buildings

MDPI

Article

Semantic Web Technologies for Indoor Environmental Quality: A Review and Ontology Design

Alex Donkers *, Dujuan Yang, Bauke de Vries and Nico Baken

Information Systems in the Built Environment, Eindhoven University of Technology, Groene Loper 6, 5600MB Eindhoven, The Netherlands
* Correspondence: a.j.a.donkers@tue.nl

Abstract: Indoor environmental quality (IEQ) affects occupants' satisfaction, health, productivity, comfort, and well-being. IoT developments enable better monitoring of IEQ parameters; however, integrating the various types of heterogeneous data from both the IoT and BIM domains is cumbersome and capital intensive, and therefore, limits the potential of smart buildings. Semantic web technologies can reduce heterogeneity issues, which is necessary to facilitate complex IEQ models. An ontology integrating data related to a building's topology and its static and dynamic properties is still lacking. The outline of this research is twofold. First, a systematic literature review was conducted to find state-of-the-art semantic web technologies related to building topology, static properties, and dynamic properties from the IoT and BIM domains. By graphically reviewing various ontologies, their valuable patterns, commonalities, and best practices were revealed. Secondly, those results were used to develop a new ontology that integrates topological building information with static and dynamic properties. This Building Performance Ontology (BOP) provides a generic upper-level description of properties and two lower-level ontologies representing observations and actuation. The ontology results in intuitive queries and is both horizontally and vertically extensible. Multiple levels of detail are introduced to ensure practical applicability and efficient patterns based on the data modeler's needs. BOP opens up a new range of research opportunities in the IEQ domain.

Keywords: semantic web; linked data; ontology; indoor environmental quality; Building Performance Ontology

check for updates

Citation: Donkers, A.; Yang, D.; de Vries, B.; Baken, N. Semantic Web Technologies for Indoor Environmental Quality: A Review and Ontology Design. *Buildings* **2022**, *12*, 1522. https://doi.org/10.3390/buildings12101522

Academic Editors: Andrej Tibaut and Salvatore Antonio Biancardo

Received: 13 July 2022
Accepted: 21 September 2022
Published: 23 September 2022

1. Introduction

The indoor environment hugely affects our daily quality of life. Air quality and thermal, visual, and acoustic comfort largely influence occupants' satisfaction [1,2]. Poor performing buildings could lead to symptoms of sick building syndrome [3], such as asthma [4], the short absence of knowledge workers [3], and lower health conditions in social housing [5]. Next to this, the indoor environment affects occupants' productivity [6] and students' learning performance [7]. Improving the indoor environmental quality (IEQ) improves the comfort and well-being of occupants [8] and should, therefore, be a priority in real estate. Various technology-driven developments such as the digitization of the built environment and thriving IoT developments could enable building performance improvements. Integrating the Internet of Things (IoT) and building information models (BIM) [9,10] could lead to the better monitoring and management of buildings, leading to improved indoor environments [11,12].

This does not come without challenges. Monitoring IEQ is often cumbersome and capital intensive [13], whereas building managers lack the time, skills, and manpower to monitor, control, and optimize their buildings [14]. Current IEQ models do not necessarily reflect the perceived environment [15] and high scores on building codes and certificates do not undoubtedly lead to higher user satisfaction [16,17].

Modeling perceived environments requires an interdisciplinary approach combining multiple research disciplines, methodologies, and input variables [18], considering not only physical but also personal and psychological input [19]. This results in a range of highly heterogeneous data and methods, which are frequently divided into four indicators, namely, thermal comfort, visual comfort, acoustic comfort, and air quality [20–22]. Al Horr et al. [8] added the sick building syndrome (SBS) as a fifth indicator, whereas others consider SBS to be a result of poor IEQ [20].

Thermal comfort refers to the satisfaction with the thermal parameters. The predicted mean vote (PMV) and predicted percentage dissatisfied (PPD) methods are widely used to model thermal comfort [8,21] and are used in major standards. The PMV model combines environmental parameters, including air temperature, mean radiant temperature, relative humidity, and air velocity, with personal parameters, namely, metabolic rate and clothing insulation [8,23]. Factors of local discomfort influence the PMV results. Various standards implement different levels of detail to model this influence, such as EN-16789 and ASHRAE-55. EN-16798 includes air pressure, radiant temperature asymmetry, draft, vertical air temperature difference, and floor difference temperature, whereas ASHRAE-55 also includes solar gain, ankle draft, and the CBE vertical temperature gradient [23]. Tartarini et al. developed both a browser-based tool [23] and a Python library [24] to calculate whether a situation complies with both standards. The PPD method calculates the percentage of people likely to be dissatisfied with the thermal comfort based on PMV results. The adaptive method [25] is an alternative to the PMV/PPD methods and is characterized by the ability of occupants to adapt to changing thermal environments by changing activities, clothing, expectations, and acclimatization. People's adaptive abilities widen the range of conditions that are deemed comfortable [21]. Both ASHRAE-55 and EN-16798 have a calculation model for the adaptive approach [23]. Frontczak and Wargocki [21] stated that preferences for IEQ are influenced by country, outdoor climate, building type, and ventilation system. Those variations are not yet fully integrated into the IEQ models.

Visual comfort refers to the satisfaction with visual parameters. It covers multiple physical parameters, such as illuminance, luminance distribution, glare, lighting color, color rendering, and the amount of daylight and artificial light [8,21,26]. Next to these, visual comfort is influenced by design parameters, such as views and the daylight factor [27]. Standards, therefore, combine both design indicators and performance tests to assess the visual comfort of a building [27,28]. Kim and Dear [2] argue that visual privacy is a major factor in workplace satisfaction. Although a wide range of measurement tools to measure the physical parameters exist [20], a combination with static building information (such as spatial design, geometry, finishing materials, textures, color) as well as the psychological dimension is necessary to obtain a full understanding of visual comfort [8,21].

In the operational phase, providing good acoustic comfort is achieved by protecting occupants from acoustic discomfort [8,21]. Acoustic discomfort could occur due to poor physical noise parameters, as well as due to a lack of sound privacy. Loomans et al. [27] define two physical performance indicators, the A-weighted background noise level and the reverberation time. Frontczak and Wargocki [21] differentiate short-term and long-term sound pressure levels and argue that sound frequency is an important factor. WELL v2 [28] adds multiple material characteristics to their evaluation, such as the sound insulation of building elements. Although Al Horr et al. state that acoustics are not prioritized in building design, multiple guidelines include norms for acoustic comfort [8,28]. Sound privacy is critical for workspace satisfaction [2] and refers to the ability to have private conversations. Spatial design, partitioning elements, material characteristics, and separation of acoustic zones could improve sound privacy [2,8,28,29]. Whether an adequate acoustic comfort level is met is therefore dependent on both the static building information as well as dynamic physical properties [21]. Evaluation methods include expert reviews, surveys, and quantitative measurements [27].

Air quality has many dimensions and is important because it involves potential discomfort and health risks [8,21]. It involves odors and air pollution due to internal or external sources which could be solved by proper mechanical or natural ventilation. Ventilation strategies differ per building type and location. Most of the pollutants such as CO, CO_2, TVOCs, and particulate matter are measurable by sensors. Next to outdoor polluting sources (such as nearby highways or smog), buildings have indoor polluting sources. Building elements, furniture, and cooking applications, as well as the occupants themselves, influence the air quality [8]. Loomans et al. [27] assess the air quality based on the capacity to keep pollutants (CO, CO_2, $PM_{2.5}$, and PM_{10}) within an acceptable range. They also address the risk of mold growth due to a combination of long-term temperature and relative humidity levels enabling mold to grow. Frontczak and Wargocki [21] argue that the assessment of air quality should be based on occupant dissatisfaction rates. WELL v2 [28] adds detailed requirements for TVOC concentrations and various managerial requirements, such as prohibiting smoking and allowing individuals to open windows.

These state-of-the-art IEQ models require the integration of topological building information (describing relationships between a building and its subelements) with static properties (such as material characteristics and geometry) and dynamic properties (such as temperature, CO_2, and illuminance). An integrated approach to semantically describe the heterogeneous IEQ-related information from both the IoT and BIM domains is necessary to facilitate the complex algorithms to model the IEQ. Semantic web technologies have proven to be able to integrate various heterogeneous data sources using ontologies [30]. For example, O'Donnell et al. [14,31] integrated sensor data and building data for energy analysis, whereas Corry et al. [32] integrated building data, simulation data, and sensor data to assess thermal comfort. Hu et al. [33] also integrated weather and calendar data to analyze building energy performance using their previously developed performance metrics ontology [34]. Reinisch et al. [35] integrated heterogeneous data for energy-related simulations in smart homes. Despite these examples, there is no comprehensive, widely accepted approach to integrate IoT and BIM modeling for IEQ management.

Therefore, this paper focuses on creating an ontology that integrates topological building information with static and dynamic properties. The outline of this research is twofold. First, a systematic literature review will show state-of-the-art semantic web technologies related to building topology, static properties, and dynamic properties. Best practices form the foundation for the second part of this paper, which presents the Building Performance Ontology as an ontology to integrate both static and dynamic properties with topological building information.

2. Materials and Methods

This section describes the methods used in this paper. First, a systematic literature review was conducted. The review presents an in-depth review of semantic web technologies related to a building's topology, static properties, and dynamic properties. Best practices and common nomenclature and patterns formed a knowledge base for developing the Building Performance Ontology. The development of this ontology was based on various ontology development methodologies and is described in the second part of this section.

2.1. Systematic Literature Review

A systematic literature review gives a coherent view of the state-of-the-art semantic web technology in the building performance domain. The paper selection procedure was designed to reduce bias in the selection process. It uses an advanced Web of Science (WoS) query to find ontologies from 15 SCI-indexed journals, which were selected based on other literature reviews related to linked data in the AEC domain [12,30]. The query used to find relevant ontologies combined a set of keywords related to IEQ with "semantic web" or "ontology" to filter semantic web-related research (Listing 1). Next to these parameters, filters were applied to filter SCI-indexed papers written in English. Since semantic web technologies for the AEC domain started to seriously develop after 2009 [36], articles before 2009 were filtered out.

Listing 1. Web of Science query.

```
TS=((“semantic web” OR “ontology”) AND (“building performance” OR
“building regulations” OR “building guidelines” OR “indoor environmental
quality” OR “thermal comfort” OR “ visual comfort” OR “sick-building
syndrome” OR “acoustic comfort” OR “air quality” OR “building topology”
OR “linked building data” OR “BIM” OR “material characteristics” OR
“building geometry” OR “HVAC” OR “indoor IoT” OR “facility management”))
AND PY=(2009–2021) AND SO=(AUTOMATION IN CONSTRUCTION OR ADVANCED
ENGINEERING INFORMATICS OR JOURNAL OF COMPUTING IN CIVIL ENGINEERING OR
EXPERT SYSTEMS WITH APPLICATIONS OR COMPUTER AIDED DESIGN OR JOURNAL OF
BUILDING ENGINEERING OR SEMANTIC WEB OR SENSORS OR INTERNET OF THINGS OR
ADVANCES IN ENGINEERING SOFTWARE OR JOURNAL OF WEB SEMANTICS OR APPLIED
ONTOLOGY OR LECTURE NOTES IN COMPUTER SCIENCE OR BUILDING RESEARCH AND
INFORMATION OR ENERGY AND BUILDINGS)
```

The query resulted in 86 papers. Papers were filtered out if they did not present an ontology related to topological information, static properties, or dynamic properties, resulting in 37 ontologies. A literature grid was built using an abstract search and the presented ontologies were categorized per data category.

Next to this query, this paper consulted multiple literature reviews and conference proceedings, adding another 33 ontologies that fit the data categories but were not yet present in the WoS query results. We do acknowledge the work of others outside of academia, as many ontologies are not necessarily published in academic papers. However, an academic foundation of the ontologies is thought to be crucial for the development of a new ontology for building performance.

The resulting ontologies were reviewed and compared to find commonalities, best practices, and valuable patterns. The focus of this review is on the terminology of various classes and their descriptions, the object properties, restrictions (adding domain knowledge to the graph), and the resulting ontology patterns. The pattern discovery was strengthened by grouping common concepts (or equivalents) by color. Graphically reviewing the structure of the ontologies revealed common patterns and best practices.

2.2. Building Performance Ontology Development

Table 1 compares various ontology development methodologies. Although some approaches focus on the overall process of ontology development and others focus on the technical design of the ontology itself, many similarities can be found in the approaches. Based on this knowledge, the following steps were taken to develop the Building Performance Ontology:

1. Specification: Determining the scope and purpose of the ontology, including its position in the semantic web landscape. Competency questions are formulated in this phase.
2. Knowledge acquisition: Defining the exact domain knowledge that the ontology should cover, based on a literature review.
3. Requirement specification: Making a list of classes, instances, object properties, and restrictions. Choices are based on the competency questions and the knowledge acquisition phase.
4. Building: Building the ontology using Stanford Protégé 5.5.0 [37].
5. Evaluation: Testing the ontology in different use-cases. Various SPARQL queries were designed to check the competency questions. Next to this, a set of evaluation criteria was created to test the ontology using a case study. Finally, the OntOlogy Pitfall Scanner! (OOPS!) [38] was used to detect common pitfalls in the ontology.
6. Integration: Integrating the ontology with existing standards and ontologies.
7. Documentation and publication: Creating HTML documentation using WIDOCO [39]. Multiple practical examples were elaborated on. An example dataset, instantiating the ontology, was openly published [40] under the CC-BY 4.0 license.

Table 1. Step-by-step ontology development approaches.

Step	Fernández et al.	Uschold and Gruninger	Gomez-Perez et al.	Harald Sack	Donkers et al.
1	Specification	Identify purpose and scope	Knowledge acquisition	Determine scope	Specification
2	Knowledge Acquisition	Building: ontology capture	Requirement specification	Consider reuse	Knowledge acquisition
3	Conceptualization	Building: ontology coding	Conceptualization	Enumerate terms	Requirement specification
4	Integration	Building: integration	Implementation	Define classes	Building
5	Implementation	Evaluation	Evaluation	Define properties	Evaluation
6	Evaluation	Creating guidelines	Documentation	Define property constraints	Integration
7	Documentation			Define instances	Documentation

The systematic literature review was used to gather knowledge for both the knowledge acquisition and requirement specification phases. Based on this knowledge, we collected relevant classes, instances, object properties, datatype properties, and the restrictions and characteristics of those properties. Daniele et al. [41] used a similar approach when developing a common ontology language for smart applications. Multiple researchers evaluate their ontology mainly based on competency questions (e.g., [32,35,42]). To ensure a fair evaluation, the Building Performance Ontology was evaluated against a predefined set of criteria, namely:

1. Query efficiency: evaluating query execution time and the simplicity and intuitiveness of creating queries.
2. Practical applicability: evaluating if the ontology fits practical use-cases and standards, whether it makes use of current data structures, and whether data mapping could be carried out efficiently.
3. Pattern efficiency: evaluating if the ontology remains efficient, concise, and flexible, while not cutting back on semantic expressivity.
4. Extensibility: evaluating if the ontology can be extended for future cases both in horizontal and vertical directions.

3. Semantic Web Technologies for Indoor Environmental Quality

Table 2 gathers a list of ontologies based on the WoS search. It divides all ontologies among the three data categories and some subcategories, based on commonalities. The following subsections review and compare the ontologies.

Table 2. Key data categories and the corresponding ontologies.

Category	Subcategories	Ontologies from WoS	Other Ontologies
Topology	Building topology	BOT [43], TDO [44], SBIM [45], BIMSO [46], muso [47], ifcOWL [36,48], IFC ontology [49], building object ontology [50]	ILONA [51], OntoNav [52], SimpleBIM [53], OntoFM [54–56], ThinkHome building ontology [35], ogbxml [57]
	Building taxonomy	BFHO [58], building object ontology [50]	
	Element topology	BPO [59], nameless ontology [60], construction-oriented product ontology [61], nameless ontology [62], IFC IR ontology [63]	

Table 2. *Cont.*

Category	Subcategories	Ontologies from WoS	Other Ontologies
Topology	Element taxonomy	NRM 1 ontology [64], building ontology [65], IDM ontology [66], CProduct [67], STG Product [68], nameless ontology [60]	PRODUCT (https://github.com/w3c-lbd-cg/product, accessed on 22-09-2022), BEO (http://pi.pauwel.be/voc/buildingelement, accessed on 22-09-2022), MEP (https://pi.pauwel.be/voc/distributionelement, accessed on 22-09-2022), freeClass OWL [69]
	Geospatial topology	Preconstruction operation ontology [70]	
Static properties	Geometry	V4D [71]	OMG [72], FOG [73]
	Material characteristics	GBMTO [74], Defect Ontology [75]	
	Environmental data	INIESOnto [67], QuartzOnto [67], SDO [76]	
	General properties	OPM [77], FP [78], multiple ontologies [79], BIMDO [46], mudo [47], ifcOWL [36,80], BAUKOM ontology [81], IFC IR ontology [63], nameless ontology [82]	PROPS (https://github.com/maximelefrancois86/props, accessed on 22-09-2022), ifcWoD [83]
Dynamic properties	Sensor data	UDSA [84], TDO [44], HERO [85], EEPSA [86], SBMS [45], WISDOM [87]	SAREF4Health [88], SOSA [89], SSO ODP [90], AAE ODP [91], SAN [91], AffectedBy ODP [92], EEP [92], BOnSAI, [42], PowerOnt [93], SEAS [94], SmartEnv [95]
	Database integration	DB-RDF [96]	
	System ontologies with patterns for dynamic properties		Brick [97], SAREF [41], SSN [98], IoT-O [91], FIESTA-IoT [99], DogOnt [100], OntoFM [54–56]

3.1. Topology

Early initiatives by Beetz et al. [36] led to the development of an ontology for linked building data. Their ontology, ifcOWL could be considered an OWL translation of the IFC schema. The ontology (which was enhanced in [80]) could be considered a one-file approach, integrating both topological, geometrical, and non-geometrical information.

Beetz et al. [101] argued that geometric descriptions add computational complexity and proposed to separate geometry models from other metadata. They argued that topological representations are necessary for O&M use-cases [102]. Ye et al. [103] distinguished the geometric representation, the symbolic representation (describing human-readable topology information), and a hybrid representation by combining the two. Topological information describes elements and spaces in buildings, including their relationships and orientation.

Following the critics on extensive construction ontologies, researchers proposed more concise construction-related ontologies. Pauwels and Roxin [53] and Rasmussen et al. [104] developed simplified AEC ontologies, describing the core concepts of a building. This resulted in the development of the Building Topology Ontology (BOT) [43]. Bonduel et al. [105] describe how a topological ontology could be used as a core model of linked building data. In the next subsections, various topologies and taxonomies (hierarchical classifications of topological building elements) on both the building and element scale are compared.

3.1.1. Building Topology

BOT [43] functions as a core topology and describes buildings as a set of zones and elements. Elements are physical objects of which the building exists, such as walls and floors. Zones are imaginary parts of the real world, often encapsulated by elements. BOT describes different classes for sites, buildings, spaces, and stories, superclassed by the bot:Zone-class. The BIM Shared Ontology (BIMSO) [46] also describes classes for buildings, spaces, and levels; however, those classes do not share a common superclass. Dibley et al. [54–56] developed a building ontology as part of their OntoFM project. OntoFM focuses on describing contextual information of indoor sensors, because it mainly focuses on describing zones and their characteristics. No general element class was invented, although various building elements (walls, doors, openings) appear in the ontology as physical boundaries of a zone. The scope of the zone class is limited to physically bounded areas. The building industry typically uses these zones as the feature of interest of building performance assessments. The definition of a zone should, therefore, be broader than merely rooms and floors. Mitterhofer et al. [106] used such a broad definition to model both rooms and thermal zones.

Next to zones, BOT describes elements using bot:Element. The SBIM ontology by Kučera and Pitner [45] is designed for building automation use-cases and extends this class with sbim:Device to describe devices in buildings. A device-type hierarchy is rooted from the device class based on the IFC4 specification. Lee and Jeong [50] distinguished a construction unit and a functional unit. Their ontology differentiates between structural building elements (walls, slabs, columns) and functional elements (furniture, HVAC).

The reviewed ontologies differ in the way they describe relationships between zones and elements, resulting in differences in semantic expressiveness. The Tunnel Diagnosis Ontology by Hu et al. [44] uses tdo:NearBy, tdo:RelAggregate, and tdo:ComposeOf to describe the relationships between elements. The object properties are based on the ifcOWL version of Pauwels and Terkaj [48] and show similarities with spatial relationships in BOT (such as bot:adjacentElement, bot:containsElement, and bot:intersectsElement). Lee and Jeong [50] only defined two object properties (“kind of” and “part of”), limiting the semantic expressiveness of their ontology.

BOT acknowledged a widespread IFC problem, being the fact that walls are often interfacing multiple indoor zones, whereas various use-cases require (geometrical) information of the interface rather than of the entire wall. BOT [43] introduces the interface class to describe the surface between building elements and/or zones. Defining the interface and its geometry requires defining the space boundary between the zone and the element. This geometric information is required for various building performance assessments, including thermal comfort [57] and building energy modeling [107]. Recent research initiatives focus on the automatic specification of second-level space boundaries [108] and their geometric and semantic information [107].

The gbXML schema implements the representation of space boundaries. It focuses on information related to energy simulations and calculations. The ThinkHome [35] project inherits its building information from the gbXML schema. XSLT documents were used to translate the gbXML schema to the OWL format. Schneider aligned their ontology to the IFC-based BOT ontology [109]. A similar gbXML-based topology was introduced by Van Gool et al. [57].

3.1.2. Building Taxonomy

Those aforementioned ontologies describe building information on a conceptual level. Multiple other ontologies follow a taxonomical structure and could be used to define more specific information on a building level. Taxonomies are typically hierarchical constructions consisting of classes and subclass relationships. Mohamed et al. [58] created the building facilities’ hierarchical ontology to describe the functions of each room. Lee and Jeong [50] created the even more extended building object ontology, describing, amongst others, different types of rooms, their functions, and how to access them. Both ontologies were mainly designed for facility management use-cases.

3.1.3. Element Topology

Although the building-scale topology ontologies usually stop at the level of an element, some use-cases require a further decomposition of these elements. It is for this purpose that multiple smaller-scale topology ontologies have been developed. One of them is the Building Product Ontology (BPO) [59]. BPO decomposes topological building elements. It describes the relationship between building elements and their parts and has been applied to describe building-integrated solar thermals and photovoltaics [110].

Xu et al. [60] defined a member-of relationship to describe whole/part-relationships of building elements. Similar to other ontologies [59,61], elements are superclassed by a product class. However, Liu et al. [61] introduced more object properties to define the relationships between elements and their parts, such as proOnto:isPartOf, proOnto:hasSubComponent, and proOnto:hasIntersection.

Another whole-part ontology was defined by Venugopal et al. [62] based on the IFC model. They argued that the current IFC structure leaves space for inconsistencies and that the distinctions between some IFC concepts were not well enough defined. Gao et al. [63] created a more extensive ontology based on IFC and OntoSTEP to structure BIM-specific knowledge. Their IFC IR ontology was used in a semantic search engine to enable querying BIM objects.

3.1.4. Element Taxonomy

Similar to the taxonomies developed to extend building topology ontologies, element taxonomies further specify elements. Lee et al. [66] developed the IDM ontology as part of their ontology-based information delivery management tool. Using IFC as a starting point allowed for easier interaction with other software tools. The ontology has four main classes: entities, elements, attributes, and relations. The element class is subclassed by a taxonomy of multiple types of elements. Gao et al. [63] implemented the freeClass OWL [69] ontology as a taxonomy for building materials. Based on this ontology, the BauDataWeb dataset was created using real building products, containing over 88 million triples.

The PRODUCT ontology was derived from the IFC standard and describes simple aggregations of building elements, mechanical elements, and furniture. Multiple ontologies are derived from the PRODUCT ontology. A domain-specific ontology for architectural building elements (BEO) and another for mechanical, electrical, and plumbing elements (MEP) are developments of PRODUCT. Another variation, named STG Product, has been created by Werbrouck et al. [68]. Many of the taxonomical ontologies found in the literature were hardly designed and published as formal ontologies, but mainly designed for a specific use-case. Examples could be found in the works of Xu et al. [60] and Djuedja et al. [67]. Djuedja et al. [67] developed CProduct as a taxonomy of terms and is used as a reference for other environmental ontologies (INIESOnto [67]). The taxonomical structure was based on INIES, a French database of environmental and health declarations. It contains a rich number of terms but is—unfortunately—only available in French. Similarly, the BAUKOM ontology [81] is a direct translation of the BAUKOM catalog, storing BIM-compatible building component information. Zhang and El-Gohary [65] used a taxonomy of building elements to perform automated building code checking. A natural language processing algorithm linked words in building code documents to their corresponding class in the ontology by matching the results of multiple NLP techniques to the names of classes in the ontology. The NRM 1 ontology [64] is an RDF version of the UK's NRM, which is a set of guidelines for storing data about embodied energy. The ontology covers a large decomposition of elements and their parts and has been mapped to Revit's material descriptions. Different from other ontologies, the NRM 1 ontology captures the material of an element directly in its class, for example, by using the "Roof wood" class.

Many of the topological ontologies which were reviewed also describe some use-case-specific subclasses, blurring the dividing line between the domains of topology and taxonomy [60,61]. Although adding taxonomical classes to the ontology might help a researcher who uses the ontology for a specific use-case, those specific taxonomical classes might complicate using the ontology outside the scope of the original use-case.

3.1.5. Topology in Other Ontologies

Zhang and Issa [49] argued that ifcOWL is not flexible enough since it strictly follows the IFC specification. Their IFC ontology follows the same hierarchical structure as IFC (and thus, ifcOWL) but extends in the taxonomical dimension by introducing elements that were not part of IFC.

The ILONA ontology [51] is designed for indoor navigation and uses topological elements to model the structure of a building. It distinguishes zone groups and gateways. Zone groups represent information about physical and non-physical zones, with the building being the greatest zone unit. Gateways (doors, stairways, escalators, elevators, and virtual crossings) represent the connection between two zones. A different approach was introduced by Anagnostopoulos et al. [52], which focuses on describing the paths between topological elements rather than describing the topological elements themselves. It describes passages (similar to gateways in [51]), exits, and corridor segments as subclasses of the PathElement class.

The preconstruction operation ontology [70] describes a broader scale topology and considers the building as an entity within a larger spatial network. It is designed to enable transferring data from BIM to and from GIS environments and decouples BIM-related classes (such as site and building) from GIS-related classes (e.g., topography, terrain, and vegetation).

3.2. Static Properties

Various ontologies enrich information about topological entities with more detailed information about the entity by describing the characteristics of those entities. Different definitions are used in the literature, such as property [46,77], quality [92], and attribute [59,111], but in general, there is agreement on the definition that a property is a measurable characteristic of a feature of interest (FOI). In the AEC domain, the FOI is often a topological entity, such as an element or a space. Some ontologies focus on the more general definition of properties [77,92], whereas others focus on a specific use-case, such as geometry [71,72], material characteristics [74,112], or environmental information [67]. The following subsections compare the different approaches by visually reviewing design patterns. Classes will appear as colored blocks in the visualizations of the ontologies, corresponding to commonly found concepts. Equivalents of a feature of interest will appear in orange, properties in apple green, executors in yellow, executions in pink, results in blue, procedures in navy, units of measure in purple, and databases in dark green.

3.2.1. Generic Properties

The OWL translation of IFC (ifcOWL) [36,80] contains a structure to describe properties of IfcObjects (Figure 1). Following the IFC schema, IfcProperties are part of an IfcPropertySet. The schema predefined various standard sets, e.g., to describe typical properties of a door. These sets and their related properties are described using IfcPropertySetDefinition and IfcPropertyDefinition, respectively.

Mendes de Farias et al. [83] argue that many properties in ifcOWL are not defined in the ontology's Tbox, but as string values of the property ifc:Name, increasing the complexity of retrieving information. Defining those properties in the ontology would solve this problem. Their ifcWoD ontology translated over 400 IFC property sets to OWL object properties.

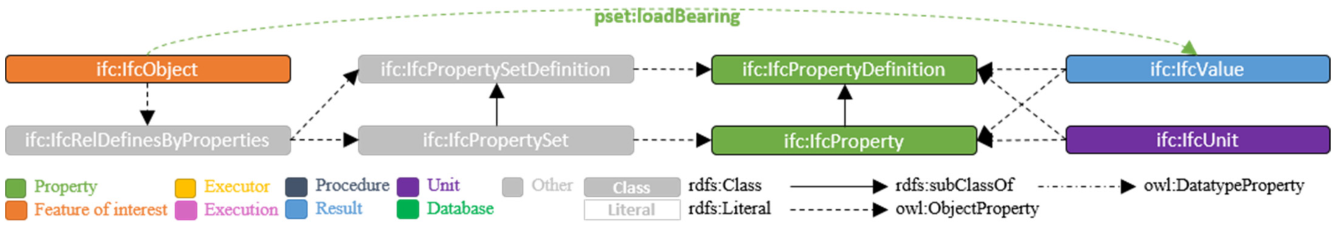


Figure 1. ifcOWL's design pattern linking properties to property sets.

Zhang et al. [79] developed multiple ontologies to enhance the query possibilities of ifcOWL. Vocabularies were developed for schema semantics (schm), instance semantics (pset and qto), product geometry (pdt), and spatial reasoning (spt). Those vocabularies are designed to create simpler SPARQL queries mainly by introducing new object properties and datatype properties between instances, serving as shortcuts. Figure 1 shows how pset:loadBearing creates such a shortcut.

Sadeghineko and Kumar [82] state that ifcOWL cannot describe all kinds of nongeometrical data and argue that additional domain ontologies are necessary to cover a variety of asset/facility management practices. They directly translated CSV files into triples using a custom-built converter. The CSV documents are based on an asset information requirements model. The simple RDF representations are linked to a core linked building data file based on ifcOWL.

Ren et al. [78] developed the FP ontology based on IFC4. It describes properties of building elements to support the information exchange between various stakeholders. Various sets of object properties and related classes were defined to link additional information to building elements, such as fp:hasGlobalId fp:GlobalId and fp:hasLocalPlacement fp:LocalPlacement. The PROPS ontology, which is also defined based on IFC, includes similar properties. However, it does mainly use datatype properties instead of object properties. The IFC IR ontology [63] uses datatype properties to represent EXPRESS simple attributes and object properties for named attributes. A similar design pattern is applied by the BIM Design Ontology (BIMDO) [46]. It extends BIMSO and aims to describe design properties of building elements, which are limited to element identities, sizes, and material properties (Figure 2). It replaces the mudo ontology [47]. Identity-related properties are modeled using datatype properties. The generic BIMDO:hasIdentity has multiple subproperties, such as BIMDO:hasDescription and BIMDO:hasManufacturer. Geometry and material characteristic properties are defined as classes and linked to an element using object properties. Different classes are defined for qualitative (e.g., material strength) and quantitative (e.g., material type) properties. Next to mudo, Niknam and Karshenas [47] also presented a cost estimating ontology (mueo), designed to help make cost estimations for assembling building elements. Properties in mueo also follow the pattern presented in Figure 2.

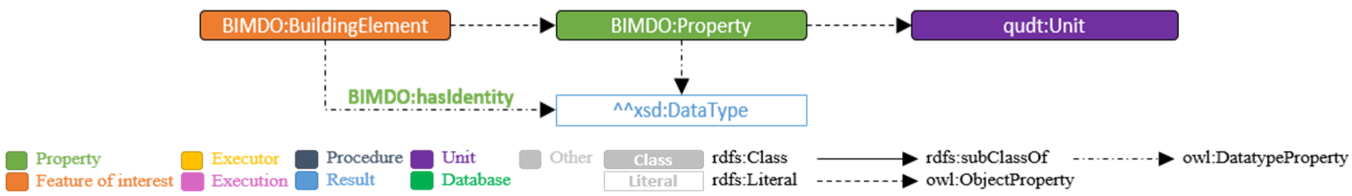


Figure 2. BIMDO describes properties using object properties and datatype properties.

Multiple characteristics of buildings and building elements are evolving, although with such a low frequency that storing those changes in an RDF format is not considered redundant. Typical examples are system defects and damage. Hamdan et al. [112] created the Damage Topology Ontology to describe different types of damage and presented extensions for natural stone (NSD), concrete (Concrete Damage Ontology) and mechanical parameters (Damage Mechanics Ontology). The versioning of damages is an important aspect of damage and is yet to be developed.

The ontology for property management (OPM) [77] presented a method for such versioning (Figure 3). It introduces an opm:PropertyState class (subclassed by opm:CurrentPropertyState) to represent the state of a property at a given time. Opm:hasPropertyState is a subproperty of seas:evaluation.

Figure 3. OPM's pattern for versioning of properties.

3.2.2. Geometry

Wagner et al. [113] performed an extensive literature review on the different descriptions of geometrical information using semantic web technologies. They identified four approaches: RDF-based geometry descriptions, JSON-LD for web geometry, non-RDF geometry as RDF literals, and linking to non-RDF geometry files. Following those approaches, Wagner et al. [72] developed the ontology for geometry management (OMG) to describe the relationship between building objects and their geometry. Similar to PROPS and OPM, it uses multiple levels to link geometry descriptions to objects (Figure 4). Level 1 directly links an object to a geometry description, level 2 uses an intermediate omg:Geometry class, and level 3 introduces the omg:GeometryState. The omg:hasGeometry and omg:hasGeometryState are both inverse functional properties, stating that an omg:Geometry can only be linked to one object and an omg:hasGeometryState can only be linked to one geometry.

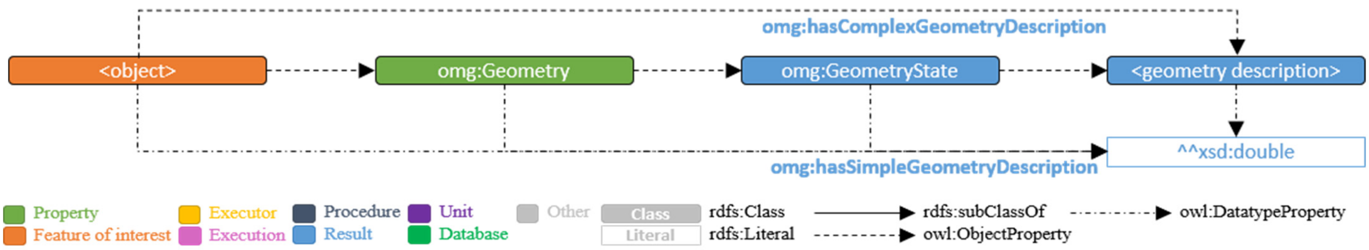


Figure 4. Four levels of detail to describe properties in OMG.

OMG is extended by the ontology for geometry file formats (FOG) [73]. FOG covers the description of geometry formats by introducing specific datatype properties for a wide range of geometry formats. These properties are designed as subproperties of OMG properties and allow users to query specific geometry formats without increasing the graph's verbosity. V4D [71] semantically enriches 2D imagery and 3D geometric models to make them useful for the architecture and game design domains. It describes mesh geometry, which is derived from omg:Geometry and is semantically enriched by reusing concepts from BOT and FOG. The mesh components are described using GOM [114] terminology.

3.2.3. Material Characteristics

GBMTO [74] defines material properties based on green building material information data. The ontology helps to infer a standardized material name based on the properties of a building element. Like BIMDO's property structure, GBMTO uses both object properties and datatype properties to connect a building element with its properties. There is no clear argumentation of why some objects are literals and why others are strings. A similar structure was used by Lee et al. [75] when designing their Defect Ontology.

3.2.4. Environmental Data

Based on the PRODUCT ontology, Djuedja et al. [67] developed IENIESOnto (based on a French national reference database in RDF) and QuartzOnto (based on the Quartz Common Products Database) to store environmental data. Both ontologies mainly use datatype properties to directly link a product to a string. Zhang et al. [76] developed a structural design ontology to facilitate better decision-making during the design phase of a building. They applied a similar strategy as Djuadja et al. Typical static properties from the structural design domain, such as weight, volume, span, and density, are linked to building elements using datatype properties. Materials are linked using object properties and have their own classes. This enables linking environmental specifications to specific materials.

3.3. Dynamic Properties

Different from the static properties, many properties are variables that constantly change over time. Dynamic properties are typically measured by sensors and have a complex spatiotemporal resolution. Hu et al. [44] consider dynamic properties to be the result of events, including a broader range than just monitoring. Inspections, repairs, and maintenance also produce dynamic properties. This subsection presents dynamic property patterns found in the dynamic properties and system ontologies in Table 2.

Arguably one of the most popular ontologies related to sensor measurements is SSN [98]. The ontology, which is recommended by W3C, describes sensors and their observations. It is built upon two conceptual models by the Open Geospatial Consortium, observations and measurements (O&M) [115] and SensorML [116], which created syntactic interoperability [98] for sensors to be machine-understandable, automated, and easily shared [116]. SSN, being an OWL2 ontology, adds the semantic richness that is necessary to use the model in the semantic web. SSN made use of a core ontology, SSO [90], that describes sensors, their stimulus to observe, and the observation. However, from the beginning of the SSN development, discussions took place about fundamental concepts [98], leading to a replacement of SSO by SOSA [89], which describes sensors, observations, samples, and actuators (Figure 5). SOSA is a lightweight ontology that is easily extendable and supports a wide range of use-cases and domains. Mainly, the observations, features of interest, and property modules of SSN/SOSA have been reused by many ontologies (https://w3c.github.io/sdw/ssn-usage/, accessed on 22-09-2022). When combined with SSN, SOSA has some owl:minCardinality restrictions involved. Although SOSA used the less restrictive schema:rangeIncludes and schema:domainIncludes (which only expects a certain class), SSN added owl:allValuesFrom restrictions on most object properties (requiring properties to point to a certain class).

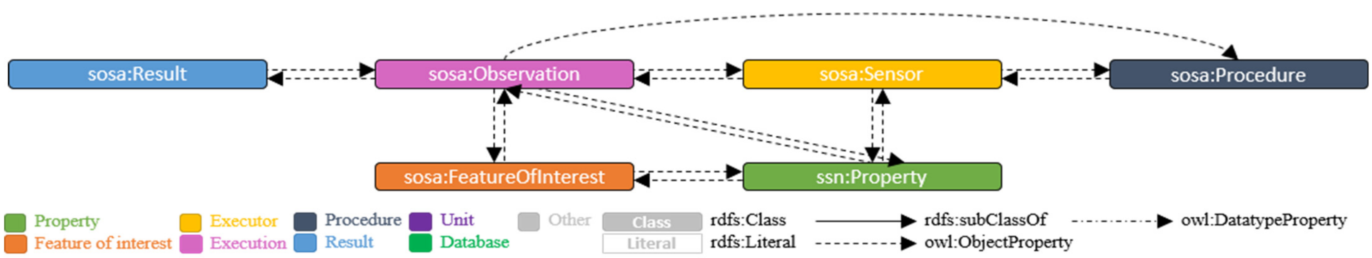


Figure 5. SSN/SOSA's pattern to describe dynamic properties measured by sensor observations.

Kuster et al. [84] extended SSN to the urban scale in their UDSA ontology. They added a spatial component to both the sensor and the observation to describe the location of the sensor and the region for which the observation is valid. Unfortunately, UDSA integrates an outdated version of SSN.

Meng et al. [85] created another SSN extension to describe human health risks. Next to the ssn:Property, their HERO ontology provides a separate hero:Phenomena class. It represents similar information as ssn:Property and is, therefore, redundant.

The SSO ODP inspired Seydoux et al. [91] to create a similar pattern for actuators: the Actuation-Actuator-Effect (AAE) ODP (Figure 6). AAE complements the pattern of SSO; where sensors observe the state of real-world (dynamic) properties through stimuli, actuators influence the state of these properties. AAE led to the development of the SAN (semantic actuator network) ontology [91], which is a complementary counterpart of SSN.

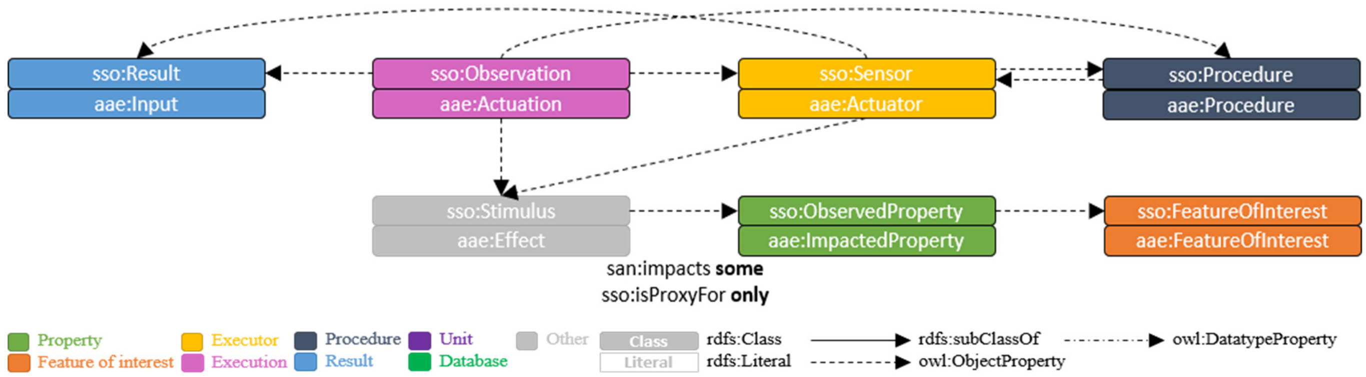


Figure 6. SSO (**top**) and AAE (**bottom**) using similar design patterns.

Esnaola-Gonzalez et al. [92] designed the AffectedBy ODP as a building block to model variables that may affect indoor conditions. They found an inadequacy in SSN. The ssn:isPropertyOf object property is not a functional property, resulting in the fact that a property could be the property of multiple features of interests. This conflicts with the intrinsicness of ssn:Property. Therefore, aff:belongsTo—which is an equivalent of ssn:isPropertyOf—is a functional property.

Using the AffectedBy ODP, Esnaola-Gonzalez et al. [92] created the EEP ontology as a building block to describe the systems that measure the variables affecting indoor conditions (Figure 7). EEP generalizes the observation-sensor-procedure pattern from SOSA by introducing their respective superclasses; eep:Execution, eep:Executor, and eep:Procedure. The eep:Execution class has an owl:someValuesFrom relationship to eep:Executor, eep:Quality, and eep:Procedure. The EEPSA ontology [86] extends the EEP ontology and is a combination of multiple ODPs related to the energy domain. These mainly contain taxonomical structures to enhance query possibilities.

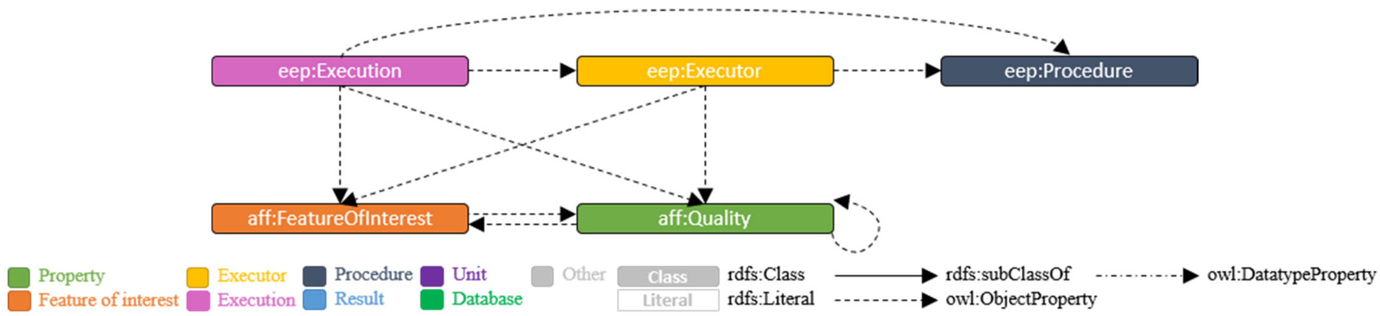


Figure 7. Generalization and improvement of SSN/SOSA, SSO, and AAE using the EEP ontology.

Hu et al. [44] created the TDO ontology for tunnel diagnosis (Figure 8). The concepts show similarities with indoor diagnosis concepts. TDO extends the idea of execution (tdo:Event) by creating multiple subclasses of tdo:Event. These subclasses, including monitoring, inspections, repairs, daily maintenance, and state assessments, can happen on an element and produce a tdo:Property as output. The broader definition of tdo:Event, compared to earlier ontologies, also implies that the property class does not purely represent dynamic properties. Inspections, for example, do not necessarily take place as frequently as sensor measurements.

Figure 8. The TDO pattern extends eep:Execution with tdo:Event.

The ThinkHome [35] ontology introduces various property classes. "Parameter" describes outdoor environmental properties, "BuildingParameter" describes properties of the building, and EquipmentParameter describes properties related to technical equipment, such as HVAC. None of these classes are linked with the BuildingElement class, narrowing the scope for describing building control-related parameters and risking possible irregularities between the various parameter classes. The sensor ontology of OntoFM [54–56] used a different approach. The ontology, which is based on OntoSensor, does not introduce a property class. Instead, a taxonomy of subclasses of the sensor class is introduced, indicating the property which is measured by the sensor (Figure 9). The sensor itself is directly linked to a building entity it measures. For example, a motion sensor could be linked to an opening in a wall. Although this reduces the size of the graph, it introduces unintuitive SPARQL constructs for basic competency questions.

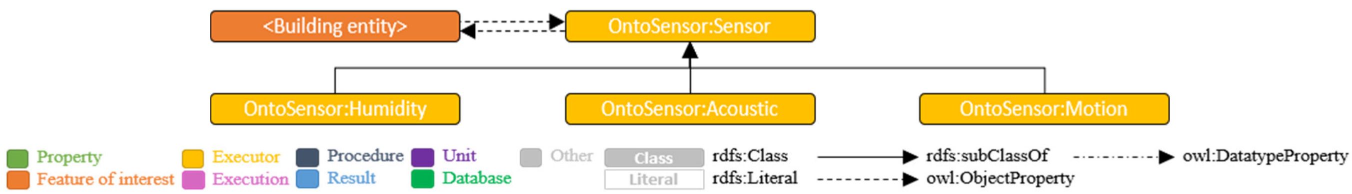


Figure 9. The absence of properties in OntoFM.

SEAS [94] also introduces a dynamic property pattern (Figure 10). It narrowed down DUL's definition of dul:Quality by stating that a property should be an observable or operable quality of an event or object. This specification is common in the context of dynamic properties since the ontologies often describe typical time-series information which is observable by sensors. The term "observable" is, however, rather loosely defined, as one could argue that static properties (such as color and length) are also observable. SEAS distinguishes seas:value (linking static property values) and seas:evaluation (linking dynamic property values) and includes various options to describe a property value as an IRI, blank node, or literal. However, since seas:value and seas:evaluation both link a seas:Property to an IRI, one of the options seems redundant.

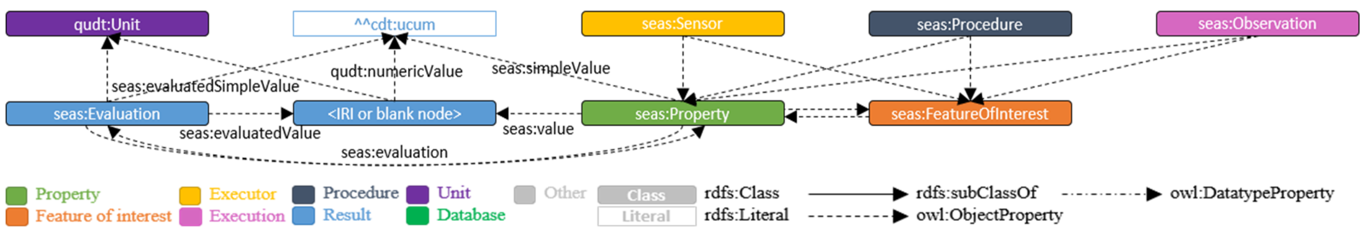


Figure 10. Design pattern of SEAS including multiple methods to describe property values.

A property in SEAS can be a feature of interest itself so that properties of properties can be described. SEAS inherits the Procedure Execution (PEP) [94] ontology as a high-level device ontology. It has a similar class structure as the EEP pattern but uses different object property restrictions (Figure 11). SEAS makes use of a temporal and spatial context, referring to the time (using time:TemporalEntity) and location (using geo:SpatialThing) of an evaluation. This strategy is different from other ontologies, which are more likely to link the spatial context to the feature of interest [41,117].

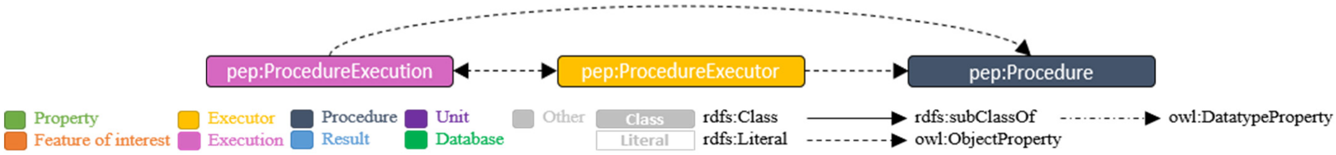


Figure 11. Procedure execution pattern of PEP, generalizing SEAS.

Next to this, SEAS directly links the result class (seas:Evaluation) to the property class. This pattern resembles opm:PropertyState and PowerOnt. PowerOnt [93] is a very lightweight ontology that uses a specific syntax to model the energy consumption of electrical devices (Figure 12). The ontology serves as an extension of multiple other ontologies in the smart building domain, such as DogOnt [100] and IoT-O [91].

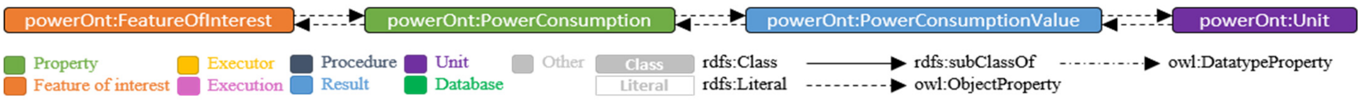


Figure 12. Linking the property (PowerConsumption) and result (PowerConsumptionValue) class using PowerOnt.

Alirezaie et al. [95] combined eight ODPs to develop their SmartEnv Ontology. The ontology's goal is to represent smart, sensorized environments. It contains building blocks for objects, events, situations, sensing, networks, place, geometry, and time. Multiple ontologies, such as SSN, SOSA, and DUL, were integrated. By integrating ODPs, Alirezaie et al. tried to avoid the design issues that appear when integrating complex ontologies. Although succeeding in this goal, the object properties between the ODPs contain too few object restrictions to answer complex competency questions [92]. A SmartObject is introduced, representing instances that are both physical objects and features of interest. Actuation is not included in SmartEnv.

Another attempt to integrate multiple ODPs is IoT-O [91]. It combines a sensing ontology (SSN), an actuation ontology (SAN), a lifecycle ontology (Lifecycle (https://vocab.org/lifecycle/schema, accessed on 22-09-2022)), a service ontology (MSM [118]), and an energy ontology (PowerOnt [93]). It uses SSN to represent properties. Properties could be observed (using ssn:observes) as well as acted on (using san:actsOn).

A risk of combining too many ontologies is that they might not be designed for the same purpose. Definitions might conflict, leading to inconsistencies in the dynamic property patterns. The FIESTA-IoT [99] project aimed to integrate multiple IoT-related ontologies. The ontology integrates a sensor and device ontology (SSN), an ontology to represent resources, entities, and services (IoT-lite [119]), a taxonomy of devices (M3-lite [120,121]), and other ontologies to represent context (DUL, Time, qu, and WGS84). The dynamic property pattern (Figure 13) is, therefore, composed out of three ontologies and shows some redundant classes.

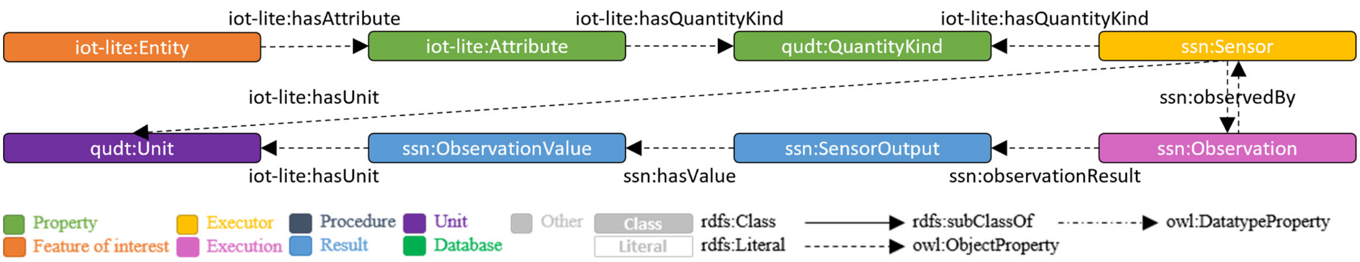


Figure 13. Redundant classes in the Fiesta-IoT pattern.

The Smart Applications Reference ontology (SAREF) [41,122] was created to enable interoperability between smart applications in the built environment, mainly to reduce energy consumption. The ontology was developed in close interaction with the industry using various workshops. Different from other patterns [89,91], SAREF does not distinguish a measurement and a result class (Figure 14). The saref:Measurement class covers both the measurement as well as the measured value. A measurement has exactly one value. The saref:hasTimestamp object property, linking a measurement with a timestamp, does not have this cardinality restriction. Although, based on their descriptions, the feature of interest and property are intrinsic to each other, the object properties connecting them are not functional object properties. The saref:Property class is subclassed by multiple commonly used properties in the HVAC domain, such as saref:Humidity and saref:Temperature. SAREF's pattern for actuation is different from the measurement pattern. The pattern includes multiple classes, including saref:Task, saref:Service, saref:Function, saref:Command, and saref:State. Those classes are not directly linked to the feature of interest and the property.

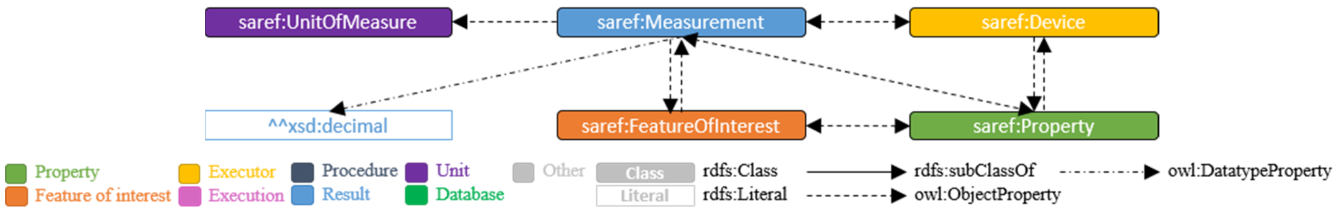


Figure 14. SAREF combines the result and execution class into saref:Measurement.

SAREF was extended by many ontologies. SAREF4BLDG [123] extended SAREF with subsets from the Industry Foundation Classes (IFC) [124] to fill the gap between building information models and the SAREF ontology. It is a rather small ontology, creating a link between the saref:Device and a physical object from the building model. The ontology implements the WGS84 system for describing location, using the GEO ontology. The ontology describes geometric location using altitude, longitude, and latitude.

In SAREF, each measurement is stored as a package of information, containing a unit and a value. Moreira et al. [88] argue that such representations are not suitable for exchanging real-time sensor data, as the payload per message is too high. They concluded that no IoT ontologies could provide an adequate balance between semantic richness and efficient exchange of real-time time-series data. Their solution (the SAREF4health extension) introduces an s4ehaw:TimeSeriesMeasurement class which links to an array of xsd:float values using an s4ehaw:hasValues datatype property (Figure 15). The measurement class in SAREF can only link to a single float number.

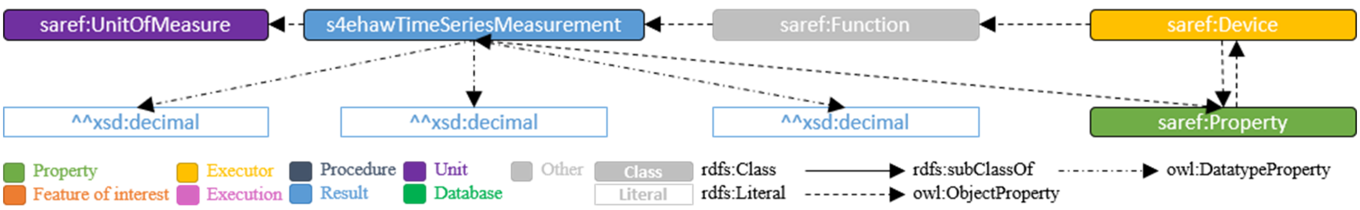


Figure 15. Storing time-series measurements as an array of xsd:float values using SAREF4health.

Stavropoulos et al. [42] designed an early smart building ontology. Their BOnSAI ontology includes concepts related to devices and their functionality, quality of service, users, and their context. Similar to SAREF, BOnSAI also differentiates the sensor structure from the actuator structure (Figure 16). Although sensors are linked to bonsai:Parameter, actuators are linked to bonsai:Action, which is a part of a larger cluster of classes related to actuators. BOnSAI also includes a class for multisensors (such as sensor arrays) and a bonsai:ActuatorSensor class for dual-purpose devices. BOnSAI separated (service-enabled) devices, such as sensors and actuators, from (non-service-enabled) appliances. Some inconsistencies could be found here, as devices that have actuation functionalities (air conditioning, radiator, lighting) are modeled as subclasses of bonsai:Appliance.

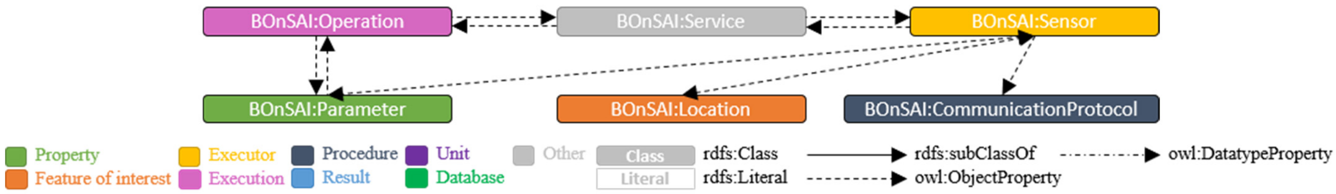


Figure 16. Dynamic properties in BOnSAI.

The Brick schema [97] is an attempt to capture metadata that is used by building management systems and facility managers into a common data representation. It covers classes related to building information, systems (such as HVAC and lighting), sensors and their measurements, and BMS-related constructs. It has been specifically designed to be implemented by the industry. Brick is built upon Project Haystack (https://project-haystack.org/, accessed on 22-09-2022) and SAREF [41]. Since Brick is not developed from an IFC perspective, it has some fundamental differences from the previously discussed ontologies. Therefore, the colors in Figure 17 might show some inadequacies. Brick models three different entities. Physical entities have a physical presence, such as HVAC equipment and spatial elements. Virtual entities have a digital presence and mostly relate to sensing, actuation, and status points. Logical entities are concepts based on rules and are used in the HVAC domain, such as thermal zones and metadata.

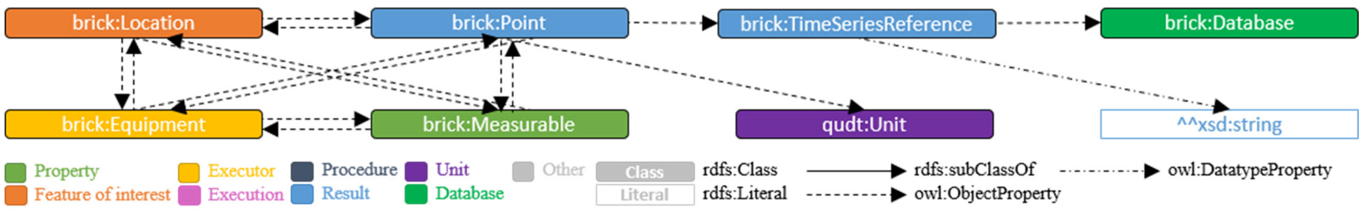


Figure 17. Brick's pattern for describing time-series data that is stored in an external database.

This leads to five top-level classes: brick:Equipment, brick:Location, brick:Measurable, brick:Point, and brick:System. A taxonomical structure, based on Haystack, has been designed using subclasses. Object properties, defining the relationships between the classes, are mainly designed between those top-level classes and are thus also applicable to the subclasses.

Brick:Equipment describes physical devices, whereas brick:Location describes physical locations. In Brick, sensors are not modeled as physical devices, but as datapoints, being a subclass of brick:Point. They, therefore, do not have a physical location, but rather serve as the output point of a physical measurement instrument. These physical measurement instruments are, however, not part of the Brick ontology. A brick:TimeSeriesReference class is introduced to add metadata describing where and how data is stored. The brick:Database class describes the location, whereas a literal describes a reference identifier.

WISDOM [87], which mainly extends an older version of SSN, also integrates a wis:Database class to represent time-series information in smart water use-cases. It is a central concept in the ontology (Figure 18). Object properties link the database with the result (wis:SensorOutput), the feature of interest (wis:Asset), the property, the sensor, and a unit. Different from Brick, wis:Unit is linked to the database class.

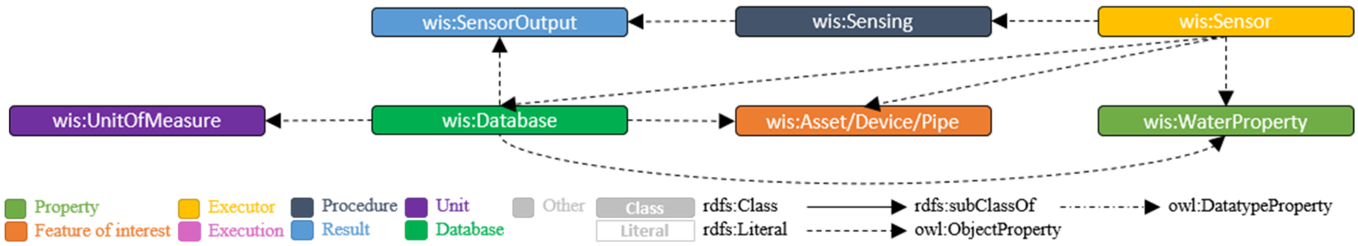


Figure 18. Representation of an external database in the WISDOM ontology.

Hu et al. [96] argued that more information about database schemas is necessary to connect various databases and obtain time-series data from them. They designed the DB-RDF ontology, a structure describing database schemas. It extends a database class by introducing classes to represent tables, columns, parameters, data formats, and translations.

SBMS extends the SSN ontology for the BMS domain (Figure 19). By introducing the sbms:Address class, the ontology aims to describe entities in BMS systems. Similar to Brick, sbms:Datapoint is introduced which represents sensor readings, actuator in- or output, or other scalar values representing the state of a property. The class is not intended to store the sensor readings but is merely an identifier to the original data in a BMS system. Next to datapoints, SBMS requires all topological building elements and devices to be tagged with an ID using sbms:hasBMSId so that they can be linked to their representations in other databases. Similar to the IfcPropertySet, SBMS defines the sbms:PropertyDomain, which enables grouping various properties of a certain type, such as properties related to water or electricity.

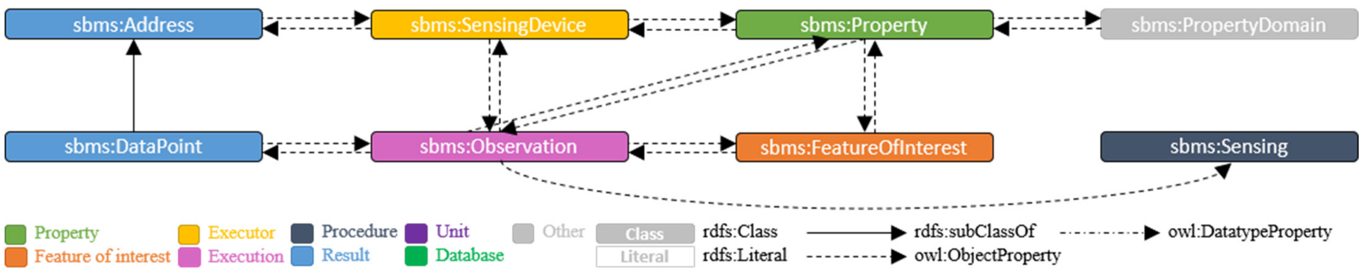


Figure 19. Using datapoints to represent dynamic property values in SBMS.

3.4. Design Patterns and Best Practices

The goal of this literature review is to find commonalities in design patterns and best practices in ontologies related to topological building information and static and dynamic properties. This subsection describes the findings and functions as foundational knowledge for designing an ontology that integrates a building's topology with static and dynamic properties.

We conclude that many ontologies describe topological building information and that there is no scientific consensus on the semantics of a core building topology ontology yet. Schneider [109] showed the complexity of aligning topological and taxonomical ontologies. Based on this literature review, fixating on a single ontology would be irrational. Ontology design patterns for static and dynamic properties should, therefore, be flexible enough to extend multiple topology schemes.

We did find commonalities in the ontologies that describe static and dynamic properties. Based on this review, we found design patterns that ontologies that describe properties would benefit from. An ontology should not limit the data modeler too much. This can be reached by introducing multiple semantic levels of detail to describe a :Property and its value(s), including the possibility to describe the value of a :Property using datatype properties and literals, as well as using object properties. The OPM ontology introduced the versioning of properties by using a :CurrentState-property. An ontology would benefit from a high-level generic :Property-class that could be extended by data modelers for specific use-cases. Creating the ability to group properties into a :PropertySet would enhance querying capabilities. A SPARQL query could, for example, use this :PropertySet-class to find all properties related to thermal comfort.

There is a central pattern consisting of four classes (:Executor, :Execution, :FeatureOfInterest, and :Property) that is commonly used in the reviewed ontologies. Although many ontologies implement use-case-specific classes such as :Sensor or :Actuator, we believe that an ontology that describes a wide range of static and dynamic properties should have an extendable top-level pattern of generic classes. A :Property and :FeatureOfInterest should be intrinsic to each other, which could be reached by using a functional object property to link them to classes. To model a wider range of properties, some ontologies implemented an object property equivalent to :hasSubProperty that allows splitting up a complex :Property-class in simpler subproperties. Furthermore, the :Property-class would benefit from a geographical reference point, as is implemented by sosa:Platform. Some properties relate to time-series measurements, whereas others relate to static values. A pattern that can describe both and results in similar SPARQL queries is desirable for use-cases that use both types of properties. Various ontologies changed the :Result-class for a time-series reference point, including an identifier and information related to the time-series database.

In general, four levels of detail could be distinguished to describe properties and their values (Figure 20). Level 1 uses a datatype property to directly link a property value to a feature of interest. The datatype property typically covers the name of the property and could be mapped as an rdfs:subproperty of an equivalent of :hasProperty. Level 2 uses an intermediate :Property-class that covers the name of the property that is being described. Level 2 descriptions are used in various ontologies that describe static properties (BIMDO [46], OMG level 2 [72,73], and BPO [59]), but cannot easily store temporal properties and their provenance data. OPM [77] uses a third level of detail and introduces the opm:PropertyState-class. Instances of this class can store provenance data, such as the time of generation and the agent that generated the value. Multiple ontologies that describe dynamic properties implement a fourth class that describes the process of creating a property state. The class often represents an observation or actuation and is generalized to :Execution in the EEP ontology [92]. An ontology that describes both static and dynamic properties should integrate all four levels of detail so that data modelers have the freedom to choose the complexity that is necessary for a specific use-case.

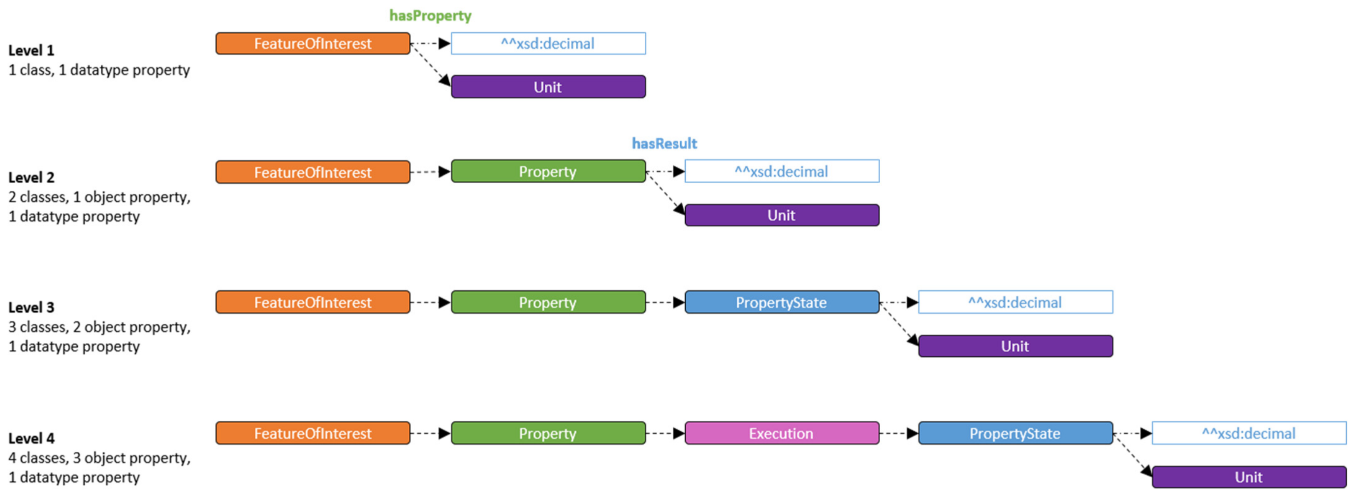


Figure 20. Levels of detail in ontologies describing static and dynamic properties.

4. Building Performance Ontology

Considering the growing interest in optimizing building performance, there is a need for an ontology that can integrate static and dynamic property values with topological building information in a uniform way to support the great variety of building performance assessments. Based on this need, we set up five competency questions:

CQ1 What is the current state of all properties related to an indoor environmental quality parameter?

CQ2 Where is the time-series data related to some of the indoor environmental quality properties stored?

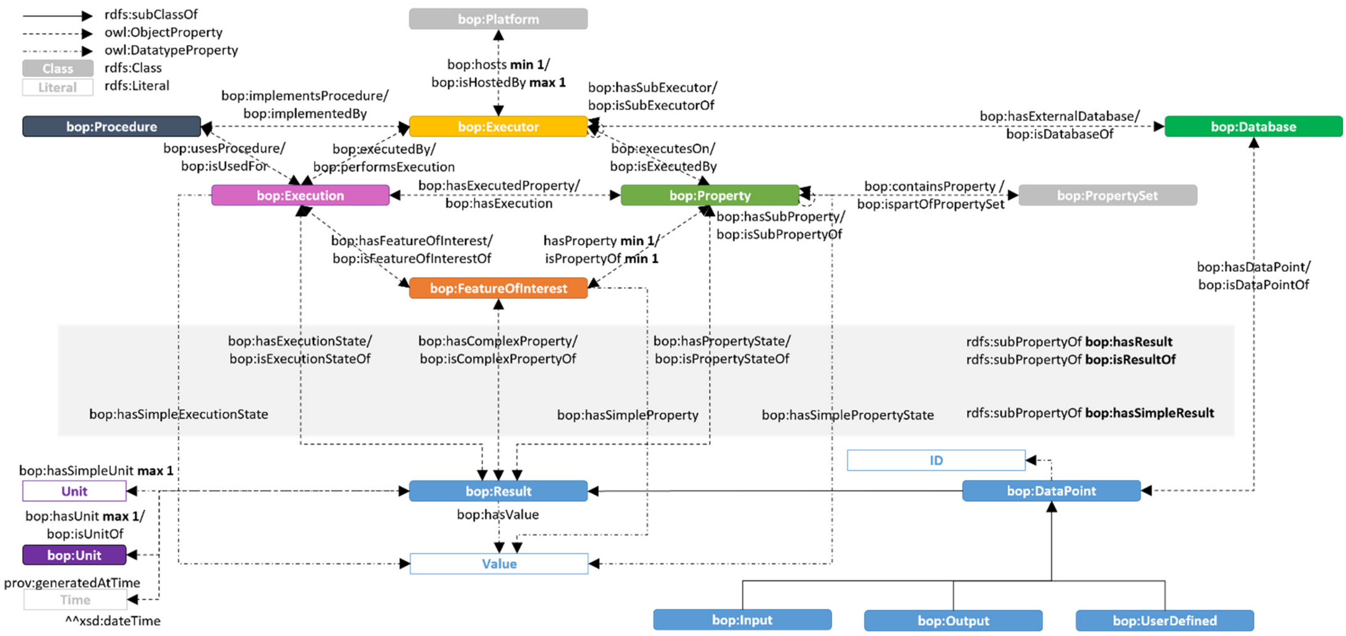
CQ3 What devices observe or act on properties related to an indoor environmental quality parameter?

CQ4 What is the datapoint of these properties in building management systems?

CQ5 What is the spatiotemporal resolution of the properties related to indoor environmental quality?

The aim of the Building Performance Ontology (BOP) is to enable the integration of topological building information with static and dynamic properties to create a homogeneous data environment used by complex building performance assessments. It can help building managers and their software to deal with the large heterogeneity of information. To remain stable over time, BOP is documented at https://w3id.org/bop and uses the namespace https://w3id.org/bop#. Its prefix, bop:, is registered at prefix.cc. The OntOlogy Pitfall Scanner! (OOPS!) [38] was used to detect common pitfalls in the ontology. It returned no pitfalls.

BOP is based on commonalities in design patterns and best practices found in the literature study in Section 3. It consists of a lightweight upper ontology that describes a generic structure of static and dynamic properties, and lower-level ontologies designed for specific use-cases (Figure 21). BOP's core consists of four classes, bop:Executor, bop:Execution, bop:Property, and bop:FeatureOfInterest. The four levels of detail (as described in Figure 20) are implemented so that data modelers can choose the level of complexity that fits their project. Figure 22 shows how bop:Execution, bop:FeatureOfInterest, and bop:Property are all linked to the resulting value by both a datatype property and an object property. Since all those separate routes are rdfs:subPropertyOf, bop:hasSimpleResult, bop:hasResult, and bop:isResultOf, one could simply query the results from those separate routes using one of these super-properties. The bop:DataPoint-class could be used to create a reference to a datapoint in a time-series database and is a subclass of bop:Result.



Legend. Property, Feature of interest, Executor, Execution, Procedure, Result, Unit, Database, Other.

Figure 21. The Building Performance Ontology's upper ontology.

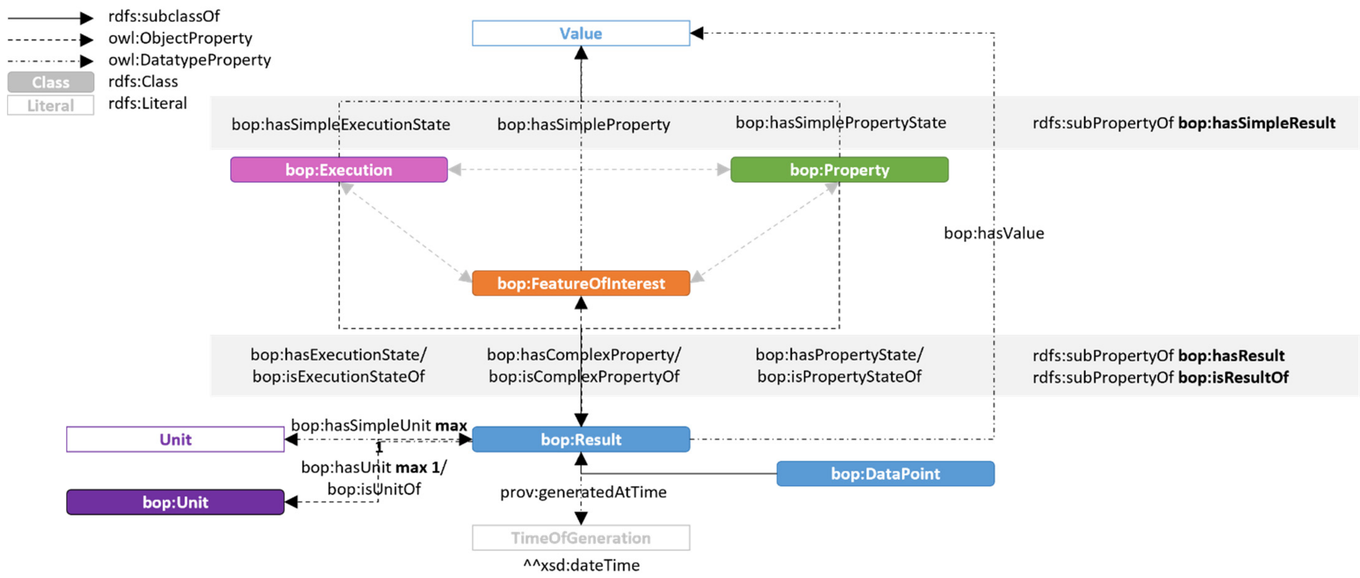


Figure 22. Result pattern: integrating multiple levels of detail to describe static and dynamic properties.

Table 3 presents the class definitions of the upper ontology of BOP. The definitions are composed by decomposing equivalent class definitions of the reviewed ontologies in Section 3. These were gathered by using the rdfs:comment or dcterms:description of the class.

Table 3. Class definitions of BOP.

	Class	Definition
(green)	bop:Property	A measurable and intrinsic characteristic of a feature of interest.
(orange)	bop:FeatureOfInterest	An abstraction of a real-world phenomenon which could be described in terms of its properties.
(yellow)	bop:Executor	An agent that can implement a procedure to perform an execution.
(pink)	bop:Execution	An act of carrying out a procedure by an executor on a property.
(dark grey-blue)	bop:Procedure	A workflow, protocol, plan, algorithm, or computational method specifying how to perform an execution.
(blue)	bop:Result	The outcome of an execution.
(purple)	bop:Unit	A particular quantity value that has been chosen as a scale for measuring other quantities of the same kind.
(green)	bop:Database	A collection of data.

The green class represents bop:Property, which is a measurable and intrinsic characteristic of a feature of interest. It represents both static and dynamic properties. Properties can be part of a bop:PropertySet. Properties can also be decomposed using the (transitive) bop:hasSubProperty object property.

Properties are intrinsic to a bop:FeatureOfInterest, represented by the orange block. The feature of interest (FOI) could be any real-world phenomenon. In practice, it will often be used to represent elements, zones, or buildings. The generic definition allows for a wide range of integration possibilities with other ontologies. FOIs are intrinsic to properties and can never exist without them. Multiple properties can be linked to a single FOI.

Properties are acted on by a bop:Executor, represented by the yellow block. Executors are agents that can implement a procedure to perform an execution. The executor class is subclassed by a bop:Sensor and a bop:Actuator, which, respectively, observe or act on a property. The introduction of multifunctional devices (such as multisensors) requires the introduction of a (transitive) bot:hasSubExecutor object property. It allows the description of multiple executive functions of devices. Executors are typically hosted by something; sensors could be hosted by a ceiling and radiators could be hosted by a wall. Bop:Platform describes this host and should host at least one executor.

The execution performed by an executor is represented by bop:Execution, the pink block. The execution is subclassed by bop:Observation (performed by a sensor) and bop:Actuation (performed by an actuator). The execution class should be carefully used. It is advised to limit its use for time-series measurements since they are best stored in external time-series databases. However, for less-frequent executions, such as site surveys observing the state of a construction or painting work to change the color of a wall, the class comes in handy. Specific procedures implemented by the executor to make the execution are described using bop:Procedure.

Results of a bop:Execution could be described as an IRI or a literal (Figure 22). Level 1 describes the value of a bop:Property by using bop:hasSimpleProperty to link bop:FeatureOfInterest to the result literal. Future extensions of BOP could create a multitude of subproperties of bop:hasSimpleProperty to increase querying possibilities. Examples could be bop-ext:hasLength, bop-ext:hasThermalInsulance, or bop-ext:hasColor. Level 2 uses the intermediate bop:Property and the bop:hasSimplePropertyState datatype property, which increases querying possibilities. Level 3 adds an instance of bop:Result that can be used to link provenance data related to the state of the property value. The highly complex level 4 description integrates the bop:Execution to describe the process of generating a property state.

As it is desirable to store high-frequency time-series data (earlier referred to as dynamic properties) in external databases, the result class has been subclassed by bop:DataPoint. It describes the point in a BMS system or other external database which represents the time-series data. Three subclasses of bop:DataPoint were introduced, being bop:Input, bop:Output, and bop:UserDefined. They represent common BMS vocabulary [45] and allow for more complex queries related to monitoring and controlling the IEQ in buildings. Each bop:DataPoint can be tagged with an ID using bop:hasID. This object property is an rdfs:subPropertyOf bop:hasValue so that the IDs of externally stored dynamic properties can be queried simultaneously with the values of static properties. The database could be described using bop:Database and can contain multiple bop:DataPoints. The database could be linked to an executor using bop:hasExternalDatabase.

Units of measure could be described as IRIs or literals, by using bop:hasUnit or bop:hasSimpleUnit, respectively. It is recommended to use an IRI and also refer to classes of specialist units of measure ontologies, such as QUDT or OM [125]. When results are stored as literals, custom datatypes could be used [126]. Since the results of time-series measurements stored in external databases are referred to as bop:DataPoint, which is a subclass of bop:Result, bop:Unit could be linked to the datapoint. The unit only needs to be stored once, reducing the latency of the time-series measurements. Similar to SOSA [89], bop:Unit is linked to the result rather than the property. The argument of not linking the unit to bop:Property is that multiple executions by multiple executors could be performed on a single property, with possible variations in the unit of measure. The other option—linking the unit to bop:Execution—would imply that an execution IRI is needed for every time-series measurement.

4.1. Using BOP in Practice

Figure 23 shows how BOP could be instantiated to represent static and dynamic properties in a real built-environment scenario. Using the sensor pattern and the static properties pattern (documented at [127]), multiple properties with a different nature could be represented. Other domain ontologies could be easily integrated; Figure 23 shows how BOT was used to specify topological relationships. Those BOT relationships could be created using the IFC-to-LBD converter [128] that converts IFC files to the RDF format.

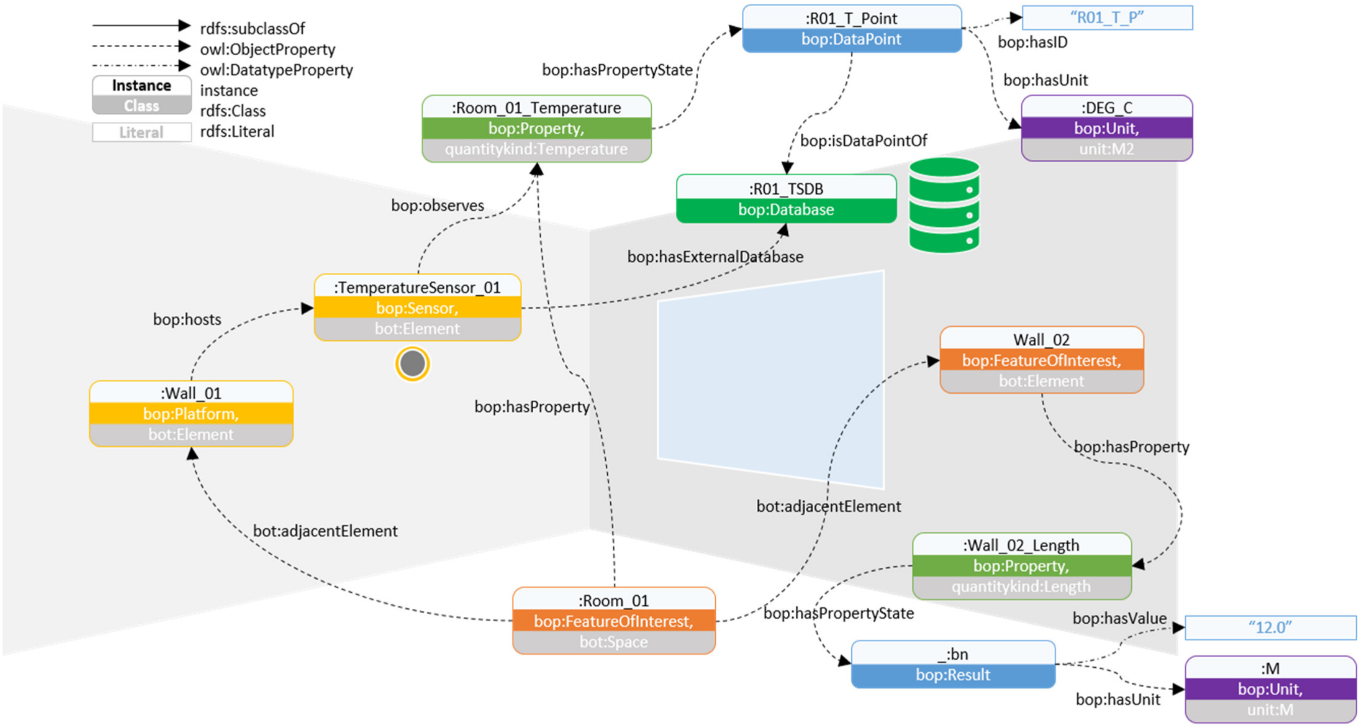


Figure 23. Instantiation of BOP.

4.1.1. Querying Static and Dynamic Properties

The shared structure of describing static and dynamic properties in BOP results in concise queries with high practical applicability, as was required in **CQ1**. Listing 2 shows how those queries could be used in practice. The query would result in an array of rooms, the static and dynamic properties associated with those rooms, and their results.

Listing 2. Querying static and dynamic properties.

```
PREFIX bop: <https://w3id.org/bop#>
PREFIX bot: <https://w3id.org/bot#>
SELECT ?room ?property ?result
WHERE {
?room a bop:FeatureOfInterest, bot:Space .
?room bop:hasProperty ?property .
?property bop:hasPropertyState ?result .
}
```

4.1.2. Spatiotemporal Resolution

CQ5 requires the possibility of describing the spatiotemporal resolution of a property. Listing 3 shows how the host of a sensor, its viewpoint, and their relationship could be queried based on the data in Figure 23. Geometric information related to the host and the viewpoint could further specify the spatial resolution of the measurement.

Listing 3. Querying the spatiotemporal resolution of a temperature measurement.

```
PREFIX bop: <https://w3id.org/bop#>
PREFIX bot: <https://w3id.org/bot#>
PREFIX quantitykind: <http://qudt.org/2.1/vocab/quantitykind>
SELECT ?host ?sensor ?viewpoint ?relationship ?currentState
WHERE {
?sensor a bop:Sensor .
?sensor bop:observes ?temperature .
?temperature a quantitykind:Temperature .
```

Listing 3. *Cont.*

```
?temperature bop:hasPropertyState ?currentState .
?sensor bop:isHostedBy ?host .
?temperature bop:isPropertyOf ?viewpoint .
?host ?relationship ?viewpoint .
}
```

4.1.3. System Control

Aligning different systems in buildings is essential for smart building control. The generic definition of bop:Executor allows for the integration of a multitude of systems using this class. Figure 24 presents the integration of a sensor and a radiator by linking them to the same bop:Property node. The object property linking the executor and the property describes how the executor influences the property. Tools can now query properties of an FOI and directly find the related executors to stimulate actions (Listing 4), as was required in **CQ3**. By describing the host of an executor, spatial context is added to the graph; not only does the graph specify the feature of interest of a property, but it also specifies a viewpoint from which an executor acts on this property.

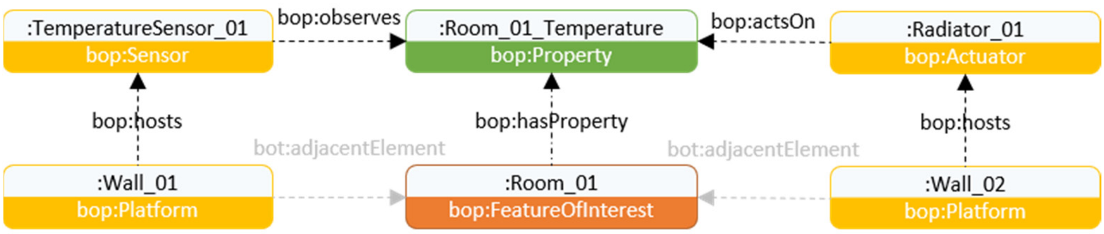


Figure 24. Aligning HVAC systems using BOP.

Listing 4. Querying a property and its related executors.

```
PREFIX bop: <https://w3id.org/bop#>
PREFIX bot: <https://w3id.org/bot#>
SELECT ?room ?property ?executor

WHERE {
?room a bop:FeatureOfInterest, bot:Space .
?room bop:hasProperty ?property .
?executor bop:executesOn ?property .
}
```

4.1.4. Database Alignment

External databases could be aligned with the linked building data using bop:Database and bop:DataPoint (Figure 25). The datapoint is a subclass of bop:Result, resulting in similar queries for static and dynamic properties, and increasing the flexibility of using those properties in complex algorithms. By aligning the database and datapoint with the linked data graph, applications using the data could query the database for each sensor and automatically create a corresponding API request to access the latest sensor measurement from the external database. The alignment could be created within the linked building data RDF file or in a separate alignment graph.

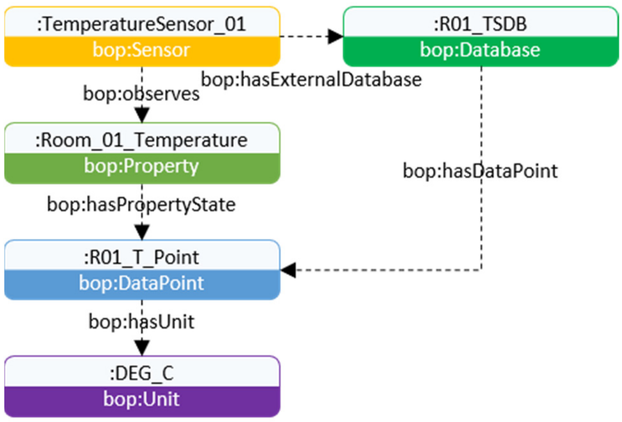


Figure 25. Aligning dynamic properties with external databases.

Listing 5 shows how a simple SPARQL query results in a property and its external storage location. Rooms and properties could easily be filtered by reusing other ontologies (e.g., quantitykind; see Listing 5). Listing 6 then uses this information to automatically query the corresponding time-series data from InfluxDB. WHERE clauses and other filters could also be created by using the SPARQL results. It fulfills the requirements of **CQ2** and **CQ4**.

Listing 5. Querying a property and its external storage location.

```
PREFIX bop: <https://w3id.org/bop#>
PREFIX bot: <https://w3id.org/bot#>
PREFIX quantitykind: <http://qudt.org/2.1/vocab/quantitykind>
SELECT ?property ?datapoint ?database
WHERE {
?room a bop:FeatureOfInterest, bot:Space .
?room bop:hasProperty ?property .
?property bop:hasPropertyState ?datapoint .
?property a quantitykind:Temperature .
?datapoint bop:isDataPointOf ?database .
}
```

Listing 6. Querying time-series data based on the SPARQL results.

```
USE ?database
SELECT ?property FROM ?datapoint WHERE ?property < 18
```

4.2. Evaluation

BOP is evaluated based on four evaluation criteria, namely, query efficiency, practical applicability, pattern efficiency, and extensibility. Figure 23 shows how a practical use-case could be modeled using BOP vocabulary. The various listings in Section 4 show how simple, short SPARQL queries could be built based for this ontology, without the need for complex SPARQL constructs. The simplicity allows starting data modelers to create SPARQL queries themselves. All queries listed in this paper were tested in a local GraphDB environment using the OpenSmartHome dataset [40] (also presented in [129]). All queries were executed within 0.1 s (the smallest measurable query execution time in GraphDB).

BOP is designed to integrate data for complex building performance algorithms and is based on state-of-the-art IEQ models and standards. Those methods were tested in a real-life use-case in earlier research [129]. The subclasses of bop:Result fit state-of-the-art BMS system architecture, allowing the integration of BIM and BMS. BOP is annotated using the dcterms and vann vocabularies. All classes and object properties are annotated using rdfs:label and rdfs:comment in both English and Dutch, consistently making use of language tags. Source documentation is openly available at https://github.com/alexdonkers/bop (accessed on 13 July 2022), where anyone could contribute or reuse the ontology. By storing data in their original format, BOP argues for less data mapping.

By integrating various levels of detail, the data modeler can choose how the result of a property should be modeled. The conciseness of the graph is, therefore, variable and fully based on the requirements of the data modeler. Based on the literature review, we believe the best payoff between semantic expressivity and efficiency is achieved when sensor data is stored in its native format, whereas a linked building data graph represents the context of the sensor data. BOP was designed in such a way that a data modeler needs to define this context only once, after which it could be used for every sensor measurement individually, resulting in concise but semantically rich graphs. A feature of BOP is the rich spatiotemporal resolution of data, which is obtained by separating the physical location of the executors' host and the feature of interest, adding a viewpoint to the data.

The lightweight upper ontology describes a generic data structure and is, therefore, easily extensible in depth. This version of BOP presents two lower-level patterns, which are a sensor and actuator ontology. Since the ontology is rather concise, it can also be extended by other ontologies for specific use-cases. Various extensions were made, including an extension to semantically describe database schemas (BOPDB [130]) and one for describing IEQ standards and their parameters (BAO [131]). Since the ontology is designed based on common design principles found in the literature review, many ontologies in Table 2 could also extend BOP. We encourage extending the ontology with taxonomical vocabularies, such as the buildingSMART Data Dictionary. Figure 23 shows how BOP could be extended in practice, by integrating it with BOT and QUDT's unit ontology. Alignment modules to other ontologies have not been made. Janowicz et al. [89] warn that alignment modules could introduce ontological commitments that are too strong, reducing the usability of the ontology.

5. Conclusions

There is a clear need for better monitoring of our indoor environments. State-of-the-art IEQ models require the integration of a wide range of heterogeneous data sources. Currently, this process is cumbersome and requires too much knowledge and manpower. Semantic web technologies can solve the heterogeneity issues. This requires an ontology that integrates topological building information with static and dynamic properties to create a homogeneous data environment that can be used for complex building performance assessments.

A literature review was conducted to find the commonalities in semantic data models related to IEQ from the IoT and BIM domains. Our pattern discovery approach visually revealed common design patterns and best practices in state-of-the-art ontologies. We concluded that the optimal ontology should uniformly integrate static and dynamic properties. By creating multiple levels of detail, the data modeler has the freedom to choose the semantic complexity that fits the project best. The pattern discovery reveals a central pattern of four classes (:Executor, :Execution, :FeatureOfInterest, and :Property) that are commonly used. Ontologies stay flexible for future extensions by using generic class names and definitions, and extending some of these generic classes (for example :Sensor and :Actuator) increases practical applicability. Based on this literature review, we concluded that an integrated model that semantically integrates static and dynamic properties with topological building information is currently lacking.

Therefore, we designed the Building Performance Ontology (BOP) by combining the common design patterns and best practices that were found in the literature review. It contains a generic upper ontology and multiple lower-level ontologies designed for specific use-cases. BOP excels in intuitiveness in both the resulting linked data structure and the resulting SPARQL queries. Simple SPARQL constructs can be used to query semantically rich information with an acceptable query execution time. The ontology has multiple levels of detail, ensuring applicability for multiple use-cases by data modelers with different preferences and levels of experience. A payoff between semantic expressivity and message payload was made by storing time-series data in external databases while describing the context of these measurements in the graph. The lightweight upper ontology of BOP is easily extensible in both horizontal and vertical directions, which was shown by integrating it with BOT and QUDT and extending it with the BOPDB and BAO ontologies. Based on some practical use-cases, we conclude that BOP fulfills the requirements of the competency questions.

BOP enables a uniform integration of static and dynamic properties and could be used to support a great variety of algorithms to model the indoor environmental quality of a building. The data model could be used to integrate the heterogeneous IEQ-related information from both the IoT and BIM domains. The resulting web of data could be used for various use-cases in the IEQ-domain, including the assessment of thermal comfort, visual comfort, acoustic comfort, and air quality [129]. By integrating IoT and BIM data, BOP supports the transition towards semantic digital twins and strives to be the next step toward automation of building management systems.

However, this research is ongoing. Near future developments will investigate the following aspects. First, a better integration with BMS systems should be realized. BOP has the potential to structure both input and output data in BMS systems and integrate this data with building information models. However, BOP was designed from a BIM perspective, and more knowledge from the BMS perspective is necessary to efficiently integrate input and output points. Second, a better integration with the occupant should be realized. A gap exists in the modeling of perceived IEQ and the integration of occupants, their actions, opinions, and preferences could lead to closing this gap. Finally, practical tools helping building managers and occupants to realize a higher IEQ should be created. Instantiating BOP is still a manual practice, and converters that automate this process would significantly improve the practical applicability of the ontology. Simultaneously, generalized IEQ calculation tools should be developed which calculate the state of various IEQ parameters based on linked data.

Author Contributions: Conceptualization, A.D., D.Y., B.d.V., and N.B.; literature review, A.D.; ontology development, A.D.; writing—original draft preparation, A.D.; writing—review and editing, A.D., D.Y., B.d.V., and N.B.; supervision, D.Y., B.d.V., and N.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Eindhoven University Technology, KPN (TKI-HTSM 19.0162), and the Netherlands Enterprise Agency, as part of the "SmartTWO: Maverick Telecom Technologies as Building Blocks for Value Driven Future Societies" project (TK | L912P06).

Data Availability Statement: An online repository, containing the BOP ontology, example queries, examples of how to instantiate BOP, and open linked building data is available at https://w3id.org/bop (accessed on 13 July 2022).

Conflicts of 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.

References

1. Kim, J.; de Dear, R. Nonlinear Relationships between Individual IEQ Factors and Overall Workspace Satisfaction. *Build. Environ.* **2012**, *49*, 33–40. [CrossRef]
2. Kim, J.; de Dear, R. Workspace Satisfaction: The Privacy-Communication Trade-off in Open-Plan Offices. *J. Environ. Psychol.* **2013**, *36*, 18–26. [CrossRef]
3. Fisk, W.J.; Black, D.; Brunner, G. Benefits and Costs of Improved IEQ in U.S. Offices. *Indoor Air* **2011**, *21*, 357–367. [CrossRef] [PubMed]
4. Fabian, M.P.; Adamkiewicz, G.; Stout, N.K.; Sandel, M.; Levy, J.I. A Simulation Model of Building Intervention Impacts on Indoor Environmental Quality, Pediatric Asthma, and Costs. *J. Allergy Clin. Immunol.* **2014**, *133*, 77–84. [CrossRef] [PubMed]
5. Diaz Lozano Patino, E.; Siegel, J.A. Indoor Environmental Quality in Social Housing: A Literature Review. *Build. Environ.* **2018**, *131*, 231–241. [CrossRef]
6. Al Horr, Y.; Arif, M.; Kaushik, A.; Mazroei, A.; Katafygiotou, M.; Elsarrag, E. Occupant Productivity and Office Indoor Environment Quality: A Review of the Literature. *Build. Environ.* **2016**, *105*, 369–389. [CrossRef]

7. Lee, M.C.; Mui, K.W.; Wong, L.T.; Chan, W.Y.; Lee, E.W.M.; Cheung, C.T. Student Learning Performance and Indoor Environmental Quality (IEQ) in Air-Conditioned University Teaching Rooms. *Build. Environ.* **2012**, *49*, 238–244. [CrossRef]
8. Al horr, Y.; Arif, M.; Katafygiotou, M.; Mazroei, A.; Kaushik, A.; Elsarrag, E. Impact of Indoor Environmental Quality on Occupant Well-Being and Comfort: A Review of the Literature. *Int. J. Sustain. Built Environ.* **2016**, *5*, 1–11. [CrossRef]
9. Tang, S.; Shelden, D.R.; Eastman, C.M.; Pishdad-Bozorgi, P.; Gao, X. A Review of Building Information Modeling (BIM) and the Internet of Things (IoT) Devices Integration: Present Status and Future Trends. *Autom. Constr.* **2019**, *101*, 127–139. [CrossRef]
10. Rhayem, A.; Mhiri, M.B.A.; Gargouri, F. Semantic Web Technologies for the Internet of Things: Systematic Literature Review. *Internet Things* **2020**, *11*, 100206. [CrossRef]
11. Burak Gunay, H.; Shen, W.; Newsham, G. Data Analytics to Improve Building Performance: A Critical Review. *Autom. Constr.* **2019**, *97*, 96–109. [CrossRef]
12. Boje, C.; Guerriero, A.; Kubicki, S.; Rezgui, Y. Towards a Semantic Construction Digital Twin: Directions for Future Research. *Autom. Constr.* **2020**, *114*, 103179. [CrossRef]
13. Adeleke, J.A.; Moodley, D. An Ontology for Proactive Indoor Environmental Quality Monitoring and Control. In *Proceedings of the 2015 Annual Research Conference on South African Institute of Computer Scientists and Information Technologists, Stellenbosch, South Africa, September 28-30, 2015*; Barnett, R.J., Cleophas, L., Kourie, D.G., Le Roux, D.B., Watson, B.W., Eds.; Association for Computing Machinery: New York, NY, USA, 2015.
14. O'Donnell, J.; Corry, E.; Hasan, S.; Keane, M.; Curry, E. Building Performance Optimization Using Cross-Domain Scenario Modeling, Linked Data, And Complex Event Processing. *Build. Environ.* **2013**, *62*, 102–111. [CrossRef]
15. Cheung, T.; Schiavon, S.; Parkinson, T.; Li, P.; Brager, G. Analysis of the Accuracy on PMV—PPD Model Using the ASHRAE Global Thermal Comfort Database II. *Build. Environ.* **2019**, *153*, 205–217. [CrossRef]
16. Altomonte, S.; Saadouni, S.; Kent, M.G.; Schiavon, S. Satisfaction with Indoor Environmental Quality in BREEAM and Non-BREEAM Certified Office Buildings. *Archit. Sci. Rev.* **2017**, *60*, 343–355. [CrossRef]
17. Altomonte, S.; Schiavon, S. Occupant Satisfaction in LEED and Non-LEED Certified Buildings. *Build. Environ.* **2013**, *68*, 66–76. [CrossRef]
18. Appel-Meulenbroek, R.; Clippard, M.; Pfnür, A. The Effectiveness of Physical Office Environments for Employee Outcomes: An Interdisciplinary Perspective of Research Efforts. *J. Corp. Real Estate* **2018**, *20*, 56–80. [CrossRef]
19. Willems, S.; Saelens, D.; Heylighen, A. Comfort Requirements versus Lived Experience: Combining Different Research Approaches to Indoor Environmental Quality. *Archit. Sci. Rev.* **2020**, *63*, 316–324. [CrossRef]
20. Heinzerling, D.; Schiavon, S.; Webster, T.; Arens, E. Indoor Environmental Quality Assessment Models: A Literature Review and a Proposed Weighting and Classification Scheme. *Build. Environ.* **2013**, *70*, 210–222. [CrossRef]
21. Frontczak, M.; Wargocki, P. Literature Survey on How Different Factors Influence Human Comfort in Indoor Environments. *Build. Environ.* **2011**, *46*, 922–937. [CrossRef]
22. Leccese, F.; Rocca, M.; Salvadori, G.; Belloni, E.; Buratti, C. Towards a Holistic Approach to Indoor Environmental Quality Assessment: Weighting Schemes to Combine Effects of Multiple Environmental Factors. *Energy Build.* **2021**, *245*, 111056. [CrossRef]
23. Tartarini, F.; Schiavon, S.; Cheung, T.; Hoyt, T. CBE Thermal Comfort Tool: Online Tool for Thermal Comfort Calculations and Visualizations. *SoftwareX* **2020**, *12*, 100563. [CrossRef]
24. Tartarini, F.; Schiavon, S. Pythermalcomfort: A Python Package for Thermal Comfort Research. *SoftwareX* **2020**, *12*, 100578. [CrossRef]
25. De Dear, R.; Brager, G.S. Developing an Adaptive Model of Thermal Comfort and Preference. *ASHRAE Trans.* **1998**, 104.
26. Leccese, F.; Salvadori, G.; Rocca, M.; Buratti, C.; Belloni, E. A Method to Assess Lighting Quality in Educational Rooms Using Analytic Hierarchy Process. *Build. Environ.* **2020**, *168*, 106501. [CrossRef]
27. Loomans, M.G.L.C.; Huovila, P.; Lefebvre, P.-H.; Porkka, J.; Huovila, A.; Sharan, Y.; Desmyter, J.; Vaturi, A.; Steskens, P. *D1.6: Optimal Indoor Performance Indicators (KIPI Framework)*; Eindhoven University of Technology: Eindhoven, The Netherlands, 2011.
28. International WELL Building Institute. In *The WELL Building Standard V2.0*; International WELL Building Institute: New York, NY, USA, 2020.
29. Leccese, F.; Rocca, M.; Salvadori, G. Fast Estimation of Speech Transmission Index Using the Reverberation Time: Comparison between Predictive Equations for Educational Rooms of Different Sizes. *Appl. Acoust.* **2018**, *140*, 143–149. [CrossRef]
30. Pauwels, P.; Zhang, S.; Lee, Y.C. Semantic Web Technologies in AEC Industry: A Literature Overview. *Autom. Constr.* **2017**, *73*, 145–165. [CrossRef]
31. Curry, E.; O'Donnell, J.; Corry, E. Building Optimisation Using Scenario Modeling and Linked Data. In Proceedings of the International Workshop on Linked Data in Architecture and Construction (LDAC 2012), Ghent, Belgium, 28–29 March 2012.
32. Corry, E.; Pauwels, P.; Hu, S.; Keane, M.; O'Donnell, J. A Performance Assessment Ontology for the Environmental and Energy Management of Buildings. *Autom. Constr.* **2015**, *57*, 249–259. [CrossRef]
33. Hu, S.; Wang, J.; Hoare, C.; Li, Y.; Pauwels, P.; O'Donnell, J. Building Energy Performance Assessment Using Linked Data and Cross-Domain Semantic Reasoning. *Autom. Constr.* **2021**, *124*, 103580. [CrossRef]
34. Hu, S.; Corry, E.; Horrigan, M.; Hoare, C.; Dos Reis, M.; O'Donnell, J. Building Performance Evaluation Using OpenMath and Linked Data. *Energy Build.* **2018**, *174*, 484–494. [CrossRef]

35. Reinisch, C.; Kofler, M.J.; Kastner, W. ThinkHome: A Smart Home as Digital Ecosystem. In Proceedings of the 4th IEEE International Conference on Digital Ecosystems and Technologies, Dubai, United Arab Emirates, 13–16 April 2010.
36. Beetz, J.; Van Leeuwen, J.; De Vries, B. IfcOWL: A Case of Transforming EXPRESS Schemas into Ontologies. *Artif. Intell. Eng. Des. Anal. Manuf. AIEDAM* **2009**, *23*, 89–101. [CrossRef]
37. Musen, M.A. The Protégé Project. *AI Matters* **2015**, *1*, 4–12. [CrossRef]
38. Poveda-Villalón, M.; Gómez-Pérez, A.; Suárez-Figueroa, M.C. OOPS! (OntOlogy Pitfall Scanner!): An on-Line Tool for Ontology Evaluation. *Int. J. Semant. Web Inf. Syst.* **2014**, *10*, 7–34. [CrossRef]
39. Garijo, D. WIDOCO: A Wizard for Documenting Ontologies. In *Lecture Notes in Computer Science, Proceedings of the 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017*; D'Amato, C., Fernandez, M., et al., Eds.; Springer: New York, NY, USA, 2017.
40. Donkers, A. OpenSmartHome. Available online: Github.com/AlexDonkers/OpenSmartHome (accessed on 12 May 2021).
41. Daniele, L.M.; den Hartog, F.T.H.; Roes, J.B.M. *Study on Semantic Assets for Smart Appliances Interoperability: D-S4: FINAL REPORT*; European Union: Luxembourg, 2015.
42. Stavropoulos, T.G.; Vrakas, D.; Vlachava, D.; Bassiliades, N. BOnSAI: A Smart Building Ontology for Ambient Intelligence. In *ACM International Conference Proceeding Series, Proceedings of the 2nd International Conference on Web Intelligence, Mining and Semantics, Craiova, Romania, June 13-15, 2012*; Association for Computing Machinery: New York, NY, USA, 2012.
43. Rasmussen, M.H.; Lefrançois, M.; Schneider, G.F.; Pauwels, P. BOT: The Building Topology Ontology of the W3C Linked Building Data Group. *Semant. Web* **2020**, *12*, 143–161. [CrossRef]
44. Hu, M.; Liu, Y.; Sugumaran, V.; Liu, B.; Du, J. Automated Structural Defects Diagnosis in Underground Transportation Tunnels Using Semantic Technologies. *Autom. Constr.* **2019**, *107*, 102929. [CrossRef]
45. Kučera, A.; Pitner, T. Semantic BMS: Allowing Usage of Building Automation Data in Facility Benchmarking. *Adv. Eng. Inform.* **2018**, *35*, 69–84. [CrossRef]
46. Niknam, M.; Karshenas, S. A Shared Ontology Approach to Semantic Representation of BIM Data. *Autom. Constr.* **2017**, *80*, 22–36. [CrossRef]
47. Niknam, M.; Karshenas, S. Integrating Distributed Sources of Information for Construction Cost Estimating Using Semantic Web and Semantic Web Service Technologies. *Autom. Constr.* **2015**, *57*, 222–238. [CrossRef]
48. Pauwels, P.; Terkaj, W. EXPRESS to OWL for Construction Industry: Towards a Recommendable and Usable IfcOWL Ontology. *Automation in Construction* **2016**, *63*, 100–133. [CrossRef]
49. Zhang, L.; Issa, R.R.A. Ontology-Based Partial Building Information Model Extraction. *J. Comput. Civ. Eng.* **2013**, *27*, 576–584. [CrossRef]
50. Lee, J.; Jeong, Y. User-Centric Knowledge Representations Based on Ontology for AEC Design Collaboration. *CAD Comput. Aided Des.* **2012**, *44*, 735–748. [CrossRef]
51. Kun, D.P.; Varga, E.B.; Toth, Z. Ontology Based Navigation Model of the ILONA System. In Proceedings of the SAMI 2017—IEEE 15th International Symposium on Applied Machine Intelligence and Informatics, Herl'any, Slovakia, 26–28 January 2017.
52. Anagnostopoulos, C.; Tsetsos, V.; Kikiras, P.; Hadjiefthymiades, S.P. OntoNav: A Semantic Indoor Navigation System. In *CEUR Workshop Proceedings, Proceedings of the First International Workshop on Managing Context Information in Mobile and Pervasive Environments, Ayia Napa, Cyprus, 9 May 2005*; Mansoor, W., Khedr, M., et al., Eds.; CEUR-WS.org: Aachen, Germany, 2005.
53. Pauwels, P.; Roxin, A. SimpleBIM: From Full IfcOWL Graphs to Simplified Building Graphs. In *eWork and eBusiness in Architecture, Engineering and Construction; Proceedings of the 11th European Conference on Product and Process Modelling, ECPPM*; CRC Press: Boca Raton, FL, USA, 2016.
54. Dibley, M.; Li, H.; Rezgui, Y.; Miles, J. An Ontology Framework for Intelligent Sensor-Based Building Monitoring. *Autom. Constr.* **2012**, *28*, 1–14. [CrossRef]
55. Dibley, M.J. An Intelligent System for Facility Management. PhD Thesis, Cardiff University, Wales, UK, October. 2011.
56. Dibley, M.J.; Li, H.; Miles, J.C.; Rezgui, Y. Towards Intelligent Agent Based Software for Building Related Decision Support. *Adv. Eng. Inform.* **2011**, *25*, 311–329. [CrossRef]
57. van Gool, S.; Yang, D.; Pauwels, P. Integrating Sensor and Building Data Flows: A Case Study of the IEQ of an Office Building in the Netherlands. In *13th European Conference on Product and Process Modeling 2020–2021*; CRC Press: Moscow, Russia, 2021; p. 6.
58. Gouda Mohamed, A.; Abdallah, M.R.; Marzouk, M. BIM and Semantic Web-Based Maintenance Information for Existing Buildings. *Autom. Constr.* **2020**, *116*, 103209. [CrossRef]
59. Wagner, A.; Rüppel, U. BPO: The Building Product Ontology for Assembled Products. In *CEUR Workshop Proceedings, Proceedings of the 7th Linked Data in Architecture and Construction Workshop, Lisbon, Portugal, 19–21 June 2019*; Poveda-Villalón, M., Pauwels, P., De Klerk, R., Roxin, A., Eds.; CEUR-WS.org: Aachen, Germany, 2019; Volume 2389, pp. 106–119.
60. Xu, Z.; Zhang, H.; Lu, X.; Xu, Y.; Zhang, Z.; Li, Y. A Prediction Method of Building Seismic Loss Based on BIM and FEMA P-58. *Autom. Constr.* **2019**, *102*, 245–257. [CrossRef]
61. Liu, H.; Lu, M.; Al-Hussein, M. Ontology-Based Semantic Approach for Construction-Oriented Quantity Take-off from BIM Models in the Light-Frame Building Industry. *Adv. Eng. Inform.* **2016**, *30*, 190–207. [CrossRef]
62. Venugopal, M.; Eastman, C.M.; Teizer, J. An Ontology-Based Analysis of the Industry Foundation Class Schema for Building Information Model Exchanges. *Adv. Eng. Inform.* **2015**, *29*, 940–957. [CrossRef]

63. Gao, G.; Liu, Y.S.; Wang, M.; Gu, M.; Yong, J.H. A Query Expansion Method for Retrieving Online BIM Resources Based on Industry Foundation Classes. *Autom. Constr.* **2015**, *56*, 14–25. [CrossRef]
64. Abanda, F.H.; Oti, A.H.; Tah, J.H.M. Integrating BIM and New Rules of Measurement for Embodied Energy and CO_2 Assessment. *J. Build. Eng.* **2017**, *12*, 288–305. [CrossRef]
65. Zhang, J.; El-Gohary, N.M. Integrating Semantic NLP and Logic Reasoning into a Unified System for Fully-Automated Code Checking. *Autom. Constr.* **2017**, *73*, 45–57. [CrossRef]
66. Lee, Y.C.; Eastman, C.M.; Solihin, W. An Ontology-Based Approach for Developing Data Exchange Requirements and Model Views of Building Information Modeling. *Adv. Eng. Inform.* **2016**, *30*, 354–367. [CrossRef]
67. Tchouanguem Djuedja, J.F.; Abanda, F.H.; Kamsu-Foguem, B.; Pauwels, P.; Magniont, C.; Karray, M.H. An Integrated Linked Building Data System: AEC Industry Case. *Adv. Eng. Softw.* **2021**, *152*, 102930. [CrossRef]
68. Werbrouck, J.; Pauwels, P.; Bonduel, M.; Beetz, J.; Bekers, W. Scan-to-Graph: Semantic Enrichment of Existing Building Geometry. *Autom. Constr.* **2020**, *119*, 103286. [CrossRef]
69. Radinger, A.; Rodriguez-Castro, B.; Stolz, A.; Hepp, M. BauDataWeb: The Austrian Building and Construction Materials Market as Linked Data. In *ACM International Conference Proceeding Series, Proceedings of the 9th International Conference on Semantic Systems, Graz, Austria, September 4-6, 2013*; Sabou, M., Blomqvist, E., Di Noia, T., Sack, H., Pellegrini, T., Eds.; Association for Computing Machinery: New York, NY, USA; p. 2013.
70. Karan, E.P.; Irizarry, J. Extending BIM Interoperability to Preconstruction Operations Using Geospatial Analyses and Semantic Web Services. *Autom. Constr.* **2015**, *53*, 1–12. [CrossRef]
71. Bassier, M.; Bonduel, M.; Derdaele, J.; Vergauwen, M. Processing Existing Building Geometry for Reuse as Linked Data. *Autom. Constr.* **2020**, *115*, 103180. [CrossRef]
72. Wagner, A.; Bonduel, M.; Pauwels, P.; Uwe, R. Relating Geometry Descriptions to Its Derivatives on the Web. In *Proceedings of the 2019 European Conference on Computing in Construction, Chania, Greece, 10-12 July, 2019*; European Council on Computing in Construction (EC3): Rhodes, Greece, 2019. [CrossRef]
73. Bonduel, M.; Wagner, A.; Pauwels, P.; Vergauwen, M.; Klein, R. Including Widespread Geometry Formats in Semantic Graphs Using RDF Literals. In *Proceedings of the 2019 European Conference on Computing in Construction, Chania, Greece, 10-12 July, 2019*; European Council on Computing in Construction (EC3): Rhodes, Greece. [CrossRef]
74. Hong, S.H.; Lee, S.K.; Yu, J.H. Automated Management of Green Building Material Information Using Web Crawling and Ontology. *Autom. Constr.* **2019**, *102*, 230–244. [CrossRef]
75. Lee, D.Y.; Chi, H.-l.; Wang, J.; Wang, X.; Park, C.S. A Linked Data System Framework for Sharing Construction Defect Information Using Ontologies and BIM Environments. *Autom. Constr.* **2016**, *68*, 102–113. [CrossRef]
76. Zhang, J.; Li, H.; Zhao, Y.; Ren, G. An Ontology-Based Approach Supporting Holistic Structural Design with the Consideration of Safety, Environmental Impact and Cost. *Adv. Eng. Softw.* **2018**, *115*, 26–39. [CrossRef]
77. Rasmussen, M.H.; Lefrançois, M.; Bonduel, M.; Hviid, C.A.; Karlshø, J. OPM: An Ontology for Describing Properties That Evolve over Time. In *CEUR Workshop Proceedings, Proceedings of the 6th Linked Data in Architecture and Construction Workshop, London, United Kingdom, 19–21 June 2018*; Poveda-Villalón, M., Pauwels, P., Roxin, A., Eds.; CEUR-WS.org: Aachen, Germany, 2018.
78. Ren, G.; Li, H.; Ding, R.; Zhang, J.; Boje, C.; Zhang, W. Developing an Information Exchange Scheme Concerning Value for Money Assessment in Public-Private Partnerships. *J. Build. Eng.* **2019**, *25*, 100828. [CrossRef]
79. Zhang, C.; Beetz, J.; De Vries, B. BimSPARQL: Domain-Specific Functional SPARQL Extensions for Querying RDF Building Data. *Semant. Web* **2018**, *9*, 829–855. [CrossRef]
80. Pauwels, P.; Krijnen, T.; Terkaj, W.; Beetz, J. Enhancing the IfcOWL Ontology with an Alternative Representation for Geometric Data. *Autom. Constr.* **2017**, *80*, 77–94. [CrossRef]
81. Costa, G.; Madrazo, L. Connecting Building Component Catalogues with BIM Models Using Semantic Technologies: An Application for Precast Concrete Components. *Autom. Constr.* **2015**, *57*, 239–248. [CrossRef]
82. Sadeghineko, F.; Kumar, B. Development of Semantically Rich 3D Retrofit Models. *J. Comput. Civ. Eng.* **2020**, *34*, 1–17. [CrossRef]
83. Mendes de Farias, T.; Roxin, A.; Nicolle, C. IfcWoD, Semantically Adapting IFC Model Relations into OWL Properties. *arXiv* **2015**, arXiv:1511.03897.
84. Kuster, C.; Hippolyte, J.L.; Rezgui, Y. The UDSA Ontology: An Ontology to Support Real Time Urban Sustainability Assessment. *Adv. Eng. Softw.* **2020**, *140*, 102731. [CrossRef]
85. Meng, X.; Wang, F.; Xie, Y.; Song, G.; Ma, S.; Hu, S.; Bai, J.; Yang, Y. An Ontology-Driven Approach for Integrating Intelligence to Manage Human and Ecological Health Risks in the Geospatial Sensor Web. *Sensors* **2018**, *18*(11), 3619. [CrossRef] [PubMed]
86. Esnaola-Gonzalez, I.; Bermúdez, J.; Fernandez, I.; Arnaiz, A. Semantic Prediction Assistant Approach Applied to Energy Efficiency in Tertiary Buildings. *Semant. Web* **2018**, *9*, 735–762. [CrossRef]
87. Howell, S.; Rezgui, Y.; Beach, T. Integrating Building and Urban Semantics to Empower Smart Water Solutions. *Autom. Constr.* **2017**, *81*, 434–448. [CrossRef]
88. Moreira, J.; Ferreira Pires, L.; van Sinderen, M.; Daniele, L. SAREF4health: IoT Standard-Based Ontology-Driven Healthcare Systems. In *Frontiers in Artificial Intelligence and Applications, Proceedings of the 10th International Conference on Formal Ontology in Information Systems, Cape Town, South Africa, 17-21 September, 2018*; Borgo, S., Hitzler, P., Kutz, O., Eds.; IOS Press: Amsterdam, The Netherlands, 2018.

89. Janowicz, K.; Haller, A.; Cox, S.J.D.; Le Phuoc, D.; Lefrançois, M. SOSA: A Lightweight Ontology for Sensors, Observations, Samples, and Actuators. *J. Web Semant.* **2019**, *56*, 1–10. [CrossRef]
90. Janowicz, K.; Compton, M. The Stimulus-Sensor-Observation Ontology Design Pattern and Its Integration into the Semantic Sensor Network Ontology. In *CEUR Workshop Proceedings, Proceedings of the 3rd International Workshop on Semantic Sensor Networks (SSN10), Shanghai, China, 7 November 2010*; Taylor, K., Ayyagari, A., De Roure, D., Eds.; CEUR-WS.org: Aachen, Germany, 2010.
91. Seydoux, N.; Drira, K.; Hernandez, N.; Monteil, T. Iot-O, a Core-Domain IoT Ontology to Represent Connected Devices Networks. In *Lecture Notes in Computer Science, Proceedings of the 20th International Conference on Knowledge Engineering and Knowledge Management, Bologna, Italy, November 19-23, 2016*; Blomqvist, E., Ciancarini, P., Poggi, F., Vitali, F., Eds.; Springer: Berlin/Heidelberg, Germany, 2016.
92. Esnaola-Gonzalez, I.; Bermüdez, J.; Fernandez, I.; Arnaiz, A. Two Ontology Design Patterns toward Energy Efficiency in Buildings. In *CEUR Workshop Proceedings, Proceedings of the 9th Workshop on Ontology Design and Patterns, Monterey, USA, 9 October 2018*; Skjæveland, M.G., Hu, Y., Hammar, K., Svátek, V., Ławrynowicz, A., Eds.; CEUR-WS.org: Aachen, Germany, 2018.
93. Bonino, D.; Corno, F.; De Russis, L. Poweront: An Ontology-Based Approach for Power Consumption Estimation in Smart Homes. In *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST, Proceedings of the First International Summit of IoT360, Rome, Italy, October 27-28, 2014*; Giaffreda, R., Vieriu, R.-L., Pasher, E., Bendersky, G., Jara, A.J., Rodrigues, J.J.P.C., Mandler, B., Dekel, E., Eds.; Springer: Berlin/Heidelberg, Germany, 2015.
94. Lefrançois, M. Planned ETSI SAREF Extensions Based on the W3C&OGC SOSA/SSN-Compatible SEAS Ontology Patterns. In *CEUR Workshop Proceedings, Joint Proceedings of SEMANTICS 2017 Workshops, Amsterdam, Netherlands, 11 and 14 September 2017*; Fensel, A., Daniele, L., Eds.; CEUR-WS.org: Aachen, Germany, 2017.
95. Alirezaie, M.; Hammar, K.; Blomqvist, E. SmartEnv as a Network of Ontology Patterns. *Semant. Web* **2018**, *9*, 903–918. [CrossRef]
96. Hu, S.; Corry, E.; Curry, E.; Turner, W.J.N.; O'Donnell, J. Building Performance Optimisation: A Hybrid Architecture for the Integration of Contextual Information and Time-Series Data. *Autom. Constr.* **2016**, *70*, 51–61. [CrossRef]
97. Balaji, B.; Bhattacharya, A.; Fierro, G.; Gao, J.; Gluck, J.; Hong, D.; Johansen, A.; Koh, J.; Ploennigs, J.; Agarwal, Y.; et al. Brick: Towards a Unified Metadata Schema for Buildings. In *Proceedings of the 3rd ACM Conference on Systems for Energy-Efficient Built Environments, BuildSys, Palo Alto, CA, United States, November 16-17, 2016*; Association for Computing Machinery: New York, NY, USA, 2016.
98. Compton, M.; Barnaghi, P.; Bermudez, L.; García-Castro, R.; Corcho, O.; Cox, S.; Graybeal, J.; Hauswirth, M.; Henson, C.; Herzog, A.; et al. The SSN Ontology of the W3C Semantic Sensor Network Incubator Group. *J. Web Semant.* **2012**, *17*, 25–32. [CrossRef]
99. Agarwal, R.; Fernandez, D.G.; Elsaleh, T.; Gyrard, A.; Lanza, J.; Sanchez, L.; Georgantas, N.; Issarny, V. Unified IoT Ontology to Enable Interoperability and Federation of Testbeds. In Proceedings of the 2016 IEEE 3rd World Forum on Internet of Things, WF-IoT 2016, Reston, VA, USA, 12–14 December 2017.
100. Bonino, D.; Corno, F. DogOnt—Ontology Modeling for Intelligent Domotic Environments. In *Lecture Notes in Computer Science, Proceedings of the 7th International Semantic Web Conference, Karlsruhe, Germany, October 26-30, 2008*; Sheth, A., et al., Eds.; Springer: Berlin/Heidelberg, Germany, 2008.
101. Beetz, J.; de Vries, B.; van Leeuwen, J.P. RDF-Based Distributed Functional Part Specifications for the Facilitation of Service-Based Architectures. In Proceedings of the 24th CIB W78 Conference, Maribor, Slovenia, 27–29 June 2007.
102. Beetz, J.; van Leeuwen, J.; de Vries, B. Towards a Topological Reasoning Service for IFC-Based Building Information Models in a Semantic Web Context. In Proceedings of the 11th International Conf. on Computing in Civil and Building Engineering ICCCBE-XI, Montreal, Canada, 14–16 June 2006.
103. Ye, J.; Coyle, L.; Dobson, S.; Nixon, P. A Unified Semantics Space Model. In *Lecture Notes in Computer Science, Proceedings of the Third International Symposium of Location- and Context-Awareness, Oberpfaffenhofen, Germany, September 20-21, 2007*; Hightower, J., Schiele, B., Strang, T., Eds.; Springer: Berlin/Heidelberg, Germany, 2007.
104. Rasmussen, M.H.; Pauwels, P.; Hviid, C.A.; Karlshøj, J. Proposing a Central AEC Ontology That Allows for Domain Specific Extensions. In *Joint Conference on Computing in Construction, Heraklion, Crete, 04-07 April, 2017*; Bosche, F., Brilakis, I., Sacks, R., Eds.; Heriot-Watt University: Edinburgh, UK, 2017.
105. Bonduel, M.; Vergauwen, M.; Klein, R.; Rasmussen, M.H.; Pauwels, P. A Novel Workflow to Combine Bim and Linked Data for Existing Buildings. In *eWork and eBusiness in Architecture, Engineering and Construction, Proceedings of the 12th European Conference on Product and Process Modelling, Copenhagen, Denmark, 12-14 September, 2018*; Karlshøj, J., Scherer, R., Eds.; CRC Press: London, UK, 2018.
106. Mitterhofer, M.; Schneider, G.F.; Stratbücker, S.; Steiger, S. Semantics for Assembling Modular Components in a Scalable Building Performance Simulation. *J. Build. Perform. Simul.* **2019**, *12*, 145–159. [CrossRef]
107. Ying, H.; Lee, S. Generating Second-Level Space Boundaries from Large-Scale IFC-Compliant Building Information Models Using Multiple Geometry Representations. *Autom. Constr.* **2021**, *126*, 103659. [CrossRef]
108. Lilis, G.N.; Giannakis, G.I.; Rovas, D.V. Automatic Generation of Second-Level Space Boundary Topology from IFC Geometry Inputs. *Autom. Constr.* **2017**, *76*, 108–124. [CrossRef]
109. Schneider, G.F. Towards Aligning Domain Ontologies with the Building Topology Ontology. In Proceedings of the 5th LDAC Workshop, Dijon, France, 13–15 November 2017.

110. Wagner, A.; Möller, L.K.; Leifgen, C.; Rüppel, U. Solconpro: Describing Multi-Functional Building Products Using Semantic Web Technologies. In *eWork and eBusiness in Architecture, Engineering and Construction, Proceedings of the 12th European Conference on Product and Process Modelling, Copenhagen, Denmark, 12-14 September, 2018*; Karlshøj, J., Scherer, R., Eds.; CRC Press: London, UK, 2018.
111. Farias, T.M.; Roxin, A.; Nicolle, C. COBieOWL, an OWL Ontology Based on COBie Standard. In Proceedings of the Lecture Notes in Computer Science, Proceedings of the OTM 2015 Conferences, October 26-30, 2015; Debruyne, C., Panetto, H., Meersman, R., Dilon, T., Weichart, C., An, Y., Ardagna, C.A., Eds.; Springer: Berlin/Heidelberg, Germany, 2015.
112. Hamdan, A.H.; Bonduel, M.; Scherer, R.J. An Ontological Model for the Representation of Damage to Constructions. In *CEUR Workshop Proceedings, Proceedings of the 7th Linked Data in Architecture and Construction Workshop, Lisbon, Portugal, 19–21 June 2019*; Poveda-Villalón, M., Pauwels, P., De Klerk, R., Roxin, A., Eds.; CEUR-WS.org: Aachen, Germany, 2019.
113. Wagner, A.; Bonduel, M.; Pauwels, P.; Rüppel, U. Representing Construction-Related Geometry in a Semantic Web Context: A Review of Approaches. *Autom. Constr.* **2020**, *115*, 103130. [CrossRef]
114. Vasilakis, G.; Garcia-rojas, A.; Papaleo, L.; Catalano, C.E.; Spagnuolo, M.; Robbiano, F.; Vavalis, M.; Pitikakis, M. A Common Ontology for Multi-Dimensional Shapes. In *SAMT 2007 2nd International Conference on Semantic and Digital Media, Genoa, Italy, December 5-7, 2007*; Falcidieno, B., et al., Eds.; Springer: Berlin/Heidelberg, Germany, 2007; Vavalis, M.
115. Cox, S.J.D.; OGC Implementation Standard 2011. Observations and Measurements - XML Implementation. Available online: https://portal.ogc.org/files/?artifact_id=41510 (accessed on 20 September 2022).
116. Botts, M.; Robin, A.; Hirschorn, E. OGC®SensorML: Model and XML Encoding Standard. Available online: http://docs.ogc.org/is/12-000r2/12-000r2.html (accessed on 12 May 2021).
117. Viktorović, M.; Yang, D.; de Vries, B. Connected Traffic Data Ontology (Ctdo) for Intelligent Urban Traffic Systems Focused on Connected (Semi) Autonomous Vehicles. *Sensors* **2020**, *20*, 2961. [CrossRef]
118. Pedrinaci, C.; Domingue, J. Toward the next Wave of Services: Linked Services for the Web of Data. *J. Univers. Comput. Sci.* **2010**, *16*, 1694–1719. [CrossRef]
119. Bermudez-Edo, M.; Elsaleh, T.; Barnaghi, P.; Taylor, K. IoT-Lite: A Lightweight Semantic Model for the Internet of Things. In *13th IEEE International Conference on Ubiquitous Intelligence and Computing, 13th IEEE International Conference on Advanced and Trusted Computing, 16th IEEE International Conference on Scalable Computing and Communications, Toulouse, France, 18-21 July 2016*; IEEE: New York, NY, USA, 2016.
120. Gyrard, A.; Datta, S.K.; Bonnet, C.; Boudaoud, K. Standardizing Generic Cross-Domain Applications in Internet of Things. In *2014 IEEE Globecom Workshops, GC Wkshps, Austin, TX, United States, 08-12 December, 2014*; IEEE: New York, NY, USA, 2014.
121. Gyrard, A.; Bonnet, C.; Boudaoud, K. Enrich Machine-to-Machine Data with Semantic Web Technologies for Cross-Domain Applications. In Proceedings of the 2014 IEEE World Forum on Internet of Things, WF-IoT 2014, Seoul, Korea, 6–8 March 2014.
122. Daniele, L.; den Hartog, F.; Roes, J. Created in Close Interaction with the Industry: The Smart Appliances REFerence (SAREF) Ontology. In *Lecture Notes in Business Information Processing*; Springer: Berlin/Heidelberg, Germany, 2015.
123. Poveda-Villalón, M.; García-Castro, R. *Extending the SAREF Ontology for Building Devices and Topology, In CEUR Workshop Proceedings, Proceedings of the 6th Linked Data in Architecture and Construction Workshop, London, United Kingdom, 19–21 June 2018*; Poveda-Villalón, M., Pauwels, P., Roxin, A., Eds.; CEUR-WS.org: Aachen, Germany, 2018.
124. *ISO 16739*; ISO Industry Foundation Classes (IFC) for Data Sharing in the Construction and Facility Management Industries. 2018. Available online: https://www.iso.org/standard/70303.html (accessed on 20 September 2022).
125. Rijgersberg, H.; Van Assem, M.; Top, J. Ontology of Units of Measure and Related Concepts. *Semant. Web* **2013**, *4*, 3–13. [CrossRef]
126. Lefrançois, M.; Zimmermann, A. The Unified Code for Units of Measure in RDF: Cdt:Ucum and Other UCUM Datatypes. In *Lecture Notes in Computer Science, Proceedings of the ESWC 2018 Satellite Events, Heraklion, Greece, June 3-7, 2018*; Gangemi, A., Gentile, A.L., Nuzzolese, A.G., Rudolph, S., Maleshkova, M., Paulheim, H., Pan, J.Z., Alam, M., Eds.; Springer: Berlin/Heidelberg, Germany, 2018.
127. Donkers, A.; Yang, D.; de Vries, B.; Baken, N. Building Performance Ontology. Available online: https://w3id.org/bop (accessed on 12 May 2021).
128. Bonduel, M.; Oraskari, J.; Pauwels, P.; Vergauwen, M.; Klein, R. The IFC to Linked Building Data Converter—Current Status. In *CEUR Workshop Proceedings, Proceedings of the 6th Linked Data in Architecture and Construction Workshop, London, United Kingdom, 19–21 June 2018*; Poveda-Villalón, M., Pauwels, P., Roxin, A., Eds.; CEUR-WS.org: Aachen, Germany, 2018.
129. Donkers, A.; Yang, D.; De Vries, B.; Baken, N. Real-Time Building Performance Monitoring Using Semantic Digital Twins. In *CEUR Workshop Proceedings, Proceedings of the 9th Linked Data in Architecture and Construction Workshop, Luxembourg, Luxembourg, 11–13 October 2021*; Poveda-Villalón, M., Pauwels, P., Eds.; CEUR-WS.org: Aachen, Germany, 2021.
130. Donkers, A. BOP Database Ontology. Available online: https://alexdonkers.github.io/bopdb (accessed on 12 May 2021).
131. Donkers, A. Building Assessment Ontology. Available online: https://alexdonkers.github.io/bao (accessed on 12 May 2021).