2023 European Conference on Computing in Construction
40th International CIB W78 Conference
Heraklion, Crete, Greece
July 10-12, 2023

EXTENSIBLE REAL-TIME DATA ACQUISITION AND MANAGEMENT FOR IOT ENABLED SMART BUILDINGS

Lasitha Chamari, Ekaterina Petrova and Pieter Pauwels
Eindhoven University of Technology, The Netherlands

Abstract

Smart buildings utilize real-time and historical sensor data to deploy data-driven application. Unlike Building Management Systems, where the data formats are somewhat aligned with industry standards, IoT is more flexible and less standardized. When multiple IoT solutions are present, they often use different methods to format the messages and represent metadata, hindering the effective usage of IoT data in smart building applications. This study proposes a framework and its software implementation for acquiring and managing real-time data using a microservices-based system architecture. To create a standardized semantic information model, the proposed framework employs smart building ontologies.

Introduction

Internet of Things (IoT) has rapidly evolved and is dominant in many smart buildings where devices, sensors, actuators and smart building applications are connected via the Internet. These smart building applications typically enable smart Indoor Environmental Quality (IEQ) monitoring, space booking and maintenance, car parking, automated lighting, implementation of Demand Side Management (DSM) strategies, etc. The IoT ecosystem is quite diverse, with a large variety of hardware, protocols, middleware, interfaces and platforms, making heterogeneity an inherent characteristic of IoT. Most buildings utilise multiple IoT platforms for different use cases, e.g., one platform for Indoor Air Quality (IAQ) monitoring, another one for space utilisation, etc. Every IoT platform provides different ways to connect and receive the data, which makes the already heterogeneous smart building ecosystems even more complex in terms of finding, querying, integrating and utilising IoT data for both humans and machines. Hence, acquiring, managing and accessing the highly heterogeneous data efficiently is essential for the development of smart buildings applications that depend on and utilise data from various domains.

Many existing approaches rely on historical data extracted from a data platform or Building Management System (BMS) that is fed into a data analytical pipeline (Teizer et al., 2017; Chen et al., 2021; Martín-Garín et al., 2020; Bashir and Gill, 2016). Furthermore, most approaches utilising time series data result in standalone applications, which means that they have little to no interactions with other systems in buildings. Recently, the more widespread use of IoT and cloud computing in buildings has led to an abundance of sensor data available in real-time. In addition, the use of web services makes it possible to integrate IoT data with other systems, so they could utilise the knowledge across multiple domains (such as BMS, Building Information Models (BIM), Energy Management Systems (EMS)) for decision making. However, the heterogeneity of the data models of the different IoT systems, and the heterogeneity of the data models of other systems such as BMS, BIM and EMS, makes this integration extremely challenging and the data cannot be seamlessly shared between different systems and across domains.

Therefore, this study proposes a framework encompassing 1) a collection of data acquisition modules (data providers), 2) a module for unifying schemata and semantics (message processor), 3) Publish/Subscribe (Pub/Sub) and real-time streaming service, 4) an ontology-based information model and 5) an Application Programming Interface (API). The paper also presents a proof-of-concept, where a web application has been developed to demonstrate the functionality and the potential of the proposed framework. The aim is to provide an opportunity for client applications to access real-time data without having to write individual software applications to access the different types of data from different sources. The proposed framework is extensible, i.e., more IoT devices can be easily added at any given point in time.

The remainder of this paper is organised as follows. Section 2 highlights some fundamental aspects and common IoT applications in buildings, as well as the usage of real-time sensor data in these applications. Section 3 outlines the main challenges that this study addresses, namely, 1) the heterogeneity in acquiring data from multiple IoT platforms and 2) the differences in data models and semantics in different platforms. Section 4 describes the five components of the proposed framework and its system architecture. Section 5 presents the implementation of the framework and a proof-of-concept web application that consumes the data from multiple IoT platforms. Section 6 concludes the paper and gives directions for future work.

Background

Real-time data updates continuously with new data points and results in continuous data streams that provide an indication of the behaviour and performance of the built environment. Building operation has complex dynamics that depend on various influences such as changes in external conditions, occupant behaviour, system operation, etc. Understanding these dynamics and utilising the existing data can positively influence and inform decision-making related to architectural design, HVAC system design, control strategies, etc. Time series data from sensors and sensor networks in buildings is often used in dashboards, charts and reports for visualisation of information related to energy consumption, monitoring of IEQ parameters, predictive maintenance, Model Predictive Control (MPC), prediction of building occupancy and modelling

of occupant behaviour, etc. Also, sensor data is used to detect abnormal conditions (e.g., bad air quality, malfunctioning devices, etc.) (Chen et al., 2021; Donkers et al., 2021). This leads to initiation of predefined tasks such as sensing alerts to facility managers or maintenance contractors.

Energy consumption data is often used in data-driven models to forecast energy demand and aid energy savings in individual buildings (Gómez-Omella et al., 2021; Leprince and Zeiler, 2020). It is also done for a group of buildings (i.e., to cluster and benchmark buildings by building type). They are further used to develop strategies to flatten peak demand in buildings for DSM (Cox et al., 2020; Walker et al., 2020). Another application is analysing energy usage data to identify daily, weekly and seasonal patterns in energy consumption (Miller et al., 2015).

Fault Detection and Diagnosis (FDD) is another application domain focusing on detecting and diagnosing faults and anomalous behaviour in HVAC systems (Mirnaghi and Haghighat, 2020). FDD approaches aim to identify and prevent faults that could lead to, for instance, high energy consumption. As such, FDD methods are mostly data-driven and typically rely on historical time series datasets. Most of the required data (supply and return temperatures, humidity, air pressure, flow, gas, electricity meter data, CO_2 concentration, signals from actuators (opening/closing), positioning of valves, etc.) is collected from the BMS, where occupant interactions (door and window opening, light level adjustment, temperature adjustment) and occupant thermal comfort data are usually collected from IoT devices, smartwatches and smartphones. Besides FDD, such data can contribute to understanding occupant behaviour, improving building operation and understanding the patterns in energy use.

Measured performance data from buildings is also used to improve the accuracy of simulation model input and output (Mirnaghi and Haghighat, 2020). In this regard, energy consumption data can be used for autonomous tuning of building energy models and continuous model calibration. Other applications include using real-time sensor data in combination with BIM, Virtual Reality (VR) and Augmented Reality (AR) in evacuation scenarios (Chen et al., 2021); for visualization of Indoor Air Quality (IAQ) with the help of AR (Hadj Sassi and Chaari Fourati, 2020), etc. Such applications aim to improve the health and safety in buildings. A large number of the studies focusing on IoT smart building applications rely either on a custom micro-controller based sensor network installed specifically for the purpose (Hadj Sassi and Chaari Fourati, 2020; Dave et al., 2018; Kang et al., 2018; Martín-Garín et al., 2020; Zahid et al., 2021; Gao et al., 2021), or on collecting data from one particular platform only (Riaz et al., 2014; Hosamo et al., 2022; Tan et al., 2022), or on virtual sensors that generate homogeneous data (Bashir and Gill, 2016), thereby avoiding the necessity to deal with a large number of heterogeneous platforms. Integrating data from heterogeneous platforms across domains is still in its infancy.

In this regard, semantic technologies have shown significant potential for integrating different domains and heterogeneous data in smart buildings. When it comes to IoT, ontologies such as SSN/SOSA[1] (Haller et al., 2018) directly address the standardization of vocabulary of sensors, observations, and their relationships. Other ontologies such Brick[2] (Balaji et al., 2018), RealEstateCore[3] (Hammar et al., 2019) and Haystack[4] also share metadata representations related to IoT. Since IoT data is related to the building and its components as well (devices and spaces), it is possible to integrate these types of data. IoT data can be linked to BIM related concepts using Linked Building Data (LBD) ontologies[5]. However, the existing studies that use ontologies for integrating the above domains (Dey et al., 2015; Mavrokapnidis et al., 2021; Kučera and Pitner, 2018; Park et al., 2022) have so far not used the semantic graphs to automate the discovery of IoT sensors and connecting to their real-time data streams.

Heterogeneity in IoT platforms and data models

IoT devices in buildings usually communicate to different IoT platforms (such as Tuya[6], Azure IoT Hub[7], openHAB[8]). Also, some IoT devices can publish their sensor data to a local message broker, and the data can be directly accessed by accessing the message broker locally. Both scenarios are shown in Fig.1. Communication of sensor data and control commands is achieved either by using lightweight real-time messaging protocols like Message Queuing Telemetry Transport (MQTT), Constrained Application Protocol (CoAP) or Representational State Transfer (REST) over Hypertext Transfer Protocol (HTTP) (Fig. 1). Different IoT devices are usually produced by different vendors and, therefore, even within the same building, sensor data is often contained within different platforms (Fig. 1).

It is common to subscribe to a particular IoT platform's RESTful API to get access to the sensor data. For example, platform 1 needs a GET request with the query `{{url}}/v1.0/devices/{{device_id}}/status` and platform 2 needs a GET request with `{{url}}/api/tk/query_now?token={{token}}`. These two examples clearly show how the methods to access the data are different for the different APIs, because there is no common standard API specification in the IoT industry. Therefore, when a new smart building application needs to access data from multiple platforms, the application developer needs to implement different methods to access the data for each new application.

An example message received for an IAQ sensor node is

[1] https://www.w3.org/TR/vocab-ssn/
[2] https://brickschema.org/ontology/
[3] https://www.realestatecore.io/
[4] https://project-haystack.org/doc/lib-phIoT/index
[5] https://w3c-lbd-cg.github.io/lbd/
[6] https://www.tuya.com/
[7] https://azure.microsoft.com/en-us/products/iot-hub/
[8] https://www.openhab.org/

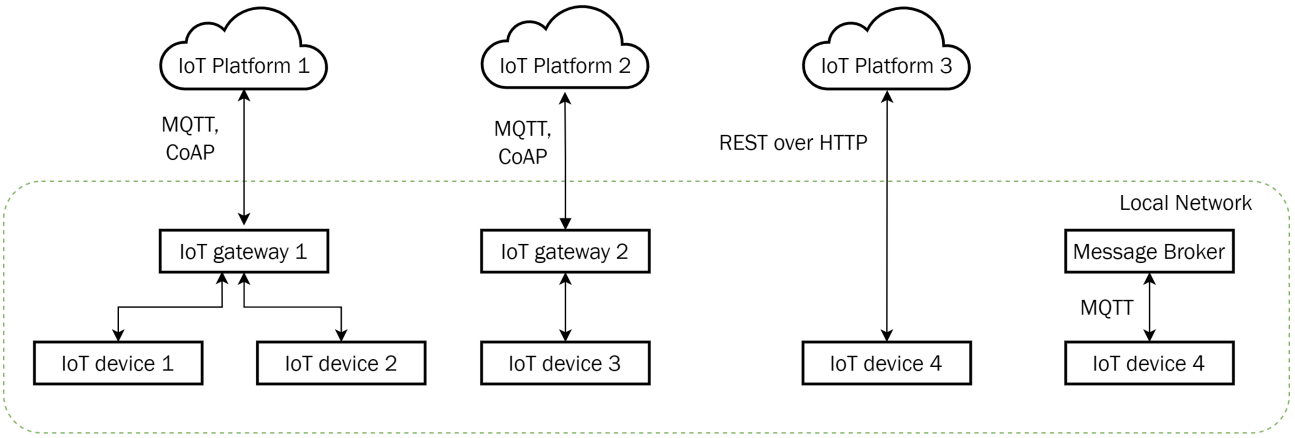

Figure 1: Multiple IoT platforms

shown in Listing 1.

```
{
    "status":"ok",
    "entries":
        [{"area":"",
        "name":"Airbox.Setup.FE",
        "fw_ver":"v1.16",
        "lat":51.4477128,
        "lon":5.4946999,
        "model":"AI-2004WP",
        "odm":"acelink.edimax",
        "h":46.8, "hcho":0,
        "pm1":0,"pm10":6,
        "pm25":6,
        "t":21.95, "tvoc":0.05,
        "co":0,"co2":588,
        "type":"indoor-airbox",
        "time":"2022-09-29T17:14:29+08:00",
        "status":"online","adf_status":0}],
    "exclusion":null,"total":1
}
```

Listing 1: IAQ sensor message payload

Listing 1 shows that message payload contains the sensor reading and additional metadata. In the payload, unambiguous keys (such as 'co_2' and 'tvoc') and ambiguous keys (such as 'h' and 't') are both used. These keys could also be different for another IoT platform. Listing 2 shows the message body received from another IoT platform for a smart plug data. In here, 'add-ele' refers to the energy consumption in kWh, 'cur_current' refers to the electric current in mA and so on.

```
{
    "result": [
    {"code": "switch_1", "value": false },
    {"code": "countdown_1", "value": 0 },
    {"code": "add_ele","value": 19 },
    {"code": "cur_current", "value": 230 },
    {"code": "cur_power", "value": 417 },
    {"code": "cur_voltage", "value": 2029 }
    ],
    "success": true,
    "t": 1664442846776,
    "tid": "11ca62a63fd711edb87eca0f69d78a9e"
}
```

Listing 2: Smart plug message payload

These examples show that the data schema and semantics are customised for each platform and there isn't a standard representation. However, for a smart building application developer, having to deal with different schemata and semantics for each application is a barrier for integrating this data.

This study proposes a standard framework to harmonise the data originating from various IoT devices and platforms. The framework not only supports harmonisation of IoT data itself, but also enables this IoT data to be integrated with other data sources using a semantic model based on existing smart building ontologies. The proof-of-concept application presented in this paper demonstrates how IoT data originating from multiple sources are unified and integrated with a BIM model of a building. This integration brings up the possibility for various interactive applications such as Digital Twins (Tan et al., 2022), monitoring the building in real-time and providing visual analytics (Dave et al., 2018), BIM based predictive maintenance, fire safety, FDD in HVAC systems (Hosamo et al., 2022), and so on.

Proposed Framework and System Architecture

The proposed framework comprises five components, each of which is responsible for a set of functionalities. The five components are 1) a collection of data acquisition modules (data providers), 2) a module for unifying schemata and semantics (message processor), 3) Pub/Sub and real-time streaming service (based on MQTT and WebSocket), 4) an ontology based information model and 5) an API. Figure 2 illustrates these five components. This architecture is mainly focused on providing real-time sensor data for web applications and, therefore, relies on WebSocket, which is a protocol that enables bi-directional data exchange between browser and server via a persistent connection. That makes it particularly suitable for services that require continuous data exchange.

Data providers.

The proposed framework relies on a collection of data providers, each of which is a microservice running independently. This microservice-based approach provides the opportunity to implement data connectors for different IoT devices separately according to their different protocols

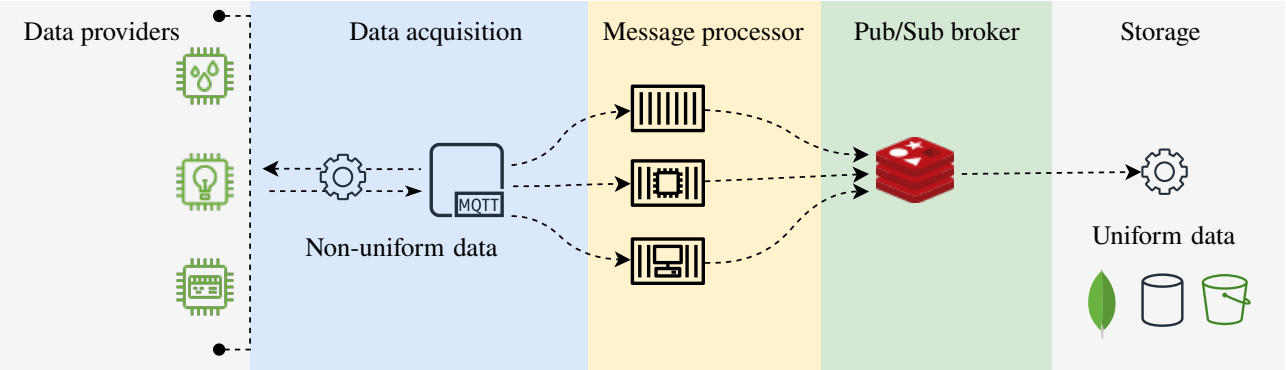

Figure 2: Proposed framework for IoT data management

and data models. Since they are independent, they can be added or removed without affecting other data providers. The main function of this data provider is to acquire IoT sensor data from different platforms and publish them to the Pub/Sub message broker. Each uniquely identifiable IoT sensor node publishes its data to a unique topic. As shown in Fig. 3, three types of IoT sensor nodes have been used in this study.

Figure 3: IoT Nodes (Indoor Air Quality sensor, smart plug and custom sensor node

IoT Node 1: An Indoor Air Quality sensor from Edimax measuring CO_2, temperature, humidity, CO, TVOC, PM2.5, PM10, and HCHO. Sensor data can be accessed via their API hourly, daily, weekly, or in real-time.
IoT Node 2: A smart socket from BlitzWolf measuring voltage, current, and power. Sensor data can be accessed via their API.
IoT Node 3: A custom built ESP32-based sensor node measuring temperature, humidity, CO_2, TVOC, illumination and battery voltage.
Each node needs a microservice to publish the received sensor data (from an API or directly from the sensors) to the local Pub/Sub message broker. After the proposed framework is implemented, adding this microservice is the only change that is required when introducing a new IoT device/platform. Then the data will be automatically fed into the forthcoming services.

Pub/Sub message broker.

Pub/Sub is a messaging pattern where any message published to a topic by a sender is immediately received by all of the receivers who have subscribed to that particular topic. The sender publishes the messages not for a particular receiver, but for a topic. The connection between the publisher and the subscriber is handled by the broker. Eclipse Mosquitto[9] is used as the message broker in this study. It is based on the MQTT protocol which is a lightweight, publish-subscribe, machine to machine network protocol.

Message processor.

The broker receives messages in the original format that they are published in by each IoT node. Therefore, the received messages are in various formats (e.g., nested, flat map). Therefore, this service converts the messages into a uniform format. At this stage, all messages are rearranged into non-nested key:value pairs. A sample of a processed message is shown in Listing 3. These processed messages are then published to a message broker again. Redis Pub/Sub broker[10] is used for this process. Since the proposed framework focuses on managing data for web based applications, socket.io[11],a WebSocket library, is used. This library already uses the Redis Pub/Sub broker. The topics in the MQTT broker and the 'Room' in the socket.io implementation are analogous, i.e., socket.io's 'Room' is analogous to MQTT broker's topic and is used to connect to a particular MQTT topic. Therefore, joining a socket.io Room means subscribing to the real-time data stream of a particular IoT sensor node (note that each IoT node publishes their messages into a unique topic).

```
{
  topic: 'esp32/083AF266DD84/pub',
  keys: [ 'co2', 'tvoc',
  'te', 'rh', 'lux', 'vbat' ],
  values: [ 872, 71, 27.3233,
  56.39648, 586.6666, 4.1569 ],
  timestamp: 1662298546131
}
```

Listing 3: Messages processed to uniform format

Information model.

In this study, the information model is semantically described using the SSN/SOSA, Brick and LBD ontologies for making it possible to access, integrate and extract information across different data providers (IoT and non-IoT such as BIM, BAS, etc.). A sample of the information model adopted by the proposed framework is shown in

[9]https://mosquitto.org/
[10]https://redis.io/docs/manual/pubsub/
[11]https://socket.io/

Listing 4. The same model is visually represented in Fig. 4. The full model is available in this repository[12].

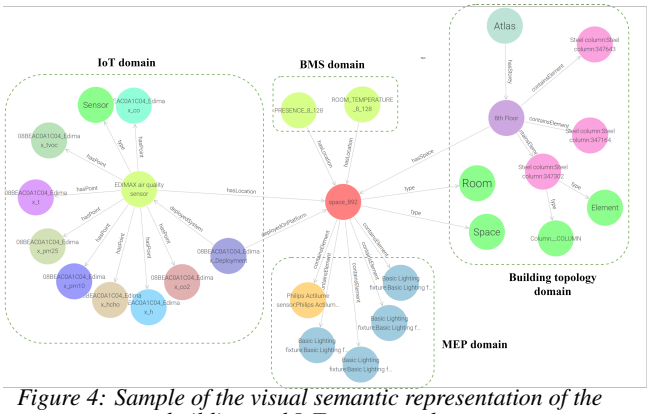

Figure 4: Sample of the visual semantic representation of the building and IoT sensor nodes

This semantic model makes use of Brick's time series identification. The MQTT topics are included using the Brick ontology's *brick:hastimeseriesId* relationship, because each sensor node publishes its sensor data into a unique topic. This way, real-time data from an IoT node located in a particular space in a building can be uniquely identified using the semantic information model.

In an IoT infrastructure, it is much more common to have a sensor node with multiple sensors attached to the same sensor node. For example, the IAQ sensor node hosts eight sensors. In the Brick ontology, the *brick:hasPoint* relationship can be used to uniquely identify these eight sensors. In the SSN ontology, the *ssn:hosts* relationship can be used to describe a node's individual sensors. The ability to uniquely identify these sensors is also important to get the identifier to access real-time/ historical data from a particular sensor in an IoT node. This is shown in the information model in Listing 4. The *brick:hasLocation* and *ssn:deployedOnPlatform* relationships can be used to locate the sensor in a particular room or a wall that belongs to a particular room.

```
inst:08BEAC0A1C04_Edimax a brick:Sensor ;
  rdfs:label "EDIMAX air quality sensor";
  rdfs:seeAlso "https://www.edimax.com/
  edimax/mw/cufiles/files
  /download/datasheet/
  AirBox_AI-1001W_V2_Datasheet_English.pdf" ;
  brick:hasPoint
    inst:08BEAC0A1C04_Edimax_co2,
    inst:08BEAC0A1C04_Edimax_co,
    inst:08BEAC0A1C04_Edimax_h,
    inst:08BEAC0A1C04_Edimax_pm25,
    inst:08BEAC0A1C04_Edimax_pm10,
    inst:08BEAC0A1C04_Edimax_tvoc,
    inst:08BEAC0A1C04_Edimax_hcho,
    inst:08BEAC0A1C04_Edimax_t ;
  brick:hasLocation inst:space_892 ;
  brick:timeseries [
        brick:hasTimeseriesId "edimax/08BEAC0A1C04/pub" ;
    ] .
inst:08BEAC0A1C04_Edimax_co2
  a brick:CO2_sensor .
inst:08BEAC0A1C04_Edimax_co2_Obs a sosa:Observation ;
        rdfs:label "CO2" ;
        rdfs:comment "The realtime observation key is CO2" ;
        sosa:observedProperty inst:Carbon_Dioxide ;
        sosa:madeBySensor inst:08BEAC0A1C04_Edimax_co2 ;
        sosa:hasResult [
        a qudt:QuantityValue ;
      qudt:unit unit:ppm ] .
```

Listing 4: Sample of the semantic representation of the building and IoT sensor nodes

Data storage.

Many applications need to access historical sensor data. Therefore, the processed messages can be recorded in a suitable data storage such a timeseries database, or can be archived in a long term archival such as blob storage for later usage as shown in Fig. 2. When recording the real-time data, a suitable database schema should be used including the MQTT topic and each of the keys in the rearranged messages.

Application Programming Interface.

When a client requests data, a WebSocket connection is made between the API server and the client (in this case, a web application) and the last message of the requested sensor is retrieved from the Redis database. The purpose of providing the last message is to avoid waiting until the next message to see the sensor data. This is achieved by using the socket.io/redis-adapter and Redis database. The redis-adapter relies on the Redis Pub/Sub mechanism. Then, the real-time sensor data is communicated via the WebSocket connection. The API also provides the end point to execute SPARQL Protocol and RDF Query Language (SPARQL)[13] queries on the information model of the building and sensors, as described in the Results section.

Results

Framework implementation

Software implementation of the proposed framework is done using microservices. Three IoT nodes were selected for the implementation. Two nodes communicate to their own IoT platforms, and the third IoT node publishes its messages to the Mosquitto message broker directly. First, three data provider microservices (Node.js applications) are implemented reflecting the three device categories - the smart socket, air quality sensor and ESP32 sensor node (Bullet 1 Fig. 5).

Second, the three data providers publish their sensor readings into Mosquito message broker (Bullet 2 Fig. 5). Third, the message processor service (another Node.js application) subscribe to all the messages, rearrange them and publish to the Redis Pub-/Sub broker (Bullet 3 Fig. 5). At this stage, it is also possible to write the sensor reading to a time series database or an archive. The fourth component is the semantic graph stored in the GrapdDB database (Bullet 4 Fig. 5). The fifth component in the API (Nest.js backend) which handles the requests from client applications, authentication and authorisation (Bullet 5 Fig. 5). The next section describes the implementation of the proof-of-concept web application.

Proof-of-concept: accessing real-time sensor data using a web application

A web application (example client application as in Fig. 5) is implemented to demonstrate how to utilise the proposed framework to develop smart building applications which require access to sensor data from various platforms. The aim of this web application is to discover the available sensors in a building and subscribe to their real-time data (Fig. 6).

[12]https://github.com/ISBE-TUe/atlas-building-graph

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

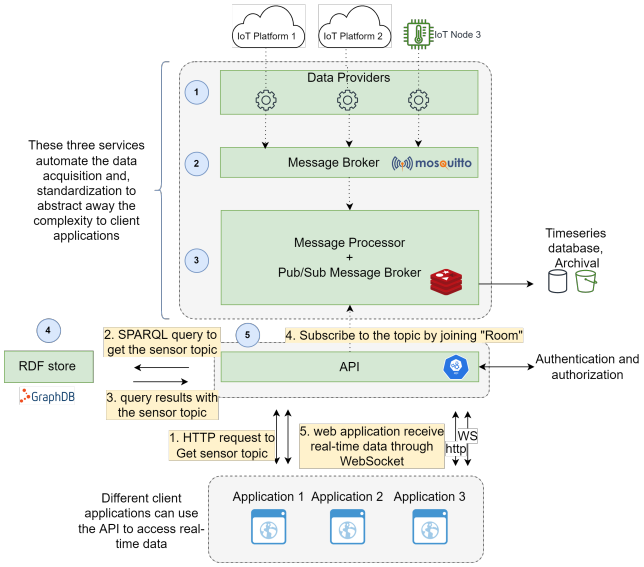

Figure 5: Detailed implementation of the framework

Discovering the sensors and observations by using their type, location or any other relationships can be done by executing SPARQL queries. For example, the query in Listing 5 fetches the number of sensors and number of sensors with real-time data. The query in Listing 6 returns the topic that the selected sensor publishes its data to. This topic is then used to join the relevant socket.io 'Room', which allows to communicate the sensor data in real-time via the WebSocket. By using formal ontologies for discovering the sensors, the developer can avoid the complexity of having to deal with non-standard and ambiguous naming conventions used by different IoT platforms.

```
SELECT ?Room ?Name (count(?sensor) as ?TotalSensors)
(count(?topic) as ?OnlineSensors) WHERE {
    {
        <\${selectedFloor['Floor']}> bot:hasSpace ?Room .
        ?Room props:name ?Name .
        ?sensor brick:hasLocation ?Room .
    }
    OPTIONAL {
         ?sensor brick:timeseries ?arr .
        ?arr brick:hasTimeseriesId ?topic .
    }
}
group by ?Room ?Name
order by desc(?OnlineSensors)
```

Listing 5: Get sensor count

```
SELECT ?topic WHERE {
        <\${selectedSensor.Sensor}> brick:timeseries ?arr .
        ?arr brick:hasTimeseriesId ?topic .
}
```

Listing 6: Get real-time sensor data topic

The above queries fetch the relevant data from the GraphDB triple store. The web app in Fig. 6 is populated with the RDF query results. Once a sensor is selected from the sensors table in Fig. 6, its real-time topic is obtained from the semantic graph via the query in Listing 6, which is then used to subscribe to the real-time data stream using the socket.io implementation. Fig. 7 shows the real-time data stream of that sensor.

The implementation of this example client application shows that the developer does not have to deal with the complexity of having to understand how to acquire data from each IoT platform and harmonise them.

Discussion

IoT systems in buildings are highly dynamic and heterogeneous. Therefore, discovering the data and metadata depends highly on the IoT platform used by a given IoT device. This poses a challenge when integrating these sensor data with other building data. Therefore, it is important to provide client applications (web applications, users, middleware) with a generalised way of accessing the data and metadata of IoT devices. Due to the ambiguity in IoT messages provided by each device, it is required to harmonise them before providing them to the client applications.

This study addresses the heterogeneity concern by a framework that harmonises IoT data and metadata. The framework enables the unification of IoT data, publishing them via a message broker, and also providing the semantics of IoT devices using an information model based on well-established existing ontologies. According to the SSN/SOSA ontology, the sensor reading and the timestamp can also be recorded in the graph under the *sosa:Observation* using *sosa:hasResult* and *sosa:resultTime* relationships. The proposed information model avoids recording the timeseries data in the graph because of the high volume of timeseries data generated by IoT devices, and the graph can quickly become unmanageable. While some existing studies embed all the timeseries data in the semantic graph (proven to be inefficient), other studies introduce the connection to timeseries database using the sensor identifiers (Kučera and Pitner, 2018; Balaji et al., 2018). Additionally, some studies include the connection string to the actual database in the semantic graph, which could lead to a security risk. We believe the connection strings should be kept in the scope of the API, and not in the semantic graph to avoid such risks. The proposed framework demonstrates how to embed the links to real-time time series data in the information model in an efficient way, and how they can be directly used to connect to real-time data streams. However, introducing the 'topic' of a real-time data stream is not explicitly implemented in any ontology and this study used the Brick ontology's *brick:hasTimeseriesId* relationship to implement it.

The proof-of-concept client application demonstrates how the IoT sensor data can be used independent of the IoT platform configurations. The complexity of having to deal with different platforms is significantly reduced by the proposed framework. The developer can execute a SPARQL query to obtain the necessary sensor metadata and subscribe to its real-time data stream simply by communicating with the API.

Conclusion

The utilization of data-driven applications is essential for improving the performance of smart buildings, and the abundance of real-time data provided by IoT devices is crucial for such applications. However, the flexible message formats and data models inherent in IoT technology pose challenges for data integration. To address this issue, this study proposes a framework for acquiring, unifying, and semantically enriching IoT data, and demonstrates its implementation using microservices. This practical approach to data integration can accelerate the development of data-driven applications in smart buildings.

Future research should aim to enable bi-directional communication between IoT devices and data-driven applications, as the current framework only allows for one-way communication, with the client application consuming data but not issuing commands

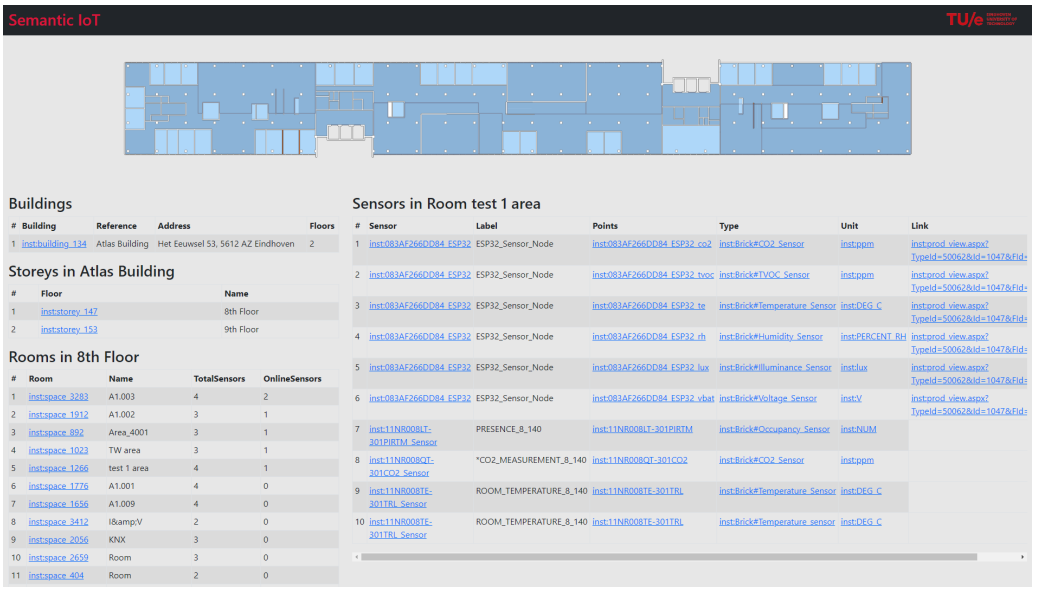

Figure 6: Web app demonstrating Query results for building, storeys, sensor count, and sensor metadata

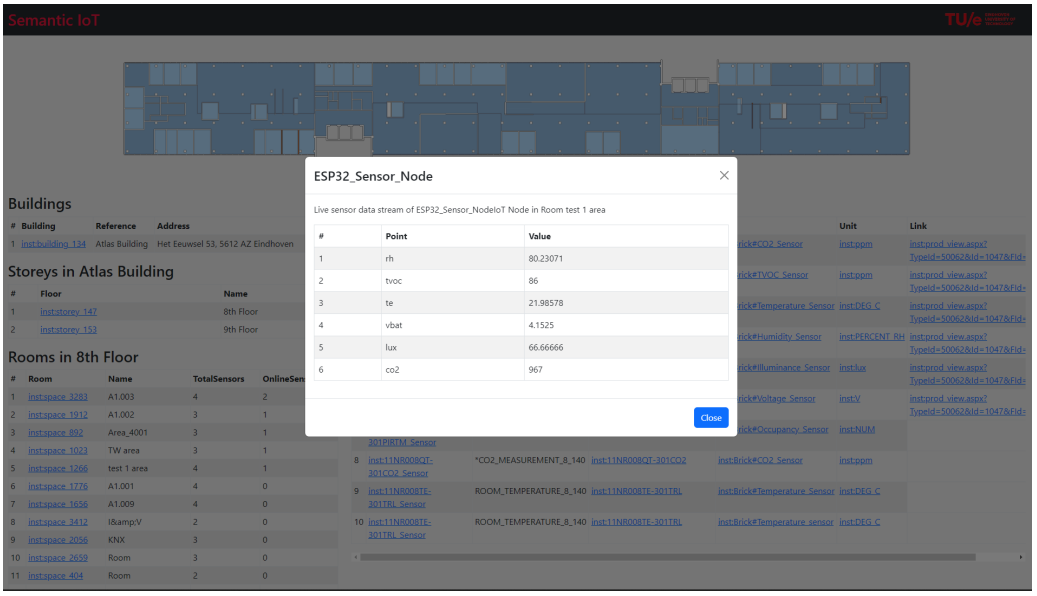

Figure 7: Real-time data stream of selected sensor

to IoT devices. Additionally, future work could improve the semantic information model by leveraging metadata such as sensor operating range (minimum and maximum values), operating conditions, accuracy, sensitivity, and other relevant factors using a semantic approach. Future research also need to provide systematic and scalable methods to semi-automatically generate this kind of a semantic model in a systematic and scalable manner for buildings with hundreds if not thousands of IoT sensors.

Acknowledgments

Brains for Buildings project received funding from the Dutch Ministry of Economic Affairs and Climate Policy and Ministry of the Interior and Kingdom Relations under the MOOI program.

References

Balaji, B., Bhattacharya, A., Fierro, G., Gao, J., Gluck, J., Hong, D., Johansen, A., Koh, J., Ploennigs, J., Agarwal, Y., Bergés, M., Culler, D., Gupta, R. K., Kjærgaard, M. B., Srivastava, M., and Whitehouse, K. (2018). Brick : Metadata schema for portable smart building applications. Applied Energy, 226:1273–1292.

Bashir, M. R. and Gill, A. Q. (2016). Towards an IoT Big Data Analytics Framework: Smart Buildings Systems. In 2016 IEEE 18th International Conference on High Performance Computing and Communications, pages 1325–1332. IEEE.

Chen, H., Hou, L., Zhang, G. K., and Moon, S. (2021). Development of BIM, IoT and AR/VR technologies for fire

safety and upskilling. Automation in Construction, 125(January):103631.

Cox, R., Walker, S., van der Velden, J., Nguyen, P., and Zeiler, W. (2020). Flattening the electricity demand profile of office buildings for future-proof smart grids. Energies, 13(9).

Dave, B., Buda, A., Nurminen, A., and Främling, K. (2018). A framework for integrating BIM and IoT through open standards. Automation in Construction, 95:35–45.

Dey, S., Jaiswal, D., Dasgupta, R., and Mukherjee, A. (2015). Organization and management of Semantic Sensor information using SSN ontology: An energy meter use case. In 2015 9th International Conference on Sensing Technology (ICST), volume 2016-March, pages 468–473. IEEE.

Donkers, A., Yang, D., De Vries, B., and Baken, N. (2021). Real-Time Building Performance Monitoring using Semantic Digital Twins. In CEUR Workshop Proceedings, volume 3081, pages 55–66.

Gao, X., Pishdad-Bozorgi, P., Shelden, D. R., and Tang, S. (2021). Internet of Things Enabled Data Acquisition Framework for Smart Building Applications. Journal of Construction Engineering and Management, 147(2):04020169.

Gómez-Omella, M., Esnaola-Gonzalez, I., Ferreiro, S., and Sierra, B. (2021). k-Nearest patterns for electrical demand forecasting in residential and small commercial buildings. Energy and Buildings, 253:111396.

Hadj Sassi, M. S. and Chaari Fourati, L. (2020). Architecture for Visualizing Indoor Air Quality Data with Augmented Reality Based Cognitive Internet of Things. In Advanced Information Networking and Applications, pages 405–418, Cham. Springer International Publishing.

Haller, A., Janowicz, K., Cox, S. J., Lefrançois, M., Taylor, K., Le Phuoc, D., Lieberman, J., García-Castro, R., Atkinson, R., and Stadler, C. (2018). The modular SSN ontology: A joint W3C and OGC standard specifying the semantics of sensors, observations, sampling, and actuation. Semantic Web, 10(1):9–32.

Hammar, K., Wallin, E. O., Karlberg, P., and Hälleberg, D. (2019). The RealEstateCore Ontology. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), volume 11779 LNCS.

Hosamo, H. H., Svennevig, P. R., Svidt, K., Han, D., and Nielsen, H. K. (2022). A Digital Twin predictive maintenance framework of air handling units based on automatic fault detection and diagnostics. Energy and Buildings, 261:111988.

Kang, K., Lin, J., and Zhang, J. (2018). BIM- and IoT-based monitoring framework for building performance management. Journal of Structural Integrity and Maintenance, 3(4):254–261.

Kučera, A. and Pitner, T. (2018). Semantic BMS: Allowing usage of building automation data in facility benchmarking. Advanced Engineering Informatics, 35(January 2017):69–84.

Leprince, J. and Zeiler, W. (2020). A Robust Building Energy Pattern Mining Method and its Application to Demand Forecasting. In 2020 International Conference on Smart Energy Systems and Technologies (SEST), pages 1–6. IEEE.

Martín-Garín, A., Millán-García, J., Baïri, A., Gabilondo, M., and Rodríguez, A. (2020). IoT and cloud computing for building energy efficiency. In Start-Up Creation, Woodhead Publishing Series in Civil and Structural Engineering, pages 235–265. Elsevier, second edi edition.

Mavrokapnidis, D., Katsigarakis, K., Pauwels, P., Petrova, E., Korolija, I., and Rovas, D. (2021). A linked-data paradigm for the integration of static and dynamic building data in digital twins. In Proceedings of the 8th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation, pages 369–372, New York, NY, USA. ACM.

Miller, C., Nagy, Z., and Schlueter, A. (2015). Automated daily pattern filtering of measured building performance data. Automation in Construction, 49:1–17.

Mirnaghi, M. S. and Haghighat, F. (2020). Fault detection and diagnosis of large-scale HVAC systems in buildings using data-driven methods: A comprehensive review. Energy and Buildings, 229:110492.

Park, E., Park, S., and Stratbücker, S. (2022). An ontology-based approach for building automation data analysis. pages 1–8.

Riaz, Z., Arslan, M., Kiani, A. K., and Azhar, S. (2014). CoSMoS: A BIM and wireless sensor based integrated solution for worker safety in confined spaces. Automation in Construction, 45:96–106.

Tan, Y., Chen, P., Shou, W., and Sadick, A.-M. (2022). Digital Twin-driven approach to improving energy efficiency of indoor lighting based on computer vision and dynamic BIM. Energy and Buildings, 270:112271.

Teizer, J., Wolf, M., Golovina, O., Perschewski, M., Propach, M., Neges, M., and König, M. (2017). Internet of Things (IoT) for Integrating Environmental and Localization Data in Building Information Modeling (BIM). In ISARC 2017 - Proceedings of the 34th International Symposium on Automation and Robotics in Construction, number Isarc, pages 603–609.

Walker, S., Khan, W., Katic, K., Maassen, W., and Zeiler, W. (2020). Accuracy of different machine learning algorithms and added-value of predicting aggregated-level energy performance of commercial buildings. Energy and Buildings, 209:109705.

Zahid, H., Elmansoury, O., and Yaagoubi, R. (2021). Dynamic Predicted Mean Vote: An IoT-BIM integrated approach for indoor thermal comfort optimization. Automation in Construction, 129:103805.