ifcOWL+: a semantic extension to Industry Foundation Classes (IFC) towards applications in building automation

Max Berktold[a,b], Martin Rätz[a], Dirk Müller[a]

[a] *RWTH Aachen University, E.ON Energy Research Center,*
Institute for Energy Efficient Buildings and Indoor Climate, Germany
[b] *max.berktold@eonerc.rwth-aachen.de, CA*

Abstract:

Model predictive control, fault detection and diagnosis, and energy monitoring have demonstrated significant potential for improving the energy efficiency of buildings. However, their widespread adoption is hindered by the extensive manual engineering effort required for implementation, largely due to fragmented building information stored in non-machine-readable formats. While the Industry Foundation Classes (IFC) have advanced digitalisation in construction, they remain focused on planning rather than building automation. To address this, researchers are leveraging the Semantic Web to promote interoperability across various engineering disciplines. Supporting this direction, BuildingSMART introduced ifcOWL, a representation of the IFC schema based on the Web Ontology Language. However, the inherent complexity of the IFC schema makes querying such knowledge graphs difficult in practice. To overcome this limitation, we propose the ifcOWL+ ontology, which extends ifcOWL by introducing semantic shortcuts that simplify data retrieval. These shortcuts reduce the need for detailed knowledge of the underlying IFC structure while preserving semantic consistency. The proposed ontology is demonstrated using a two-story office building equipped with multiple consumers supplied by hydronic and duct-based systems. We show how the enriched knowledge graph enables simplified querying and supports the implementation of a basic fault detection algorithm. Ultimately, the ifcOWL+ ontology represents a step towards the automated deployment of applications in building automation and supports the digitalisation of energy-efficient building operation.

Keywords:

Building Automation; Fault Detection and Diagnosis; Industry Foundation Classes; Ontologies; Semantic Web Technologies.

1. Introduction

Buildings account for around one third of global energy consumption [1]. Consequently, improving the energy efficiency of buildings is a primary lever for achieving global climate goals. In this context, applications in Building Automation (BA) such as Model Predictive Control, Fault Detection and Diagnosis (FDD), and Energy Monitoring have shown significant potential for increasing the operational efficiency of buildings. However, their practical deployment remains limited, as implementation still requires substantial manual engineering effort [2, 3]. This limitation stems largely from information being scattered across various non machine-readable and heterogeneous sources such as technical drawings, text documents, and spreadsheets. Such fragmentation hinders the systematic and automated deployment of BA applications, as it requires manual data collection and integration, which is time-consuming and error-prone [4–6].

Building Information Modeling (BIM) addresses these challenges by providing a unified digital representation of building data. To ensure interoperability, buildingSMART established the Industry Foundation Classes (IFC) as the open standard for BIM-based information sharing. The IFC schema captures geometry, spatial topology, and technical system specifications across the entire building lifecycle [7]. This information, when modeled with sufficient detail, can be highly valuable for BA applications. Yet, despite the potential benefits, BIM is rarely leveraged in this domain [8].

At the same time, the Semantic Web Technologies (SWT) have been proposed to make building information machine-readable and linkable across domains [9, 10]. In this context, several ontologies have emerged to represent building information in a semantic format. Notable examples include the Real Estate Core[1] and the Brickschema[2] [11]. Other efforts focus on aligning semantic models with industry standards, such as ASHRAE Standard 223[3].

In order to integrate IFC data into the SWT ecosystem, Beetz et al. proposed an Web Ontology Language (OWL) representation of the IFC schema known as ifcOWL [12]. This translation allows for graph based querying and reasoning on IFC data while enabling interoperability beyond traditional BIM applications. This effort was later recognized by buildingSMART by publishing the official ifcOWL[4] representation of the IFC schema. Building on this foundation, Pauwels et al. further advanced the translation of IFC data within the SWT [13, 14]. Despite its standardization, the practical utility of ifcOWL is limited by its complexity and verbosity, which strictly mirrors the original EXPRESS schema.

To address this, Pauwels and Roxin [15] propose the “SimpleBIM” approach to dynamically restructure ifcOWL data into more simple representations. By implementing targeted simplification strategies, the authors demonstrated that Resource Description Framework (RDF) triple counts and file sizes could be reduced by approximately 90%. This transformation provides the lightweight building data necessary for efficient querying and reasoning in practical automation workflows. The simplification is achieved by stripping redundant geometric and presentation data while reducing complex wrapper classes into direct OWL datatype properties. Furthermore, the approach flattens the property hierarchy and replaces intermediate relationship instances with direct object properties between building elements.

Beyond general schema restructuring, specific attention has been given to the “triple explosion” caused by geometric data, which typically relies on verbose ordered collections in the standard ifcOWL ontology. To optimize these structures, Pauwels et al. [16] propose an alternative representation that serializes geometric aggregated data into literals. The authors demonstrated a reduction in triple counts of approximately 50% for representative real-world building models. This approach significantly enhances the scalability and computational efficiency of geometric data within the Semantic Web ecosystem without a total loss of geometric semantics.

Despite these advancements, directly using the ifcOWL for operational applications still remains challenging. Although IFC contains rich building information, its schema was primarily developed for planning, design, and construction processes. As a result, information that is relevant for BA applications, especially regarding the hydronic and duct systems, is often not represented in a directly accessible form. Even simple queries, such as identifying connections between devices or retrieving device locations, may require traversing several

[1] https://www.realestatecore.io/
[2] https://brickschema.org/
[3] https://docs.open223.info/
[4] https://technical.buildingsmart.org/standards/ifc/ifc-formats/ifcowl/

intermediate entities and demand detailed knowledge of the IFC schema. This complexity limits the practical reuse of IFC-derived semantic data for applications in BA.

This paper addresses this limitation by proposing the ifcOWL+ ontology, a OWL-based extension that enriches IFC-based graphs with direct semantic relationships for operational and BA related queries. The ontology introduces additional OWL classes and OWL object properties that simplify access to relevant system relations while preserving compatibility with IFC-derived data. These relationships are inferred using Shapes and Constraints Language (SHACL), enabling more concise SPARQL Protocol and RDF Query Language (SPARQL) queries and reducing the effort required to derive application-relevant information.

The remainder of the paper is structured as follows. Section 2 presents the proposed ontology extension and inference workflow. In Section 3, we apply the ontology to a example building and demonstrate how the enriched Knowledge Graph (KG) facilitates information retrieval for FDD use cases. While the evaluation focuses specifically on FDD, the topological queries enabled by ifcOWL+ are broadly applicable to other applications in BA. Finally, Section 4 summarizes the findings and outlines future work.

2. Introducing the ifcOWL+ ontology

The proposed ifcOWL+ ontology extends ifcOWL with additional classes and object properties that simplify the retrieval of information relevant for use cases in BA. The ontology uses both the OWL and SHACL rules to infer new relations that are not explicitly represented in the original IFC schema. This enriched KG enables direct queries for building-automation-oriented information such as spatial context, metadata, and distribution-system dependencies. As the ontology is an extension it does not lose any information from the original IFC KG, but rather adds new relations that are more directly accessible. The ontology is available on GitHub: `https://github.com/RWTH-EBC/ifcOWLplus`.

2.1. Core classes

BIM captures the precise 3D routing of distribution networks, including exact pipe lengths and individual fittings for coordination and clash detection purposes. However, for use cases in BA, this granular geometric data is often secondary to the functional connectivity and logical wiring of devices. For example, determining which devices are affected by a valve closure requires traversing dozens of intermediate pipe segments and fluid ports in plain ifcOWL. To facilitate such operational queries, ifcOWL+ emphasizes the functional connectivity of devices rather than the precise geometric routing. This is formalized through the distinction between active functional devices and passive carrier elements, represented by two upper-level OWL classes:

Flow Carrier The `ifcOWL+:FlowCarrier` represents all network elements whose primary purpose is to route a medium. Specifically, this class comprises `ifcOWL:FlowSegment` and `ifcOWL:FlowFitting`. The subclasses `ifcOWL+:Pipe` and `ifcOWL+:Duct` are used to further specialize the carrier type.

Device The `ifcOWL+:Device` includes all remaining operationally relevant distribution elements of type `ifcOWL:IfcDistributionElement`. It covers components such as pumps, valves, and dampers, and also monitoring/control equipment such as sensors and thermostats.

A visual representation of the distinction between devices and flow carriers is shown in Figure 1. The `ifc` prefix is used for classes and properties from the original ifcOWL, while the `ifcPlus` prefix indicates the new classes and properties introduced by ifcOWL+. The relations connecting devices and flow carriers are further described in the next section.

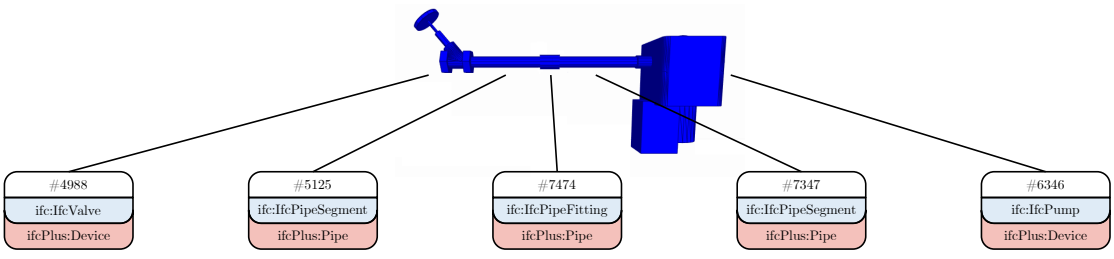

Figure 1. *Device and flow carrier distinction in ifcOWL+ at the example of a valve connected to a pump through an intermediate pipe and fitting.*

2.2. Shortcut properties

In addition to the new classes, ifcOWL+ introduces a set of OWL object properties that directly expose relations otherwise hidden in the nested ifcOWL representation. These properties are listed in Table 1 and grouped into five categories. Each category addresses specific operational querying needs in BA applications, enabling more efficient and intuitive access to relevant information without requiring deep knowledge of the underlying IFC schema. The relations are presented by category in this section.

Table 1. *Summary of ifcOWL+ shortcut relations and their operational motivation.*

Category	Property	Inverse	Description
Properties	`HasPropertySet`	`IsPropertySetOf`	Access of metadata.
Spatial	`HasLocation`	`IsLocationOf`	Fast localization and zone-based aggregation.
Spatial	`HasWall`	`IsWallOf`	Direct access to bounding walls.
Port	`HasPort`	`IsPortOf`	Direct access to Ports.
Port	`FeedsPort`	`IsFedByPort`	Provide oriented connectivity between ports.
Flow	`Feeds`	`IsFedBy`	Directed adjacency.
Flow	`FeedsIndirectly`	`IsFedByIndirectly`	Reachability along carrier networks.
Connectivity	`IsConnectedTo`	`IsConnectedFrom`	Collapse carriers to device-level dependencies.
Connectivity	`IsConnectedToByPipe`	`IsConnectedFromByPipe`	Medium-specific device connectivity.
Connectivity	`IsConnectedToByDuct`	`IsConnectedFromByDuct`	Medium-specific device connectivity.

Port relations To bypass the nested ifcOWL relationship structure and support causal reasoning, ifcOWL+ flattens port-level connectivity: The property `ifcOWL+:HasPort` creates a direct link between an `ifcOWL:IfcDistributionElement` and its `ifcOWL:IfcDistributionPort`. This simplifies model-quality checks, such as identifying elements with missing or unconnected ports. By evaluating the `ifcOWL:FlowDirection` attribute, `ifcOWL+:FeedsPort` connects a SOURCE port directly to a SINK port. This explicit orientation is essential for downstream impact analysis and upstream root-cause identification.

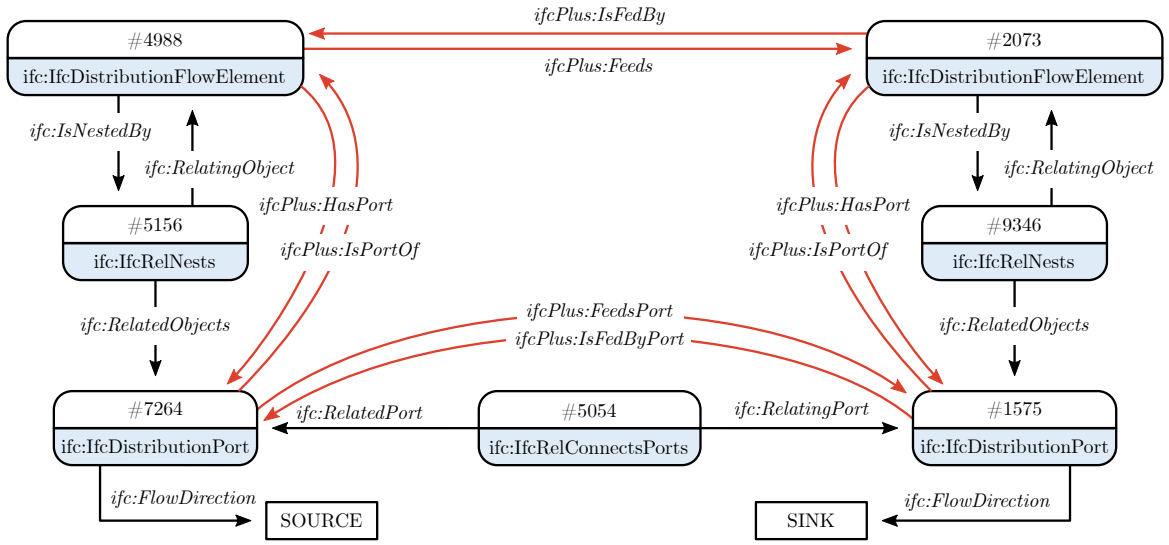

Figure 2. *Inference of the `ifcOWL+:Feeds` property. Black arrows represent standard IFC relationships; red arrows show the inferred `ifcOWL+:Feeds` relationship that directly connects two distribution elements.*

Direct flow relations To simplify system traversal, `ifcOWL+` introduces direct adjacency and reachability properties that bypass the nested IFC relationship structure. The properties `ifcOWL+:Feeds` and `ifcOWL+:IsFedBy` are inferred from port connectivity and flow direction. These allow queries to traverse supply dependencies directly between two `ifcOWL:IfcDistributionFlowElement` instances. For carrier chains like pipe or duct networks, the ontology defines `ifcOWL+:FeedsIndirectly`. Modeled as an OWL transitive property, it allows skipping intermediate segments and fittings to link functional endpoints, such as an Air Handling Unit (AHU) to a terminal unit.

Functional connectivity relations In many BA applications, the focus lies on device-level connectivity rather

than geometric routing. Extracting the device-only dependency graph for graph-based FDD propagation or querying for the nearest upstream controllable isolation device for a leaking terminal can be cumbersome to express robustly when intermediate carrier elements dominate the graph. To collapse intermediate pipes, ducts, and fittings and abstract functional connectivity at the most abstract level, ifcOWL+ introduces the OWL object property `ifcOWL+:IsConnectedTo` between instances of `ifcOWL+:Device`. These properties express that two devices are functionally connected through an intermediate carrier network. To preserve the transport medium explicitly, the ontology further defines the OWL sub-properties `ifcOWL+:IsConnectedToByPipe` and `ifcOWL+:IsConnectedToByDuct`. As demonstrated in Figure 3, the `IsConnectedTo` property directly links two active devices.

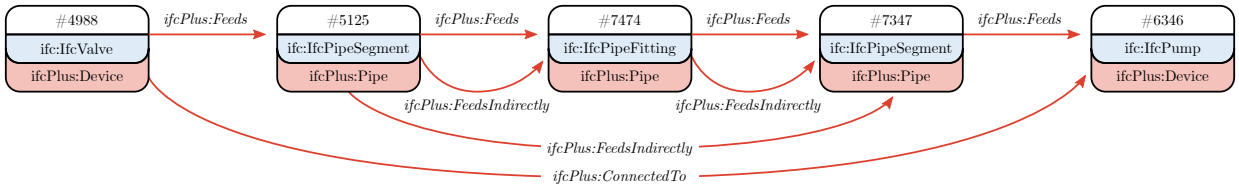

Figure 3. *Inference of the `IsConnectedTo` property, collapsing intermediate carrier elements to directly link functionally connected devices.*

Spatial relation Fast localization and grouping of assets (e.g., room- or storey-level aggregation) is critical for analytics, reporting, and spatially contextualized FDD. For example, the localization of faulty devices is necessary to support and initialize repair workflows (e.g., "where is the broken valve located?"). To simplify access to this data, the OWL object property `ifcOWL+:HasLocation` links any `ifcOWL:IfcDistributionElement` directly to the `ifcOWL:IfcSpatialStructureElement` in which it is located. The inverse OWL object property `ifcOWL+:IsLocationOf` links the spatial element back to the distribution element. This is demonstrated for an example room in Figure 4, where a radiator (`ifcOWL:IfcSpaceHeater`) is placed in a room.

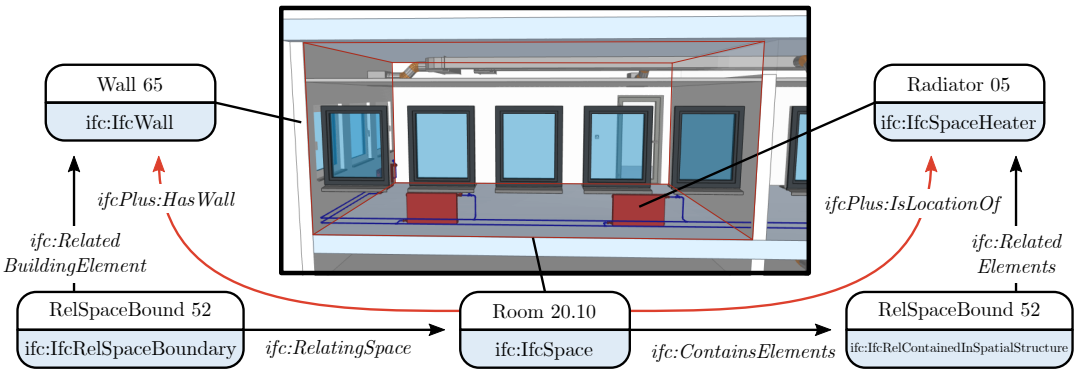

Figure 4. *Inference of `ifcOWL+:HasWall` and `ifcOWL+:IsLocationOf` to expose space-to-wall adjacency and containment relations.*

In addition, operational analyses and especially energy-related analyses often require building-envelope context, including spaces and their bounding walls. To support envelope-related queries without complex boundary traversals, ifcOWL+ introduces the OWL object property `ifcOWL+:HasWall`, which links an `ifcOWL:IfcSpace` directly to its bounding `ifcOWL:IfcWall` elements, inferred from `ifcOWL:IfcRelSpaceBoundary`. Based on this relation and the wall property-set metadata, a Boolean flag `ifcOWL+:IsExternal` is inferred for spaces that are adjacent to at least one wall marked as external.

Property set relation The deployment of BA functions often relies on device metadata such as nominal flow rates, temperature setpoints, manufacturer data, and valve coefficients. To simplify queries for such information, the OWL object property `ifcOWL+:HasPropertySet` links an `ifcOWL:IfcObject` directly to the corresponding `ifcOWL:IfcPropertySet`. This shortcut replaces the deeper nested traversal through `ifcOWL:IfcRel DefinesByProperties`.

For all the introduced properties, the inverse relations are also defined to allow for flexible querying in both

directions. The next section describes how these relations are inferred from the original IFC data using the SHACL and OWL.

2.3. Inference of implicit relations using OWL and SHACL

In the context of the SWT, inference is the process of deriving new triples from existing knowledge. utilizes OWL 2 to model fundamental semantic relationships, such as transitive properties, property chains, and subclass hierarchies. For this purpose, any reasoner compliant with the OWL 2 RL profile may be used for inference. However, certain logic discussed above requires complex pattern matching that exceeds the expressive capabilities of OWL. For these specific relationships, we utilize SHACL-SPARQL rules by leveraging the SHACL Advanced Features. Similar to OWL reasoning, a processing engine compliant with the SHACL Advanced Features specification is required for inference.

All SHACL shapes and rules for inferring the `ifcOWL+` relations are included in the aforementioned repository. Listing 1 shows an example SHACL-SPARQL rule for inferring the `ifcOWL+:Feeds` relation between two distribution elements. This inference is based on port exposure (`ifcOWL+:HasPort`) and port-to-port directionality (`ifcOWL+:FeedsPort`). The prefixes are the same for all code examples and are therefore omitted in the following listings.

```
@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix ifc: <http://ifcowl.openbimstandards.org/IFC4x3#> .
@prefix ifcOWL+: <http://ifcowl.openbimstandards.org/IFC4x3plus#> .
ifcOWL+:FeedsShape
    a sh:NodeShape ;
    sh:targetClass ifc:IfcDistributionElement ;
    sh:rule [
        a sh:SPARQLRule ;
        sh:construct """
            CONSTRUCT {
                $this ifcOWL+:Feeds ?end .
            }
            WHERE {
                $this ifcOWL+:HasPort ?start_port .
                ?end a ifc:IfcDistributionElement ;
                    ifcOWL+:HasPort ?end_port .
                ?start_port a ifc:IfcDistributionPort ;
                            ifcOWL+:FeedsPort ?end_port .
                ?end_port a ifc:IfcDistributionPort .
            }
        """ ;
    ] .
```

Listing 1: SHACL-SPARQL rule for inferring directed flow adjacency (`ifcOWL+:Feeds`) from port exposure and directionality.

3. Demonstration of ifcOWL+ on a example building

This section demonstrates how ifcOWL+ simplifies information retrieval on IFC-derived graphs. We first present representative examples that show how the shortcut relations enable concise SPARQL queries for typical information retrieval tasks. As a secondary demonstration, we show how the same enriched relations can be used to automatically apply a simple graph-based FDD algorithm to the example building.

The example building takes inspiration from the main building of the E.ON Energy Research Centre in Aachen, Germany, and is designed to represent a two-storey office building with a basement. Radiators with regulating valves provide heating to all rooms. Additionally, two roof-mounted AHUs supply conditioned air through a duct system to terminal units in all rooms. The rooms are also fitted with temperature and CO_2 sensors, which enable room-level air quality control.

The hydronic system is supplied by a central heat source, which distributes thermal energy to the radiator branches and AHU heating coils. All systems are modelled in detail, incorporating geometric and asset

information. This includes precise duct and pipe routing, as well as the locations of pumps, valves and sensors. Property sets that specify the desired room temperature set points and air change rates are also included for all rooms. Figure 5 shows a rendering of the building and excerpts of the system topology.

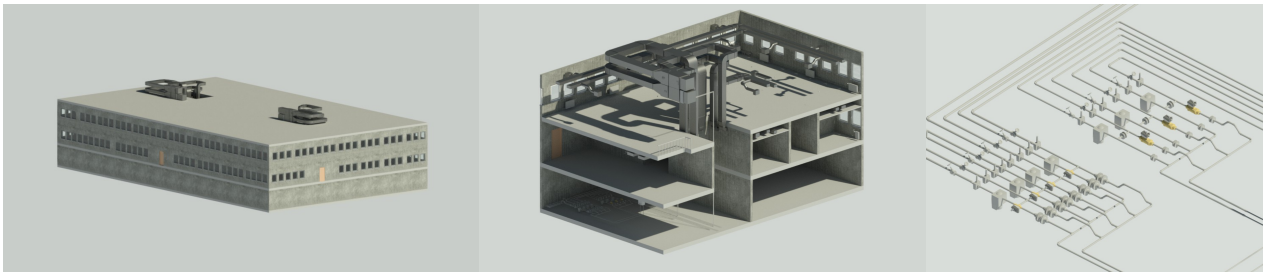

Figure 5. *3D rendering of the E.ON Energy Research Center (envelope, ventilation, hydronic)*

The building is authored in Autodesk Revit and exported to IFC 4.3. The IFC file is then converted to an RDF graph using the IFCtoRDF[5] converter, that is in conformance with standard translation workflow recommended by buildingSMART. The resulting base graph contains approximately 1.5×10^6 triples before reasoning. Considering the size and complexity of the resulting graph, we recommend using a dedicated triple store. In this study, we utilize Ontotext GraphDB for its robust native reasoning performance. Within GraphDB, the SHACL rules from Section 2.3 are compiled into a custom ruleset (PIE), combining a lightweight subset of standard RDF and OWL entailments with the ifcOWL+-specific rules. This setup allows for efficient materialization of the ifcOWL+ relations and classes from Section 2 while maintaining compatibility with the original ifcOWL graph structure.

3.1. Facilitated information retrieval for FDD use cases

We demonstrate the practical utility of ifcOWL+ for information retrieval by applying it to typical queries relevant for FDD use cases. For each of the following three tasks, we present the query using ifcOWL+ shortcut relations, and then compare it to the equivalent query without ifcOWL+. All queries and outputs from the following examples are included in the aforementioned GitHub repository.

Task A: device localization Many FDD algorithms require associating field devices with their spatial context. In this example, we link room temperature sensors to terminal heating units (e.g., radiators) by leveraging their shared room location. This association supports room-level FDD, for instance by relating room temperature measurements to radiator valve positions to detect faults such as stuck valves or faulty sensors.

Listing 2 shows how to retrieve this mapping using the shortcut relation `ifcOWL+:HasLocation`. The query binds both the radiator and the temperature sensor to the same `ifcOWL:IfcSpace`, avoiding explicit traversal of the underlying spatial containment structure as seen in figure 4. Executing the query against the KG returns 68 unique room-sensor-radiator tuples. Table 2 provides an excerpt including the room name and the IFC GUIDs of the associated radiator and temperature sensor.

```
SELECT ?spaceName ?radiatorId ?tempId
WHERE {
    ?space a ifc:IfcSpace ;
        ifc:Name ?spaceName .
    ?radiator a ifc:IfcSpaceHeater ;
        ifc:PredefinedType "RADIATOR" ;
        ifcOWL+:HasLocation ?space ;
        ifc:GlobalId ?radiatorId .
    ?temp a ifc:IfcSensor ;
        ifc:PredefinedType "TEMPERATURESENSOR" ;
        ifcOWL+:HasLocation ?space ;
        ifc:GlobalId ?tempId .
}
```

Listing 2: Task A with ifcOWL+: room association via `ifcOWL+:HasLocation`.

[5] `https://github.com/pipauwel/IFCtoRDF`

The query uses N=10 triple patterns. In plain ifcOWL, the two `ifcOWL+:HasLocation` patterns are replaced by four containment patterns, yielding N=12 triple patterns. Besides the increased number of patterns, the ifcOWL+ terms are more intuitive and directly express the spatial association.

Table 2. *Task A result excerpt: room-level association between radiators and temperature sensors.*

Space	Radiator (IFC GUID)	Temperature sensor (IFC GUID)
"67"	"0pEWkNLhjE3xM3DTIDgrDQ"	"0CZbfj7uz0LfSTXWMcvPLd"
"69"	"0pEWkNLhjE3xM3DTIDgr1O"	"0CZbfj7uz0LfSTXWMcvPAG"
"69"	"0pEWkNLhjE3xM3DTIDgr3S"	"0CZbfj7uz0LfSTXWMcvPAG"
"69"	"0pEWkNLhjE3xM3DTIDgr5W"	"0CZbfj7uz0LfSTXWMcvPAG"
"68"	"0pEWkNLhjE3xM3DTIDgrRe"	"0CZbfj7uz0LfSTXWMcvPB7"
"62"	"0pEWkNLhjE3xM3DTIDgrTz"	"0CZbfj7uz0LfSTXWMcvPBv"

Task B: downstream device reachability for impact analysis In FDD, the identification of downstream dependencies is crucial for impact analysis, as it allows to determine which devices and spaces are affected by a fault in a given component. This example retrieves the rooms downstream of a given ahu heating coil and returns the corresponding room-level temperature sensor identifiers. Such a query is a common prerequisite for air-side FDD, as it delineates the impacted set for an AHU-side fault (e.g., a stuck heating coil valve or insufficient heating power) and identifies the room sensors required to assess the resulting effects.

Listing 3 shows the query formulated in `ifcOWL+`. Using the `ifcOWL+:IsConnectedToByDuct*` property path, the query traverses the inferred device-level topology from the selected coil to all downstream air terminals, and subsequently resolves their locations and any associated temperature sensors. Here, the asterisk operator denotes transitive reachability (zero or more hops) over the connectivity graph. Applied to the KG, the query identifies 48 rooms supplied by the specified heating coil and returns the corresponding temperature sensor identifiers for each room.

```
SELECT DISTINCT ?spaceName ?tempId
WHERE {
    ?coil a ifc:IfcCoil ;
        ifc:GlobalId "1izbNUApXAqP4X3VNz0DYj" ;
        ifcOWL+:IsConnectedTo* ?airTerminal .
    ?airTerminal a ifc:IfcAirTerminal ;
        ifcOWL+:HasLocation ?space .
    ?space a ifc:IfcSpace ;
        ifc:Name ?spaceName .
    ?tempSens a ifc:IfcSensor ;
        ifc:PredefinedType "TEMPERATURESENSOR" ;
        ifcOWL+:HasLocation ?space ;
        ifc:GlobalId ?tempId .
}
```

Listing 3: Task B with ifcOWL+: device reachability plus direct spatial lookup.

The ifcOWL+ query uses N=11 triple patterns. In plain ifcOWL, even a *single* directed hop between two distribution elements (element → port → connection → port → element) requires N_{hop}=14 triple patterns. An explicit k-hop downstream traversal therefore scales as $N \approx 12 + 14k$ triple patterns. Additionally, the ifcOWL version requires detailed knowledge of the port and connection structure, while the ifcOWL+ version directly expresses the intended reachability query.

Task C: upstream device reachability for root cause candidate selection In FDD, the identification of upstream dependencies is crucial for root cause candidate selection, as it allows for the determination of which components supply a specific device. By isolating the specific branch of the distribution network serving a failing component, the search space for potential faults is significantly reduced.

In this example the air temperature in room “42” is persistently below the set-point, indicating a potential fault in the heating system. To identify the root cause, the query retrieves all devices functionally upstream of the radiator in room 42 that could influence its operation. Listing 4 shows the query to traverse the connection from the radiator to the boiler using the `ifcOWL+:IsConnectedFromByPipe` property. First, the query identifies

the source radiator in room 42 and the destination boiler using the ifcOWL+:HasLocation property. As native SPARQL does not support path queries we leverage GraphDB's Paths service to compute the path between the symptomatic device and the heat source. The path computation is delegated to SERVICE path:search, which traverses through the ifcOWL+:IsConnectedFromByPipe property.

```
SELECT ?path ?index ?end ?name ?type
WHERE {
    ?dst a ifc:IfcBoiler .
    ?src a ifc:IfcSpaceHeater ;
        ifc:PredefinedType "RADIATOR" ;
        ifcOWL+:HasLocation ?space .
    ?space a ifc:IfcSpace ;
        ifc:Name "42" .
    SERVICE path:search {
        <urn:path>
        path:findPath path:shortestPath ;      path:sourceNode ?src ;
        path:destinationNode ?dst ;               path:pathIndex ?path ;
        path:resultBindingIndex ?index ;         path:startNode ?start ;
        path:endNode ?end .
        SERVICE <urn:path> {
            ?start ifcOWL+:IsConnectedFromByPipe ?end .
        }
    }
    OPTIONAL {?start ifc:Name ?name . }
    OPTIONAL {?start ifc:PredefinedType ?type}
}
ORDER BY ?path ?index
```

Listing 4: Task C with ifcOWL+ and GraphDB Paths: shortest-path search on the inferred device-only graph with direct localization.

The query result, shown in Table 3, reveals an eight node dependency chain. This approach identifies critical actuators such as the three way mixing valve at Index 2 and the circulation pump at Index 3. By providing a transparent and ordered list of upstream dependencies, the enriched graph replaces manual engineering drawing reviews with a structured and prioritized diagnostic checklist for physical inspection. While this specific query identifies the supply path, the return path can also be analysed by replacing ifcOWL+:IsConnected FromByPipe with ifcOWL+:IsConnectedToByPipe in the query.

Table 3. *Task C result: shortest path from boiler to a radiator located in room "42".*

Path	Index	Id	Name	Type
0	0	#260554	"Boiler 12 kW (Natural Gas)"	"WATER"
0	1	#260164	"Check valve (flanged, std.)"	"CHECK"
0	2	#260263	"3-way motorized ball valve 3/4 230V"	"MIXING"
0	3	#241506	"Circulation pump DN25 (threaded)"	"CIRCULATOR"
0	4	#151126	"Pipe temp. sensor DN32 G1Ï/4"	"TEMPERATURESENSOR"
0	5	#115282	"Globe valve DN32 (threaded)"	"SAFETYCUTOFF"
0	6	#117320	"Check valve (flanged, std.)"	"CHECK"
0	7	#117142	"Globe valve DN15 (threaded)"	"REGULATING"

3.2. Operational use case: automated fault diagnosis

This section demonstrates the practical application of the ifcOWL+ for a diagnostic workflow. While these relations support fully automated FDD, they also significantly lower the technical barrier for facility management to perform manual ad-hoc queries. The operational data used in this scenario are generated from a Modelica-based simulation with injected faults.

In our scenario, a persistent comfort violation in room “42” is detected on the 4th January. The measured air temperature has declined steadily over a 24-hour period and is deviating from the setpoint as shown in Figure 6. The diagnosis proceeds through the following structured steps:

1. **Contextualization and Asset Identification:** By utilizing `ifcOWL+:HasLocation`, the symptomatic sensor is mapped to its related `ifcOWL:IfcSpace`. From this space entity, the planned minimal room air temperature setpoint is retrieved from the `Pset_SpaceHVACDesign` property set via `ifcOWL+:Has PropertySet` to confirm the magnitude of the thermal deviation. Simultaneously, following the logic demonstrated in Task A, the suspect heating devices are identified by querying for radiators located within that same spatial context.
2. **Upstream Dependency Traversal:** To narrow down the root cause, the hydronic distribution chain is then analyzed to identify components that influence the radiator. As shown in Task C the property `ifcOWL+:IsConnectedFromByPipe` allows for a direct trace of the supply path back to the heat source. This replaces the manual review of 2D engineering drawings with a structured, prioritized checklist of intermediate mixing valves and pumps for inspection. The resulting dependency for room “42” is shown in Table 3.
3. **Cross-Validation and Diagnosis:** With this knowledge of the relevant equipment the corresponding operational data points are derived. Either by manual inspection or by directly linking the data points to the equipment in the KG. Figure 6 shows the time series of the supply temperature, pump status and valve position for the identified components. While the boiler and pumps are confirmed to be operational, the mixing valve from the secondary heating cycle is stuck at approx. 65% open. This leads to a deviation of the primary supply temperature from the set-point and therefore insufficient heating power at the radiator, which explains the observed comfort violation.

This focused localization significantly reduces the need for manual interpretation of complex engineering drawings. By directing maintenance efforts to the most probable physical failure point through a structured dependency chain, the ifcOWL+ ontology facilitates a more efficient and accessible approach to building energy management.

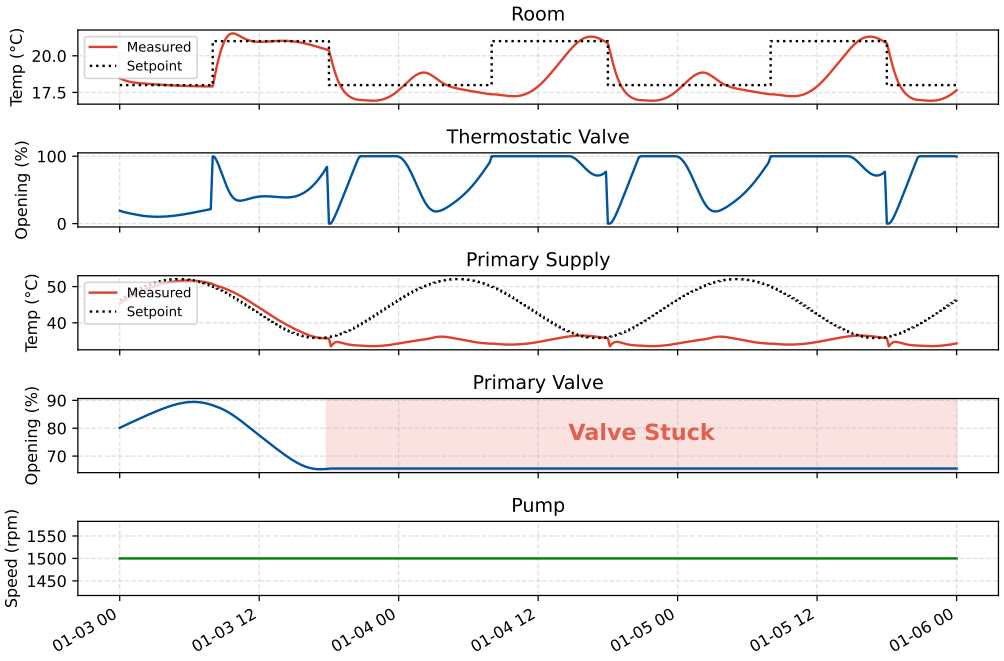

Figure 6. *Time series of supply temperature, pump status, valve command and valve position for the identified components.*

4. Discussion and outlook

The ifcOWL+ ontology addresses the inherent complexity and nested nature of standard ifcOWL representations, which hinder the use for applications in BA. By introducing direct shortcut relations and functional abstractions, we have demonstrated a significant reduction in the complexity of SPARQL queries required for typical FDD tasks. The case study illustrates that these semantic enrichments allow both automated systems and facility managers to traverse building systems based on functional logic rather than being encumbered by granular geometric and carrier-network data. Another advantage of the proposed approach is its non-

destructive nature. As an extension, ifcOWL+ preserves all original IFC data while adding a layer of high-level, machine-understandable relationships. Furthermore, the use of SHACL-SPARQL rules for inference ensures that the logic is explicit and transferable across different graph processing environments.

However, some limitations remain. The current workflow relies on the initial quality and detail of the BIM export. First, incomplete or inconsistent IFC models, specifically regarding port connectivity and flow direction attributes can result in incomplete inferred graphs. Second, the relation of devices to datapoints is not yet standardized in IFC, which limits the direct applicability. Third, as ifcOWL+ is an extension to ifcOWL it adds triples to the KG, which can lead to increased storage and reasoning time.

Future work will focus on developing robust fallback rules to infer connectivity in models with missing or poorly defined port data. Additionally, we aim to expand the ontology to include datapoint-to-device relations, for example by aligning with the Brick or BACnet ontologies, to support direct live operations and real-time data integration. Finally, we intend to investigate the computational performance of SHACL materialization on extremely large-scale building portfolios to assess the scalability of the approach beyond the current case study.

Acknowledgments

The authors thank their colleagues at the Institute for Energy Efficient Buildings and Indoor Climate for valuable feedback during the development and evaluation of this work. This work was supported by the Federal Ministry for Economic Affairs and Climate Action (BMWK), promotional reference 03EN1075A (STIFT - Standardisation of interfaces and functionalities for Internet of Things (IoT)-based building automation).

References

[1] United Nations Environment Programme and Global Alliance for Buildings and Construction. *Not just another brick in the wall: The solutions exist - Scaling them will build on progress and cut emissions fast. Global Status Report for Buildings and Construction 2024/2025*. 2025. URL: https://wedocs.unep.org/handle/20.500.11822/47214;jsessionid=AB7326F86C878449897D5D0473288953.

[2] E. Saloux, K. Zhang, and J. A. Candanedo. A Critical Perspective on Current Research Trends in Building Operation: Pressing Challenges and Promising Opportunities. In: *Buildings* 13.10 (2023). ISSN: 2075-5309. DOI: 10.3390/buildings13102566. URL: https://www.mdpi.com/2075-5309/13/10/2566.

[3] W. Kim and S. Katipamula. A review of fault detection and diagnostics methods for building systems. In: *Science and Technology for the Built Environment* 24.1 (2018). ISSN: 2374-4731. DOI: 10.1080/23744731.2017.1318008.

[4] H. Dibowski, J. Ploennigs, and K. Kabitzsch. Automated Design of Building Automation Systems. In: *IEEE Transactions on Industrial Electronics* 57.11 (2010). ISSN: 0278-0046. DOI: 10.1109/TIE.2009.2032209.

[5] S. Runde and A. Fay. Software Support for Building Automation Requirements Engineering—An Application of Semantic Web Technologies in Automation. In: *IEEE Transactions on Industrial Informatics* 7.4 (2011). ISSN: 1551-3203. DOI: 10.1109/TII.2011.2166784.

[6] X. Cai, Z. Jin, H. Li, A. Kümpel, and D. Müller. Automated PLC Code Generation for the Implementation of Mode-Based Control Algorithms in Buildings. In: *Buildings* 14.1 (2024). ISSN: 2075-5309. DOI: 10.3390/buildings14010073. URL: https://www.mdpi.com/2075-5309/14/1/73.

[7] S. Azhar. Building Information Modeling (BIM): Trends, Benefits, Risks, and Challenges for the AEC Industry. In: *Leadership and Management in Engineering* 11.3 (2011). ISSN: 1532-6748. DOI: 10.1061/(ASCE)LM.1943-5630.0000127. URL: https://ascelibrary.org/doi/10.1061/%28ASCE%29LM.1943-5630.0000127.

[8] R. Vieira, P. Carreira, P. Domingues, and A. A. Costa. Supporting building automation systems in BIM/IFC: reviewing the existing information gap. In: *Engineering, Construction and Architectural Management* 27.6 (2020). ISSN: 0969-9988. DOI: 10.1108/ECAM-07-2018-0294. URL: https://www.emerald.com/ecam/article/27/6/1357/93436/Supporting-building-automation-systems-in-BIM-IFC.

[9] P. Pauwels, S. Zhang, and Y.-C. Lee. Semantic web technologies in AEC industry: A literature overview. In: *Automation in Construction* 73 (2017). ISSN: 0926-5805. DOI: 10.1016/j.autcon.2016.10.003. URL: https://www.sciencedirect.com/science/article/pii/S0926580516302928.

[10] G. Fierro, A. Saha, T. Shapinsky, M. Steen, and H. Eslinger. Application-driven creation of building metadata models with semantic sufficiency. In: *Proceedings of the 9th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*. Ed. by J. Ortiz. New York, NY, USA: ACM, 2022. ISBN: 9781450398909. DOI: 10.1145/3563357.3564083.

[11] B. Balaji, A. Bhattacharya, G. Fierro, J. Gao, J. Gluck, D. Hong, A. Johansen, J. Koh, J. Ploennigs, Y. Agarwal, M. Bergés, D. Culler, R. K. Gupta, M. B. Kjærgaard, M. Srivastava, and K. Whitehouse. Brick : Metadata schema for portable smart building applications. In: *Applied Energy* 226 (2018). ISSN: 0306-2619. DOI: 10.1016/j.apenergy.2018.02.091.

[12] J. Beetz, J. van Leeuwen, and B. de Vries. IfcOWL: A case of transforming EXPRESS schemas into ontologies. In: *AI EDAM* 23.1 (2009). ISSN: 0890-0604. DOI: 10.1017/S0890060409000122.

[13] P. Pauwels and D. van Deursen. IFC-to-RDF: Adaptation, Aggregation and Enrichment. In: *First International Workshop on Linked Data in Architecture and Construction (LDAC)*. Ghent, Belgium, 2012. URL: http://multimedialab.elis.ugent.be/ldac2012/documents/LDACworkshopreport.pdf.

[14] P. Pauwels and W. Terkaj. EXPRESS to OWL for construction industry: Towards a recommendable and usable ifcOWL ontology. In: *Automation in Construction* 63 (2016). ISSN: 0926-5805. DOI: 10.1016/j.autcon.2015.12.003. URL: https://www.sciencedirect.com/science/article/pii/S0926580515002435.

[15] P. Pauwels and A. Roxin. SimpleBIM : from full ifcOWL graphs to simplified building graphs. In: *eWork and ebusiness in architecture, engineering and construction*. Ed. by S. E. Christodoulou and R. J. Scherer. Boca Raton: CRC Press Taylor & Francis Group, 2016. ISBN: 978-1-138-03280-4. URL: https://biblio.ugent.be/publication/8041826.

[16] P. Pauwels, T. Krijnen, W. Terkaj, and J. Beetz. Enhancing the ifcOWL ontology with an alternative representation for geometric data. In: *Automation in Construction* 80 (2017). ISSN: 0926-5805. DOI: 10.1016/j.autcon.2017.03.001. URL: https://www.sciencedirect.com/science/article/pii/S0926580517301826.