Design and integration of the project-specific ontology for data analytics support

Miloš Šipetić[1], Reinhard Jentsch[1], Judit Aizpuru[1] and Jan Kurzidim[1]

[1] Austrian Institute of Technology

Abstract. Data analytics projects often have access to little or no metadata associated with data collected and intended for processing, data analysis or machine learning. Even when metadata is available, it is usually poorly described, structured and linked to the corresponding data. In H2020 project InBetween, metadata description problem is approached by designing the project ontology on top of established ontologies BOT, SAREF and SAREF4BLDG. An approach for structuring and integrating all the collected data about occupants, their dwellings, appliances, sensors and locations into the project knowledge base is described. Building monitoring data, weather observation and weather forecast streams stored in SQL-based and time-series databases are semantically linked to the information about their respective source sensors stored in the knowledge base. Two data analysis use cases are discussed, along with improvements and benefits enabled by using the linked data approach. Discussed use cases are virtual occupancy sensor and electricity load profiling service. Linked data approach and use of common and standardized dictionaries help designing data analysis workflows that are easier to test, reuse and reason about.

Keywords: Linked data, Data analytics, Machine learning, Building operational phase, BOT, SAREF, SAREF4BLDG

1 Introduction

InBetween project [1] aims beyond available ICT technologies used for inducing the end user behavior change towards more energy efficient lifestyles by simultaneously assisting users to identify energy wastes, showing how they could conserve energy and motivating them to act. The overarching technological objective is to deliver cost-effective solution that brings added value, without significant disruption of everyday activities, through InBetween platform's advanced energy services. It allows users to integrate their building's connected devices and systems with advanced energy analytics and optimization services to create a comprehensive recommendation and feedback solution which will facilitate the behavior change towards more energy and cost-efficient daily routines. Developed services are designed to optimize energy flows by using renewable energy systems (RES), shaving peak loads and identifying inefficiencies and losses.

Utilized approach relies on defining and implementing a set of loosely coupled services that discover new knowledge about individual apartments or buildings through executing a predefined analytics workflow and storing and linking results to the corresponding apartment or building entities. This allows downstream services to use results generated by services earlier in the pipeline.

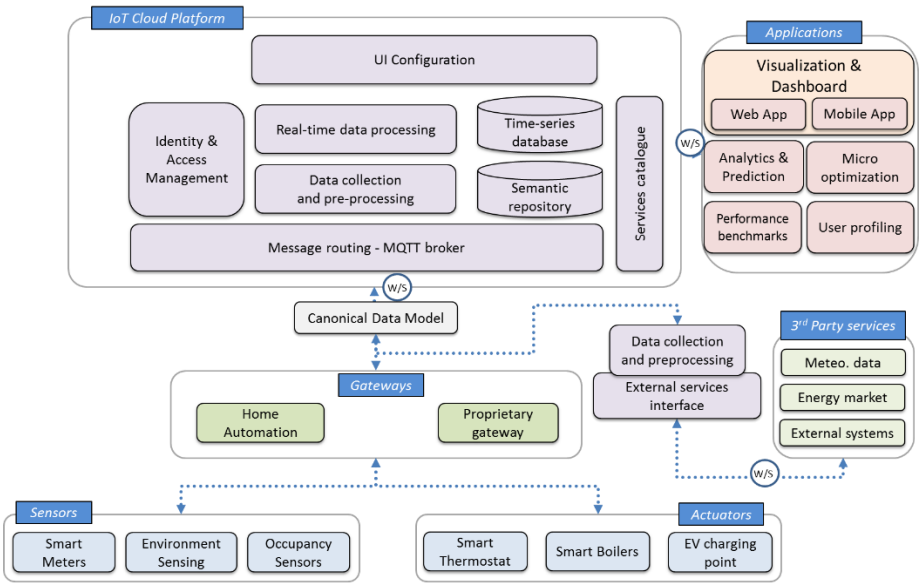

Fig. 1. InBetween platform architecture

At the very inception of the project it was obvious that it will be necessary to collect a lot of data from demonstration sites in order to realize the vision of the project. Furthermore, it was clear that for numerous data points collected through the monitoring campaign it will be necessary to collect a lot of metadata about datapoints themselves, and about their context. This refers predominantly to the locational context, but also to others, such as social, geographical, weather. Such contextual information can be described and represented using domain-specific ontologies. Ontologies covering different domains are aligned and linked together to form a complex graph of concepts and properties to represent their rich interrelationships. This work describes the design of the project ontology used to model and link data and metadata of all demonstration site entities relevant to the project. This is elaborated in paragraph 2. In the project, rich representations thusly obtained are used to realize use-cases that would otherwise be difficult to implement. Specifically, it becomes possible to develop data analysis workflows that do not target columns of excel worksheets, csv files or database tables, as is state of the art in the domain, but rather nodes in the knowledge graphs. Entities and relationships between them do not need to be specified exactly. Rather it is possible to describe analyses targets in terms of graph patterns and use abstractions to get to the necessary data instead of having to reference it explicitly and directly. Examples of realization of such use cases are described in paragraph 3.

Fig. 1 illustrates the architecture of InBetween platform. Components of interest in this work are semantic repository within the **IoT cloud platform** group along with analytics/prediction service and user profiling service within the **Applications** group [2].

2 Methodology

Many European projects involving demonstration sites used similar approaches of characterizing the demonstration sites as a first step, where all the relevant information is identified and described to characterize the demonstration site's technical, structural and social composition. Planning the action to be performed comes next, followed by performing the planned actions, and ultimately, the monitoring and evaluating the effects of performed actions. In recent years, there has been an increase in the use of ontologies to capture the information collected within such projects. Project EcoShopping [3] addresses various issues related to renovation of buildings, with a specific focus on shopping mall objects. Here the data collected about the demonstration site is structured without the use of ontology. This information is used to develop a simple simulation model of the shopping center, that is then utilized to predict the outcomes of the renovation to help with the decision making process for selection of optimal renovation option. In the CASCADE project [4] several airport facilities and their corresponding energy systems are modeled using abstract concepts from Suggested Upper Merged Ontology (SUMO). Link thusly established between data collected from various building automation systems (BEMS) and the ontological model of the facility and its systems is then used for smarter control and ultimately energy saving. RESPOND project [5] linked collected monitoring data with the demonstration sites metadata by developing the project ontology, which reused BOT ontology for space topology modelling and SAREF and SEAS Features of Interest ontologies for modelling of monitoring equipment and appliance.

2.1 Collecting the data sources

Main guiding force while designing the project ontology were use-cases described in the original data proposal. They informed a process of selecting which data to collect, and what metadata to describe the collected data. This included occupant information, energy consumption data, building information, space topology information, building elements, existing building equipment and appliances, sensor and actuator data and metadata, networking equipment connectivity information, historical and forecasted weather data and metadata.

Initial design data was collected in the first phase of the project, during the demonstration sites' characterization. Data was collected via surveys (occupant information, energy bills, existing appliances and energy-related equipment) and floorplan drawings (topology, building orientation, walls, doors, windows). As the total number of apartments was not excessive, floorplans and surveys were semi-automatically converted into excel sheets capturing the space containment hierarchy, space connected-

ness, door/window locations, relevant energy equipment descriptions and locations and occupant information.

In the second phase of the project, based on the collected data, and subsequent analysis of optimization potential, a monitoring plan was created which consisted of a list of locations for sensors and actuators. Installation location, sensor type and associated gateway connection were harvested from this phase of the project. One demo site responsible partner performed the "mock" commissioning of complete set of monitoring equipment on their premises. Mock commissioning included association of gateways and individual sensors and monitoring time series storage creation before deploying them in their final locations, thereby making sure that on-site connectivity wouldn't prevent the installation completion.

In the third phase monitoring gateways were installed, connected to network, sensors and actuators were commissioned in monitored apartments, and data recording began. In the remaining demonstration buildings identification numbers of sensors and gateways were recorded upon installation and associated with their target measurements and fields in the influx database.

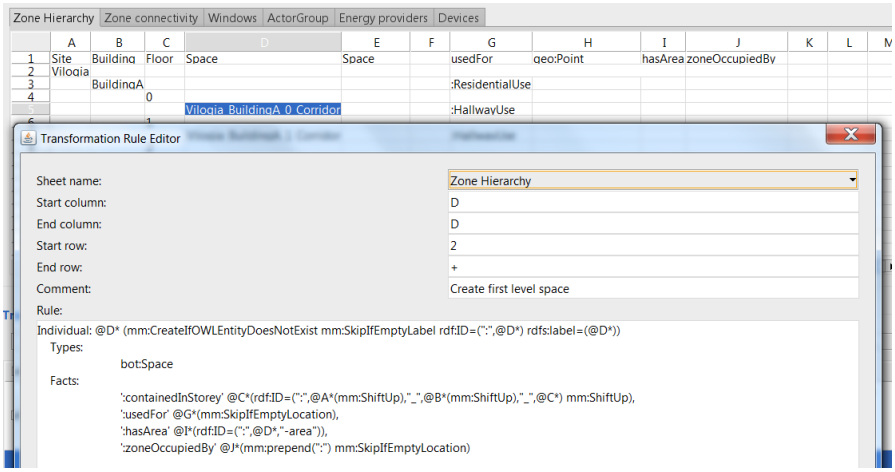

Fig. 2. Rule definition for instantiation of spaces and relationships related to them

All the data from Excel spreadsheets was ultimately transformed into OWL ontology using the Protégé plugin Cellfie [6] and a set of custom designed rules. Cellfie is a plugin for instantiating knowledgebases from data structured in tables. **Fig. 2** presents one example of a rule for creating **bot:Space** instances and their properties. Basic parts of a rule are **rule definition** and **range of target columns and rows** that should be processed. The instantiation engine iteratively processes each cell in the specified range (only column D in this example), and for each encountered value creates a series of axioms as defined in the text of the rule under "Facts:" section. Axiom definition can contain absolute values, as is the case with the definition of created entity type (i.e. "Types: bot:Space"), or references to other cells in the input table, as is the case with the row "':usedFor' @G*(mm:skipIfEmptyLocation)",

where **@G*** references the cell from column G belonging to the same row as the original encountered value. All the entities of InBetween ontology are prefixed with double colon (":").

2.2 Ontologies selected for reuse

After careful deliberation and review of available ontologies it was decided to base the project ontology on a number of well-known ontologies. Those are Building topology ontology (BOT) [7], Smart appliances reference ontology (SAREF) [8], and Saref extension for building devices ontology (SAREF4BLDG) [9].

Bot. Building Topology Ontology [10] is an ontology that describes the main topological concepts of a building. This minimal and extendable ontology provides the context of the building to devices, components and sub-components within the building and to the outside. Furthermore, relationships between the building sub-components are also defined. BOT defines several major classes: **bot:Element**, **bot:Interface** and **bot:Zone**. Sub types of **bot:Zone** are: **bot:Site**, **bot:Building**, **bot:Storey** and **bot:Space**. For those classes following object properties are available: **bot:adjacentElement**, **bot:adjacentZone**, **bot:containsElement**, **bot:containsZone**, **bot:interfaceOf**, **bot:intersectingElement**, **bot:intersectsZone**, **bot:has3DModel**, **bot:hasBuilding**, **bot:hasElement**, **bot:hasSubElement**, **bot:hasSpace**, **bot:hasStorey** and **bot:hasZeroPoint**.

SAREF. The Smart Appliances REFerence (SAREF) is an ontology for smart appliances that offers semantic interoperability through shared concepts. SAREF defines concepts for devices in households, offices and public buildings. Devices have functions and states also modeled as concepts. Devices can provide services, combine and compose their functions and make them discoverable, registerable and remotely controllable by other devices in the network. There are several extensions of SAREF such as SAREF4ENER [11] for the energy domain, SAREF4ENVI [12] for environment and SAREF4BLDG for the building domain. While SAREF does not include a library of appliance and sensor definitions, the vocabulary it defines allows describing sensors and appliances in detail.

SAREF4BLDG. SAREF4BLDG is an extension of the SAREF ontology. The original ontology is extended with a subset of the Industry foundation classes (IFC) standard related to devices and appliances. The idea is to provide neutral device descriptions supporting the IFC data model that could be shared among various stakeholders to ease the communication between smart appliances. SAREF4BLDG offers extensions to the original SAREF in the domain of topology representation.

BOT ontology was reused for describing the space topologies, and for compositions of complex devices. Such devices include combined sensors that measure multi-

ple properties. An example of such a sensor is a combined temperature, humidity and pressure sensor in one small package. sensors and actuators as defined in SAREF4BLDG ontology were used as base classes for all sensors and actuators in the project. SAREF properties were used as a starting point for property modelling and additional necessary properties were added to this base list to satisfy our data modelling needs.

2.3 Additional requirements and design.

In addition to classes and properties reused from selected ontologies, a number of new classes and properties were defined to cover the remaining concepts. A selection of new concepts is presented in this section. The exhaustive description of additional classes and properties is outside the scope of this work.

From the very beginning of the modelling process, it was clear that currently available semantic technologies cannot satisfy non-functional requirements for performance and scale for storing and querying time series monitoring data from demo sites and for measured and forecasted weather data. It was therefore decided to store monitoring data in the Influx time series database, and to subsequently continue using part of the TICK stack (Telegraph, InfluxDb, Chronograph, Kapacitor) [13] for data preprocessing and visualization. InfluxDb is a database custom designed for use cases where storing and querying large amounts of time series data is the main requirement. Similarly, measured and forecasted weather data was collected in the relational database. One of the reasons for disparate implementations of sensor data and weather data storage was difference in types of preprocessing (as illustrated in **Fig. 1**) that needed to take place for different data sources. Additionally, the ability to reference data coming from different sources was considered beneficial as it is a common occurrence in the industry.

Classes **inbetween:InfluxDbTimeSeries** and **inbetween:SqlDbTimeSeries** were defined to store information about the location of time series data. All time series data was stored outside the knowledge base. References to their location in the external databases was realized using these two classes. **inbetween:InfluxDatabase** and **inbetween:SqlDatabase** instances describe specific instances of external databases. Each **inbetween:TimeSeries** instance is associated to one **inbetween:Database** instance. This allowed referencing of external data coming from multiple databases of the same type which can be useful in testing or similar scenarios.

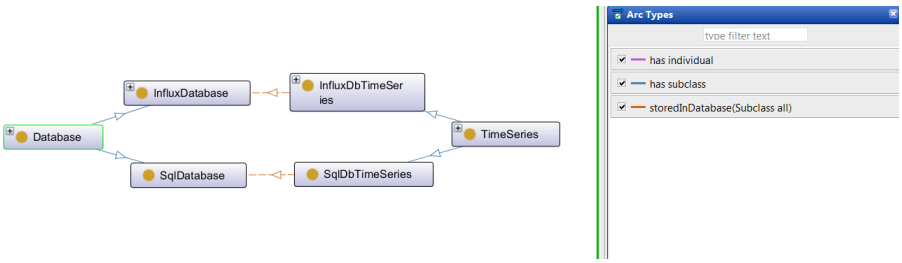

Fig. 3. Time series reference contains custom properties depending on the database where it is stored

It may be useful to stress the difference between weather forecast data and weather monitoring data. **WeatherSensor** class has been created to reference weather monitoring readings stored in the SQL database. Weather data is collected for all the demonstration sites from a third party service every hour and each invocation of this data collection process yields a list of readings (in that sense, it could be considered a virtual sensor, as it does not physically exist at the demonstration site).

On the other hand, one weather forecast instance contains forecasts of **all properties** of interest **for every hour in the next 24 hours** period. This data is needed for the operation of energy consumption prediction services. As short term weather forecasts change every hour, it is necessary for the purpose of training and testing of our consumption prediction algorithms that forecasts collected every hour are retained and available. In other words, while **weather data** is a **vector** of property values for present time, **weather forecast** is a **matrix** of property values, where second dimension of the matrix encodes different forecast horizons (i.e. property values 1 hour from now, 2 hours from now, etc). Weather forecasts are collected from the third party service and are stored in the SQL database. A subclass of **Sensor** called **inbetween:WeatherForecastSensor** was created. Similarly to **inbetween:WeatherSensor** it contains the reference to the containing SQL table and column where readings are stored. **inbetween:WeatherForecastSensor** also has object properties **inbetween:isMeasureadIn**, **inbetween:hasTimeSeries** and **inbetween:measuresProperty**. Important difference from **inbetween:WeatherSensor** is that **inbetween:WeatherForecastSensor** is qualified with **inbetween:hasForecastHorizon** property, that describes forecast horizon of the time series it references. For each demonstration site and for each property 24 **inbetween:WeatherForcastSensor** entities are instantiated in the knowledge base.

inbetween:WeatherForecastSensor only stores the reference to the actual time series location in the SQL database, and does not store the forecast data itself.

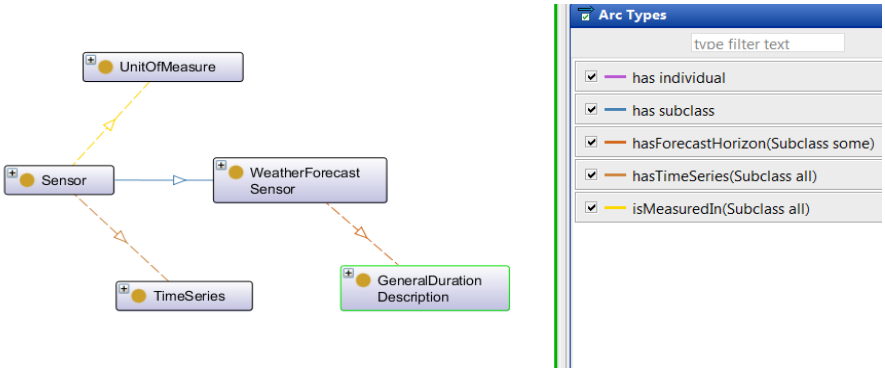

Fig. 4. Definition of weather forecast sensor

A number of classes inheriting **saref:Property** subclasses has been added to be able to represent all the properties used in InBetween. These additional properties include **inbetween:AirQuality**, **inbetween:Area**, **inbetween:BatteryState**, **inbetween:Direction**, **inbetween:Length**, **inbetween:Mass**, **inbetween:OnOff**, **inbetween:SolarInsolation**, **inbetween:SpecificHeat**, **inbetween:WindSpeed**, **inbetween:ElectricityExportPrice**, **inbetween:ElectricityImportPrice**, **inbetween:GasPrice** and **inbetween:HotWaterPrice**.

Electricity, heating and cooling demand profiles are represented using the Profile class. SAREF4ENER extension of SAREF ontology describes detailed scheme for representing the load profile of a device depending on the use case or selected mode of usage of a device. For usage in InBetween we opted for simpler approach as we do not have detailed specifications of power sequences nor the ability to pause or skip activities (slots) within a power sequence of otherwise non-smart appliances. For that reason the class **inbetween:SimpleEnergyProfile** is used, which is derived from **inbetween:SimpleProfile** (subclass of **saref:Profile**), which **saref:isAbout saref:Energy**, has specific sampling interval (**inbetween:hasSampleInterval** of type **xsd:duration**), and **saref:isMeasured** in a **saref:UnitOfMeasure** (in case of energy – kWh). Load forecasts are similarly modeled, as subclass of **inbetween:SimpleProfile**, and the only difference being link to the **inbetween:PredictionModel**. **inbetween:PredictionModel** references machine learning model trained within the InBetween platform and used for energy consumption prediction. **inbetween:PredictionModel** is stored as a binary blob and includes both the model itself and metadata describing inputs and outputs of the model.

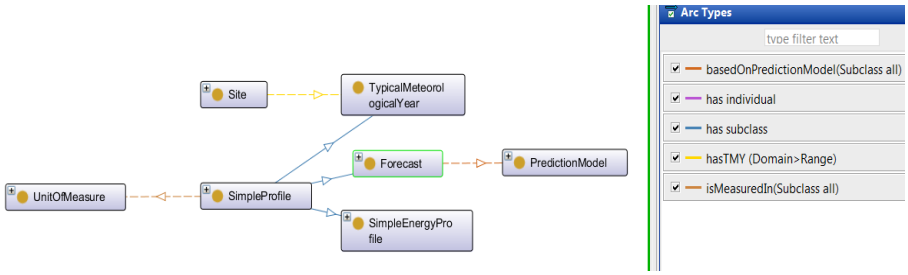

Fig. 5. Relationships used to link Forecast-related classes

Additionally, a number of object and data properties were added to cover different modelling needs. In the case of complex devices, the relationships between container device and contained device is modelled with a **inbetween:consistsOfElement** object property which is a subclass of **bot:hasSubElement** object property.

Inbetween:hostsElement is another subclass of **bot:hasSubElement** and is aimed at modelling the relationships between different building elements and sensors. Specifically, it was used to model the relationship between doors and windows and sensors embedded in them (as described in 3.1).

2.4 Ontology evaluation

Ontology evaluation has been performed mostly through manual inspection. Expressivity of the ontology as its ability to represent all the data collected and needed to realize use-cases specified in project documentation has also been positively evaluated, as it has been used throughout the project without complaints about missing concepts. Small improvements have been implemented during the process of evaluation. The ontology was deployed to production using the Fuseki server. In terms of feasibility, some use-cases produced queries which turned out to be too complex to execute in realtime on production systems. Offline datasets have been made available to better satisfy performance requirements of those use-cases.
Even though the project ontology reuses a number of actively developed ontologies, we expect that it will be feasible to stay aligned with those in the future.
InBetween ontology is not publicly available at the time of writing.

3 Use cases

3.1 Virtual occupancy sensor

Method. Virtual occupancy sensor is a software service that uses data available from physical sensors and available contextual data to infer if a person is present in the monitored space. For directly detecting occupancy, the active infrared sensors, usually in the form of thermographic cameras, could be used to detect and track heat-emitting objects in spaces. Their cost however prohibits wider use. Much more widely available and used are motion sensors, usually utilizing passive infrared (PIR) sensors, that detect motion by observing changes in infrared radiation in the monitored volume of space.

Fig. 6. Develco Door/Windows sensor

In InBetween in addition to motion sensors, access to data from door sensors is available. Door sensors are simple two-part sensors of which one part is placed on the door frame and the second part is placed on the door itself, opposite of each other (**Fig. 6**).

Information about the status of the door (i.e. opened or closed) is reported to the monitoring system on each change event.

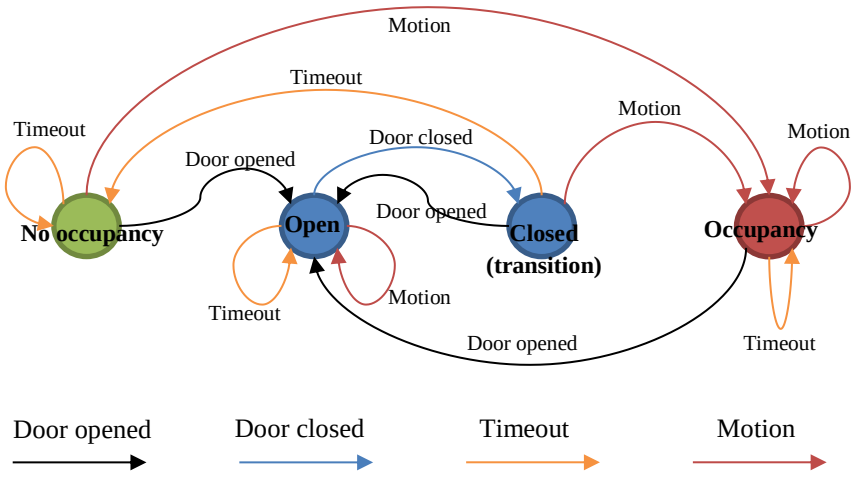

Fig. 7. State change diagram for occupancy detection

Occupancy detection systems based solely on motion sensor data typically introduce time threshold after motion events to detect situations when there is no presence in the space. This is prone to errors, most obviously during night hours when occupants are asleep and motionless. Thresholds between detected movements are usually exceeded in these situations, and unless additional logic is introduced, systems based solely on motion sensors have no way of inferring occupancy after the threshold period has passed.

Availability of dwelling topology information, sensor locations and sensor time series enables us to detect occupancy using logic defined using the state change diagram in **Fig. 7**. Result of running the occupancy detection over 2 day data slice can be seen in **Fig. 8**. Blue line represents the detected occupancy. As no information about the actual apartment occupancy was available, the ability of the algorithm to detect occupancy has been validated by selecting random periods for individual apartments and visually cross-checking the sensor information with detected occupancy time series. As can be seen in the method seems to detect the actual occupancy well and maintains the “occupied” status even during prolonged periods of inactivity.

Data needs. For implementation of virtual occupancy detection sensor for the specific apartment, it is necessary to collect data from all motion sensors and a door sensor mounted on the entrance of the apartment.

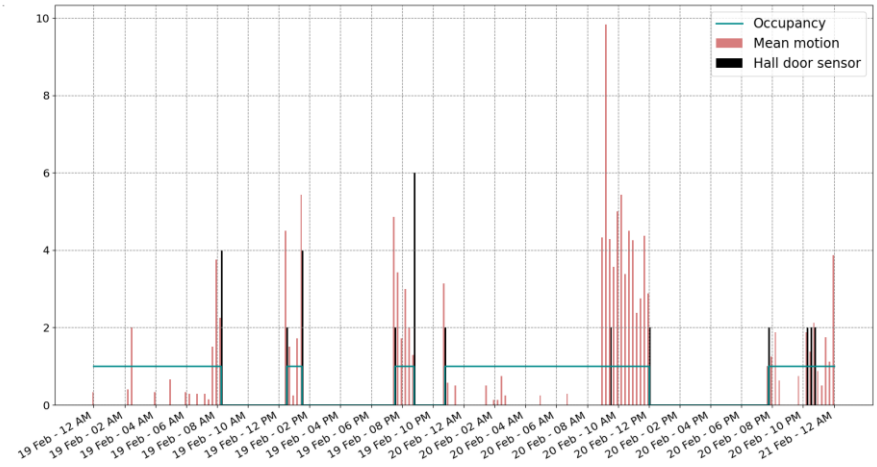

Fig. 8. Occupancy detection using motion sensing and apartment entrance door state changes

The references to relevant motion sensors' timeseries are collected using the following query:

```
SELECT ?apartment ?sensor ?timeseries_ref WHERE {
  ?apartment a inbetween:Apartment.
  ?apartment bot:containsElement ?sensor.
  ?sensor bot:hasSubElement ?motionsensor.
  ?motionsensor a inbetween:MotionSensor.
  ?motionsensor inbetween:hasTimeSeries ?timeseries_ref}
```

Another query for obtaining the relevant door sensor time series reference:

```
SELECT ?apartment ?timeseries_ref WHERE {
  ?apartment a inbetween:Apartment.
  ?apartment bot:containsZone ?space.
  ?space bot:adjacentElement ?wall.
  ?wall a inbetween:InternalWall.
  ?wall  inbetween:hostsElement ?door.
  ?door inbetween:hostsElement ?sensor.
  ?sensor bot:hasSubElement ?doorsens.
  ?doorsens a inbetween:DoorWindowSensor.
  ?doorsens inbetween:hasTimeSeries ?timeseries_ref
}
```

As can be seen from examples, accessing door and window sensors is more complicated, as they are not simply linked to the respective space. Instead, they are linked to a wall that is located between the apartment space and common corridor.

3.2 Electricity load profiling service

Method. Electricity load profiling service relies on time series clustering methods to detect patterns of users' electricity loads in monitored dwellings.
An extensive set of tests and checks need to be performed prior to using the monitoring data for unattended training of machine learning algorithms due to frequent data gaps that occur in realistic deployments. After cleaning repairing the data, it is prepared for training by doing preprocessing steps. Using the context information stored in the knowledge base, relevant time series are grouped based on location and type of consumption. Type of consumption is extracted from the knowledge base and is inferred based on the type of device connected to the corresponding smart plug. In In-Between, grouping was performed for domestic hot water production (DHW), heating (Radiators) and other consumers are grouped in "Rest" group. The entities and relationships describing the demonstration site down to the level of smart plugs and devices plugged into them enable this fine grained analysis. Thusly grouped time series are then split into daily load curves and machine learning methods are used to cluster daily per-apartment loads based on their similarity. Each apartment's daily load pattern is labeled with a cluster label. **Fig. 9** visualizes detected cluster medians and overlays interquartile ranges to give a better overview of range of daily apartment loads grouped into one cluster.

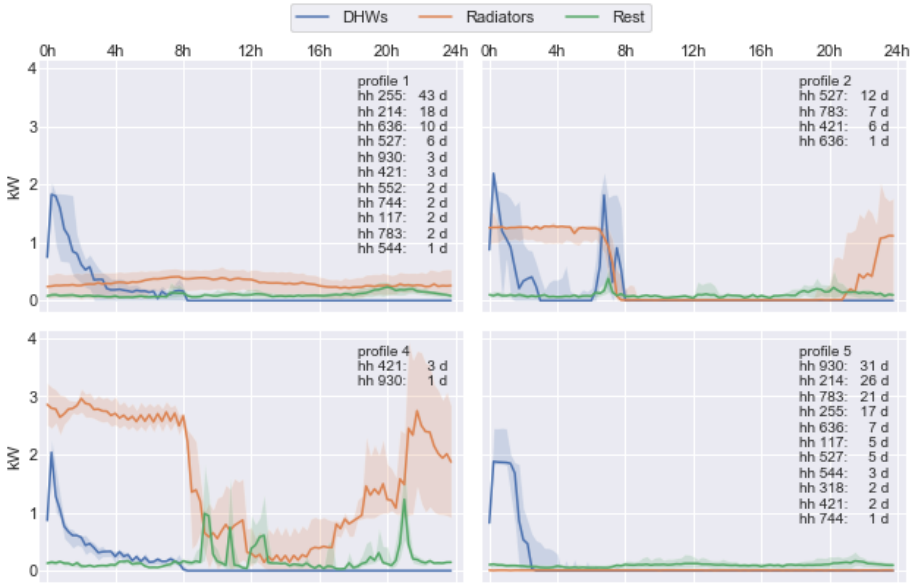

Fig. 9. Clusters of daily electricity loads

Additionally, legend inside each cluster visualization indicates the number of days each of the apartments was represented in that cluster. It is important to note that dis-

covered load profiles are not user-individual profiles, but rather specific load patterns of certain apartments on certain days.

Data needs. It is necessary to collect all different types of loads to be able to group them together in the apartment. In this case we have the total consumption data for each apartment (collected through metering sensor installed on the main electricity meter in the apartment), smart plug sensors installed on radiators and smart cables installed on electricity boilers. "Rest" load group is calculated by subtracting heating and hot water loads from the total load.

Here is an example of a query used to extract metering time series references for all apartments.

```
SELECT ?apartment ?sensor ?subsensor ?timeseries_ref
WHERE {
  ?apartment a inbetween:Apartment.
  ?sensor inbetween:metersZone ?apartment.
  ?sensor bot:hasSubElement ?subsensor.
  ?subsensor a inbetween:PowerSensor.
  ?subsensor inbetween:hasTimeSeries ?timeseries_ref
}
```

and an example of a query for extracting heating loads

```
SELECT ?apartment ?timeseries_ref WHERE {
  ?apartment a inbetween:Apartment.
  ?apartment bot:containsElement ?radiator.
  ?smartplug inbetween:controlsElement ?radiator.
  ?smartplug bot:hasSubElement ?sensor.
  ?sensor a inbetween:PowerSensor.
  ?sensor inbetween:hasTimeSeries ?timeseries_ref
}
```

4 Conclusions and future work

Informal internal evaluation of the implemented ontology concluded it was well suited for the task, but some shortcomings were identified. Performance can be a serious issue, with some queries taking seconds, and even minutes to run. This required tuning of reasoning axioms, splitting the knowledge base into separate graphs to bring performance to levels required for production, and ultimately generating static tables for some services to use to avoid longer delays in operation.

One issue not directly related to the knowledge base itself is tracking of demonstration site changes, which is not well supported by the current design. Scenario where users move sensors around the apartment or connect different devices to smart plugs are realistic and probable. Location with a time dimension would need to be tracked within the knowledge base to accommodate such scenarios. There are some ontolo-

gies which support change tracking and they will be evaluated for usage in scenarios like this. In case of changing the device plugged into the smart plug, the time series information needs to be split at that point in time and new time series thus created needs to be linked with the new device entity in the knowledge base to preserve correctness of the stored data. Possible future solution to this may be automatic detection of significant load profile changes for smart cables or plugs when in activated state and offering suggestions for device update in user-facing applications.

Releasing the project ontology in public domain would be a beneficial step, even though we fear that some of the solutions employed are too project-specific to be of use to the wider audience. Publishing the ontology remains a possibility that we will reevaluate in the future.

ACKNOWLEDGMENT: This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 768776

References

1. Inbetween project homepage, https://www.inbetween-project.eu/.
2. M. Batić, N. Tomašević, S. Vraneš*: IoT based energy efficiency platform architecture design considerations.* 2018.
3. EcoShopping project homepage, http://ecoshopping-project.eu/.
4. Cascade project homepage, http://www.cascade-eu.org.
5. I. Esnaola-Gonzalez, F. J. Diez: *Integrating Building and IoT data in Demand Response solutions.* 2019. Proceedings of the 7th Linked Data in Architecture and Construction Workshop.
6. Cellfie homepage, https://github.com/protegeproject/cellfie-plugin.
7. M. H. Rasmussen, P. Pauwels, M. Lefrançois, J. Karlshøj, G. F. Schneider, C. A. Hviid: *Recent changes in the Building Topology Ontology.* 2017. LDAC 2017.
8. Smart Appliances REFerence (SAREF) ontology, https://sites.google.com/site/smartappliancesproject/ontologies/reference-ontology.
9. M. P. Villalón, R. G. Castro. SAREF extension for building devices. https://ontoology.linkeddata.es/publish/saref4bldg/index-en.html.
10. Building Topology Ontology, https://w3c-lbd-cg.github.io/bot/.
11. SAREF4ENER: an extension of SAREF for the energy domain. ETSI, https://saref.etsi.org/saref4ener/latest/saref4ener.html.
12. M. P. Villalón, R. G. Castro. SAREF extension for environment, http://ontoology.linkeddata.es/publish/saref4envi/index-en.html.
13. TICK Stack - Influx data, https://www.influxdata.com/time-series-platform/.