contexts remains relatively unexplored. Aside from that, while semantic models offer significant value, they do not entirely address challenges in point discovery and mapping, as these tasks are still needed for generating the models in the first place [29]. This makes it crucial to develop computational methods that automate such generation. Existing studies have explored automating the discovery and mapping of ontology concepts from BAS point lists to generate semantic models. [30,31]. However, because BAS data sources often lack contextual metadata (e.g., relationships between points, spaces, and equipment), manual effort and expertise are still needed to organize the metadata before generating the model. To address this, recent research has proposed the use of Building Information Models (BIM) to complement BAS data and further automate the generation of semantic models [29,32,33]. BIM are increasingly prevalent in the industry [34] and offer comprehensive data on buildings' physical and functional aspects, which can reduce the need for human interpretation and intervention in establishing connections between BAS points and their contextual metadata (e.g., related location) [24,35,36]. Nonetheless, these studies using both BIM and BAS have not considered the data required for DF controls nor discussed the ontologies that are suitable for them.

Moreover, the benefits of investing in methods to generate semantic models only become evident when these models are consistently used across various applications [37]. This is fully realised when the models are built upon ontologies that can meet the data needs of various applications and when they are based on established ontologies that are more likely to achieve widespread adoption in various buildings. Well-established ontologies for building applications include the Building Topology Ontology (BOT),[1] RealEstateCore (REC),[2] Brick,[3] Smart Appliances REFerence Ontology (SAREF),[4] Semantic Sensor Network (SSN) and Sensor, Observation, Sample, and Actuator (SOSA)[5] [14]. While each of them has a different purpose, there is some overlap in their scope, and collectively, they can model buildings, including devices (e.g., chillers), measurement and control points (e.g., sensors and setpoints) and topological and spatial relationships. In evaluating the vocabulary (terms and concepts) within these and related ontologies, a prior review by the authors indicated that Brick and SAREF combined may effectively meet the data needs for DF applications, particularly from the building perspective [38]. Compared to other ontologies, Brick offered a more comprehensive formal structure, providing expressive semantics for capturing operational and spatial aspects of building systems. In contrast, SAREF, along with its various extensions, offered significant potential in modelling numerous aspects related to DF. However, although both ontologies have been independently investigated to support DF applications [39–42], their alignment and comparison beyond their vocabulary have not yet been explored.

This paper aims to introduce a framework that uses semantic models to support DF supervisory control applications in real buildings. We expand upon our previous research [40], which proposed a framework to enable the portability of DF control applications using Brick-based semantic models. In this paper, we present a novel alignment between Brick and SAREF to accommodate and assess DF applications based on both ontologies. We also provide open-source algorithms as computational methods to automate the generation of semantic models. The models adhere to concepts from the two ontologies and map metadata from BIM and BAS sources. Moreover, we introduce a web platform to develop, configure and deploy DF controls in real buildings while leveraging their semantic models. The platform follows an actor and microservices architecture [43,44], enhancing the benefits of semantic models by allowing a modular design for controls. This approach simplifies decoupling control logic from building-specific data access methods and metadata, such as data point descriptions, and enables applications to be easily reused, extended, and customised. While developing this, our paper addresses the following research questions:

1. How can Brick and SAREF concepts be aligned and fulfil the data needs of DF applications?
2. How can BIM, through the use of semantic models, facilitate the configuration and deployment of DF applications?
3. What are the benefits of deploying DF applications using the proposed framework?

The remainder of this paper is structured as follows: Section 2 provides a literature review on related works. Section 3 describes the proposed methodology to deploy our extended framework. Section 4 demonstrates the implementation of the semantics-driven framework in a real building. Section 5 discusses the case study findings and examines the research questions. Finally, Section 6 outlines concluding remarks and future research directions.

2. Literature review

Many existing initiatives propose reference infrastructure and conceptual architectures to assist in managing, exchanging, or securing data across different domains. One such initiative is the Data Exchange Reference Architecture (DERA) 3.0, introduced by the Bridge initiative and supported by the European Commission. DERA 3.0 aims to provide a systematic foundation for developing standard smart grid solutions, including DF [45]. DERA 3.0 uses the Smart Grid Architecture Model (SGAM) dimensions to define potential data exchanges

---

[1] https://w3c-lbd-cg.github.io/bot/.

[2] https://dev.realestatecore.io/ontology/.

[3] https://brickschema.org/ontology.

[4] https://saref.etsi.org/core/v3.1.1/.

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

between multiple building and grid systems [46]. Among the SGAM dimensions, the interoperability dimension represents information management through multiple layers (component, communication, information, function and business). Different interoperability layers should be considered according to the use case and business area of a smart grid solution.

This paper explores the use of semantic models within the SGAM information layer to facilitate an interoperable and scalable deployment of HVAC-based DF applications in buildings. To support this work, we review previous academic studies covering the new and existing ontologies that can support data needs in DF applications, the different methods available for generating semantic models based on these ontologies, and existing semantics-driven frameworks for such applications. Each of these topics is approached in the subsections of this literature review, followed by an analysis of the existing research gaps and a summary of the proposed contributions of this paper.

2.1. Ontologies for demand flexibility

Semantic models are built on (or instantiated from) ontologies that rely on technologies such as Resource Description Framework (RDF),[6] Web Ontology Language (OWL)[7] and Shapes Constraint Language (SHACL).[8] Through the ontologies, they aim to formalise the representation of domain knowledge and link heterogeneous data in a meaningful, consistent and machine-readable manner [47]. To be effective, though, the underlying ontologies of the semantic models must cover the data needs for the applications they are supporting [48]. While each DF application may have unique data requirements, prior studies on characterising and specifying DF controls in buildings have identified a common set of data inputs and outputs [14,38,40,49–54]. These typically include BAS data points and configuration parameters, such as indoor environment metrics, power and energy data, preferred setpoints, weather conditions, scheduling information, activation timing, threshold values triggering control actions, DF operation parameters and control commands. Capturing these types of data and their contextual metadata (e.g., related location or equipment) within ontologies is critical to ensure that semantic models can help configure and deploy DF applications.

Several studies have introduced new or expanded existing ontologies to support DF [12,41,55–64]. One such example is the OpenADR ontology [57], which was created to model grid communication concepts based on the internationally recognised OpenADR standard (v2.0) for Automated Demand Response. However, some of these ontologies are poorly documented or undisclosed, which hinders their widespread reuse [12,55,56,58,59,61]. Even when they are open and well-documented, many, including OpenADR, lack tools for instantiation or maintenance support from a consortium or standardisation body.

Furthermore, many custom-built ontologies for DF fall short in terms of expressivity or alignment with other schemas to represent HVAC system information as needed to support DF controls for HVAC effectively. Although Hviid et al. (2022) acknowledge the use of Brick for building information representation, their work focuses on introducing a novel ontology for describing related building analytic applications (e.g., weather prediction, flexibility estimation) rather than providing comprehensive alignment with Brick [61]. As such, there is a need to investigate the use of more established and expressive ontologies that can represent HVAC-specific information for DF controls, along with tools and methodologies that facilitate their generation into semantic models.

2.2. Brick and SAREF ontologies

Brick and SAREF are well-established ontologies that have been explored to support DF applications. Brick has gained significant traction in the research community across diverse use cases [14,17,22,27,37,65] and its modelling concepts have been integrated with the upcoming ASHRAE 223 standard [66]. In recent research, Brick supported the configuration and portability of DF applications [39,40]. Meanwhile, SAREF is supported by the European Telecommunications Standards Institute (ETSI) and has proven a reference ontology to promote interoperability between several standards within the DF domain [67]. Owing to this, SAREF has been recommended as an essential asset of the DERA 3.0 [45]. This ontology was also used to enhance the interoperability of DF applications in recent work [41,42]. However, while both ontologies demonstrate promise and potential paths to adoption in Europe (SAREF) and the US (Brick), their alignment and comprehensive comparison have not yet been pursued. The absence of a mapping between both ontologies hinders the ability to comprehensively compare them or to reuse applications developed based on one of them across buildings modelled based on the other. For instance, Brick-driven portable applications, such as those developed in [39,40], are not easily deployed in buildings with SAREF-driven models. The opposite applies to the SAREF-driven applications, as proposed in [41,42]. This highlights the need and potential for proposing an alignment between the two.

2.3. Semantic models generation

Many works on the automated generation of semantic models rely on Rule Markup Language (RML) [68–70], which uses mapping rules to link data within different formats to corresponding ontology classes or relationships. However, the unstructured data point descriptions in BAS [71] and the syntactic complexity of BIM data following standard exchange data formats, such as Industry Foundation Classes (IFC) [72], render the use of RML-based tools impractical, as RML does not easily support their needed complex mapping rules. Some existing studies have proposed tools to generate Brick-based semantic models based on unstructured BAS descriptions [30,31,73]. These tools, however, naturally centre around ontology classes derived from Brick or have not demonstrated support for BIM mapping. In contrast, a number of studies have investigated the development of semantic models that map data from BIM [74–76], but they have not considered the integration with BAS data points. To bridge this gap, other efforts have emerged aiming to generate semantic models linking BIM and BAS data while linking a set of well-known ontologies, such as Brick [29,32,33]. These studies, however, do not explore data required in DF control applications and their supported ontologies do not include SAREF. As a result, novel approaches focusing on DF data needs and SAREF concepts need to be proposed.

2.4. Semantics-driven frameworks

A number of studies introduced frameworks for linking DF applications to required data sources based on generated semantic models [12,39–42,55,56,58,59,61,77]. Because some of these works introduced new, bespoke ontologies, the potential to reuse their proposed frameworks remains constrained due to inadequate documentation, discontinued maintenance or the absence of tools to effectively instantiate their new ontologies [12,55,56,58,59,61,63,64]. Furthermore, some of these proposed methodologies require storing temporal data such as device readings, weather information, or grid requests as instances of their ontologies [12,41,42,55,56,58,59]. This is often the case for approaches that use semantic web inference notations and rules to trigger DF actions [78]. However, this approach is not considered ideal, particularly when dealing with large data streams or more complex control logic such as required for HVAC systems [79–81].

---

[6] https://www.w3.org/RDF/.

[7] https://www.w3.org/OWL/.

[8] https://www.w3.org/TR/shacl/.

Studies presenting field and simulation demonstrations of semantics-driven DF control applications supported by the Brick ontology have also been published [39,40]. One of these works proposed a software stack leveraging Brick to facilitate integrating and deploying a DF application in a real building [39]. The other study proposed a framework using Brick to enable the portability of an application across multiple buildings [40]. These papers emphasised the reduction in effort required to configure an application. However, neither demonstrates how to integrate applications that use the SAREF ontology, which is prevalent in Europe. Furthermore, neither study has investigated the use of contextual data from BIM to generate their semantic models. In fact, the semantic models from most buildings evaluated in [40] were manually generated and limited in the number of data points to the scope of the DF applications being tested. Moreover, in terms of the implementation of DF portable controls, while these works introduced the idea of a modular design approach, they have not explored purpose-built platforms that facilitate the development of such controls.

2.5. Research gaps and contributions

Overall, existing efforts establish the foundation for semantics-driven DF applications, but notable research gaps remain for further enhancement. While most of these studies propose new ontologies, their usefulness and scalability may be impacted. Much of the reviewed literature considers temporal data stored in the semantic models instead of appropriated databases, which may have negative impacts on the management of these data. None of the DF-driven efforts considers BIM data, which could minimise efforts in establishing a mapping between data points and their contextual metadata to generate models. Moreover, the alignment between the Brick and SAREF ontologies has not been demonstrated, with the coverage and benefits of both ontologies remaining unclear and not verified. Finally, there remains a gap in exploring the use of purpose-built platforms that can support the development, configuration and deployment of DF control applications in a modular, easily replicable and adaptable way.

Motivated by these remarks, our research proposes to develop an approach that supports DF controls using semantic models. To this purpose, the contributions of this paper are three-fold:

- Novel alignment between Brick and SAREF to capture data exchange needs for DF controls applied to HVAC systems, mostly at the zone level.
- Design open-source algorithms[9] as computational methods to generate semantic models that adhere to Brick and SAREF while mapping metadata from heterogeneous sources (BIM and BAS).
- Demonstrate the deployment of DF applications in a real building, leveraging the framework's outputs, such as a semantic model generated for the building and a web platform that implements controls based on an actor and microservices architecture.

3. Methodology and framework design

The methodology devised to design our proposed semantics-driven framework was based on an active design science approach [82], involving multiple interactive developments and evidence gathered. The framework comprises five phases, as illustrated in Fig. 1, expanding our previous work that supports DF control applications using semantic models [40]. Phase 1 establishes requirements for DF controls development. Phase 2 aligns the Brick and SAREF ontologies. Phase 3 generates compliant semantic models. Phase 4 validates and formulates queries for these models. Lastly, Phase 5 deploys the controls within a web platform based on actor and microservices architecture.

Phases 2, 3 and 5 are novel and complementary to our previous framework and do not necessarily need to take place for each new control. For instance, the alignment proposed in Phase 2 can be shared among different applications. Similarly, Phase 3 is only required for a building if there is no semantic model based on Brick and SAREF. If a semantic model based on one of those ontologies exists, the model can be updated to include concepts from the missing ontology. This is important to allow the different applications relying on queries from both ontologies to access the same building. Each phase is discussed in depth in the subsequent subsections.

3.1. Phase 1: Control applications requirements

As stated in our previous work [40], developing control applications that can be portable and generalisable across heterogeneous buildings involves adhering to four key requirements. First, the controls must account for changing operating conditions that buildings might encounter [83], such as occupancy schedules, comfort status, faulty conditions and grid signals. Second, the controls need to be flexible to available data and how the data are structured in different semantic models [84], for example, allowing self-configuration based on relaxed, generic queries that use either Brick or SAREF concepts. Third, the controls' logic must be agnostic to a specific building's context, such as communication protocols, point naming practices and data access methods [21]. Finally, the controls need to be developed in a modular way, composed of self-contained functions that are easily re-used, extended and customised if needed [85]. The latter follows the modular design approach adopted by the upcoming ASHRAE 231 standard on a control logic description for control sequences [86].

Adhering to these requirements should minimise the labour-intensive process of reconfiguring control applications when re-using the same codebase across various buildings. Once the controls are developed in an adaptable, flexible, agnostic and modular manner, the subsequent step involves identifying their data exchange needs, such as measurement readings (inputs) and control (outputs) points related to a particular location or equipment. This is the foundation for generating and validating appropriate semantic models that facilitate the controls' configuration, as elaborated in the following subsections.

3.2. Phase 2: Brick and SAREF alignment

Semantic models are the core of our framework. By capturing the metadata related to the data points (inputs and outputs) required by control applications, semantic models can streamline configuration tasks, such as point discovery and mapping [21]. In our framework, this is enabled by an approach known as Ontology-based Data Access (OBDA) [87], which allows applications to uniformly query and discover the metadata (e.g., access information) associated with required data points across different buildings, and subsequently use the results to automate their mapping to each building. This mitigates the need for hard-coded, customised metadata (e.g., point naming), as further described in Phase 4. Nonetheless, for this to be scalable, our framework relies on the models being built on: (i) ontologies that can meet the data needs of various applications to represent a one-time, "future-proof" investment for building owners and operators; and (ii) well-established ontologies that can be consistently adopted across various buildings, so that control developers can easily understand their models and potentially rely on portable semantic queries. Otherwise, developers will still need to invest significant effort in aligning their inputs and outputs with the distinct underlying ontologies of models from different buildings.

Despite being strongly encouraged within recent initiatives in Europe and the United States [13,15–17,21,27,37,45,88], and the proven potential for supporting DF applications independently [38–42], Brick and SAREF have not yet been aligned. To address this, in this phase of the framework, we propose an alignment method between Brick (v1.3)

---

[9] https://github.com/ucl-sbde/semantics-driven_controls.git.

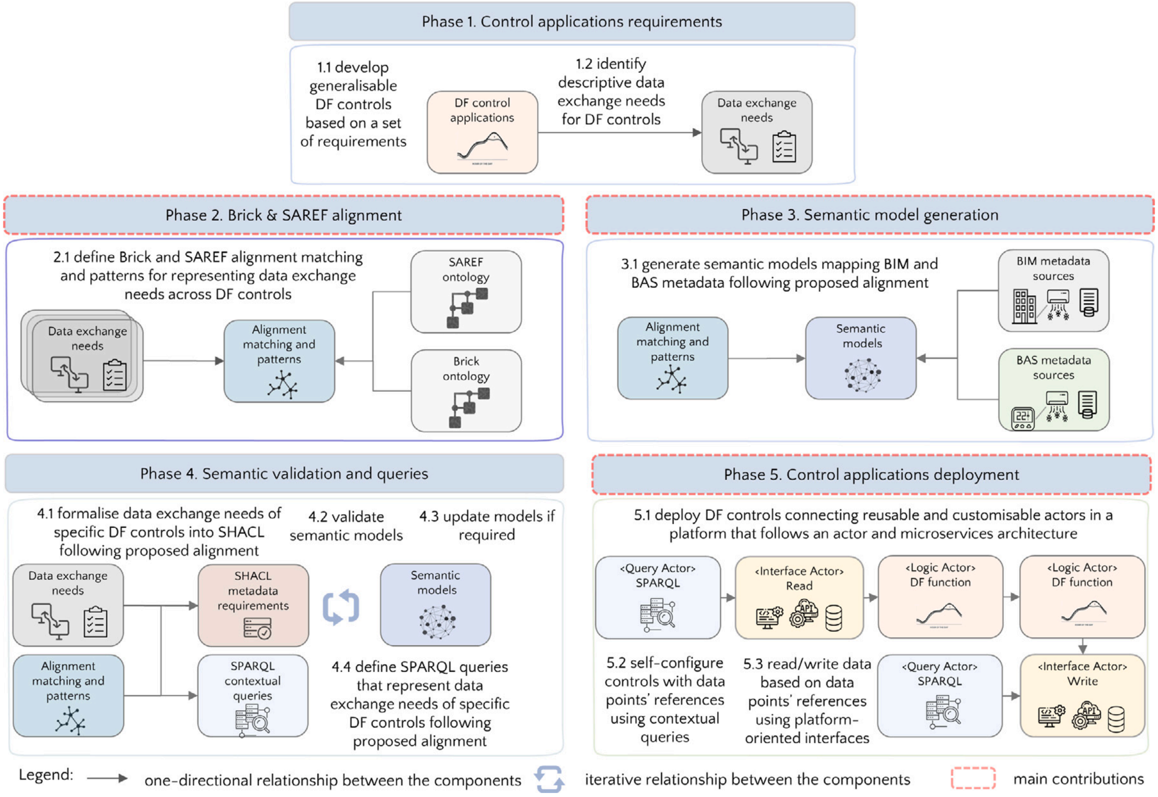


Fig. 1. Proposed semantics-driven framework to enable DF control applications in real buildings.

and SAREF (core v3.2.1, along with its extensions and recommended ontologies). We first classify their common concepts into four: spatial, device, measurement and control points. Then, we use two alignment techniques (matching and patterns) to link the concepts from both ontologies based on their modelling premises and constraints. Fig. 2 illustrates four samples of alignments, one for each type of concept with a proposed connection between them. The alignments on the top are matching-based, and those on the bottom are pattern-based.

Matching-based techniques are one of the simplest ways to create ontology alignment by identifying the one-to-one overlap of classes [89, 90]. This matching typically relies on string similarity, where the names and descriptions of classes in different ontologies are compared to find corresponding matches. Considering the modelling premises of Brick and SAREF, this method is suitable for aligning their spatial and device concepts. Fig. 2 illustrates one alignment between *brick:Space* and *s4bldg:BuildingSpace* based on their names, and another between the *brick:Thermostat* and *s4bldg:UnitaryControlElement* based on their names and descriptions. A list with all alignment matching between Brick and SAREF is available in Appendix A.

Although matching-based techniques can simplify alignment tasks, they are limited to finding basic correspondences between ontologies and cannot capture semantic nuances or implicit relationships between entities. Given that SAREF aims for higher levels of abstraction, focusing on core (broader) concepts [91], while Brick tends to be more expressive and granular [92], more complex alignment approaches are required to model these concepts. Ontology patterns [93] facilitate the specification of complex ontological alignments. They achieve this by examining the meaning of similar concepts within different ontology fragments and then constructing patterns representing their correspondences.

Therefore, in order to align the Brick and SAREF concepts for measurement and control points (e.g., sensors and setpoints), we use a pattern-based technique grounded in a one-to-many strategy. This involves establishing alignments between Brick classes and sets of ontological classes and relationships from SAREF. For example, as illustrated in Fig. 2, for measurement points such as *brick:Temperature_Sensor*, we propose an alignment with a pattern composed by the *saref:Sensor* class, the *saref:observes* relationship and the *quantitykind:Temperature*, which is a suggested subclass of the *saref:Property* class by SAREF. Similarly, for control points such as *brick:Temperature_Setpoint*, we propose an alignment with a pattern composed by the *saref:Actuator* class, the *saref:controls* relationship and the *quantitykind:Temperature* class. Although more abstract and implicit, the definition of such a set of SAREF classes and relationships aligns with the classes from Brick. A list with all alignment patterns between Brick and SAREF is also available in Appendix A.

The classification into four common concepts, along with the matching and pattern-based techniques, was performed manually, guided by the authors' expertise, a comprehensive review of existing literature and the ontologies' documentation, and insights from practical demonstrations. While we recognise the inherent subjectivity and potential uncertainties in this approach, as well as the possible influence of specific HVAC configurations or existing alignments (such as those resulting from Brick's recent harmonisation with REC), this analysis serves as a starting point for initiating a much-needed discussion and guiding future research that integrates these two ontologies.

To carry out the proposed alignments, we employ an instance-level approach that follows the principles from the Linked Building Data (LBD) community [17]. This method facilitates mapping individual data instances across different ontologies, provided their chosen classes do not pose semantic conflicts. The following subsection creates mapping algorithms that implement this instance-level approach leveraging the matching and patterns listed in Appendix A.

3.3. Phase 3: Semantic model generation

Semantic models of specific buildings are created by instantiating (populating) an ontology (or a set of ontologies) based on the buildings'

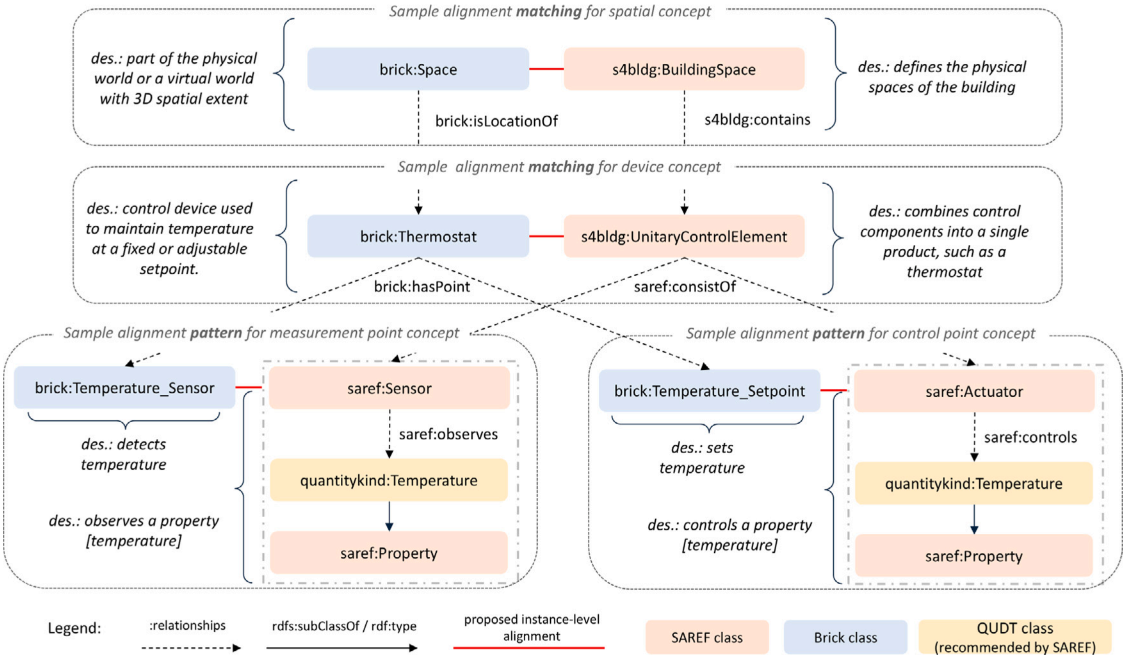


Fig. 2. Sample of suggested alignment matching and patterns between sets of concepts from Brick (v1.3) and SAREF (core v3.2.1 along with extensions and recommended ontologies). The alignments for spatial and device concepts are based on string matching, while the alignments for measurement and control points are based on patterns.

data sources and operational staff knowledge [16]. Among examples of data sources are BAS point lists with point names, types, data units and short descriptions, as well as design stage models (e.g., BIM) with functional and spatial relationships between equipment, points and spaces. In isolation, these sources provide relevant information, but while BIM sources do not often include data point information, BAS sources usually offer ambiguous or lack functional and spatial relationships about points. Integrating these sources becomes crucial as their combined impact complements and outweighs their individual contributions. This paper proposes computational methods that map and link relevant information from these data sources to generate semantic models based on Brick and SAREF concepts.

To streamline the generation of semantic models, our paper introduces open-source algorithms leveraging the robust capabilities of the Apache Jena framework.[10] Our choice of Jena stems from its extensive library set for data management and modelling tasks of RDF-based ontologies. Using these libraries, we employ an iterative process to map metadata from BIM (via IFC models) and BAS (via point lists in Comma-Separated Value (CSV) format) to Brick and SAREF concepts according to their alignment. This approach generates semantic models of specific buildings, serialised as Terse RDF Triple Language (TTL) files. Both algorithms, called BAS-to-RDF and BIM-to-RDF, generate distinct models for each source, which can be merged by linking shared instances. Due to the infrequent representation of spatial elements in BAS sources, our BAS algorithm only accommodates device and measurement and control points. Conversely, owing to the inherent lack of measurement and control points in BIM sources, our BIM algorithm only incorporates spatial and device concepts. Consequently, the device instances will serve as the main link between the models generated from both sources. Tables 1 and 2 provide an overview of the number of mappings currently facilitated by our BIM and BAS algorithms across different concept types and ontologies. A complete list of our supported concepts indicating their mapping to ontology classes (from Brick and SAREF) and the data source (either BIM or BAS) from which they are mapped is provided in Appendix A.

Table 1
Number of BAS concepts mapped to Brick and SAREF.

Concept type	Subtotal	Brick class	SAREF class
Device	13	13	13
Measurement point	19	19	12
Control point	26	26	13
Total	58	58	38

Table 2
Number of BIM concepts mapped to Brick and SAREF.

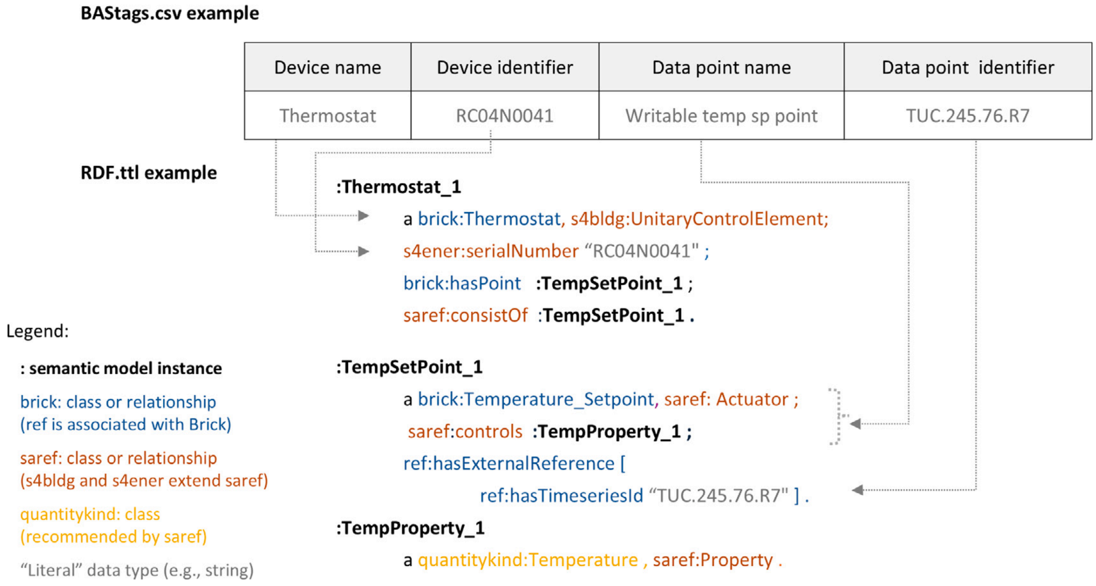
Concept type	Subtotal	Brick class	SAREF class
Spatial	5	5	2
Device	13	13	13
Total	18	18	12

Starting with the BAS-to-RDF mapping algorithm. As illustrated in Fig. 3, our proposed script is based on a simple CSV template following a structure of how some BAS point lists can often be exported from BAS tools. The template has a header row stating each column's corresponding classes and entity properties. These include **Device name** that refers to descriptions for classifying devices/equipment, **Device identifier** that adds given identifiers to each device (e.g., serial numbers), **Data point name** that refers to descriptions for measurement and control points related to the devices and **Data point identifier** that informs the external reference identifier that provides data reading and writing access to each of the points.

The process provided by our algorithm for mapping BAS metadata within this CSV template into RDF involves multiple steps. First, the script identifies the unique **Device identifier** and, based on their corresponding **Device name**, associates them with a Brick and SAREF class. Second, for each unique identified device, the script iterates over the rows of the CSV, creating individual instances of their respective data points. Then, the script identifies the Brick and SAREF concepts corresponding to each data point using the description within the **Data point name** column. It also adds their external references for each data point based on the **Data point identifier**. Finally, the script links each data point with the respective device. A pseudo-code for this is provided in Appendix B, and the full script is available in our repository[9].

---

[10] https://jena.apache.org/



Device name	Device identifier	Data point name	Data point identifier
Thermostat	RC04N0041	Writable temp sp point	TUC.245.76.R7

```
:Thermostat_1
        a brick:Thermostat, s4bldg:UnitaryControlElement;
        s4ener:serialNumber "RC04N0041" ;
        brick:hasPoint  :TempSetPoint_1 ;
        saref:consistOf  :TempSetPoint_1 .

:TempSetPoint_1
        a brick:Temperature_Setpoint, saref: Actuator ;
         saref:controls  :TempProperty_1 ;
        ref:hasExternalReference [
                ref:hasTimeseriesId "TUC.245.76.R7" ] .
:TempProperty_1
        a quantitykind:Temperature , saref:Property .
```

Fig. 3. Sample template mapping from BAS (CSV) into RDF (TTL) for instantiating the proposed alignment between Brick and SAREF, generating the semantic model of a given building.

The mapping between device and data point names to Brick and SAREF concepts is handled through rule-based conditions that link given labels to corresponding concepts (e.g., *"Writable temp sp point"* referring to a temperature setpoint). To address inconsistencies in naming across different BAS data sources, the code includes various synonymous terms based on prior research on common labels [52]. Although this approach may not cover all possible labels, the algorithm can be extended to include new terms as needed. It is also important to note that because our template requires the point name to be separated from their related device/equipment identifier, some pre-processing may be needed in cases where available BAS point list combines them into one label (e.g., *"FCU2.targetTemperature"* referring to the temperature setpoint related to fan coil unit #2). Moreover, in the absence of a BAS tool to provide the required metadata for the CSV template, manual completion will be necessary. This may require consulting supplementary documentation (e.g., mechanical drawings) and seeking assistance from building operators. Although labour-intensive, such effort is likely a one-time cost, as generating semantic models should greatly reduce or eliminate future repetition of these manual tasks.

As expected, the result of this BAS into RDF mapping leads to a preliminary semantic model covering only classes related to devices and measurement and control points. To supplement this initial model with additional BIM metadata, we developed the BIM-to-RDF mapping algorithm. This algorithm builds upon a previous work by the authors [32].

In the BIM-to-RDF script, the proposed mapping template follows the IFC hierarchy for retrieving information on spatial and device concepts. Fig. 4 illustrates the template covering all the spatial aspects (site, building, storey, space and zone) while providing an example for device mapping. For the generation of the semantic model, the algorithm parses the IFC, iterating over its instances to check whether they are instances of a given entity while using **IfcRelAggregates** to find their relationships. In order to identify the connection between the devices and the spatial concepts, we use the **IfcRelContainedInSpatialStructure** entity, which allows us to state that devices are contained within spaces and related to zones. Although such a relationship might not be the most suitable for connecting more complex HVAC systems and zones, both SAREF and Brick lack comprehensive and granular concepts to capture the interconnections between HVAC components, such as duct and pipe segments, as depicted in IFC. While further elaboration on this aspect is needed, it is beyond the scope of this paper. A pseudo-code for this algorithm is also provided in Appendix B, and the full script is available in our repository[9].

When integrating the semantic models generated separately through the BIM and BAS algorithms, their data sources need to include common device instances. To identify these instances, we use the devices' unique identifiers to ensure consistent Uniform Resource Identifiers (URIs) across both models. In the CSV template, such an identifier is located in the **Device identifier** column, whereas in IFC, it is represented using the **IfcPropertySingleValues**. This notably demands consistency in the CSV inputs and IFC modelling. The latter, in particular, should also follow best practices and guidelines, as outlined in [94], to ensure the availability and appropriate modelling of the required information.

3.4. Phase 4: Semantic validation and queries

Once semantic models are generated (via our mapping algorithms or by other means), they can be validated to ensure completeness to given DF applications' data needs and adherence to a consistent modelling approach. To accomplish this, the inputs and outputs necessary for DF applications must be translated into SHACL shapes that comply with both Brick and SAREF according to our proposed alignment. This is essential because although SHACL shapes can be independently constructed using Brick or SAREF concepts and be used to validate existing models based on either ontology, these models must still adhere to our proposed modelling approach.

Listing 1 and Listing 2 give examples of formalised definitions of data needs using Brick and SAREF-driven SHACL shapes. They require a semantic model to have spaces (brick:Thermostat and s4bldg:UnitaryControlElement) with at least one (sh:qualifiedMinCount 1) point (brick:hasPoint and saref:consistOf) of the type "temperature sensor" (brick:Temperature_Sensor and saref:Sensor - saref:observes - quantitykind:Temperature). These examples can be further enhanced to represent other classes of devices and points, as well as relationships, such as related locations. Once defined for all inputs and outputs of a specific control application, SHACL shapes can be used to validate that a given building, described by a semantic model, can support such an application. To automate this validation process, we leverage BuildingMOTIF,[11] as detailed in [40]. This process offers clear and

---

[11] https://nrel.github.io/BuildingMOTIF

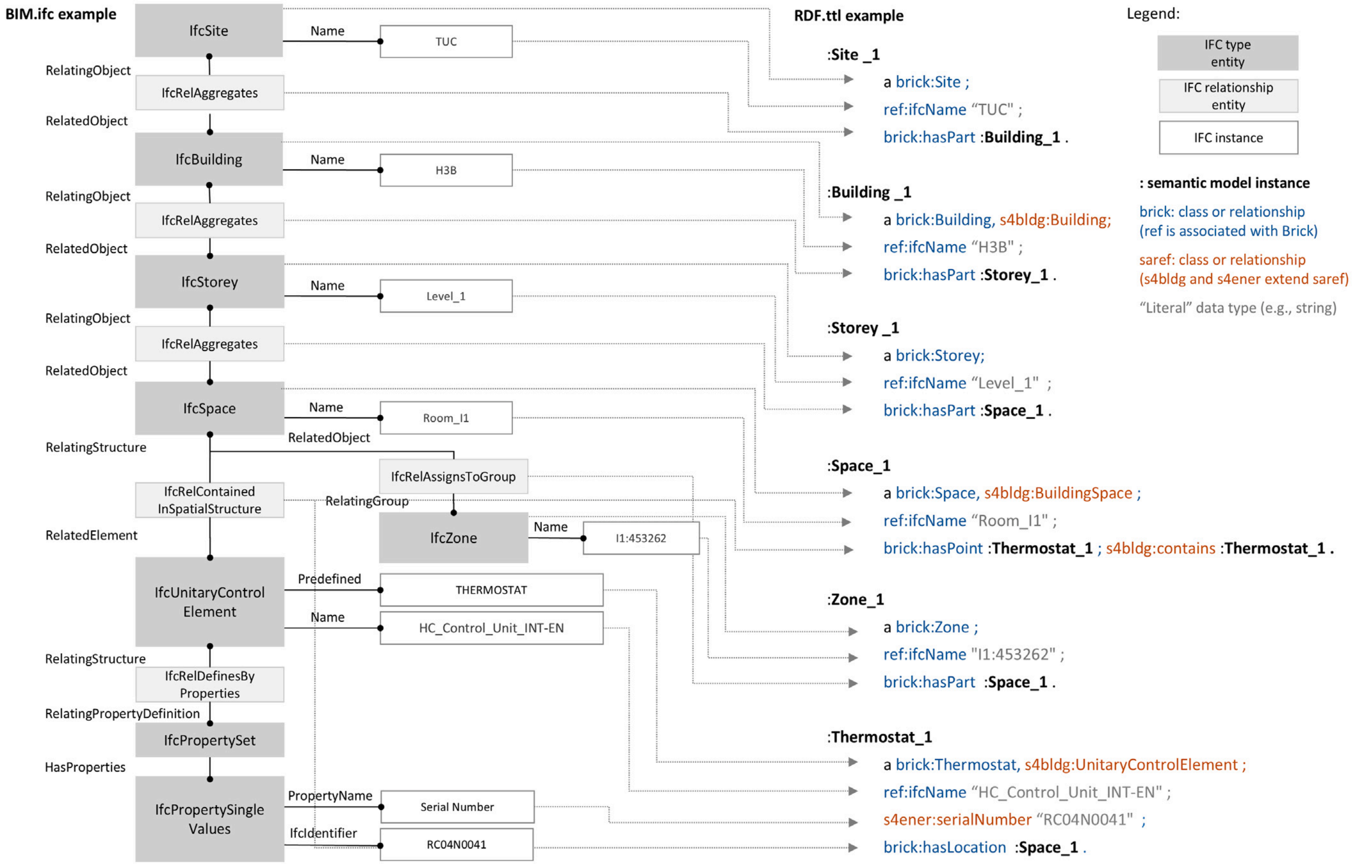


Fig. 4. Sample template mapping from BIM (IFC) into RDF (TTL) for instantiating the proposed alignment between Brick and SAREF, generating the semantic model of a given building.

actionable feedback on missing metadata, helping to expedite error identification and, when possible and required, update the semantic models accordingly.

```
:thermostat  a sh:NodeShape, owl:Class;
  sh:targetClass brick:Thermostat;
  sh:property [
    sh:path brick:hasPoint ;
    sh:qualifiedValueShape [
      sh:class brick:Temperature\
          _Sensor ];
    sh:qualifiedMinCount 1; ]; .
```

Listing 1: Snippet of a formalised definition of data needs using Brick-driven SHACL shapes.

```
:thermostat  a sh:NodeShape, owl:Class;
  sh:targetClass s4bldg:
      UnitaryControlElement;
  sh:property [
    sh:path saref:consistOf ;
    sh:qualifiedValueShape [
      sh:class saref:Sensor ];
    sh:qualifiedMinCount 1; ]; .

:Tproperty a sh:NodeShape ;
    sh:targetClass saref:Sensor;
    sh:property [
        sh:path saref:observes ;
        sh:minCount 1;
        sh:nodeKind sh:BlankNode ;
        sh:property [
            sh:path quantitykind:
                Temperature ;
            sh:minCount 1;
            sh:maxCount 1; ] ; ] ; .
```

Listing 2: Snippet of a formalised definition of data needs using SAREF-driven SHACL shapes.

To self-configure controls using validated semantic models from specific buildings, we leverage the ODBA approach that allows semantic (SPARQL[12]) queries to retrieve the accessing information (i.e., external references such as foreign key, time series identifier, pub-sub topics) of required data points based on their context and common vocabulary provided by Brick and SAREF. While the actual values of these points are fetched from BAS connectors (e.g., message brokers or gateways) or purpose-built storage platforms (e.g., relational or time series databases), their references allow the applications to determine how to read or write the intended data according to the interfaced source [17,40]. For instance, using suitable SQL[13] queries for time series databases or making MQTT requests to BAS brokers.

Listing 3 illustrates a SPARQL example using Brick concepts. It looks for external references related to zone-level temperature sensors by explicitly stating the target Brick class *brick:Temperature_Sensor* while implicitly stating its relationship to the zone (lines 5–8). Meanwhile, the Listing 4 illustrates a corresponding SAREF-driven SPARQL example. As noted, in the SAREF query, the concepts for zone and external

---

[12] https://www.w3.org/TR/rdf-sparql-query/

[13] https://www.w3schools.com/sql/

references have been adopted from Brick since they are not inherently part of SAREF.

```
SELECT ?zone ?data_point_identifier WHERE {
  ?sensingPoint a brick:Temperature_Sensor;
    ref:hasExternalReference/ref:hasTimeseriesId ?data_point_identifier .

  ?zone a brick:Zone ;
    brick:hasPart ?space .

  ?space brick:isLocationOf/brick:hasPoint ?sensingPoint .
}
```

Listing 3: Brick-driven example of a SPARQL query to obtain the external references from temperature sensor data points per zone

```
SELECT ?zone ?data_point_identifier WHERE {
  ?sensingPoint a saref:Sensor;
                saref:observes ?property ;
                ref:hasExternalReference/ref:hasTimeseriesId ?data_point_identifier .

  ?property a quantitykind:Temperature, saref:Property .

  ?zone a brick:Zone ;
        brick:hasPart ?space .

  {?space s4bldg:contains/saref:consistOf ?sensingPoint .}
}
```

Listing 4: SAREF-driven example of a SPARQL query to obtain the external references from temperature sensor data points per zone

The result of both queries should be the same: arrays of zones with external references to access associated temperature sensors. The distinction between them lies in the set of triple patterns used in their SPARQL syntax, which reflects the differences in their underlying modelling approaches (also seen in their SHACL above). SAREF offers better abstraction than Brick, which could be helpful for users unfamiliar with the varied classifications of Brick. For instance, allowing users to rely on common concepts for *saref:Sensor* and *quantitykind:Temperature* to query a temperature data point independent if it has been modelled with a similar, but not the equal, class such as *brick:Temperature_Sensor*, *brick:Air_Temperature_Sensor* or *brick:Zone_Air_Temperature_Sensor*. It could also assist users to change the property type of an existing query (e.g., from *quantitykind:Temperature* to *quantitykind:Occupancy*) without having to change the class of the sensor. Nonetheless, although these benefits can be significant in some cases, they may not always align with the needs of certain applications. For instance, applications that require queries for specific data points, such as *brick:Supply_Air_Temperature_Sensor* would benefit from the Brick expressiveness to avoid ambiguity while querying the models. This is further discussed in Section 5.

3.5. Phase 5: Control applications deployment

To enable the configuration and deployment of controls using the generated and validated semantic models (as proposed in phases 3 and 4), we introduce a novel control platform. The platform enhances existing efforts proposed in [95]. While using a set of open-source Java libraries and frameworks, including Akka[14] and Spring,[15] the platform implements an actor and microservices architecture. Akka actors were used to decoupling application logic from specific building data access methods and metadata. This was enabled by creating three types of actors serving different functions: query, interface and logic actors. Each actor, further explained in the following paragraph, can concurrently receive data, perform tasks and communicate with the other actors. Meanwhile, the Spring framework was used to assemble these actors into microservice applications to facilitate their communication with specific buildings via standardised protocols and messaging queues.

A basic interaction among the three actor types is illustrated in Phase 5 of Fig. 1. The **query actors** are responsible for self-configuring the controls using semantic queries against building-specific semantic models to retrieve required data points' external references. This is a generic actor capable of processing SPARQL queries from input text files. While the actor itself is easily portable, the portability of the SPARQL text files depends on the buildings having suitable semantic models (e.g., Brick and/or SAREF compatible) so that the same queries can be re-used. The **interface actors** are responsible for communicating with the buildings to read or write in their data points using the references obtained from the semantic queries. The portability of these actors varies with the data access methods in different buildings and the requirements in different applications (e.g., historical or real-time data). For instance, we can have actors with reading roles tailored to make HTTP endpoint requests, subscribe to data streams or access databases via query languages (e.g., SQL or native ones). Buildings with the same data access methods can share the same actors. Finally, the **logic actors** can access the readings from the interface actors with reading roles or write in the interface actors with the writing roles. Although the logic actors essentially do not rely upon and are not hard coded to any building-specific details (such as point names), their portability can still be constrained by other aspects, such as the building's baseline control or HVAC system configuration. To maximise the portability of our logic actors, each actor shall offer a basic set of functions (such as detecting occupancy status and comfort level) that can be easily re-used, extended and customised without impacting other actors when connected.

4. Case study

To assess the benefits and challenges of our proposed framework, this section describes the real building that we used as the case study and how we followed the phases described in Section 3 to implement the framework. First, based on the requirements from Phase 1, we developed two DF control applications. Second, as part of Phases 2 and 3, we generated a semantic model based on the available BIM and BAS data sources while following the proposed alignments for Brick and SAREF. Then, based on Phase 4, we validated the generated model using the data needs for both controls formalised as SHACL shapes, as well as defined SPARQL queries to support the configuration of these controls. Finally, we implemented the two control algorithms using actors within the web platform and connected it to the BAS from the real building, as proposed in Phase 5. The combined demonstration of these phases should illustrate the potential of our framework to integrate data from heterogeneous sources, deliver DF actuation in a real building and promote modularity for easy decoupling of control applications from specific buildings, as well as for their effortless reuse, extension, and customisation.

---

[14] https://akka.io/docs/

[15] https://spring.io/projects/spring-framework

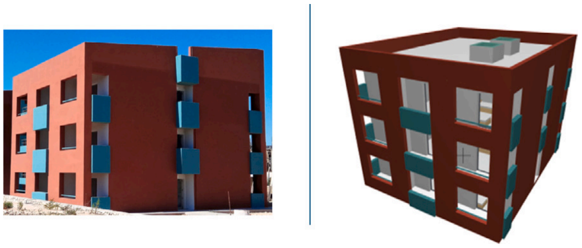


Fig. 5. Case study facility (left: real-building, right: 3D view of the IFC model).

4.1. Real building description

The case study comprises a student residence at the Technical University of Crete (TUC), Greece. The building is a three-floor complex with 20 en-suite student rooms and a shared kitchen on each floor. Each room has independent split air conditioners and smart thermostats with built-in temperature and occupancy. The building is powered by electricity and contains building-level metering and room-level sub-metering to measure consumption. There are 15 similar buildings on campus with a combined capacity of 300 student rooms and more than 6000 data points, which holds significant promise for the replicability potential of this case study.

The real building and the 3D view of the IFC model are shown in Fig. 5. The IFC adheres to the guidelines presented in [94], modelling the spatial containment of physical resources (split air conditioners and smart thermostats) within the spaces and the property set entities (with serial numbers) for the physical resources. Both are critical for the proper functioning of the proposed BIM-to-RDF mapping algorithm. Regarding the BAS, the HVAC controllers within each zone communicate through the Modbus[16] protocol and use MQTT[17] to enable their communication with a cloud middleware platform. The baseline control for the HVAC system includes basic operational settings and functionalities applied to support comfort and energy efficiency goals. This involves features such as temperature setpoint ranges based on the HVAC mode (heating or cooling) to which users can adjust their local thermostat and overarching supervisory controls such as a scheduling function based on the academic calendar.

Since we could not directly access the real grid signals of the TUC buildings, our proposed controls rely on dynamic electricity prices and carbon intensity levels obtained from utility services associated with the TUC location. The dynamic electricity prices are based on the real day-ahead market (wholesales) prices provided by the EnexGroup,[18] the utility that serves the TUC location. The dynamic carbon intensity levels are based on real carbon dioxide (CO_2) emissions retrieved from the Electricity Maps Data Portal API based on the TUC location [96].

4.2. Control applications development

To demonstrate Phase 1 of our framework, we propose two DF control applications aiming to automate the operation of flexible HVAC loads. Their difference lies in the grid signal used to control them. One of the applications reacts to pricing signals and aims to reduce the load during high-price periods, while the second responds to carbon intensity (greenhouse gas emissions) signals and aims to reduce the load during high-emission periods. Each application is rule-based and uses occupant comfort in addition to price or emission thresholds to determine when to shed load (Fig. 6). The thresholds are established by considering values exceeding the third quartile of the input price and carbon intensity distribution. Both controls set a standby mode when zones are unoccupied.

The data needed for both DF control applications include external references for occupancy sensors, temperature sensors, temperature setpoints, minimum and maximum temperature setpoints per zone and respective grid signals. While both controls are very similar, they are relevant to demonstrate how the platform we propose for their deployment allows the same base control to be easily customised and extended according to the use case (e.g., type of grid signal in this case). In the control deployment section, we further detail this and discuss how the controls follow the generalisability criteria described in Phase 1.

4.3. Semantic model generation

To demonstrate Phases 2 and 3 of our framework, we generated the semantic model for this case study using our mapping algorithms[9] that leverage our proposed Brick and SAREF alignment. Fig. 7 illustrates a snippet of the resulting model with a few ontology concepts and relationships aligning both Brick and SAREF to represent one zone. The figure shows which metadata comes from IFC and CSV and their point of connection (i.e., the HVAC controller instance), which allows the integration between BIM and BAS.

This semantic model is valuable for classifying and explaining how resources and data points are related. This allows DF applications to query measurement and control points at the zone level or find available commands to respond to DF actions. For instance, from this model snippet, a few pieces of information about this case study can be implied, including that zones are constrained by a maximum temperature value and linked to occupancy sensors and setpoint commands. We can also verify that these points are accessible from external data sources through the given references.

It is important to note that while Brick and SAREF subclasses and relationships include most of the requirements for the proposed DF application, the grid signals are not part of these two ontologies and, as such, the reference to access their datasets is custom-specified in the applications' logic. While this falls outside the scope of this paper, if one were to model such concepts using semantic models, two potential approaches could be considered. First, these two ontologies could be extended. Second, alignments with a dedicated ontology, such as the OpenADR ontology [57], could be explored. The latter, however, would require updating this ontology according to the current version of the OpenADR standard while fostering a dedicated community for its ongoing management, as it currently lacks established support.

4.4. Semantic validation and queries

To demonstrate Phase 4 of our framework, we formalised the data needs for the proposed DF controls into SHACL shapes following the Brick and SAREF alignment matching and patterns. In total, five SHACL shapes were required, specifying the need for the controls to access the external references for occupancy, temperature, temperature setpoints and minimum and maximum temperature setpoints per zone. All these shapes were successfully addressed using Brick concepts. However, since SAREF lacks classes for minimum and maximum temperature setpoints, the shapes for these two could not be fulfilled by SAREF. Additionally, because SAREF does not inherently cover concepts for zones and external references, all SAREF-driven shapes had to rely on Brick and its associated ref-schema ontology. This means that existing semantic models solely based on Brick concepts are mostly likely to satisfy the requirements for the proposed controls, while models only based on SAREF concepts may not. Although SAREF could be extended to include missing concepts, this is outside the scope of this paper, and future efforts in this direction are recommended.

The generated semantic model for the case study included the data needed by the proposed controls and successfully passed the semantic validation for both Brick and SAREF-driven shapes. To test how such a model could be used to configure and deploy the controls, we created

---

[16] https://modbus.org/.

[17] https://mqtt.org/.

[18] https://www.enexgroup.gr/web/guest/home

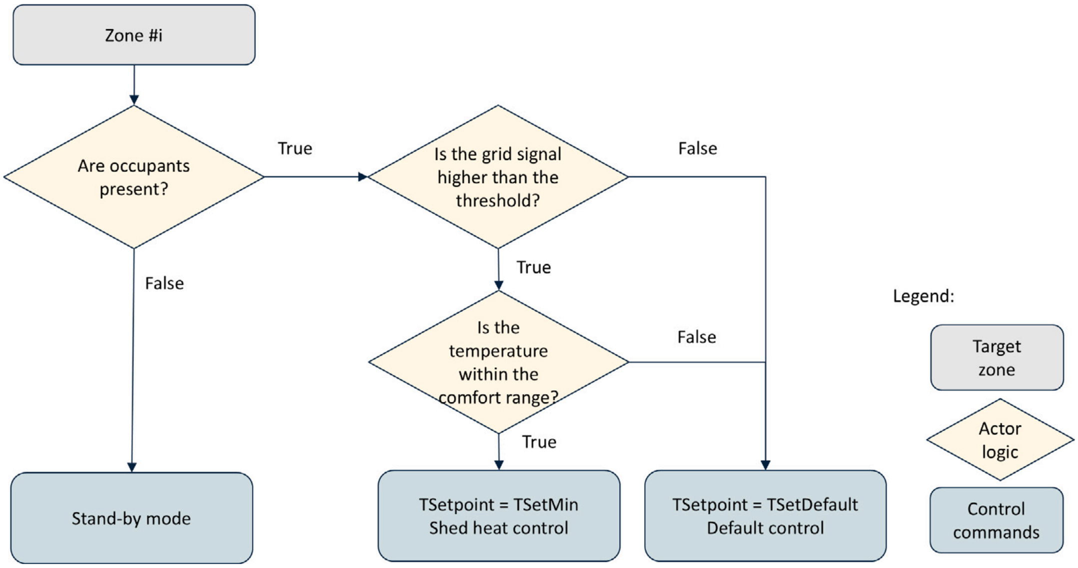


Fig. 6. Case study rule-based control flow diagram to adjust HVAC loads at zone level based on occupancy, comfort range and grid signals thresholds.

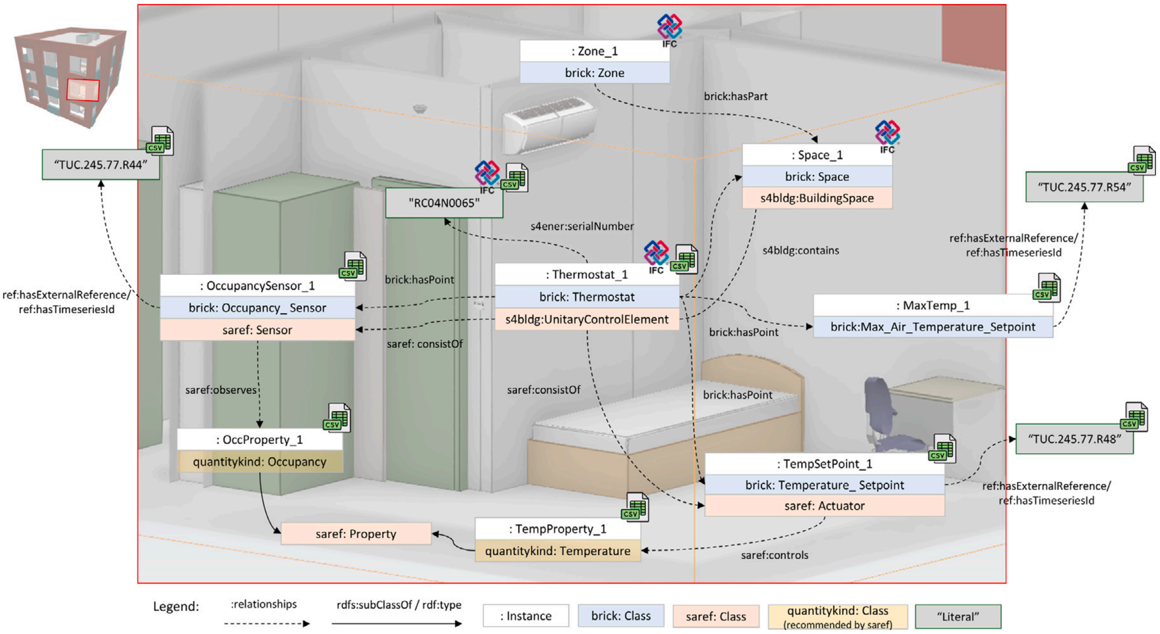


Fig. 7. Snippet of the generated semantic model for the case study with examples of Brick and SAREF concepts mapped from BIM (IFC) and BAS (CSV) within one zone.

SPARQL queries corresponding to these SHACL shapes. Both Brick and SAREF-compliant queries can be found in our online repository[9], and their use within the proposed platform is detailed in the following section.

4.5. Control applications deployment

To demonstrate Phase 5 of our framework, we deployed the DF supervisory control applications in a web platform. To achieve this, we created several actors, including SPARQL query, interface and logic type of actors. Fig. 8 illustrates platform's intuitive drag-and-drop interface with the actors and the information flows we implemented for the price-driven control application.

The **scheduler** (logic) actor is responsible for setting frequency (slots) to prompt the subsequent actors. The **SPARQL** (query) actors query information about required data points, including their access information (i.e., external references such as foreign key, time series identifier, pub-sub topics). As the name suggests, the **read BAS** (interface) actors use these external references to read the real-time values from the required data points in the BAS (i.e., TUC middleware platform) using HTTP request. Such values are sent to the **occupancy status** and **comfort status** (logic) actors, which process them and output current occupied and out-of-comfort zones, respectively. Concurrently, the **price signal** (logic) actor outputs a boolean DF signal, which is true when the current price is higher than a threshold. Then, based on the outputs of these three actors (occupancy, comfort and

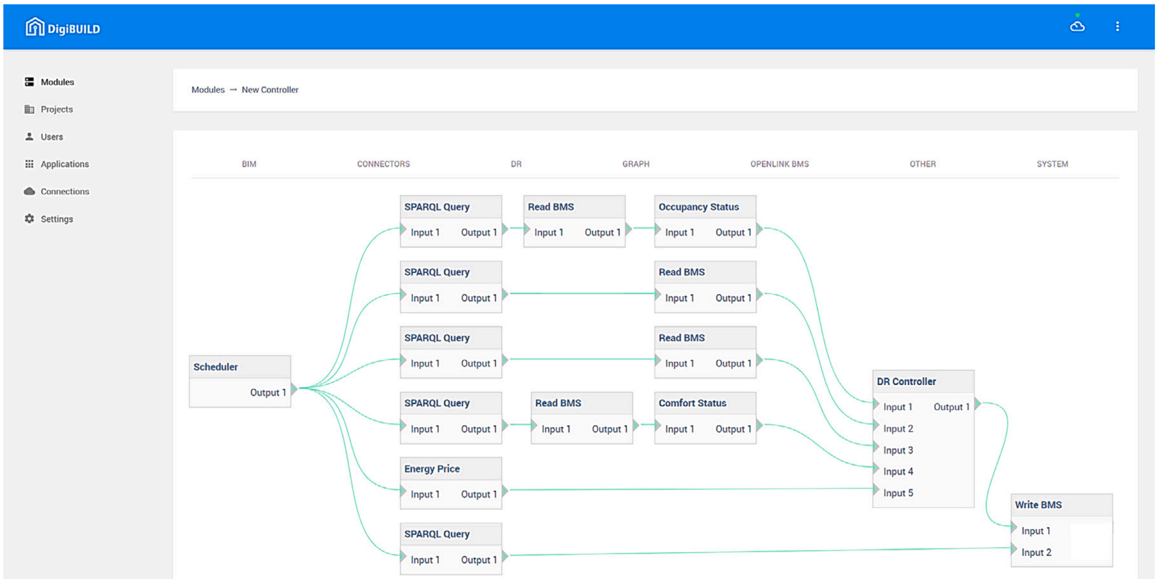


Fig. 8. Proposed platform with the DF control actors for the price-driven application.

price), the **controller** (logic) actor determines new commands for each zone. These commands and their associated values, such as the minimum and maximum setpoint values received from the BAS readers, are then transmitted to the **write BAS** (interface) actor. Upon receiving these commands and values, the BAS writer, along with SPARQL queries actors, infer the external references of corresponding actuation data points for each zone (e.g., identifiers for their setpoints and run request commands).

The features of this implementation align well with the generalisability criteria described in Phase 1. The controls are built using modular functions. They respond to changing zone operating conditions (occupancy and thermal comfort) and dynamic grid signals. The logic actors are abstract from building-specific details (such as point naming conventions) and the underlying ontologies used to model buildings and their data points. The only hard-coded data are the grid signals (energy price and emissions) and the minimum temperature setpoint during DF shedding events, as Brick and SAREF do not support these concepts. While configuring controls requires understanding SAREF and/or Brick to define relevant triple patterns to the concepts they support, our approach decouples the control logic from query definitions. This means that, even if queries need to be adapted to comply with different ontologies, the control logic actors themselves do not need modification, provided that the query results yield references to the same type of data points. The advantage of using well-established ontologies such as Brick and SAREF is that they increase the likelihood of reusing queries across different buildings. This also reduces the learning effort required from application developers, as they do not need to create new, customised queries each time they deploy their applications in a building modelled using a distinct ontology.

Finally, the controls are flexible (easily adaptable) to available data or required customisation. To demonstrate that, we replaced the price actor with an emission actor and deployed an emission-driven DF control without having to modify anything else in the underlying code. While the internal logic and data accessing method changed in the emission actor compared to the price actor, their output was the same, a Boolean value indicating a DF shed event, as needed in the controller logic. This same flexibility can also be applied to improve individual actors without affecting others. For example, the comfort status actor could be enhanced with more complex functions that identify contextual changes based on indoor and outdoor weather conditions while still returning a list of zones that are out of comfort. In that case, we would only need to rewire new inputs (and associated SPARQL) as needed to satisfy the new logic.

4.6. Control results

To evaluate the effectiveness of our framework, we analyse the main inputs and outputs of the controls to observe their responsiveness to the modular actors that assemble the DF applications. This assessment illustrates whether the framework enables the controls to operate according to the logical rules embedded within the actors while leveraging data inputs acquired through SPARQL queries.

The analysis of the proposed DF controls is based on testing the two controls in one of the buildings (H3B) of the TUC facility in February 2024, during the heating season. In the first days of testing, we fine-tuned the controls' parameters according to the field conditions. Because the zones' indoor temperatures could not reach their setpoints, even during the baseline period, we had to adjust the controls' setpoints and comfort range accordingly to allow the DF control to activate. This required to set the default temperature setpoint to 26 °C, which is normally a high value for winter conditions and reducing the comfort range's lower bound to 20 °C, which is normally a low value. Possible factors contributing to this issue include the building's low thermal mass, discrepancies in the accuracy and placement of the temperature sensors we had access to versus the sensors used by the built-in lower-level logic in the HVAC split units, the lack of capacity by the HVAC or user interventions. The latter could involve actions such as opening windows or infrequent operation of the HVAC units by the users, who retain the ability to override the controls by turning the HVAC units off.

Once the controls' parameters were tuned, we tested both DF controls. Fig. 9 shows the results for the controls in a sample zone responding to dynamic electricity prices (left plots) and carbon intensity levels (right plots). The occupancy periods (always on in this zone) are shown in all plots, while the price-driven shed periods are shown on the left and the emission-driven shed periods on the right. The top plots demonstrate how the setpoints change according to the predefined conditions based on occupancy, shed events and indoor temperature. The bottom plots show the resulting power demand changes by the setpoints and the HVAC on/off commands, which also influence the

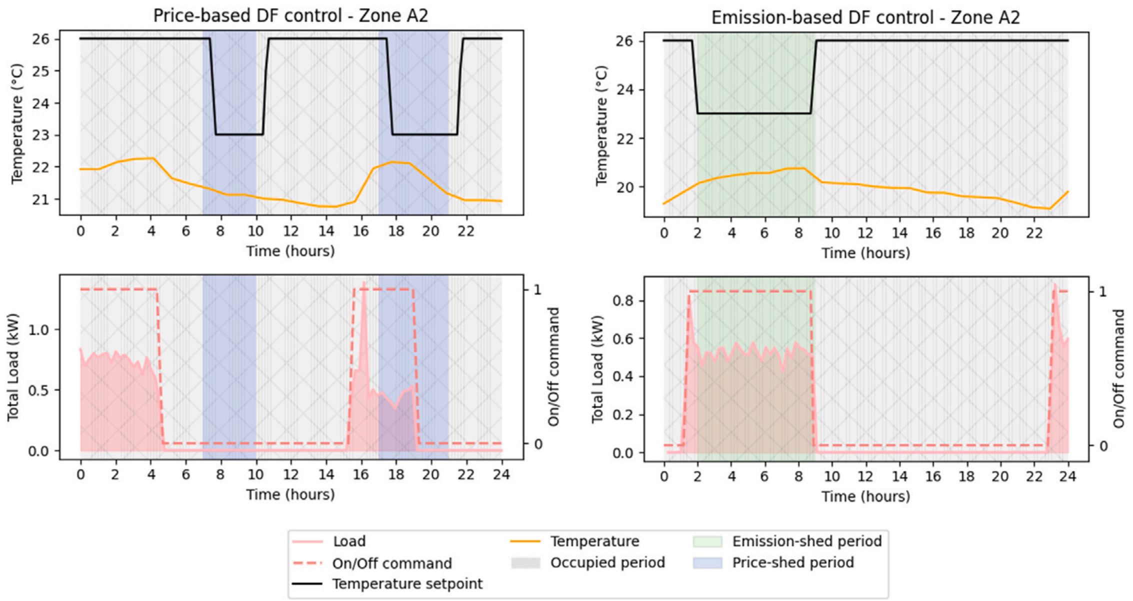


Fig. 9. DF shed activation in a sample zone (left: price-based control, right: emission-based control). Note that variations in temperature and load, driven by the setpoints sent by our controls, are only observable in the plotted data during the active operation of HVAC units. This operational aspect relies entirely on the user control.

demand. Notably, the HVAC units were often inactive for extended periods, which is the reason the temperature response to the setpoints in the top plots is limited to specific periods of the day.

Despite the mismatch between the indoor temperature and the setpoints and the HVAC that operates infrequently, the effectiveness of our controls is still demonstrated in Fig. 9. The figure showcases how our controls can adjust temperature setpoints and effectively manage the load by monitoring conditions and responding to grid signals. It also illustrates that our framework was able to seamlessly integrate various data sources while facilitating the configuration and deployment of control applications based on semantic (reusable) queries. This would streamline the reconfiguration of these same controls in another building with minimal effort. That is, once a new building's semantic model has been generated and/or validated as outlined in the previous section, only minor adjustments would be required. Specifically, it would be necessary to edit or add new **interface actors** to handle data reading and writing according to the building's specific data access methods, whether through pub/sub topics or API endpoints, for example.

While the performance evaluation of our controls falls outside the scope of this paper, the control has shown a potential decrease in the building demand of approximately 35% and 60% on average during shed events for the price and emission controls, respectively (Fig. 10). This assessment was conducted by estimating the baseline demand (labelled as "baseline total load" in the figure) using the High3of5 forecast method [97]. To reduce the influence of external factors leading to varying heating loads during the baseline and flexible periods, we normalised the baseline load by evaluating weather conditions for each period using the Heating Degree Days (HDD) method [98]. This is referred to as the "adjusted baseline total load" in the figure. It is important to note, however, that other factors may have also contributed to these outcomes, such as the occupancy status and whether users had the AC in operation.

Although simplistic, the proposed rule-based control strategies incorporate relevant characteristics from DF applications, such as dynamic load shaping based on pricing and emission signals, which enables the benefit of DF to be achieved [1]; occupancy, which can notably affect overall building energy consumption [99]; and thermal comfort which can significantly impact users acceptance of DF applications [100]. In addition, given the prevalence of rule-based DF strategies as seen in various real-world deployments [9,51], the domain may greatly benefit from the reuse of this control with the support of the proposed approach to facilitate its deployment.

5. Discussion

The implementation of the proposed framework is intended to enhance current control deployment workflows [11], while supporting the advance on research towards new reference architectures using semantic models [39,101–104]. The proof-of-concept undertaken in this work demonstrates the viability of our framework while uncovering a number of important findings that provide insights to examine the research questions outlined in Section 1.

5.1. Brick and SAREF alignment for DF needs

Our framework introduced an approach to generate semantic models that can support the integration of diverse data sources for configuring DF controls. Due to their potential coverage and growing global support [15,17,22,37,45,102], the approach was designed to leverage Brick (v1.3) and SAREF (core v3.2.1, along with its extensions and recommended ontologies). To achieve this, we proposed an instance-level alignment between the two ontologies, allowing instances of building concepts (e.g., sensors) to be mapped to the relevant classes and relationships from both of them. The proposed alignment was derived using approaches based on string similarity and ontology design patterns that facilitate the specification of correspondent classes by examining their names, descriptions and meanings within ontology fragments. The intent was not to provide exhaustive coverage but rather to set the stage for further analysis. For example, 'What main concepts do they lack to support DF?', 'Can Brick and SAREF be fully aligned?', and 'What are their strengths and weaknesses?'.

Following the implementation of our framework, it is evident that both ontologies' classes and relationships can fairly support common inputs and outputs for DF applications, as for the scope of this paper. However, neither ontology can fully describe DF-related requirements. These include grid signals and DF settings (e.g., minimum/maximum temperature setpoint allowed during DF shedding events, or horizon time and allowed setpoint adjustment value as required in load shifting applications). Although Brick includes a few classes that could assist DF settings, such as load shed command,[19] those are not expressive enough to support the controls in this paper. Meanwhile, even though

---

[19] https://brickschema.org/ontology/1.3/classes/Load_Shed_Command/.

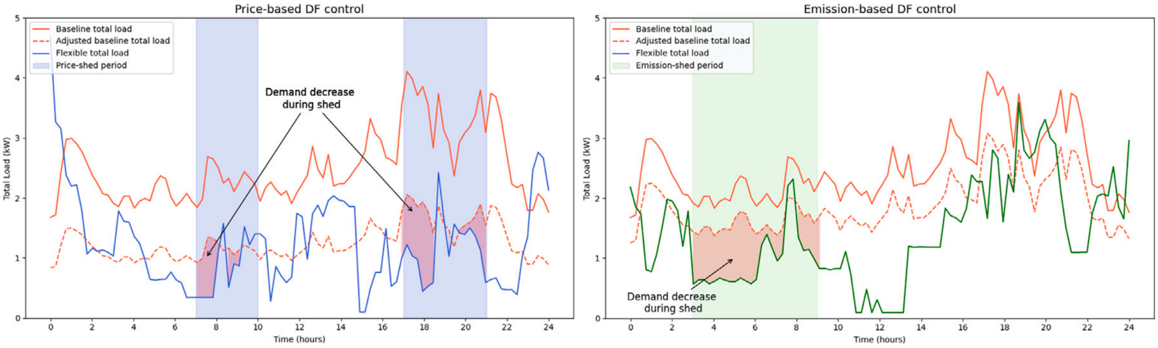


Fig. 10. Building-level comparison between baseline, adjusted baseline, and flexible loads from the emission and price-driven DF controls. While the baseline is the same for both, the adjusted baseline was normalised according to the testing days of each control.

the SAREF energy extension (v1.2.1) includes DF-related concepts, such as a flexibility profile class that enables the maximum power allowed for a given device in a certain period to be specified, it does not model concepts that support temperature adjustment strategies, as needed in this paper. While extending these ontologies or aligning them with others, such as OpenADR for modelling grid signals [57], could address these identified gaps, this was not the primary objective of this work. However, future efforts in this direction are encouraged.

Furthermore, while aligning Brick with SAREF holds significant potential, we could not align all of their concepts due to semantic conflicts. For example, S4BLDG models the concepts *operationTemperatureMin* and *operationTemperatureMax* as object properties with the restriction that they must be linked to the *saref:Measurement* class. In contrast, in Brick, these concepts are modelled as classes. Due to this distinct representation and given that the *saref:Measurement* class was deprecated in the latest version of the SAREF core (while the S4BLDG extension has not been updated yet), we could not align these concepts and decided to use only their corresponding Brick classes in our case study. It is also important to note that because SAREF and Brick have different levels of expressiveness (i.e., SAREF has more generic classes while Brick includes more specific ones), some of the proposed alignments may not always be suitable for all use cases. For example, we proposed to use a set of SAREF concepts (*saref:Sensor*, *saref:observes*, *saref:Property* and *quantitykind:Temperature*) to align with multiple Brick classes, such as *brick:Temperature_Sensor*, *brick:Supply_Air_Temperature_Sensor* and *brick:Return_Air_Temperature_Sensor* classes. Without extending SAREF, this approach may introduce ambiguity and might not be ideal for applications requiring SAREF-driven queries for specific data points.

Finally, when considering a single-ontology solution, Brick and SAREF present their own distinct strengths and weaknesses. Overall, SAREF provides an enhanced abstraction compared to Brick, which could benefit users who may not be familiar with the diverse classifications employed by Brick. However, as detailed in Appendix A, from the 63 concepts we identified to represent common data needs for DF applications, SAREF could only model 40 of them. From those, 25 are for representing multiple measurement and control points using the same sets of classes, which could lead to ambiguity issues if no extensions are considered. Moreover, while adopting our hybrid-storage methodology that does not store temporal data within the semantic models, SAREF relies on the ref-schema associated with Brick to capture data point references/identifiers as stored external data sources. In contrast, Brick offers enhanced expressiveness compared to SAREF, and can model all the concepts outlined in Appendix A. Nevertheless, Brick-driven queries are constrained by specific classes and relationships, requiring users to understand the ontology and the details of model creation. In addition, Brick currently lacks classes to represent more detailed topological information within HVAC systems, such as ducts, pipes, and electrical wiring. This limitation may pose challenges for supporting DF applications in buildings with systems that require more complex modelling than what is covered in this paper. As such, future efforts should focus on the alignment with more suitable ontologies for this matter, such as the Flow Systems Ontology (FSO), as initially investigated in [105] or the upcoming ASHRAE 223[20] standard.

5.2. BIM support for DF configuration

Without the introduction of semantics, the manual, labour-intensive process of integrating and mapping data points to control applications can take most of controls development time. This still remains one of the major obstacles to deploying supervisory control applications, as reported in previous research in this area [15,16]. Connecting data sources with applications in each building demands considerable effort, expertise and time, which becomes particularly crucial during the final phases of commissioning and handover of buildings. Consequently, this increases the overall expenses and hinders the scalability of these applications. Semantic models support and simplify this process by enabling the interoperability and configuration of controls based on semantic, reusable queries.

Several approaches have been proposed to automate the generation of semantic models based on BAS metadata. One issue they face is that this often demands specialised knowledge to decode the spatial and functional context of equipment and their data points, often found scattered across unstructured documentation such as technical drawings and manuals. This is where BIM sources become invaluable. Integrating BIM with BAS data sources facilitates the creation of semantic models. By leveraging BIM's rich spatial context, it minimises the need for human interpretation and intervention in establishing a mapping between data points and their contextual information. This may result in a more efficient, automated and accurate semantic model generation process. Nevertheless, the key to allowing BIM information to be mapped together with BAS data sources for generating semantic models is having a common instance presented in both. In our approach, we integrated BIM and BAS data sources based on the device serial number, which required both to include such an identifier. This was accomplished by generating isolated semantic models and then merging them based on the identifiers of such instances.

While our proposed tools are designed to automate the generation of semantic models, it is important to note that they reduce, but do not entirely eliminate, the need for human involvement. Still, although some manual efforts are still expected to ensure correct inputs (from IFC models and CSV templates) and check for correct outputs (generated models), it may be a one-time investment. In addition, although the

---

[20] https://docs.open223.info/intro.html.

results of this study provide a comprehensive mapping approach for BIM and BAS, specific applications may require particular features, affecting the suggested computational methods. In a sense, the proposed CSV and IFC templates and their corresponding algorithms may be considered an alpha version. If they prove unsuitable for a particular case, they need to be adapted accordingly. For instance, as the current focus is on IFC and CSV, additional developments would be necessary to support other data sources, formats, or use cases that do not have BIM. The current method also focuses exclusively on relationships between points, equipment, and spaces/zones. Enhancements are needed to support more comprehensive topological relationships, including those between equipment. Similarly, any modifications to the underlying ontology (i.e., new versions) will require adaptation in the algorithms. To manage these adaptations across various use cases, distinct versions of the templates and algorithms can be created and systematically tracked and documented, ensuring that each version is preserved and available for reuse.

5.3. Framework benefits for DF deployment

Existing studies on semantic models have been undertaken to address the labour-intensive and ad-hoc issues inherent in business-as-usual workflows to develop, configure and deploy building applications. While these studies lay a solid foundation for semantics-driven applications, they present several research gaps. Most works have focused on analytic purposes rather than controls. Our framework addresses this by supporting controls, particularly in the context of DF, while also having the potential for broad adaptation across other use cases, including analytics and different controls. Moreover, existing works supporting controls often propose new ontologies, which may lead to limited scalability due to the challenges of maintaining and adopting newly developed ontologies. Even studies that advocate for existing ontologies often fall short by not providing sufficient tools to effectively manage them. Our framework not only encourages the use of existing, established ontologies but also provides an approach and the required tools for aligning and generating semantic models based on them.

Some existing works also embed time series data and control logic directly within semantic models rather than using appropriate databases or algorithms. This practice is particularly unsuitable when managing large data streams or complex logic, such as required in DF controls for HVAC systems [79,80,99]. Our framework employs the OBDA approach to decouple control logic, time series data, and semantic models representing specific buildings, aiming to address these issues and facilitate controls' scalability. The framework also adopts a novel control platform, allowing for greater simplicity in the reuse, extension, and customisation of controls, either to enhance existing controls or seamlessly connect them to new buildings. This is facilitated by the platform's underlying actor and microservices architecture and an intuitive drag-and-drop interface that supports effortless reuse and modifications of controls deployed as modular actors (functions scripts).

In summary, our framework has the potential to provide scalable and interoperable DF controls while reducing time and costs. To demonstrate this, Fig. 11 compares two workflows for developing, configuring, and deploying building control applications. Drawing from previous experience, the *before* workflow outlines the main tasks involved in delivering a supervisory HVAC control (similar, though simpler, compared to our proposed control) without using the semantics-driven framework or the actor- and microservices-based platform introduced in this work. In contrast, the *after* workflow shows the main tasks for implementing the proposed DF control using the semantics-driven framework and platform. The same research team deployed both workflows in the same building featured in the case study of this paper, leveraging the same base tasks (performed once) to connect to field data points, initialise the middleware, and assess the building. The difference between them lies in how their controls were subsequently developed, configured, and deployed, and therefore, we focus on these tasks to compare such efforts.

In the *before* workflow, control-related tasks included exporting a BAS points list, manually mapping its points to a bespoke naming convention, programming the control logic, and setting up an API-based interface between control and middleware. This task required about a week of effort and resulted in a hard-coded, building-specific control that lacked portability. Mapping was the most time-intensive step, consistent with studies showing it can comprise up to 29% of control implementation efforts [106]. Industry developers often use proprietary scripts to semi-automate mapping [107]. However, although these scripts can be reused across different buildings by the same developers, they are often inaccessible or inadequate for others, leading to duplication of effort even within the same building [40,108].

In the *after* workflow, control-related tasks were divided into three: generating a semantic model, developing modular control logic actors, and configuring and deploying the control. The first task involved running the proposed open-source BAS-to-RDF and BIM-to-RDF algorithms built on established ontologies. This semi-automated process took about one hour to generate and validate the model, which other developers can easily reuse. The second task required creating control logic actors and corresponding SPARQL text files. Despite taking approximately 8 h, such actors are likely reusable as they are agnostic to specific buildings. The same applies to the SPARQL queries due to their foundation in established ontology concepts. Finally, the third task involved importing and setting up suitable actors from the platform, including newly created logic actors and existing default query and interface actors. It also involved properly setting up the actor, for example, by linking the query actor to the SPARQL text files and configuring the API key in the interface actor. This semi-automated process took around one hour and was greatly supported by the user-friendly interface provided by the actor and microservices platform.

Overall, the *after* workflow shows a 75% reduction in effort for developing, configuring, and deploying similar controls within the same building, assuming that the platform and mapping algorithms proposed in this paper are pre-designed. The estimated efforts were defined by the research team (the authors of this paper) and verified by a leading system integrator to ensure they align with industry practices. It is important to acknowledge, though, that this estimation is preliminary and depends largely on the case study building, the specific controls implemented, and the delivery process. The benefits are expected to be more significant for buildings with more data points. However, further studies across various buildings and scenarios (e.g., point naming quality and system complexity) are necessary to provide this quantitative evidence.

6. Conclusions

Improving the scalability and interoperability of DF applications is crucial to supporting the decarbonisation and digitalisation of the energy sector. In line with these goals, this paper introduces a semantics-driven framework to enable DF controls in real buildings. The framework extends our previous work presented in [40]. Its novelty lies in a new approach for aligning Brick and SAREF concepts and generating semantic models suitable for DF applications while mapping metadata from BIM and BAS sources. We also propose a novel control platform to develop, configure and deploy DF applications in a modular, easily replicable and adaptable way.

While relying on methods predominantly agnostic to particular buildings, our work offers several benefits to researchers and developers of control applications and semantic models (often led by system integrators). The framework assists non-ontology experts with predefined design decisions in specifying an alignment among Brick and SAREF to support DF data exchange needs. In addition, we offer open-source algorithms to link metadata from BIM and BAS data sources into

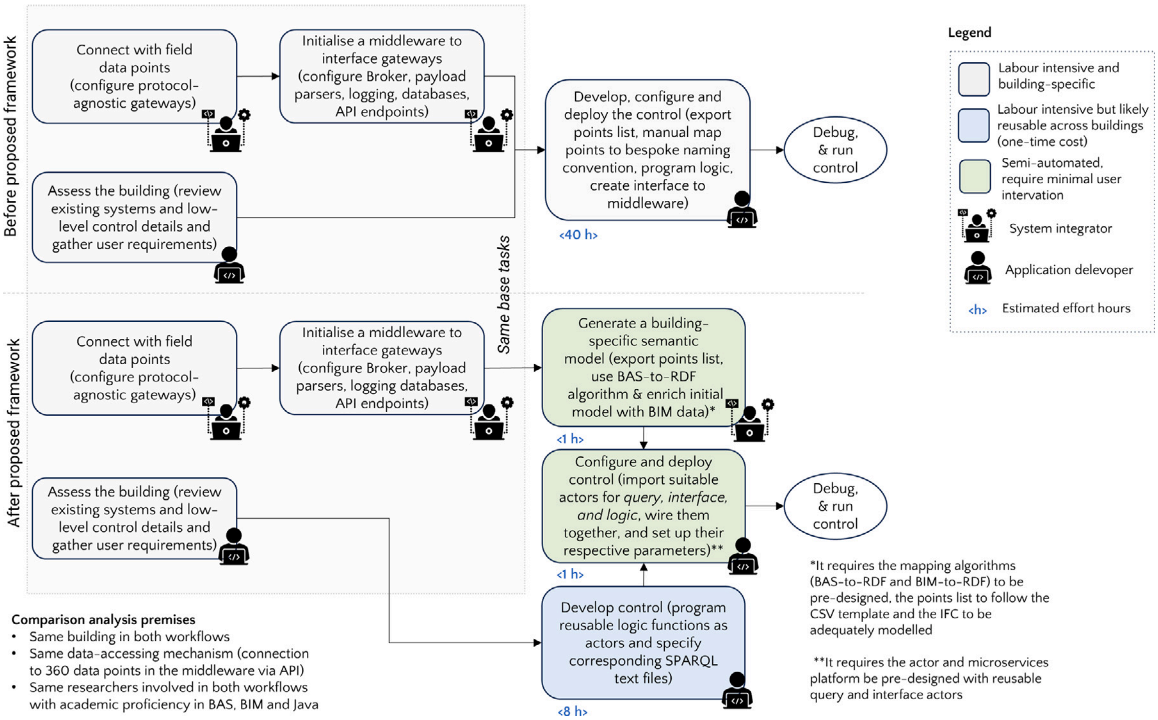


Fig. 11. Comparison between two workflows for developing, configuring, and deploying building control applications by the same research team in the same building featured in the case study of this paper. *Before* workflow outlines the main tasks without using the semantics-driven framework or the actor- and microservices-based platform introduced in this work, while *after* workflow applies them.

unified semantic models compatible with Brick and SAREF. Finally, we propose a structure on how DF control applications logic can be decoupled from specific buildings using a control platform that follows an actor and microservices architecture.

Future efforts include testing more complex control applications across different building archetypes and target systems leveraging the actor and microservices architecture. This would validate the proposed framework's robustness and portability potential while capturing nuances associated with more comprehensive real-world scenarios. We also aim to create a semantics-driven library of reference and portable DF control applications to help scale their adoption.

CRediT authorship contribution statement

Flavia de Andrade Pereira: Writing – original draft, Visualization, Validation, Software, Methodology, Data curation, Conceptualization. **Kyriakos Katsigarakis:** Writing – review & editing, Software. **Dimitrios Rovas:** Writing – review & editing, Methodology, Conceptualization. **Marco Pritoni:** Writing – review & editing, Methodology. **Conor Shaw:** Writing – review & editing. **Lazlo Paul:** Writing – review & editing. **Anand Prakash:** Writing – review & editing. **Susana Martin-Toral:** Writing – review & editing. **Donal Finn:** Writing – review & editing. **James O'Donnell:** Writing – review & editing, Supervision, Methodology, Conceptualization.

Declaration of generative AI in scientific writing

During the preparation of this work, the authors used AI-powered tools, such as Grammarly and ChatGPT, in order to improve readability and language in specific portions of the manuscript. After using this tool/service, the authors reviewed and edited the content as needed and take full responsibility for the content of the publication.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was supported by the CBIM-ETN funded by the European Union's Horizon 2020 research and innovation programme under the Marie Slodowska-Curie grant agreement No 860555, by the COGITO project co-funded by the H2020 programme, by the DigiBUILD project co-funded by the European Union's Horizon Europe research and innovation programme under the grant agreement No 101069658, by the BuildON project co-funded by the UK Research and Innovation, United Kingdom under the UK government's Horizon Europe funding guarantee grant No 10069441, by the Assistant Secretary for Energy Efficiency and Renewable Energy, Building Technologies Office, of the U.S. Department of Energy, United States of America under Contract No DE-AC02-05CH11231 and by California Energy Commission, United States of America through grant No EPC-19-013.

Appendix A. BAS and BIM mapping

Table A.1 outlines the proposed instance-level alignment between Brick (v1.3) and SAREF (v3.2.1), along with SAREF extensions and recommended ontologies, for modelling data needs for DF control applications. The table also presents the proposed mapping from these concepts to BIM and BAS metadata sources, which are the foundation for the proposed algorithms to generate semantic models compatible with the alignment and DF data needs.

Appendix B. Mapping algorithms

Algorithm 1 presents a pseudo-code for the BAS-to-RDF mapping algorithm, while Algorithm 2 presents a pseudo-code for the BIM-to-RDF mapping algorithm. Their full script is available in an open-source repository.[21]

---

[21] https://github.com/ucl-sbde/semantics-driven_controls.git.

Table A.1
Proposed alignment between Brick (v1.3) and SAREF (core v3.2.1, saref4bldg v1.1.2, and quantitykind/qudt v2.1) for DF data needs and their mapping to BIM and BAS sources.

Item	Concept	Concept type	Align type	Brick concept	SAREF concept[a]	Metadata source
1	Site	Spatial	Match	brick:Site	–	BIM
2	Building	Spatial	Match	brick:Building	s4bldg::Building	BIM
3	Storey	Spatial	Match	brick:Storey	–	BIM
4	Space	Spatial	Match	brick:Space	s4bldg:BuildingSpace	BIM
5	Zone	Spatial	Match	brick:Zone	–	BIM
6	Thermostat	Device	Match	brick:Thermostat	s4bldg:UnitaryControlElement	BIM, BAS
7	Split System	Device	Match	brick:Terminal_Unit	s4bldg:FlowTerminal	BIM, BAS
8	Air Handler Unit	Device	Match	brick:Air_Handler Unit	s4bldg:DistributionDevice	BIM, BAS
9	Pump	Device	Match	brick:Pump	s4bldg:Pump	BIM, BAS
10	Boiler	Device	Match	brick:Boiler	s4bldg:Boiler	BIM, BAS
11	Chiller	Device	Match	brick:Chiller	s4bldg:Chiller	BIM, BAS
12	Coil	Device	Match	brick:Coil	s4bldg:Coil	BIM, BAS
13	Damper	Device	Match	brick:Damper	s4bldg:Damper	BIM, BAS
14	Fan	Device	Match	brick:Fan	s4bldg:Fan	BIM, BAS
15	Heat Exchanger	Device	Match	brick:Heat_Exchanger	s4bldg:HeatExchanger	BIM, BAS
16	Valve	Device	Match	brick:Valve	s4bldg:Valve	BIM, BAS
17	Variable Air Volume Box	Device	Match	brick:Variable_Air_Volume_Box	s4bldg:FlowTerminal	BIM, BAS
18	Compressor	Device	Match	brick:Compressor	s4bldg:Compressor	BIM, BAS
19	Occupancy Sensor	Measurement point	Pattern	brick:Occupancy_Sensor	saref:Sensor saref:observes saref:Property quantitykind:Occupancy	BAS
20	Motion Sensor	Measurement point	Pattern	brick:Motion_Sensor	saref:Sensor saref:observes saref:Property quantitykind:Occupancy	BAS
21	Supply Temperature Sensor	Measurement point	Pattern	brick:Supply_Temperature_Sensor	saref:Sensor saref:observes saref:Property quantitykind:Temperature	BAS
22	Return Temperature Sensor	Measurement point	Pattern	brick:Return_Temperature_Sensor	saref:Sensor saref:observes saref:Property quantitykind:Temperature	BAS
23	Discharge Temperature Sensor	Measurement point	Pattern	brick:Discharge_Temperature_Sensor	saref:Sensor saref:observes saref:Property quantitykind:Temperature	BAS
24	Temperature Sensor	Measurement point	Pattern	brick:Temperature_Sensor	saref:Sensor saref:observes saref:Property quantitykind:Temperature	BAS
25	Humidity Sensor	Measurement point	Pattern	brick:Humidity_Sensor	saref:Sensor saref:observes saref:Property quantitykind:Humidity	BAS
26	Power Sensor	Measurement point	Pattern	brick:Power_Sensor	saref:Sensor saref:observes saref:Property quantitykind:Power	BAS
27	Supply Air Static Pressure Sensor	Measurement point	Pattern	brick:Supply_Air_Static_Pressure_Sensor	saref:Sensor saref:observes saref:Property quantitykind:Pressure	BAS
28	Exhaust Air Static Pressure Sensor	Measurement point	Pattern	brick:Exhaust_Air_Static_Pressure_Sensor	saref:Sensor saref:observes saref:Property quantitykind:Pressure	BAS
29	Discharge Air Static Pressure Sensor	Measurement point	Pattern	brick:Discharge_Air_Static_Pressure_Sensor	saref:Sensor saref:observes saref:Property quantitykind:Pressure	BAS
30	Pressure Sensor	Measurement point	Pattern	brick:Pressure_Sensor	saref:Sensor saref:observes saref:Property quantitykind:Pressure	BAS
31	Position Sensor	Measurement point	–	brick:Position_Sensor	–	BAS
32	Speed Sensor	Measurement point	–	brick:Speed_Sensor	–	BAS
33	Supply Air Flow Sensor	Measurement point	–	brick:Supply_Air_Flow_Sensor	–	BAS

(*continued on next page*)

Table A.1 (*continued*).

Item	Concept	Concept type	Align type	Brick concept	SAREF concept[a]	Metadata source
34	Discharge Air Flow Sensor	Measurement point	–	brick:Discharge_Air_Flow_Sensor	–	BAS
35	Mixed Air Flow Sensor	Measurement point	–	brick:Mixed_Air_Flow_Sensor	–	BAS
36	Flow Sensor	Measurement point	–	brick:Flow_Sensor	–	BAS
37	CO2 Sensor	Measurement point	–	brick:CO2_Sensor	–	BAS
38	Supply Air Temperature Heating Setpoint	Control point	Pattern	brick:Supply_Air_Temperature_Heating_Setpoint	saref:Actuator saref:controls saref:Property quantitykind:Temperature	BAS
39	Discharge Air Temperature Heating Setpoint	Control point	Pattern	brick:Discharge_Air_Temperature_Heating_Setpoint	saref:Actuator saref:controls saref:Property quantitykind:Temperature	BAS
40	Heating Temperature Setpoint	Control point	Pattern	brick:Heating_Temperature_Setpoint	saref:Actuator saref:controls saref:Property quantitykind:Temperature	BAS
41	Supply Air Temperature Cooling Setpoint	Control point	Pattern	brick:Supply_Air_Temperature_Cooling_Setpoint	saref:Actuator saref:controls saref:Property quantitykind:Temperature	BAS
42	Cooling Temperature Setpoint	Control point	Pattern	brick:Cooling_Temperature_Setpoint	saref:Actuator saref:controls saref:Property quantitykind:Temperature	BAS
43	Discharge Air Temperature Cooling Setpoint	Control point	Pattern	brick:Discharge_Air_Temperature_Cooling_Setpoint	saref:Actuator saref:controls saref:Property quantitykind:Temperature	BAS
44	Maximum Temperature Setpoint	Control point	Pattern	brick:Max_Air_Temperature_Setpoint	saref:Actuator saref:controls saref:Property quantitykind:Temperature	BAS
45	Minimum Temperature Setpoint	Control point	Pattern	brick:Min_Air_Temperature_Setpoint	saref:Actuator saref:controls saref:Property quantitykind:Temperature	BAS
46	Discharge Air Temperature Setpoint	Control point	Pattern	brick:Discharge_Air_Temperature_Setpoint	saref:Actuator saref:controls saref:Property quantitykind:Temperature	BAS
47	Supply Air Temperature Setpoint	Control point	Pattern	brick:Supply_Air_Temperature_Setpoint	saref:Actuator saref:controls saref:Property quantitykind:Temperature	BAS
48	Temperature Setpoint	Control point	Pattern	brick:Temperature_Setpoint	saref:Actuator saref:controls saref:Property quantitykind:Temperature	BAS
49	Pressure Setpoint	Control point	Pattern	brick:Pressure_Setpoint	saref:Actuator saref:controls saref:Property quantitykind:Pressure	BAS
50	Humidity Setpoint	Control point	Pattern	brick:Humidity_Setpoint	saref:Actuator saref:controls saref:Property quantitykind:Humidity	BAS
51	Cooling Supply Air Flow Setpoint	Control point	–	brick:Cooling_Supply_Air_Flow_Setpoint	–	BAS
52	Cooling Discharge Air Flow Setpoint	Control point	–	brick:Cooling_Discharge_Air_Flow_Setpoint	–	BAS
53	Heating Supply Air Flow Setpoint	Control point	–	brick:Heating_Supply_Air_Flow_Setpoint	–	BAS
54	Heating Discharge Air Flow Setpoint	Control point	–	brick:Heating_Discharge_Air_Flow_Setpoint	–	BAS
55	Supply Air Flow Setpoint	Control point	–	brick:Supply_Air_Flow_Setpoint	–	BAS
56	Discharge Air Flow Setpoint	Control point	–	brick:Discharge_Air_Flow_Setpoint	–	BAS
57	Flow Setpoint	Control point	–	brick:Flow_Setpoint	–	BAS
58	Speed Setpoint	Control point	–	brick:Speed_Setpoint	–	BAS
59	Frequency Setpoint	Control point	–	brick:Frequency_Setpoint	–	BAS
60	On Off Command	Control point	–	brick:On_Off_ Command	–	BAS
61	Run Request Command	Control point	–	brick:Run_Request Command	–	BAS
62	Mode Command	Control point	–	brick:Mode_Command	–	BAS
63	Lockout Command	Control point	–	brick:Lockout_Command	–	BAS

[a] Including SAREF extensions and recommended ontologies.

Algorithm 1: Pseudo-code for the BAS-to-RDF mapping algorithm

```
Input: CSVFile
Output: BASRDFModel

Step 1: Initialise RDF model and namespaces ;
rdfModel ← ModelFactory.createDefaultModel() ;
InitialiseNamespaces(rdfModel) ;

Step 2: Load CSV data ;
csvData ← LoadCSV(CSVFile) ;

Step 3: Create RDF resources for device and point concepts ;
foreach record ∈ csvData do
    Step 3.1: Identify unique device IDs ;
    uniqueDevices ← IdentifyUniqueID(device_identifier);

    Step 3.2: Iterate over the common device IDs to identify devices' classes and composition ;
    foreach deviceIdentifier ∈ uniqueDevices do

        Step 3.3: Identify devices' Brick/SAREF classes and create RDF resources ;
        deviceResource ← CreateResource(rdfModel, deviceIdentifier) ;
        deviceBrickClass ← IdentifyBrickClass(device_name) ;
        deviceResource ← AddProperty(rdfModel.Brick + deviceBrickClass) ;
        deviceSAREFClass ← IdentifySAREFClass(device_name) ;
        deviceResource ← AddProperty(rdfModel.SAREF + deviceSAREFClass) ;

        Step 3.4: Identify and link points to devices ;
        foreach pointIdentifier ∈ data_point_identifier do
            dataPointResource ← CreateResource(rdfModel, pointIdentifier) ;
            pointBrickClass ← IdentifyBrickClass(data_point_name) ;
            dataPointResource ← AddProperty(rdfModel.Brick + pointBrickClass) ;
            pointSAREFClass ← IdentifySAREFClass(data_point_name) ;
            dataPointResource ← AddProperty(rdfModel.SAREF + pointSAREFClass) ;
            dataPointResource ← AddProperty(rdfModel.Brick + "hasTimeseriesId", pointIdentifier) ;
            deviceResource ← AddProperty(rdfModel.Brick + "hasPoint", dataPointResource) ;
            deviceResource ← AddProperty(rdfModel.SAREF + "consistOf", dataPointResource) ;

Step 4: Write RDF model to file ;
BASRDFModel ← WriteRDFModel(rdfModel) ;
```

Algorithm 2: Pseudo-code for the BIM-to-RDF mapping algorithm

```
Input: CSVFile
Output: BIMRDFModel
Step 1: Initialise RDF model and namespaces ;
rdfModel ← ModelFactory.createDefaultModel() ;
InitialiseNamespaces(rdfModel) ;
Step 2: Load IFC data ;
ifcData ← LoadIFC(IFCFile) ;
Step 3: Create RDF resources for spatial and devices concepts ;
foreach ifcProject ∈ ifcData do
    Step 3.1: Identify instances of sites and iterate to identify their classes and composition) ;
    foreach site ∈ ifcSite do
        siteResource ← CreateResource(rdfModel, IfcSite_ExpressId) ;
        siteResource ← AddProperty(rdfModel.Brick + Site) ;
        Step 3.2: Identify instances of buildings and iterate to identify their classes and composition ;
        foreach building ∈ siteResource.IfcRelAggregates() do
            buildingResource ← CreateResource(rdfModel, IfcBuilding_ExpressId) ;
            buildingResource ← AddProperty(rdfModel.Brick + Building) ;
            buildingResource ← AddProperty(rdfModel.SAREF + Building) ;
            siteResource ← AddProperty(rdfModel.Brick + "hasPart", buildingResource) ;
            Step 3.3: Identify instances of storeys, and iterate to identify their classes and composition ;
            foreach storey ∈ buildingResource.IfcRelAggregates() do
                storeyResource ← CreateResource(rdfModel, IfcBuildingStorey_ExpressId) ;
                storeyResource ← AddProperty(rdfModel.Brick + Storey) ;
                buildingResource ← AddProperty(rdfModel.Brick + "hasPart", storeyResource) ;
                Step 3.4: Identify instances of spaces and iterate to identify their classes and composition ;
                foreach space ∈ storeyResource.IfcRelAggregates() do
                    spaceResource ← CreateResource(rdfModel, IfcSpace_ExpressId) ;
                    spaceResource ← AddProperty(rdfModel.Brick + Space) ;
                    spaceResource ← AddProperty(rdfModel.SAREF + BuildingSpace) ;
                    storeyResource ← AddProperty(rdfModel.Brick + "hasPart", spaceResource) ;
                    Step 3.5: Identify and link devices to spaces ;
                    foreach device ∈ spaceResource.IfcRelContainedInStapialStructure() do
                        deviceResource ← CreateResource(rdfModel, IfcElement_identifier) ;
                        deviceBrickClass ← IdentifyBrickClass(IfcElement_type) ;
                        deviceResource ← AddProperty(rdfModel.Brick + deviceBrickClass) ;
                        deviceSAREFClass ← IdentifySAREFClass(IfcElement_type) ;
                        deviceResource ← AddProperty(rdfModel.SAREF + deviceSAREFClass) ;
                        spaceResource ← AddProperty(rdfModel.Brick + "hasPoint", deviceResource) ;
                        spaceResource ← AddProperty(rdfModel.SAREF + "contains", deviceResource) ;
                    Step 3.6: Identify instances of zones as a group of spaces ;
                    foreach zone ∈ spaceResource.IfcRelAssignsToGroup() do
                        zoneResource ← CreateResource(rdfModel, IfcZone_ExpressId) ;
                        zoneResource ← AddProperty(rdfModel.Brick + Zone) ;
                        zoneResource ← AddProperty(rdfModel.Brick + "hasPart", spaceResource) ;
Step 4: Write RDF model to file ;
BIMRDFModel ← WriteRDFModel(rdfModel) ;
```

Data availability

The data and code for this study are available in our open-access repository via the following link: https://github.com/ucl-sbde/semantics-driven_controls.git.

References

[1] A. Satchwell, M. Piette, A. Khandekar, J. Granderson, N. Frick, R. Hledik, A. Faruqui, L. Lam, S. Ross, J. Cohen, K. Wang, D. Urigwe, D. Delurey, M. Neukomm, D. Nemtzow, A National Roadmap for Grid-Interactive Efficient Buildings, Tech. Rep., 2021, http://dx.doi.org/10.2172/1784302.

[2] M. Antretter, M. Klobasa, M. Kühnbach, M. Singh, K. Knorr, J. Schütt, J. Boer, O. Rolser, D. Hernandez Diaz, F. Fitzschen, A. Garcerán, R. Reina, S. Stemmer, J. Steinbach, E. Popovski, Digitalisation of Energy Flexibility, Publications Office of the European Union, 2022, http://dx.doi.org/10.2833/113770.

[3] S.O. Jensen, A. Marszal-Pomianowska, R. Lollini, W. Pasut, A. Knotzer, P. Engelmann, A. Stafford, G. Reynders, IEA EBC Annex 67 energy flexible buildings, Energy Build. 155 (2017) 25–34, http://dx.doi.org/10.1016/j.enbuild.2017.08.044.

[4] V. Reif, T.I. Strasser, J. Jimeno, et al., Towards an interoperability roadmap for the energy transition, Elektrotech. Informationstech. 140 (2023) 478–487, http://dx.doi.org/10.1007/s00502-023-01144-2.

[5] E. Commission, D.-G. for Energy, Digital Flexible Solutions for the Energy System – Terms of Reference, Publications Office of the European Union, 2022, http://dx.doi.org/10.2833/47807.

[6] E. Zhou, E. Hale, E. Present, Building flexibility revenue in modeled future bulk power systems with varying levels of renewable energy, Heliyon 8 (7) (2022) http://dx.doi.org/10.1016/j.heliyon.2022.e09865.

[7] IEA, More Efficient and Flexible Buildings are Key to Clean Energy Transitions, IEA, Paris, 2024, URL https://www.iea.org/commentaries/more-efficient-and-flexible-buildings-are-key-to-clean-energy-transitions, Licence: CC BY 4.0.

[8] J. Liu, L. Yu, R. Yin, M.A. Piette, M. Prittoni, A. Casillas, P. Schwartz, Factors influencing building demand flexibility, 2023, http://dx.doi.org/10.20357/B71882, Lawrence Berkeley National Laboratory.

[9] H. Li, Z. Wang, T. Hong, M.A. Piette, Energy flexibility of residential buildings: A systematic review of characterization and quantification methods and applications, Adv. Appl. Energy 3 (2021) http://dx.doi.org/10.1016/j.adapen.2021.100054.

[10] R. Li, A.J. Satchwell, D. Finn, T.H. Christensen, M. Kummert, J. Le Dréau, R.A. Lopes, H. Madsen, J. Salom, G. Henze, K. Wittchen, Ten questions concerning energy flexibility in buildings, Build. Environ. 223 (2022) 109461, http://dx.doi.org/10.1016/j.buildenv.2022.109461.

[11] W. Wang, M.R. Brambley, W. Kim, S. Somasundaram, A.J. Stevens, Automated point mapping for building control systems: Recent advances and future research needs, Autom. Constr. 85 (2018) 107–123, http://dx.doi.org/10.1016/j.autcon.2017.09.013.

[12] J. Koh, S. Ray, J. Hodges, Information mediator for demand response in electrical grids and buildings, in: 2017 IEEE 11th International Conference on Semantic Computing, ICSC, 2017, pp. 73–76, http://dx.doi.org/10.1109/ICSC.2017.26.

[13] T. Peffer, G. Fierro, P. Raftery, C.D. Roa, M. Pritoni, M. Wetter, et al., Skewering the silos: Using brick to enable portable analytics, modeling and controls in buildings, 2023, URL https://escholarship.org/uc/item/04w0b9n2.

[14] M. Pritoni, D. Paine, G. Fierro, C. Mosiman, M. Poplawski, A. Saha, J. Bender, J. Granderson, Metadata schemas and ontologies for building energy applications: A critical review and use case analysis, Energies 14 (7) (2021) http://dx.doi.org/10.3390/en14072024.

[15] H. Bergmann, C. Mosiman, A. Saha, S. Haile, W. Livingood, S. Bushby, G. Fierro, J. Bender, M. Poplawski, J. Granderson, M. Pritoni, Semantic interoperability to enable smart, grid-interactive efficient buildings, 2020, p. 18, http://dx.doi.org/10.20357/B7S304.

[16] C.D. Roa, P. Raftery, R. Sun, L. Paul, A.K. Prakash, M. Pritoni, G. Fierro, T. Peffer, Towards a stronger foundation: Digitizing commercial buildings with brick to enable portable advanced applications, 2022, http://dx.doi.org/10.20357/B7ZG6R.

[17] G. Fierro, P. Pauwels, Survey of metadata schemas for datadriven smart buildings (Annex 81), 2022, CSIRO, Australia, URL https://annex81.iea-ebc.org/publications.

[18] C. Hoare, R. Aghamolaei, M. Lynch, A. Gaur, J. O'Donnell, A linked data approach to multi-scale energy modelling, Adv. Eng. Inform. 54 (2022) 101719, http://dx.doi.org/10.1016/j.aei.2022.101719, URL https://www.sciencedirect.com/science/article/pii/S147403462200177X.

[19] A. Cimmino, J. Cano-Benito, A. Fernández-Izquierdo, C. Patsonakis, A.C. Tsolakis, R. García-Castro, D. Ioannidis, D. Tzovaras, A scalable, secure, and semantically interoperable client for cloud-enabled demand response, Future Gener. Comput. Syst. 141 (2023) 54–66, http://dx.doi.org/10.1016/j.future.2022.11.004, URL https://www.sciencedirect.com/science/article/pii/S0167739X22003648.

[20] D. Calvanese, L. Ding, A. Mosca, G. Xiao, Realizing ontology-based reusable interfaces for data access via virtual knowledge graphs, in: CHItaly 2021: 14th Biannual Conference of the Italian SIGCHI Chapter, ACM, Bolzano Italy, 2021, pp. 1–5, http://dx.doi.org/10.1145/3464385.3464744.

[21] A. Roth, M. Wetter, K. Benne, D. Blum, Y. Chen, G. Fierro, M. Pritoni, A. Saha, D. Vrabie, Towards digital and performance-based supervisory HVAC control delivery, in: ACEEE Summer Study on Energy Efficiency in Buildings, 2022, http://dx.doi.org/10.20357/B70G62.

[22] 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, K. Whitehouse, Brick : Metadata schema for portable smart building applications, Appl. Energy 226 (2018) 1273–1292, http://dx.doi.org/10.1016/j.apenergy.2018.02.091.

[23] N.M. Tomasevic, M.C. Batic, L.M. Blanes, M.M. Keane, S. Vranes, Ontology-based facility data model for energy management, Adv. Eng. Inform. 29 (4) (2015) 971–984, http://dx.doi.org/10.1016/j.aei.2015.09.003.

[24] A. Kucera, T. Pitner, Semantic BMS: Allowing usage of building automation data in facility benchmarking, Adv. Eng. Inform. 35 (2018) 69–84, http://dx.doi.org/10.1016/j.aei.2018.01.002.

[25] T. Li, Y. Zhao, C. Zhang, K. Zhou, X. Zhang, A semantic model-based fault detection approach for building energy systems, Build. Environ. 207 (2022) 108548, http://dx.doi.org/10.1016/j.buildenv.2021.108548, URL https://www.sciencedirect.com/science/article/pii/S0360132321009410.

[26] J. Li, N. Li, B. Yue, R. Yan, K. Farruh, A. Li, K. Li, Research on the semantic web representation for building operation with variable refrigerant flow systems, J. Build. Eng. 56 (2022) 104792, http://dx.doi.org/10.1016/j.jobe.2022.104792, URL https://www.sciencedirect.com/science/article/pii/S2352710222008051.

[27] G. Fierro, M. Pritoni, M. AbdelBaky, P. Raftery, T. Peffer, G. Thomson, D.E. Culler, Mortar: An open testbed for portable building analytics, in: Proceedings of the 5th Conference on Systems for Built Environments, BuildSys '18, Association for Computing Machinery, New York, NY, USA, 2018, pp. 172–181, http://dx.doi.org/10.1145/3276774.3276796.

[28] D. Mavrokapnidis, G. Fierro, M. Husmann, I. Korolija, D. Rovas, SeeQ: A programming model for portable data-driven building applications, in: BuildSys '23: The 10th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation, 2023, http://dx.doi.org/10.1145/3600100.3623744.

[29] J. Shi, Z. Pan, L. Jiang, X. Zhai, An ontology-based methodology to establish city information model of digital twin city by merging BIM, GIS and IoT, Adv. Eng. Inform. 57 (2023) 102114, http://dx.doi.org/10.1016/j.aei.2023.102114, URL https://www.sciencedirect.com/science/article/pii/S1474034623002422.

[30] J. Koh, B. Balaji, D. Sengupta, J. McAuley, R. Gupta, Y. Agarwal, Scrabble: Transferrable semi-automated semantic metadata normalization using intermediate representation, in: BuildSys'18, 2018, http://dx.doi.org/10.1145/3276774.3276795.

[31] G. Fierro, A. Saha, T. Shapinsky, M. Steen, H. Eslinger, Application-driven creation of building metadata models with semantic sufficiency, in: Proceedings of the 9th ACM International Conference on Systems for Energy-Efficient Built Environments, BuildSys '22, Association for Computing Machinery, Boston, Massachusetts, USA, 2022, http://dx.doi.org/10.1145/3563357.3564083.

[32] D. Mavrokapnidis, K. Katsigarakis, P. Pauwels, E. Petrova, I. Korolija, D. Rovas, A linked-data paradigm for the integration of static and dynamic building data in digital twins, in: Proceedings of the 8th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation, BuildSys '21, Association for Computing Machinery, Coimbra, Portugal, 2021, pp. 369–372, http://dx.doi.org/10.1145/3486611.3491125.

[33] L. Chamari, E. Petrova, P. Pauwels, A web-based approach to BMS, BIM and IoT integration: a case study, in: CLIMA 2022 Conference, 2022, http://dx.doi.org/10.34641/clima.2022.228.

[34] NBS national BIM report 2020, 2020, URL https://www.thenbs.com/PublicationIndex/documents/details?Pub=NBS&DocId=329337. (Accessed: 10 February 2024).

[35] F. de Andrade Pereira, C. Shaw, S. Martín-Toral, R. Sanz Jimeno, D. Finn, J. O'Donnell, Exchange requirements to support demand side management using BIM and building automation system domains, in: Computing in Construction, vol. 1, University College Dublin, 2021, pp. 253–260, http://dx.doi.org/10.35490/EC3.2021.202.

[36] J. Le Dréau, R.A. Lopes, S. O'Connell, D. Finn, M. Hu, H. Queiroz, D. Alexander, A. Satchwell, D. Österreicher, B. Polly, A. Arteconi, F. de Andrade Pereira, M. Hall, T. Kırant-Mitić, H. Cai, H. Johra, H. Kazmi, R. Li, A. Liu, L. Nespoli, M.H. Saeed, Developing energy flexibility in clusters of buildings: A critical analysis of barriers from planning to operation, Energy Build. 300 (2023) 113608, http://dx.doi.org/10.1016/j.enbuild.2023.113608.

[37] C. Roa, P. Raftery, A. Prakash, T. Peffer, Field Demonstration of the Brick Ontology to Scale up the Deployment of ASHRAE Guideline 36 Control Sequences, Tech. Rep., Lawrence Berkeley National Laboratory: Energy Technologies, 2023, URL https://escholarship.org/uc/item/53c0t2b1.

[38] F. De Andrade Pereira, C. Shaw, S. Martín-Toral, J. L. Hernández, R. Sanz Jimeno, D. Finn, J. O'Donnell, Towards semantic interoperability for demand-side management: a review of BIM and BAS ontologies, in: Computing in Construction, vol. 3, University College Dublin, 2022, http://dx.doi.org/10.35490/EC3.2022.154.

[39] L. Paul, F. De Andrade Pereira, S.w. Ham, M. Pritoni, R. Brown, J.D. Feng, Open building operating system: an open-source grid responsive control platform for buildings, Sci. Technol. Built Environ. (2023) URL https://buildings.lbl.gov/publications/open-building-operating-system-open.

[40] F. de Andrade Pereira, L. Paul, M. Pritoni, A. Casillas, A. Prakash, W. Huang, C. Shaw, S. Martin-Toral, D. Finn, J.O. Donnell, Enabling portable demand flexibility control applications in virtual and real buildings, J. Build. Eng. (2024) 108645, http://dx.doi.org/10.1016/j.jobe.2024.108645.

[41] J. Falcão, C. Cândido, D. Silva, J. Sousa, M. Pereira, D. Rua, C. Gouveia, F. Coelho, R. Bessa, A. Lucas, Enabling interoperable flexibility and standardized grid support services, in: CIRED 2021 - the 26th International Conference and Exhibition on Electricity Distribution, vol. 2021, 2021, pp. 3029–3033, http://dx.doi.org/10.1049/icp.2021.2045.

[42] M. Tosic, F.A. Coelho, B. Nouwt, D.E. Rua, A. Tomcic, S. Pesic, Towards a cross-domain semantically interoperable ecosystem, in: Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining, WSDM '22, Association for Computing Machinery, New York, NY, USA, 2022, pp. 1640–1641, http://dx.doi.org/10.1145/3488560.3508496.

[43] W. Kassab, K.A. Darabkh, A–Z survey of Internet of Things: Architectures, protocols, applications, recent advances, future directions and recommendations, J. Netw. Comput. Appl. 163 (2020) 102663, http://dx.doi.org/10.1016/j.jnca.2020.102663.

[44] J. Giao, A.A. Nazarenko, F. Luis-Ferreira, D. Gonçalves, J. Sarraipa, A framework for Service-Oriented Architecture (SOA)-based IoT application development, Processes 10 (9) (2022) http://dx.doi.org/10.3390/pr10091782, URL https://www.mdpi.com/2227-9717/10/9/1782.

[45] E. Commission, D.-G. for Energy, M. Couto, K. Kotsalos, K. Kukk, European (Energy) Data Exchange Reference Architecture 3.0, Publications Office of the European Union, 2023, http://dx.doi.org/10.2833/815043.

[46] M. Uslar, S. Rohjans, C. Neureiter, F. Andrén, J. Velasquez, C. Steinbrink, V. Efthymiou, G. Migliavacca, S. Horsmanheimo, H. Brunner, T. Strasser, Applying the smart grid architecture model for designing and validating system-of-systems in the power and energy domain: A European perspective, Energies 12 (2019) 258, http://dx.doi.org/10.3390/en12020258.

[47] T. Hartmann, A. Trappey, Advanced engineering informatics - Philosophical and methodological foundations with examples from civil and construction engineering, Develop. Built Environ. 4 (2020) 100020, http://dx.doi.org/10.1016/j.dibe.2020.100020.

[48] G. Beydoun, G. Low, A. Gill, M. Moniruzzaman, J. Shen, Tailoring ontology retrieval for supporting requirements analysis, Adv. Eng. Inform. 59 (2024) 102231, http://dx.doi.org/10.1016/j.aei.2023.102231.

[49] R.G. Junker, A.G. Azar, R.A. Lopes, K.B. Lindberg, G. Reynders, R. Relan, H. Madsen, Characterizing the energy flexibility of buildings and districts, Appl. Energy 225 (2018) 175–182, http://dx.doi.org/10.1016/j.apenergy.2018.05.037.

[50] J.-L. Hippolyte, Y. Rezgui, H. Li, B. Jayan, S. Howell, Ontology-driven development of web services to support district energy applications, Autom. Constr. 86 (2018) 210–225, http://dx.doi.org/10.1016/j.autcon.2017.10.004.

[51] A.Q. Santos, B.N. Jørgensen, H. Kazmi, R. Ruusu, A. Hasan, T. Péan, Y. Zhou, S. Cao, Y.J. Yu, J. Clauß, R.G. Junker, C. Finck, D. Christantoni, D.P. Finn, A.S. Derakhtenjani, J.A. Candanedo, A. Athienitis, A. Kathirgamanathan, Control Strategies and Algorithms for Obtaining Energy Flexibility in Buildings, Tech. Rep., 2019, URL https://www.annex67.org/media/1898/control-strategies.pdf.

[52] A.A. Bhattacharya, J. Ploennigs, D. Culler, Short paper: analyzing metadata schemas for buildings — The good, the bad, and the ugly, in: BuildSys'15, 2015, http://dx.doi.org/10.1145/2821650.2821669.

[53] L. Daniele, W. Strabbing, B. Roelofsen, A. Aalberts, P. Stapersma, Study on Ensuring Interoperability for Enabling Demand side Flexibility, Publications Office, LU, 2018, URL https://data.europa.eu/doi/10.2759/26799.

[54] O. Genest, G. Olivier, G. Roy, S. Zikos, H. Mirtaheri, D. Sebilleau, Bridge: Interoperability of flexibility assets - Data Management Working Group, 2021, European Comission, URL https://energy.ec.europa.eu/system/files/2021-06/bridge_wg_data_management_interoperability_of_flexibility_assets_report_2020-2021_0.pdf.

[55] J.L. Hippolyte, S. Howell, B. Yuce, M. Mourshed, H.A. Sleiman, M. Vinyals, L. Vanhee, Ontology-based demand-side flexibility management in smart grids using a multi-agent system, in: 2016 IEEE International Smart Cities Conference, ISC2, 2016, pp. 1–7, http://dx.doi.org/10.1109/ISC2.2016.7580828.

[56] S. Howell, Y. Rezgui, J.-L. Hippolyte, M. Mourshed, Semantic Interoperability for Holonic Energy Optimization of Connected Smart Homes and Distributed Energy Resources, 2016, pp. 259–268, http://dx.doi.org/10.1201/9781315386904.

[57] A. Fernandez-Izquierdo, A. Cimmino, C. Patsonakis, A.C. Tsolakis, R. Garcia-Castro, D. Ioannidis, D. Tzovaras, OpenADR ontology: Semantic enrichment of demand response strategies in smart grids, in: 2020 International Conference on Smart Energy Systems and Technologies, SEST, IEEE, Istanbul, Turkey, 2020, pp. 1–6, http://dx.doi.org/10.1109/SEST48500.2020.9203093.

[58] G. Santos, Z. Vale, P. Faria, L. Gomes, BRICKS: Building's reasoning for intelligent control knowledge-based system, Sustainable Cities Soc. 52 (2020) 101832, http://dx.doi.org/10.1016/j.scs.2019.101832.

[59] G. Santos, T. Pinto, Z. Vale, R. Carvalho, B. Teixeira, C. Ramos, Upgrading BRICKSthe context-aware semantic rule-based system for intelligent building energy and security management, Energies 14 (15) (2021) 4541, http://dx.doi.org/10.3390/en14154541.

[60] H. Li, T. Hong, A semantic ontology for representing and quantifying energy flexibility of buildings, Adv. Appl. Energy 8 (2022) 100113, http://dx.doi.org/10.1016/j.adapen.2022.100113.

[61] J. Hviid, A. Johansen, G. Fierro, M.B. Kjærgaard, Service portability and information discovery in building operating systems using semantic modeling, in: Proceedings of the 11th International Conference on the Internet of Things, IoT '21, Association for Computing Machinery, New York, NY, USA, 2022, pp. 110–117, http://dx.doi.org/10.1145/3494322.3494337.

[62] L. Daniele, M. Solanki, F. den Hartog, J. Roes, Interoperability for smart appliances in the IoT world, in: P. Groth, E. Simperl, A. Gray, M. Sabou, M. Krötzsch, F. Lecue, F. Flöck, Y. Gil (Eds.), The Semantic Web ISWC 2016, in: Lecture Notes in Computer Science, Springer International Publishing, Cham, 2016, pp. 21–29, http://dx.doi.org/10.1007/978-3-319-46547-0_3.

[63] I. Esnaola-Gonzalez, F.J. Díez, L. Berbakov, N. Tomasevic, P. Štorek, M. Cruz, P. Kirketerp, Semantic interoperability for demand-response programs: RESPOND project's use case, in: 2018 Global Internet of Things Summit, GIoTS, 2018, pp. 1–6, http://dx.doi.org/10.1109/GIOTS.2018.8534568.

[64] I. Esnaola-Gonzalez, F. Díez, Integrating building and IoT data in demand response solutions, in: Proceedings of the 7th Linked Data in Architecture and Construction Workshop, LDAC2019, 2019, URL https://ceur-ws.org/Vol-2389/07paper.pdf.

[65] T. Peffer, G. Fierro, P. Raftery, C.D. Roa, M. Pritoni, M. Wetter, et al., Skewering the silos: using Brick to enable portable analytics, modeling and controls in buildings, 2023, URL https://escholarship.org/uc/item/04w0b9n2, UC Berkeley: Center for the Built Environment.

[66] ASHRAE, ASHRAE BACnet committee, project haystack and Brick schema collaborating to provide unified data semantic modeling solution, 2018, URL https://www.ashrae.org/about/news/2018/ashrae-s-bacnet-committee-project-haystack-and-brick-schema-collaborating-to-provide-unified-data-semantic-modeling-solution. (Accessed: 19 August 2024).

[67] W. Strabbing, P. Stapersma, B. Roelofsen, L. Daniele, A. Aalberts, Study on ensuring interoperability for enabling demand side flexibility, in: European Commission and Directorate-General for Communications Networks, Content and Technology, Publications Office, 2019, http://dx.doi.org/10.2759/26799.

[68] World Wide Web Consortium, ConverterToRdf, 2020, URL https://www.w3.org/wiki/ConverterToRdf. (Accessed: 12 April 2024).

[69] A. Dimou, M. Vander Sande, P. Colpaert, R. Verborgh, E. Mannens, R. Van de Walle, RML: A Generic Language for Integrated RDF Mappings of Heterogeneous Data, vol. 1184, 2014, URL https://api.semanticscholar.org/CorpusID:6564357, ISSN: 1613-0073.

[70] A. Cimmino, R. García-Castro, Helio: a framework for implementing the life cycle of knowledge graphs, Semantic Web (2022) http://dx.doi.org/10.3233/SW-233224.

[71] J. Koh, D. Sengupta, R. McAuley, B. Balaji, Y. Agarwal, Scrabble: Converting unstructured metadata into brick for many buildings, in: BuildSys'17, 2017, http://dx.doi.org/10.1145/3137133.3141448.

[72] M.H. Rasmussen, M. Lefrançois, G.F. Schneider, P. Pauwels, BOT: The building topology ontology of the W3C linked building data group, Semantic Web (2021) 143–161, http://dx.doi.org/10.3233/SW-200385.

[73] G. Fierro, OpenRefine server-based reconciliation API, 2020, https://github.com/BrickSchema/reconciliation-api. (Accessed: 12 March 2024).

[74] P. Pauwels, W. Terkaj, EXPRESS to OWL for construction industry: Towards a recommendable and usable ifcOWL ontology, Autom. Constr. 63 (2016) 100–133, http://dx.doi.org/10.1016/j.autcon.2015.12.003.

[75] M. Bonduel, J. Oraskari, P. Pauwels, M. Vergauwen, R. Klein, The IFC to linked building data converter - current status, in: LDAC, 2018, URL https://api.semanticscholar.org/CorpusID:53505961.

[76] R. Sacks, Z. Wang, B. Ouyang, D. Utkucu, S. Chen, Toward artificially intelligent cloud-based building information modelling for collaborative multidisciplinary design, Adv. Eng. Inform. 53 (2022) 101711, http://dx.doi.org/10.1016/j.aei.2022.101711.

[77] L. Paul, F. De Andrade Pereira, A. Prakash, S.w. Ham, M. Pritoni, R. Brown, J.D. Feng, Open building operating system: a grid responsive semantics-driven control platform for buildings, Sci. Technol. Built Environ. (2024).

[78] R. Reda, A. Carbonaro, V. de Boer, R. Siebes, R. van der Weerdt, B. Nouwt, L. Daniele, Supporting smart home scenarios using OWL and SWRL rules, Sensors (Basel) 22 (11) (2022) 4131, http://dx.doi.org/10.3390/s22114131.

[79] P. Pauwels, A. Costin, M.H. Rasmussen, Knowledge graphs and linked data for the built environment, in: M. Bolpagni, R. Gavina, D. Ribeiro (Eds.), Industry 4.0 for the Built Environment: Methodologies, Technologies and Skills, in: Structural Integrity, Springer International Publishing, Cham, 2022, pp. 157–183, http://dx.doi.org/10.1007/978-3-030-82430-3_7.

[80] V. Kukkonen, Method for using information models and queries to connect HVAC analytics and data, J. Comput. Civ. Eng. 37 (6) (2023) 04023034, http://dx.doi.org/10.1061/JCCEE5.CPENG-5341.

[81] S. Hu, E. Corry, E. Curry, W. Turner, J. O Donnell, Building performance optimisation: A hybrid architecture for the integration of contextual information and time-series data, Autom. Constr. 70 (2016) http://dx.doi.org/10.1016/j.autcon.2016.05.018.

[82] M.K. Sein, O. Henfridsson, S. Purao, M. Rossi, R. Lindgren, Action design research, Manag. Inform. Syst. Res. Center 35 (1) (2011) 37–56, http://dx.doi.org/10.2307/23043488.

[83] K. Nweye, B. Liu, P. Stone, Z. Nagy, Real-world challenges for multi-agent reinforcement learning in grid-interactive buildings, Energy AI 10 (2022) 100202, http://dx.doi.org/10.1016/j.egyai.2022.100202.

[84] D. Mavrokapnidis, G. Fierro, I. Korolija, D. Rovas, A programming model for portable fault detection and diagnosis, in: Proceedings of the 14th ACM International Conference on Future Energy Systems, in: e-Energy '23, Association for Computing Machinery, New York, NY, USA, 2023, pp. 127–131, http://dx.doi.org/10.1145/3575813.3595190.

[85] L. Lazzari, K. Farias, Event-driven architecture and REST: An exploratory study on modularity, J. Appl. Res. Technol. (2023) 338–351, URL https://jart.icat.unam.mx/index.php/jart/article/view/1764/995.

[86] M. Wetter, P. Ehrlich, A. Gautier, M. Grahovac, P. Haves, J. Hu, A. Prakash, D. Robin, K. Zhang, OpenBuildingControl: Digitizing the control delivery from building energy modeling to specification, implementation and formal verification, Energy 238 (2022) http://dx.doi.org/10.1016/j.energy.2021.121501.

[87] G. Xiao, D. Calvanese, R. Kontchakov, D. Lembo, A. Poggi, R. Rosati, M. Zakharyaschev, Ontology-based data access: A survey, in: Proceedings of the 27th International Joint Conference on Artificial Intelligence, IJCAI, IJCAI Organization, 2018, pp. 5511–5519, http://dx.doi.org/10.24963/ijcai.2018/777.

[88] G. Fierro, A.K. Prakash, C. Mosiman, M. Pritoni, P. Raftery, M. Wetter, D.E. Culler, Shepherding metadata through the building lifecycle, in: BuildSys '20 Conference, 2020, p. 10, http://dx.doi.org/10.1145/3408308.3427627.

[89] Z. Pan, Y. Gao, F. Ponci, A. Monti, Pan ontology development framework for building energy data management, IEEE Access 11 (2023) 111991–112003, http://dx.doi.org/10.1109/ACCESS.2023.3323335.

[90] E.G. Caldarola, A.M. Rinaldi, An approach to ontology integration for ontology reuse, in: 2016 IEEE 17th International Conference on Information Reuse and Integration, IRI, 2016, pp. 384–393, http://dx.doi.org/10.1109/IRI.2016.58.

[91] R. García Castro, M. Lefrançois, M. Poveda-Villalón, L. Daniele, The ETSI SAREF ontology for smart applications: A long path of development and evolution, ISBN: 9781119899426, 2023, pp. 183–215, http://dx.doi.org/10.1002/9781119899457.ch7.

[92] C. Quinn, J. McArthur, A case study comparing the completeness and expressiveness of two industry recognized ontologies, Adv. Eng. Inform. 47 (2021) 101233, http://dx.doi.org/10.1016/j.aei.2020.101233.

[93] F. Scharffe, O. Zamazal, D. Fensel, Ontology alignment design patterns, Knowl. Inf. Syst. 40 (2014) 1–28, http://dx.doi.org/10.1007/s10115-013-0633-y.

[94] G. Giannakis, K. Katsigarakis, G. Lilis, S. Álvarez-Díaz, Guidelines for OptEEmAL BIM input files, 2018, OptEEmal Horizon 2020 project, URL https://www.opteemal-project.eu/files/guidelines_for_opteemal_bim_input_files_v11.pdf.

[95] K. Katsigarakis, G.N. Lilis, D. Rovas, S. González-Gerpe, S. Bernardos, A. Cimmino, M. Poveda-Villalón, R. García-Castro, A digital twin platform generating knowledge graphs for construction projects, in: European Semantic Web Conference, 2022, p. 12, URL https://ceur-ws.org/Vol-3291/paper3.pdf.

[96] Electricity Maps, Greece 2024 carbon intensity data, 2024, Electricity Maps Data Portal, URL https://www.electricitymaps.com/data-portal. (Accessed: 20 February 2024).

[97] H. Li, H. Johra, F. de Andrade Pereira, T. Hong, J. Le Dréau, A. Maturo, M. Wei, Y. Liu, A. Saberi-Derakhtenjani, Z. Nagy, A. Marszal-Pomianowska, D. Finn, S. Miyata, K. Kaspar, K. Nweye, Z. O'Neill, F. Pallonetto, B. Dong, Data-driven key performance indicators and datasets for building energy flexibility: A review and perspectives, Appl. Energy 343 (2023) 121217, http://dx.doi.org/10.1016/j.apenergy.2023.121217.

[98] F.C. McQuiston, J.D. Parker, J.D. Spitler, H. Taherian, Heating, Ventilating, and Air Conditioning: Analysis and Design, John Wiley & Sons, 2023.

[99] S. Hu, J. Wang, C. Hoare, Y. Li, P. Pauwels, J. O'Donnell, Building energy performance assessment using linked data and cross-domain semantic reasoning, Autom. Constr. 124 (2021) 103580, http://dx.doi.org/10.1016/j.autcon.2021.103580.

[100] B. Zeng, G. Wu, J. Wang, J. Zhang, M. Zeng, Impact of behavior-driven demand response on supply adequacy in smart distribution systems, Appl. Energy 202 (2017) 125–137, http://dx.doi.org/10.1016/j.apenergy.2017.05.098.

[101] P. Pauwels, G. Fierro, A reference architecture for data-driven smart buildings using brick and LBD ontologies, in: CLIMA 2022 conference, 2022, http://dx.doi.org/10.34641/clima.2022.425, URL https://proceedings.open.tudelft.nl/clima2022/article/view/425.

[102] E. Lambert, G. Boultadakis, K. Kukk, K. Kotsalos, N. Bilidis, European Energy Data Exchange Reference Architecture, European Comission, 2021.

[103] D.D. Eneyew, M.A.M. Capretz, G.T. Bitsuamlak, Toward smart-building digital twins: BIM and IoT data integration, IEEE Access 10 (2022) 130487–130506, http://dx.doi.org/10.1109/ACCESS.2022.3229370.

[104] L. Chamari, E. Petrova, P. Pauwels, An end-to-end implementation of a service-oriented architecture for data-driven smart buildings, IEEE Access 11 (2023) 117261–117281, http://dx.doi.org/10.1109/ACCESS.2023.3325767.

[105] D. Mavrokapnidis, G. Lilis, K. Katsigarakis, I. Korolija, D. Rovas, Semi-automated extraction of HVAC system topology from imperfect building information models, in: Proceedings of Building Simulation 2023: 18th Conference of IBPSA, 2023, http://dx.doi.org/10.26868/25222708.2023.1392.

[106] D. Blum, Z. Wang, C. Weyandt, D. Kim, M. Wetter, T. Hong, M.A. Piette, Field demonstration and implementation analysis of model predictive control in an office HVAC system, Appl. Energy 318 (2022) 119104, http://dx.doi.org/10.1016/j.apenergy.2022.119104, URL https://www.sciencedirect.com/science/article/pii/S0306261922004895.

[107] B. Urban, D. Taylor, Finding fault: How AFDD can deliver efficiency at scale for the military and beyond, in: 2024 ACEEE Summer Study on Energy Efficiency in Buildings, 2024.

[108] M. Pritoni, M. Wetter, L. Paul, A. Prakash, W. Huang, S. Bushby, P. Delgoshaei, M. Poplawski, A. Saha, G. Fierro, M. Steen, J. Bender, P. Ehrlich, Digital and interoperable: The future of building automation is on the horizon. What's in it for me? in: 2024 ACEEE Summer Study on Energy Efficiency in Buildings, 2024, URL https://escholarship.org/uc/item/191333wd.