Modular Knowledge integration for Smart Building Digital Twins

Isaac **Fatokun**[1], Arun Raveendran Nair **Sheela**[1], Thamer **Mecharnia**[2],
Maxime **Lefrançois**[2,*], Victor **Charpenay**[2], Fabien **Badeig**[2] and Antoine **Zimmermann**[2]

[1]*Mines Saint-Etienne, Institut Henri Fayol, F - 42023 Saint-Etienne France*

[2]*Mines Saint-Etienne, Univ Clermont Auvergne, INP Clermont Auvergne, CNRS, UMR 6158 LIMOS, F - 42023 Saint-Etienne France*

Abstract

It is accepted in the Linked Data for Architecture and Construction (LDAC) community that generating knowledge graphs (KGs) from the BIM model of a building enables higher level use cases such as integration with geographic information systems, operational system integration, semantic digital twins (DTs), or automatic compliance checking. However, existing approaches generate a large, monolithic knowledge graph that is difficult to integrate with other knowledge such as Thing Descriptions (TDs) of Internet of Things (IoT) devices, or information about office occupants and room occupancy schedules. In this work, we describe a set of three modular knowledge graphs that enable knowledge integration for the semantic DT of our building at Mines Saint-Étienne, leveraging the principles of Linked Building Data: (1) KG_{LBD} is automatically generated from the Revit model of our building, (2) KG_{FOAF} is semi-automatically generated from the employee directory of Mines Saint-Étienne, and (3) KG_{TD} is automatically generated from the ETS5 project file describing the KNX network in our building using the W3C TD ontology, and points to real-time and historical data. Our approach offers an alternative with respect to the state of the art such that: (1) relevant bits of the building's KG can be accessed using a simple REST-like interface, where each small KG contains links to other entities that themselves are identified by an IRI and have a small KG accessible; (2) Knowledge potentially served by different servers can be integrated in the same solution; (3) simple access control can be implemented for some parts of the global KG.

Keywords

BIM, IFC, Digital Twin, Linked Data, Knowledge Graphs, Modular Design, KNX

1. Introduction

The digitization in the AECOO (Architecture, Engineering, Construction, Owner, and Operation) industry grounds on the development of Building Information Modeling (BIM) and the Internet of Things (IoT). Linking the building's digital model with data about smart devices and intelligent applications is fast becoming indispensable [1]. As a result, BIM professionals have had to embrace various methods, one of which is the use of Digital Twins (DTs) [2]. Although there is a misconception that digitized 3D models are DTs, the concept of a DT transcends the basic idea of a computer-aided 3D model [3]. For a model to be considered a DT, there must be full integration of data between the physical object and the virtual model in both directions [3]. Thus, in the case of Building DTs, a change in the building could reflect in its DT and vice versa, all in real-time.

In practice, however, modeling changes that occur over time in building design is complex as one must define what elements are the same (with some changes) or completely new [4]. In addition, with DTs, one must be able to inject data specific to a device (such as a heater) or property (such as a room temperature) from IoT devices and other sources into the BIM model. The problem with this is that although BIM models are very useful, they were originally designed to provide data only about the building in the design and construction phase, not necessarily in the operation phase [5]. As such, there is no standard approach to use them in dynamic situations as needed in a DT. Moreover, BIM data is largely fragmented depending on the source of the data [6], thus to add data, it must be in a manner that allows this data to be associated with the right domain and to be managed independently of the overall system if needed. As a result, the use of Semantic Web technologies to aid this integration is becoming popular. With the use of Knowledge Graphs (KGs), BIM models can become more useful by integrating BIM data in KGs, or linking BIM data to KGs thus providing better contextual information [7].

Even so, integrating BIM data and IoT data poses a new set of challenges due to the monolithic nature of most KGs. For one, the existing reasoning engines are unable to effectively process large-scale KGs because they load and compute them as a whole, as demonstrated by the authors of [8]. Modular design, or modularity in design, refers to a design paradigm where components work through a combination of distinct building blocks or modules to achieve an overall objective or function [9]. For the knowledge engineering domain, d'Aquin [10] describes modularity as *a way of designing graphs to support maintenance and re-usability by the combination of self-contained, independent, and reusable knowledge components.*

This necessitates the need for ways to use subsets of KGs or more simply the use of modular design in knowledge engineering [11]. To this end, this work presents a framework to integrate BIM with data from external sources such as IoT devices, and manually curated knowledge in a modular manner. Backed by Linked Data (LD) principles, this work is motivated by a vision of a building of the future where all information is encoded such that information can be gotten from the IFC model of the building, sensors, and actuators stationed at various parts of the building, and other sources, such that every single element in the building is interconnected to enable faster data exchange and visual monitoring of user comfort parameters as well as room occupancy in real-time.

In this paper we propose and discuss an approach that is based on a large set of small, linked KGs. Our primary research problem centers on how to integrate BIM models in the IFC format with data from external sources such as IoT device configurations and live IoT data, and manually curated KGs, in a modular manner, improving overall system performance in DTs. We focus on the following research question:

Could modular graphs facilitate knowledge integration in the semantic DT of a Smart Building?

We demonstrate the benefits of our approach on the semantic DT of our smart building at Mines Saint-Étienne. The rest of this work is structured as follows. Section 2 recalls some background on the linked data principles, modular design, the Building Topology Ontology (BOT), and the WoT Thing Descriptions (TDs). Section 3 presents some related work, then Section 4 describes the Espace Fauriel building of Mines Saint-Étienne. Section 5 presents our

proposed approach, preliminary results, and a demonstration use case. Section 6 discusses the benefits of our approach, and Section 7 concludes the paper.

2. Background

The modern idea of linked data is based on the linked data principles presented by Tim Berners-Lee in a design issue note:[1]

1. Use URIs as names for things,
2. use HTTP URIs so that people can look up those names,
3. When someone looks up a URI, provide useful information, using the standards (RDF, SPARQL),
4. Include links to other URIs, so that they can discover more things.

The W3C Linked Building Data community group (LBC-CG) brings together experts in the area of BIM and Web of Data technologies, who are working to address the challenge of managing the huge amount of data that is generated across the building life cycle. They propose to adapt the linked data principles for building-related data as follows:

1. Using URIs as names for building-related things such as: rooms, walls, products, elements, enable different parties to provide complementary descriptions of the same uniquely identified entities in different knowledge graphs.
2. Using HTTP URIs for these things enables the authority responsible for these URIs to provide reference information about this entity when one looks it up on the Web. For example, products in a catalogue, building appliances, or even the building itself.
3. Providing information using common standards (RDF, SPARQL) and common KG models enable semantic interoperability between data sources.
4. Include links to other URIs can help to discover more things, such as a link to the catalogue an appliance has been chosen from, or a link to the Web of Things Servient that enables it to interact with this appliance.

This group's flagship result is the Building Topology Ontology (BOT) [12], an ontology that provides a high-level semantic description of the topology of buildings including storeys, spaces, and the building elements they contain.

The W3C Web of Things (WoT) Working Group seeks to counter the fragmentation of the IoT through standard complementing building blocks (e.g., metadata and APIs) that enable easy integration across IoT platforms and application domains.[2] TDs [13] is the main building block of the W3C's WoT architecture introduced to solve the interoperability problem in different IoT devices. The TD of an IoT device contains four main components: 1. some metadata about the device itself, 2. descriptions of interaction capabilities for the device including Properties, Actions, and Events, 3. web forms to enact these interaction capabilities, and 4. relationships to other devices.

---

[1]https://www.w3.org/DesignIssues/LinkedData.html
[2]Source: https://www.w3.org/groups/wg/wot

3. Related Work

In this section, we present a brief review of works related to the conversion of IFC models to KGs and integration with IoT data.

3.1. IFC to KGs

Although there aren't many ontologies that adequately cover the scope of IFC, ifcOWL [14, 15] is a game changer. Introduced in 2009 by Beetz et. al., ifcOWL is a direct mapping of IFC express schema into OWL thus, it has the same status as IFC's EXPRESS and XSD schemas. A downside however is that the graphs produced by the ifcOWL mapping are complex in terms of structure and size [15]. [11] provided a first attempt to modularize an ifcOWL KG. Alternative approaches such as SimpleBIM lighten ifcOWL graphs by pruning everything related to the geometry [16].

The IFC-to-LBD converter [17, 18] generates a lighter KG from an IFC file using the ontologies from the LBD-CG group. After conversion, important information about the building such as topology, elements classification e.g Wall, Space, etc, and their properties are saved in one to three large graphs. When compared to the output of the ifcOWL-based IFC-to-RDF converter [19], the graph structure becomes more concise (minimum 83% fewer triples) and substantially simpler to query.

3.2. IoT + BIM

Since IoT devices are fundamental in the Operation phase of a Smart Building lifecycle, there have been a number of studies centered around the integration of IoT data with BIM information. Tang et. al. [20] reviewed several methods of integrating building contextual data and time-series data generated in the IoT layer. Their work showed that an efficient approach would be to represent the building contextual data and sensor information as an RDF graph and also serialize time-series data from IoT devices into another RDF graph. The final step of this approach is to then link both graphs using unique identifications. The architecture of this method is shown in Figure 1.

Based on the knowledge from [20], the same researchers went on to develop a hybrid approach to facilitate information exchange between BIM-based building data, time-series data generated from the IoT devices (in a relational database), and Building Automation System (BAS) metadata using Semantic Web Technology [21]. As mentioned earlier, this work made use of the conversion mechanism identified in [20]. How-
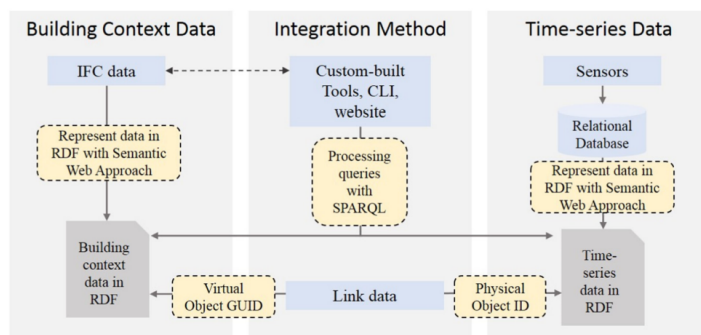

Figure 1: Integration of time-series data and building contextual data (Source: [20])

4

ever, to avoid the massive data and high computational power required to convert IoT time series data to RDF representation, the researchers used online URLs to access IoT data directly from the relational database (thus the *hybrid* approach). In 2020, Markus et. al. [22] proposed another framework to implement a novel DT for smart homes and buildings based on WoT TDs. This framework contains three components: 1. a converter (which converted device data to WoT format), 2. a triple-store to store the generated knowledge graphs, 3. and a web service to publish the data via REST API for the DT. The Architecture of the proposed framework is shown in Figure 2.

Donkers et. al. [23] presented a method for occupant-based feedback gathering and integration with building information. Their DT integrates BIM data, sensor data, and occupant feedback using the Occupant Feedback Ontology implemented in a smartwatch app. The system was evaluated and validated using the case study of an apartment building. Although their approach proved effective in collecting occupant feedback as intended, it made use of a monolithic KG.
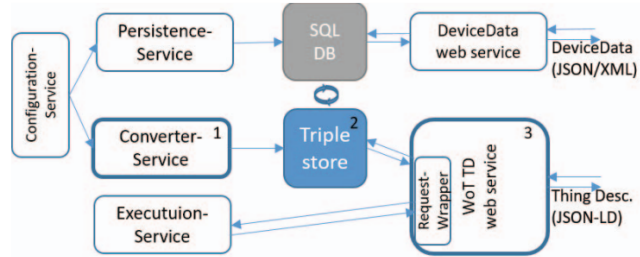

Figure 2: Workflow of DT using WoT TDs (Source: [22])

Similarly, to improve indoor condition monitoring, Desogus et. al. [24] presented a data platform for the visualization of building indoor conditions (e.g., temperature, luminance, etc.) and energy consumption parameters. It was tested using a case study in Italy. Their approach integrate the Revit software with the Dynamo visual programming platform and a specific API.

Cimmino et. al. [25] introduced eWoT, an approach that enables transparent interaction with an IoT ecosystem made up of Web-accessible heterogeneous IoT devices through SPARQL queries. To profile the various IoT devices, the technique uses TDs. However, TDs could not handle the expressivity necessary to handle the variety of IoT devices. Thus, they also presented an extension of TDs called *WoT-Mappings*. The TDs and WoT-Mappings were then stored in a single store to be used by the eWoT system to perform both context-based and content-based device discovery every time a SPARQL query is sent. Again, it is clear that every time data was needed for a particular device, the system had to query the entire triple store.

In order to support real-time data integration and dynamic decision-making in Asset Management (AM) applications, Moretti et. al. [26] presented an openBIM methodology that makes use of the IFCSHAREDFACILITIESELEMENTS schema to process and integrate geometric and semantic information of existing and newly created IFC objects. The approach incorporates data from the BIM environment, IoT platform, and Building Management Systems to support the asset management team's workflows and operations during the asset's use phase. For improved interoperability and cross-platform use, the technique was built utilizing open-source software and IFC. The case study showed how the suggested strategy could support an asset anomaly detection application and boost digital AM automation.

4. The Espace Fauriel Building of Mines Saint-Étienne

The Espace Fauriel (EF) building of Mines Saint-Étienne, built around 1920 by mail-order selling company *Manufrance* and re-empowered in 1994 for the Mines Saint-Étienne higher education institute, is an eight storeys building of 6720 m2 used for research and teaching. It includes lecture halls, classrooms, offices, meeting rooms, and a space filled with industrial robots and workshops called the *IT'M Factory*.[3] The 3D model of the EF building is made available under the CC-by 4.0 license as Revit (137 MB), IFCv4 (186 MB), and Xeokit (12 MB).[4]

Figure 3: Espace Fauriel Building of Mines Saint-Étienne

The Building Management System consists of an historical EIB bus (European Installation Bus, which became KNX ISO-IEC 14543-3) installation for controlling HVAC and alarm systems, and several IoT devices deployed to monitor comfort parameters such as humidity, temperature, luminosity, CO_2, windows opening status. In addition, smart meters are deployed to measure the electric consumption at the level of storeys, and at the level of a few individual equipments. Sensor data is published in real time to an MQTT broker, and a subset of the historical data is made publicly available under the CC-by 4.0 license.[5]

## 5. Interlinked Modular KGs for a Building's DT

Our approach aims to interlink three different KGs: (1) KG_{LBD} is automatically generated from the Revit model of the building, and describes its topology and static properties using the BOT ontology; (2) KG_{FOAF} is semi-automatically generated from Mines Saint-Étienne's employee directory, and uses identifiers from KG_{LBD} to link employees to their office and the desk they work on; (3) KG_{TD} is automatically generated from the ETS5 project file, and describes the configuration of KNX sensors and actuators using the TD ontology. KG_{FOAF} uses identifiers from KG_{LBD}, KG_{LBD} uses identifiers from KG_{TD}, and KG_{TD} describes means to access the real-time and historical IoT data. Figure 4 illustrates the general methodology of our approach, which we detail in the rest of this section. The knowldege graphs are served as static files by an Apache 2 server, and have the following entrypoints: https://ci.mines-stetienne.fr/emse/ for the LBD graph, https://ci.mines-stetienne.fr/knx/ for the KNX configuration graph, https://ci.mines-stetienne.fr/mqtt/ for the historical IoT data as CSV files.

---

[3]IT'M Factory: Virtual visit https://itm-factory.fr/index.php/objectif_et_visite_360/

[4]3D model of the EMSE EF building https://ci.mines-stetienne.fr/EMSE_EF/

[5]Raw data for smart meters and heating devices https://ci.mines-stetienne.fr/mqtt/
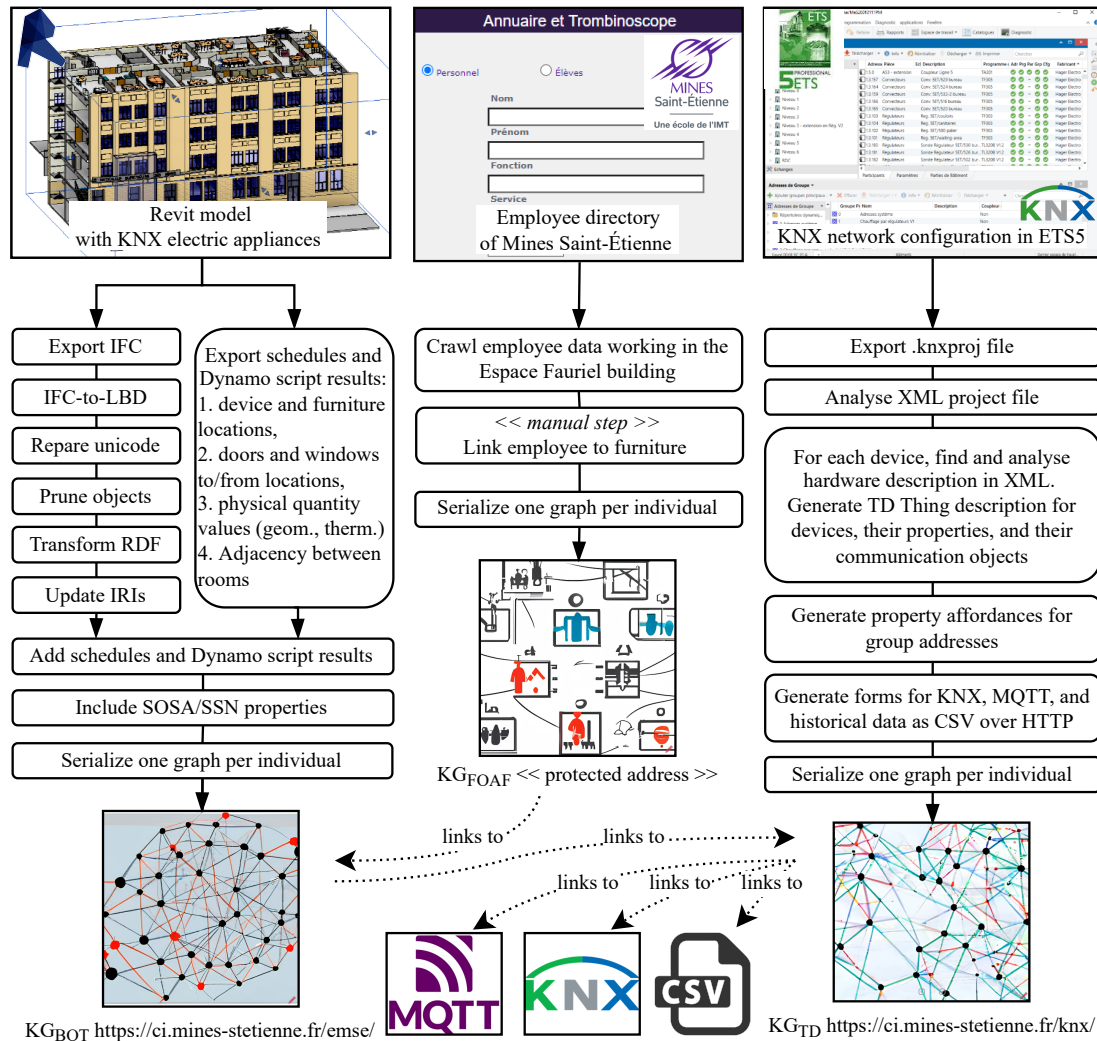
Figure 4: Overview of the methodology to integrate modular knowledge graphs for the EF Building's Digital Twin. (Sources of the Knowledge Graph images: DALL·E 2)

5.1. KG$_{LBD}$ - A Modular LBD KG

To prepare the linking of KG$_{LBD}$ to KG$_{TD}$, a 'KNX Device' family is created in Revit, and elements are positioned in different spaces and are given a KNX address parameter as shown on Figure 5. The Revit model is first exported to IFC, then converted to RDF using the IFC-to-LBD tool.[6] This results in a large 256k triples graph that is unsatisfactory for our use cases, which justifies the following sequence of post-processing steps, implemented as python scripts. The sources of our post-processing scripts and instructions to re-generate KG$_{LBD}$ are openly available.[7]

---

[6]Settings for the conversion are described in https://github.com/maximelefrancois86/databat-kgc-revit/tree/main/rvt
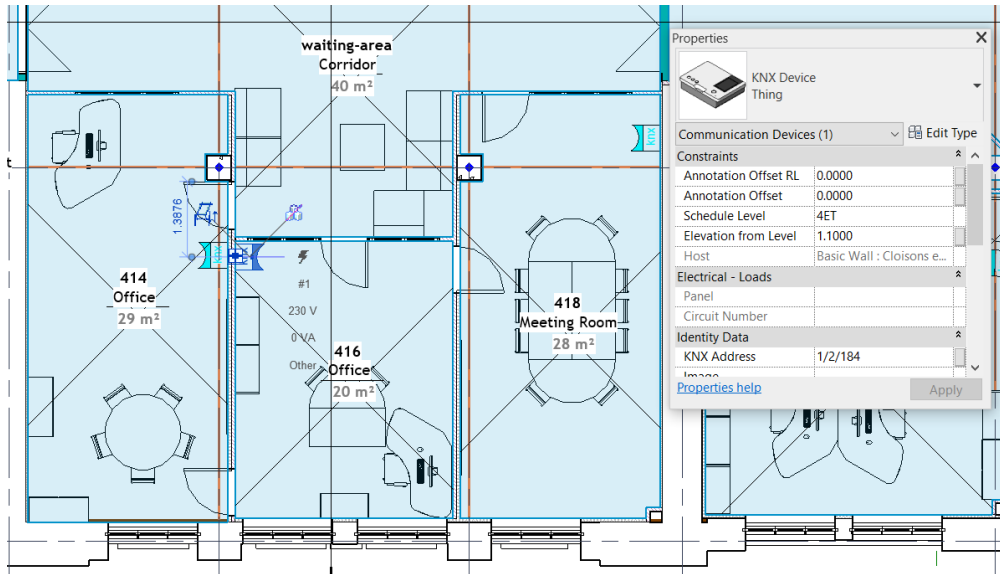[7]https://github.com/maximelefrancois86/databat-kgc-revit

Figure 5: KNX Device objects prepare the linking from the KG$_{LBD}$ to the KG$_{TD}$ knowledge graphs.

1_repareCharacters.py. Wrongly encoded Unicode characters are restored. For example: "\n" was encoded as "\\X\\0D\\X\\0A", "é" was encoded as "\\X\\00E9\\X\\0A".

2_pruneObjects.py. We only need a few type of elements for our use case. This step keeps triples with RDF terms whose IRI contain one of: "site", "building", "storey", "space", "wall", "window", "ifcowl_ifcfurniture", "electricappliance". It prunes triples with RDF terms whose IRI contain one of "airterminal", "beam", "cablecarrierfitting", "cablecarriersegment", "column", "covering", "curtainwall", "ductfitting", "ductsegment", "flowterminal", "ifcowl_ifcopeningelement", "lightfixture", "member", "plate", "railing", "ramp", "roof", "slab", "stair", "stairflight". The total number of triples after this step is 129 k.

3_transformObjects.py. This step re-generates a KG by executing a set of SPARQL construct queries that select a subset of the triples and replace properties. For example the following properties had the same literal objects: props:familyAndType_simple, props:family_simple, props:objectTypeIfcObject_attribute_simple, props:typeId_simple, props:type_simple. Only one is kept and we use property rdfs:label instead. Output contains 28,5 k triples.

4_renameObjects.py. IRIs were all under the same inst: namespace, their local name is made of a type name and a UUID. This step establishes a more natural IRI structure that reflects the organisation of the building. For example <{building}>, <{building}/{storey}>, <{building}/{storey}/{space}>, <{building}/{storey}/{space}/{knx_device_in_space}>.

5_readSchedules.py. This step generates triples from the Revit schedules and the output of Dynamo scripts, which are exported as CSV files. Its goal is to recover important knowledge that was lost somewhere in the Revit-to-IFC-to-LBD pipeline: (i) in which space furnitures and devices are located, (ii) which spaces windows and doors are adjacent to, (iii) adjacency between spaces, (iv) quantity values for properties were floored to the nearest integer, with no unit. The total number of triples after this step is 50 k.

8

Table 1

Number of graphs in KG$_{LBD}$ and example individuals.

Element	#	Example
bot:Site	1	https://ci.mines-stetienne.fr/emse#
bot:Building	1	https://ci.mines-stetienne.fr/emse/fayol#
bot:Storey	9	https://ci.mines-stetienne.fr/emse/fayol/4ET#
bot:Space	215	https://ci.mines-stetienne.fr/emse/fayol/4ET/416#
bot:Door	323	emse/fayol/4ET/door/1VMOlJ3knDeODu7af36Nwh#
:Device	137	emse/fayol/4ET/416/device/1bDMdL0k55X8oOMH5VKzRL#
:Furniture	1386	emse/fayol/4ET/416/BureauD'Angle/1VMOlJ3knDeODu7af36LS_#
:Wall	1729	emse/fayol/4ET/wall/1Qfe3f5n95yfrQ8nllTW3_#
:Window	313	emse/fayol/4ET/window/1VMOlJ3knDeODu7af36GPu#

6 _addProperties.py. This step explicits some SOSA/SSN Properties for spaces and elements, which are useful for some of our use cases. For example each space is associated a temperature, CO2, and humidity property. Each door and window is associated an open/close status. The total number of triples after this step is 53 k.

7_storeTurtleFiles.py. Finally, this step iterates over the resources in the KG, automatically extracts a small number of triples that describe each resource, and serializes these small RDF graphs in Turtle files named after the resources' IRIs.

KG$_{LBD}$ contains a total of 4.414 RDF graphs, with 19,64 triples per graph on average. The total number of triples contained in those graphs is 86.686 while the number of triples in the non-modularized graph is 52.979. Two triples per graph are just metadata, this means that 24.879 triples are duplicates and are useful to help clients navigate in the knowledge graph. The set of static RDF documents are served by an Apache 2 server with the URL of the bot:Site as an entrypoint: https://ci.mines-stetienne.fr/emse#. Listing 1 shows an excerpt of the graph that describes Office 416. Table 1 lists the number of instance (and graphs) for each main class, and gives an example one can access.

```
@base <https://ci.mines-stetienne.fr/> .
@prefix bot: <https://w3id.org/bot#> .
@prefix cdt: <https://w3id.org/cdt/> .
@prefix coswot: <https://w3id.org/coswot/> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix skos: <http://www.w3.org/2004/02/skos/core#> .

<emse/fayol/4ET#> a bot:Storey ; bot:hasSpace <emse/fayol/4ET/416#> .
<emse/fayol/4ET/416#> a bot:Space ;
    rdfs:label "416" ;
    rdfs:comment "Office" ;
    skos:hiddenLabel "2IRRGeGVD4mwnjuD6Rcm6i" ;
    bot:adjacentElement <emse/fayol/4ET/door/1VMOlJ3knDeODu7af36Nwh#>, ... ;
    bot:adjacentZone <emse/fayol/4ET/418#>, <emse/fayol/4ET/waiting-area#>, ... ;
    bot:containsElement <emse/fayol/4ET/416/device/1bDMdL0k55X8oOMH5VKzRL#> , ... ;
    coswot:canWalkTo <emse/fayol/4ET/418#>, <emse/fayol/4ET/waiting-area#>, ... ;
    coswot:hasAreaStableValue "20 m2"^^cdt:ucum ;
    coswot:hasTemperatureProperty <emse/fayol/4ET/416#temperature> ;
    ...
```

```
<emse/fayol/4ET/416/device/1bDMdL0k55X8oOMH5VKzRL#> a bot:Element, coswot:Device .
...
```

Listing 1: KG_{LBD} sample: excerpt of the RDF document available at https://ci.mines-stetienne.fr/emse/fayol/4ET/416. Note 1: the coswot vocabulary is under development. Note 2: base and namespace declarations are shared by listings.

5.2. KG_{FOAF} - Employees and their offices

Knowledge graph KG_{FOAF} is the result of crawling the Mines Saint-Étienne employee directory, generate a small RDF graph for each employee that works in the EF Building, and identify them with obfuscated IRIs for privacy reasons. As an additional manual step, each employee is linked to the desk it works at, using IRIs from KG_{LBD}. At the exception of a few, access to all RDF graphs in KG_{LBD} requires authentication. Table 2 lists the number of instances in KG_{FOAF}. Listing 2 is an example of an openly accessible RDF graph in KG_{FOAF}.

Table 2
Number of graphs in KG_{LBD} and example individuals.

Element	#	Example
foaf:Person	80	*personal data, requires authentication*

```
@base <https://ci.mines-stetienne.fr/> .
@prefix foaf: <http://xmlns.com/foaf/0.1/>.
@prefix org: <http://www.w3.org/ns/org#>.

<foaf/b54952c3-e4a6-423d-b724-ed9392f6f2a1> a foaf:Document ;
  foaf:primaryTopic _:person .

_:person a foaf:Person ;
   foaf:name "Xxx Xxx" ;
   foaf:img <data:image/png;base64, /.....> ;
   foaf:phone <tel:+xxxxxxxxxxx> ;
   foaf:mbox <mailto:xxxxxxxxxxxxxxxxx@emse.fr> ;
   org:memberOf <https://www.imt.fr/>, <https://mines-stetienne.fr/>, <foaf/fayol> ;
   coswot:worksAtDesk <emse/fayol/4ET/416/BureauD'Angle/1VMOlJ3knDeODu7af36LS_#> .
```

Listing 2: KG_{FOAF} sample: RDF document available at https://ci.mines-stetienne.fr/foaf/b54952c3-e4a6-423d-b724-ed9392f6f2a1. Note 1: the coswot vocabulary is under development. Note 2: base and namespace declarations are shared by listings.

5.3. KG_{TD} - KNX Configuration Linked to the BIM Model

Devices in a KNX network are identified with a two bytes address, which defines some topology for the network. The first four bits is the Area, the next four bits is the Line, and the remaining eight bits is the Bus Device. For example the selected KNX Device on Figure 5 in Office 416 has address 1/2/184. Devices have some configurable properties and communication objects

that may be readable and/or writable. Communication objects of devices are linked by so-called group addresses, which are also encoded on two bytes, and are used in read/write KNX messages. Group addresses may be displayed with a 3-level structure 5 bits/3 bits/8 bits, and are how messages are passed from communication objects of sensors to communication objects of actuators. For example group address 2/0/117 links Object O-12 of Device 1/2/185: the output command of a Hager TX320 room thermostat deployed in Office 423, with Object O-140 of Device 1/2/171: one the controllable relays of a Hager TYA608C 8x16A Output module (which controls the heater in Office 423).

The first step to generate KG_{TD} is to export the ETS5 configuration project to a 1 MB .knxproj file, which is an archive that contains multiple XML files, including a 7 K lines XML project file, a 450 lines XML hardware description file, and 42 hardware XML definition files. We then automatically extract from these files relevant bits of information about the KNX network: the KNX topology, devices properties and communication objects, and the list of group addresses.

Devices are modeled as instances of td:Thing, and each property and communication object is modeled as a td:PropertyAffordance. Each property affordance is further described as a js:DataSchema using the JSON Schema in RDF vocabulary.[8] Group addresses are also modeled as td:PropertyAffordance and are where td:Forms are defined: Each form describes how a certain operation can be made to interact with the property. These forms are shared by different property affordances using a property coswot:sharesFormsWith we introduced.

```
coswot:sharesFormsWith a owl:SymmetricProperty, owl:TransitiveProperty .
td:hasForm owl:propertyChainAxiom ( coswot:sharesFormsWith td:hasForm ) .
```

Listing 3: Axioms on coswot:sharesFormsWith.

There are three kinds of forms currently defined in KG_{TD}: 1. KNX forms use a proposed **knxip://** URL scheme for KNX/IP addresses, where the host and port are those of the KNX/IP gateway service, and the path defines the group address to read or write. They also use a proposed coswot:KNXMethodName property we introduced to define if the address needs to be read or written to. 2. MQTT forms use the MQTT vocabulary for RDF (mqv), which is under development.[9] 3. HTTP forms to access the historical data as CSV files. As these files have a different schema (an array of objects with value and timestamp properties), we choose to define separate property affordances for them. The sources of our post-processing scripts and instructions to re-generate KG_{LBD} are openly available.[10] Listing 4 shows how the current and historical temperature values of Office 416 are described.

```
@base <https://ci.mines-stetienne.fr/knx/emse/fayol/> .
@prefix schema: <https://schema.org/> .
@prefix td: <https://www.w3.org/2019/wot/td#> .
@prefix hctl: <https://www.w3.org/2019/wot/hypermedia#> .
@prefix mqv: <https://www.w3.org/2019/wot/mqtt#> .
@prefix htv: <http://www.w3.org/2011/http#> .
@prefix js: <https://www.w3.org/2019/wot/json-schema#> .

<1/2/184> a coswot:Device, td:Thing ;
```

---

[8]JSON Schema in RDF vocabulary https://www.w3.org/2019/wot/json-schema
[9]MQTT vocabulary for RDF - https://www.w3.org/TR/wot-binding-templates/
[10]https://github.com/maximelefrancois86/databat-kgc-knxproj

```
        rdfs:label "Sonde Reg. 4ET/416 salle" ;
        skos:hiddenLabel "1/2/184" ;
        schema:description "TX320 Thermostat" ;
        schema:model "TL320B V1.2" ;
        td:hasPropertyAffordance <1/2/184#O-10>, <1/2/184#O-10-history>, ...
        ...

<1/2/184#O-10> a js:NumberSchema, td:PropertyAffordance ;
        rdfs:label "Indication d'état" ;
        coswot:objectSize "2 Bytes" ;
        coswot:sharesFormsWith <ga/3/0/27> ;
        td:description "Température ambiante" ;
        td:title "O-10" .

<ga/3/0/27> a coswot:KNXGroupAddress, td:PropertyAffordance ;
        rdfs:label "T° amb secrétariat N4" ;
        skos:hiddenLabel "3/0/27" ;
        coswot:sharesFormsWith <1/2/184#O-10> ;
        td:hasForm [ hctl:forContentType "text/plain" ;
                hctl:hasOperationType td:observeProperty ;
                hctl:hasTarget <mqtt://193.49.165.40:1883> ;
                mqv:controlPacket "subscribe" ;
                mqv:filter "emse/fayol/4ET/416/temperature/current" ],
             [ coswot:KNXMethodName "read" ;
                hctl:hasOperationType td:readProperty ;
                hctl:hasTarget <knxip://195.83.140.67:3761/1/8/27> ] .

<ga/3/0/27#history> a js:ArraySchema, td:PropertyAffordance ;
        coswot:sharesFormsWith <1/2/184#O-10-history> ;
        js:items [ a js:ObjectSchema ;
                js:properties [ a js:NumberSchema ; js:propertyName "value" ],
                              [ a js:StringSchema ; js:propertyName "timestamp" ] ] ;
        td:hasForm [ htv:methodName "GET" ;
                hctl:forContentType "text/csv" ;
                hctl:hasOperationType td:readProperty ;
                hctl:hasTarget <https://ci.mines-stetienne.fr/mqtt/emse/fayol/4ET/416/temperature/
                    current/log.csv> ] .
```

Listing 4: Current and historical temperature affordances for Office 416. Note 1: the coswot vocabulary is under development. Note 2: base and namespace declarations are shared by listings.

The generated RDF graph contains a total of 40.721 triples, which are then distributed in 1.043 graphs with an average of 53,3 triples per graph, totalling 55.556 triples (36% redundancy). Static RDF documents in KG_{LBD} are served by an Apache 2 server, with entrypoint https://ci.mines-stetienne.fr/knx. Listings 5 shows an excerpt of how the KNX device 1/2/184 is linked to from KG_{LBD}.

```
<emse/fayol/4ET/416#> bot:containsElement <emse/fayol/4ET/416/device/1bDMdL0k55X8oOMH5VKzRL#> .

<emse/fayol/4ET/416/device/1bDMdL0k55X8oOMH5VKzRL#> a bot:Element,
        coswot:Device ;
    rdfs:label "KNX Device:Thing" ;
    owl:sameAs <knx/emse/fayol/1/2/184> ;
```

```
skos:hiddenLabel "1bDMdL0k55X8oOMH5VKzRL" ;
coswot:KNXAddress "1/2/184" ;
coswot:category "Communication Devices"  .
```

Listing 5: Graph emse/fayol/4ET/416/device/1bDMdL0k55X8oOMH5VKzRL# links the KNX
Device in Office 416 to TD Thing knx/emse/fayol/1/2/184.

Table 1 lists the number of instance (and graphs) for each main class, and gives an example one can access. The MQTT broker (Mosquitto) and the KNX/IP gateway are only available when connected to the building's network. The CSV logs are openly available under the CC-by 4.0 license.[11] These logs include the history of the outside temperature, offices temperatures, room thermostat setpoint commands, and room thermostat acknowledgement messages. Note that there exists a proposal for a KNX ontology http://schema.knx.org/2020/ontology/, however in this work we wanted to abstract from KNX and therefore rather used the W3C TD ontology.

Table 3
Number of graphs in KG_{TD} and example individuals.

Element	#	Example
td:Thing	4539	https://ci.mines-stetienne.fr/knx/emse/fayol/1/2/159
td:PropertyAffordance	215	https://ci.mines-stetienne.fr/knx/emse/fayol/3/0/20#history

5.4. Demonstration Use Case: Comfort parameter monitoring

To illustrate the combined use of the three KGs, this section describe a scenario *comfort parameter monitoring*. Preliminary demonstrations of the Territoire Databat scenario are available online[12].

An employee of Mines Saint-Étienne, Marie-Line, works in room 416 of the EF building. She logs in the Territoire Databat platform, and the camera point of view directly jumps to the center of her desk. She can then navigate in the 3D model using the arrow keys, step in front of the box that represents the KNX Device next to her office door, click on it, and visualize the current and historical state of the room thermostat. As she is registered as the user of this office, she can select a different setpoint, or request a change in the temperature setpoint.

In practice, this scenario only requires the Territoire Databat server to enable authentication, and after a successful login send to Marie-Line's browser the URL of the graph G_{FOAF1} that describes her in KG_{FOAF}. The client may then download and parse G_{FOAF1}. It can then dereference the object of the coswot:worksAtDesk property, and obtain graph G_{LBD1} with the IfcGUID of the furniture, and a link to the office it is an element of. The client can use the IfcGUID to determine the XYZ coordinates of the center of the desk. When Marie-Line clicks on the KNX Device on the wall, the client can dereference its URL, a) check it is located in Marie-Line's office, and b) fetch the graph G_{TD1} in KG_{TD} that describes the device. From G_{TD1}, it can find more graphs and ultimately obtain the forms it can use to visualize the history of the temperature in the room and the thermostat setpoints.

---

[11]Raw data for smart meters and heating devices https://ci.mines-stetienne.fr/mqtt/
[12]Demo: https://ci.mines-stetienne.fr/ldac2023/demo/demo1.mp4 - https://ci.mines-stetienne.fr/ldac2023/demo/demo2.mp4

6. Discussion

Although other studies had worked on BIM modeling using semantic web approaches, we propose to use modular knowledge graphs which we believe has several benefits compared to using monolithic graph architectures. Modular graphs allow merging independently developed KGs into a single one that is easier-to-understand, better to reason with, and also reusable as demonstrated in [27]. In addition, when integrated with real-life systems, modular graphs improve performance by loading only the needed segments, eliminating problems with querying and reasoning in large stores.

Thus, in contrast to the other works that used monolithic [15] or semi-modular [11] approaches, our work proposes a proactive approach to BIM and IoT integration by breaking down otherwise complex KGs into a set of small graphs. Instead of using complex queries, our approach allows users to interact with the system by browsing through the graphs via URLs as recommended by the Linked Data principles, making it more user-friendly. In the context of DTs, our approach allows us to distinctly select and update knowledge components without impacting the overall system. As such, problems related to querying and reasoning on the graphs as we would encounter with large stores are eliminated as we only work with subsets of the graph as needed.

Serving static documents with an Apache 2 server is more scalable than exposing SPARQL endpoints, and this is also better aligned with the vision of a hypermedia-driven Web and the Web of Linked Data. True, more processing is required on the client side to fetch and parse multiple smaller RDF documents, and query the subset of the graph that has been fetched at any given point in time. On the other hand, the application gains responsivity as little data is needed when the application starts.

Our motivation may be positioned with respect to the Linked Data Fragments spectrum [28], where on one far end the server bares all the query processing work, and on the other far end the client downloads the whole graph as a dump and queries it.[13] Arguably, our approach applies to KGs what tilemaps apply to 2D or 3D game development: the global KG is made out of small, regular-shaped graphs that we could call *LBD Knowledge Tiles*, which results in performance and memory usage gains.

7. Conclusion and Future Work

We have presented a methodology to integrate different knowledge graphs so as to contribute to the building's semantic Digital Twin vision. Our approach has been implemented on the Espace Fauriel building of Mines Saint-Étienne, and results in the creation of a set of resources that may be useful for other researchers in the community. As for future work, we aim to improve the knowledge graphs with more knowledge useful for doing thermal simulations of rooms, serve live and historic IoT data as RDF using the SOSA/SSN ontology, serve live data with MQTT over websocket, further develop the 3D visualization platform shallowly demonstrated in Section 5.4, and experiment with autonomous agents in this virtual environment.

---

[13]Linked Data Fragments - https://linkeddatafragments.org/

Acknowledgments

References

[1] K. Siountri, E. Skondras, D. D. Vergados, Developing smart buildings using blockchain, internet of things, and building information modeling, in: Research Anthology on Convergence of Blockchain, Internet of Things, and Security, IGI Global, 2022, pp. 366–381. doi:10.4018/978-1-6684-7132-6.ch022.

[2] M. Batty, Digital twins, Environment and Planning B: Urban Analytics and City Science 45 (2018) 817–820. doi:10.1177/2399808318796416.

[3] A. Fuller, Z. Fan, C. Day, C. Barlow, Digital twin: Enabling technologies, challenges and open research, IEEE Access 8 (2020) 108952–108971. URL: https://doi.org/10.1109%2Faccess.2020.2998358. doi:10.1109/access.2020.2998358.

[4] M. H. Rasmussen, M. Lefrançois, P. Pauwels, C. A. Hviid, J. Karlshøj, Managing interrelated project information in AEC knowledge graphs, Automation in Construction 108 (2019) 102956. doi:10.1016/j.autcon.2019.102956.

[5] N. Wills, J. Diaz, Integration of Real-Time data in BIM enables FM processes, WIT Transactions on The Built Environment 169 (2017) 127–133. doi:10.2495/bim170121.

[6] Z. Abu Bakar, M. Mathews Dr, A Proposal to Harmonize BIM and IoT Data Silos using Blockchain Application, Technological University Dublin, 2021. doi:10.21427/3KRX-Z768.

[7] H. Liu, M. Lu, M. Al-Hussein, Ontology-based semantic approach for construction-oriented quantity take-off from bim models in the light-frame building industry, Advanced Engineering Informatics 30 (2016) 190–207. doi:10.1016/j.aei.2016.03.001.

[8] C. Wang, S. Bi, R. Zhang, Q. Fu, T. Gan, A modular reasoning approach to knowledge graph, in: 2022 15th International Conference on Advanced Computer Theory and Engineering (ICACTE), 2022, pp. 73–77. doi:10.1109/ICACTE55855.2022.9943760.

[9] R. B. Stone, Towards a theory of modular design, The University of Texas at Austin, 1997.

[10] M. d'Aquin, A. Schlicht, H. Stuckenschmidt, M. Sabou, Ontology modularization for knowledge selection: Experiments and evaluations, in: International Conference on Database and Expert Systems Applications, Springer Berlin Heidelberg, 2007, pp. 874–883.

[11] W. Terkaj, P. Pauwels, A method to generate a modular ifcOWL ontology, in: 8th International Workshop on Formal Ontologies meet Industry, volume 2050, 2017.

[12] M. H. Rasmussen, M. Lefrançois, G. Schneider, P. Pauwels, BOT: the Building Topology Ontology of the W3C Linked Building Data Group, Semantic Web 12 (2021) 143–161. doi:10.3233/SW-200385.

[13] S. Käbisch, T. Kamiya, M. McCool, V. Charpenay, M. Kovatsch, Web of Things (WoT) Thing Description, W3C Recommendation 9 April 2020, World Wide Web Consortium (W3C), 2020. URL: https://www.w3.org/TR/wot-thing-description10/.

[14] J. Beetz, J. van Leeuwen, B. de Vries, Ifcowl: A case of transforming express schemas into ontologies, AI EDAM 23 (2009) 89–101. doi:10.1017/S0890060409000122.

[15] P. Pauwels, W. Terkaj, EXPRESS to OWL for construction industry: Towards a recommendable and usable ifcOWL ontology, Automation in Construction 63 (2016) 100–133.

[16] P. Pauwels, A. Roxin, Simplebim: From full ifcowl graphs to simplified building graphs, in: eWork and eBusiness in Architecture, Engineering and Construction, CRC Press, 2017, pp. 11–18.

[17] M. Bonduel, J. Oraskari, P. Pauwels, M. Vergauwen, R. Klein, The ifc to linked building data converter-current status, in: Proceedings of the 6th Linked Data in Architecture and Construction Workshop (LDAC 2018), volume 2159, CEUR Workshop Proceedings, 2018, pp. 34–43.

[18] J. Oraskari, M. Bonduel, K. McGlinn, P. Pauwels, F. Priyatna, A. Wagner, V. Kukkonen, S. Steyskaland, J. Lehtonen, Ifctolbd, 2023. URL: https://doi.org/10.5281/zenodo.7701792. doi:10.5281/zenodo.7701792, If you use this software, please cite it using the metadata from this file.

[19] P. Pauwels, L. J. McGibbney, B. Thurm, G. Velludo, pipauwel/ifctordf: v0.4, 2020. URL: https://doi.org/10.5281/zenodo.4008032. doi:10.5281/zenodo.4008032.

[20] S. Tang, D. R. Sheldon, C. M. Eastman, P. Pishad-Bozorgi, Z. Gao, A review of Building Information Modeling and the Internet of Things devices integration: Present Status and Future Trends, in: Automation in Construction, BCS-HCI '08, British Computer Society, Swinton, UK, UK, 2019, pp. 13–22.

[21] S. Tang, C. Zhang, J. Hao, F. Guo, A Framework for BIM, BAS, and IoT Data Exchange Using Semantic Web Technologies, in: Construction Research Congress 2022, 2022.

[22] M. Kuller, F. Kohlmorgen, N. Karoglan, M. Niemeyer, I. Kunold, H. Wohrle, Conceptual design of a Digital Twin based on Semantic Web Technologies in the Smart Home context, in: IEEE work shop, CANDO-EPE 2020, British Computer Society, Swinton, UK, UK, 2020, pp. 167–172.

[23] A. Donkers, B. de Vries, D. Yang, Creating occupant-centered digital twins using the Occupant Feedback Ontology implemented in a smartwatch app, Semantic Web (2022) 1–26. doi:10.3233/SW-223254.

[24] G. Desogus, E. Quaquero, G. Rubiu, G. Gatto, C. Perra, BIM and IoT sensors integration: A framework for consumption and indoor conditions data monitoring of existing buildings, Sustainability 13 (2021) 4496.

[25] A. Cimmino, M. Poveda-Villalón, R. García-Castro, eWoT: A semantic interoperability approach for heterogeneous IoT ecosystems based on the Web of Things, Sensors 20 (2020) 822.

[26] N. Moretti, X. Xie, J. Merino, J. Brazauskas, A. K. Parlikad, An openBIM Approach to IoT Integration with Incomplete As-Built Data, Applied Sciences 10 (2020) 8287.

[27] B. C. Grau, I. Horrocks, Y. Kazakov, U. Sattler, A Logical Framework for Modularity of Ontologies, in: International Joint Conference on Artificial Intelligence, 2007.

[28] R. Verborgh, M. Vander Sande, O. Hartig, J. Van Herwegen, L. De Vocht, B. De Meester, G. Haesendonck, P. Colpaert, Triple pattern fragments: a low-cost knowledge graph interface for the web, Journal of Web Semantics 37 (2016) 184–206.