Linked Data for Building Management

Andreas Fernbach, Igor Pelesic and Wolfgang Kastner

Vienna University of Technology
Institute of Computer Aided Automation
Email: {afernbach, ipelesic, k}@auto.tuwien.ac.at

***Abstract*—In the life cycle of buildings, plenty of different disciplines and companies are involved. Therefore, a unified information base for the whole building and its components and technical equipment is highly beneficial. One approach aiming at this goal is Building Information Modelling (BIM). There exist standards mostly focusing on architectural and building physical properties, like the Green Building XML schema (gbXML). A further improvement would be to include a description of the technical equipment and building automation components as well, and to make the resulting model accessible in a unified way. This paper aims at a holistic knowledge base for buildings providing views on static building and automation systems configuration as well as runtime information, i.e., access to live values of datapoint originating in the building automation systems. This includes sensor values, setpoints, calculated or aggregated values as well as time series of historical values.**

I. Introduction

Ontologies, i.e., the patterns behind a knowledge base, suite well for describing buildings and their components. There is already previous work done in this field like [1] and [2]. [3] provides a broad overview on existing ontologies from the domain. Besides the sophisticated modelling capabilities of ontologies, a knowledge base supports also an easily accessible semantic query (SPARQL) interface by which it allows access to all building related information useful for the whole building lifecycle. Application scenarios enabled by this approach are for example predictive maintenance, energy management and optimisation. A future application might be an intelligent control system taking the knowledge base as an information source to found its decisions on control strategies on it.

The following section gives an overview of the different technologies used in this work. After that, an ontology with the capabilities to express the desired information is explained. Section IV describes a way to automatically set up a building knowledge base using already available information. Finally, with the aid of a prototypical implementation, the concepts introduced in this work are evaluated.

II. Technologies in building automation

Information modelling in the architectural and construction domain [4] together with the concept of establishing an information base throughout a building's lifecycle form a step forward towards better integration and interoperability of the various disciplines involved. While passing the life cycle phases, the model grows with information and becomes a reliable source of knowledge with regard to the different communities. One of the most popular open building information modelling (BIM) standards in the architecture, engineering and construction (AEC) industry is the Green Building XML (gbXML) schema [5]. It is designed as a unified data exchange format between CAD and design tools and tools for energy. Leading software products like AutoCAD Architecture and EnergyPlus from Autodesk or Bentley Architecture and Bentley AECOsim Energy Simulator provide support for gbXML. The schema defines a great variety of parameters concerning the building geometry, its architecture and physical properties. Attributes are hierarchically organised and provide attributes for sites, buildings, storeys, materials, rooms, surfaces and openings like doors and windows. Structural information is enriched with properties of building physics, e.g., albedo, glaze, emittance and U-value. An XSD schema formalises and explains all available parameters in gbXML and ensures compatibility between the different software tools.

OPC Unified Architecture [6] is an open integration standard with the goal to establish interoperability between devices and networks of different technologies and vendors. It is the successor of the classic OPC standard which is still widely used in industry. OPC UA encompasses significant improvement compared to the classical OPC, mainly with respect to data transport mechanisms based on Web Services following a client-server-architecture and comprehensive information modelling capabilities. Within the so called OPC UA address space which is based on a system of nodes and references, information models can be instantiated. An information model in OPC UA can be seen as a description of how runtime data is represented in a distinct technology. An OPC UA server exposing an information model to an OPC UA client enables the latter to access runtime data of the underlying technology without the necessity of having any knowledge about its protocol specifics and the organisation of data. This way, a unified and consistent view to data is provided to an OPC UA client, independent of the underlying technologies.

The Open Building Information Exchange (OBIX) specification [7] is maintained by the Organization for the Advancement of Structured Information Standards (OASIS). The first specification oBIX 1.0 has been published in December 2006. The main goal of oBIX is to provide an open Web service interface for accessing any kind of building automation systems. As a platform independent technology, oBIX can be used on top of any existing technology. To exchange data, oBIX defines a small set of Web services that can be used over SOAP or HTTP binding. The object model is concise but very flexible due to its support for object-oriented concepts. Currently, the revised version oBIX 1.1 is under development. It mainly consists of bug fixes as well as some additional features like a binary encoding scheme for the used object model.

The most relevant technologies in the context of the semantic web are the Resource Description Framework (RDF) [8], the Resource Description Framework RDFS Schema [9], the Web Ontology Language (OWL) [10], the Rule Interchange Format (RIF) [11] and the SPARQL Protocol and RDF Query Language (SPARQL) [12]. RDF was developed to structure information in graphs for representation of knowledge and information exchange on the WWW. As such, RDF builds its structure on subject-predicate-object triples. The RDF Schema standard, as an extension of RDF adds a first set of semantic annotations to RDF and defines concepts for describing the semantic meaning of data represented in RDF. A further step into the direction of a semantically rich description of data on the Web in form of an ontology is OWL. OWL is defined as a description language to represent meanings of entities on the WWW as general as possible. This description facilitates information integration and allows sophisticated reasoning on stored knowledge. RIF is a standard to exchange logical rules between different rule systems and languages. Rules are hereby used to expressively describe knowledge about terms. As a query language, SPARQL is the same for an RDF triple-store as the SQL language for relational databases or XQUERY to an XML database.

The IEC 81346-1 [13] defines principles for structuring objects, i.e., components in plants and buildings. Following these structures, ISO/TS 81346-3 [14] further defines rules how to form reference designations in order to uniquely identify an object within a plant or a building. This results in a systematic naming convention providing information about an embedded object's functionality, location and product related aspect. The benefits of such a reference designation system are the easy identification of systems and their parts and the use of a common language for aligning information about a whole plant containing of an enclosing building structure and the technical equipment. A reference designation describes one up to three of the previously called aspects. Syntactically, each one is signed by a prefix, "=" for the functional aspect, "-" for the product aspect and "+" for the local aspect. Standardised identification letters following the prefixes are used to classify the intended application of an object. The location can be signed by an individual systematic. Considering a motor (M01) driving a pump (G01) of a cooling system (E03) located in room number 4 of building A, the reference designation would look like the following: `-E03-G01-M01+A+4` where the hierarchical structure is represented in a top-down manner.

III. An Ontology for a holistic knowledge base

Pieces of information in a knowledge base are inherently put in relation to each other in a matter that it correlates to the reality it is intended to describe. For this reason, it is also called linked data. [15] It is represented in form of a graph built on a system of subject-predicate-object relations. Subject and objects are entities whereas predicates constitute the type of relations among them. Additionally, typed entities, and also restrictions on relations like a domain and a range as well as cardinalities are supported.

The ontology used in this work mainly builds upon the ThinkHome ontology [16]. Its "Energy and Resources" part describes the technical equipment of a building like the heating, ventilation and air conditioning system (HVAC) as well as lighting applications. The control systems model, i.e. the description of building automation components based on different technologies, includes a fine-grained state model to describe the physical state of the processes and physical quantities measured and influenced by devices.

In order to realise a flexible, consistent and fine-grained access control concept it appeared most convenient to implement the necessary mechanisms within the knowledge base. To this aim, the Social Semantic SPARQL Security for Access Control (S4AC) [17] vocabulary has been integrated into the ontology and properties representing live values of building automation resources have been enriched with concepts defined in S4AC. The S4AC is based on Access Conditions formulated as SPARQL ASK queries. If such a query evaluates as true, the access condition is said to be verified. The so called Access Evaluation Context models the relation between resources and requesting users. By Access Tagging Rules the link between Access Conditions, Access Evaluation Contexts and privileges (read, update, delete, create, append) is established. A tag set defines the names of RDF graphs which are concerned by the Access Conditions. Having these concepts integrated into the ontology, it is possible to define access policies of arbitrary complexity which scale from single datapoints to any named RDF (sub-) graph.

The knowledge base following this ontology is instantiated based on information from the following sources:

Static information like architectural, geometrical, building topology, building physics properties originates from a building information model written in gbXML. A respective mapping from gbXML to an OWL model has already been defined in [16]. Another static information source included in the knowledge base is for some parts the configuration of building automation systems. Therefore, device locations, functionalities and datapoint descriptions are considered. Logical connections between devices and resulting distributed applications are neglected for the moment. The device and datapoint semantics, i.e., the underlying physical quantity, engineering units and value ranges, included in the knowledge base are gathered from OBIX and OPC UA information models. These models can be exposed by integration servers providing unified access to legacy building automation resources like KNX [18], LonWorks [19] and BACnet [20]. On the other hand, Internet of Things (IoT) devices can for example natively host an OBIX server.

The previously introduced reference designation system defined in IEC 81346 is used to achieve convergence in both static information model domains which is essential for a

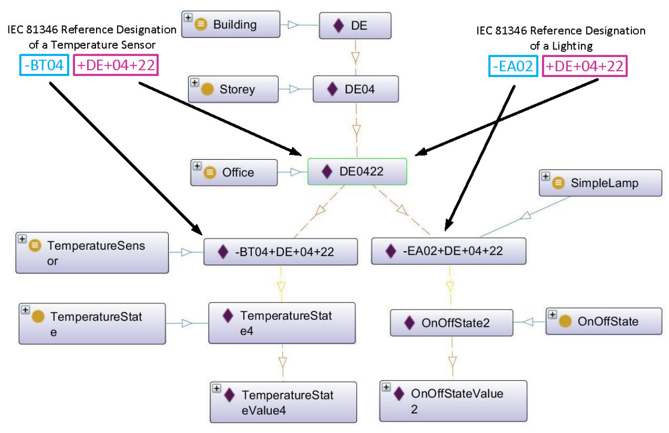


Fig. 1. Building and Device Ontology Instance aligned by IEC 81346 Reference Designations

following automated knowledge base instantiation. This is described in the following based on an example.

We consider a temperature sensor located in room number 22 of the fourth floor of building DE. The resulting reference designation of this sensor is `-BT04+DE+04+22` where `-BT04` describes the product aspect of being a temperature sensor and `+DE+04+22` the location aspect. Provided that in the devices' OBIX contract as well as in the gbXML model the same building topology naming scheme is used, the temperature sensor can be automatically associated to the respective room, floor and building. This principle is illustrated in the ontology snippet shown in Figure 1 where the yellowish circles denote OWL classes and the purple diamonds OWL instances. The instances are organised in a hierarchical way with a building individual on the top. The fourth floor of this building contains a room labelled DE0422. A temperature sensor instance labeled `-BT04+DE+04+22` is assigned to this room via a `IsIn` reference. A textual description of the device instance (e.g., Room Temperature Sensor) can be given as an OWL annotation.

A workflow based on reference designation systems throughout the whole instrumentation and control engineering process as part of BIM still needs to be elaborated in detail.

The goal of this knowledge base concept for building management is not only to consolidate static building information but also runtime data from the automation systems deployed in a building. Datapoints relevant for building management shall also be included in the knowledge base. To this aim, instances of automation components in the knowledge base also encompass state information representing the actual operating state of a device. In Figure 1 the `-BT04+DE+04+22` temperature sensor exposes a state `TemperatureState4` and a value `TemperatureStateValue4` (an ontology design concept defined in [16]). Analogously, the lighting `-EA02+DE+04+22` has a state `OnOffState02` and a value `OnOffStateValue02`. The state value individuals are kept up to date with the actual sensor or actuator value during runtime.

In order to establish a logical connection between the state value representation in the knowledge base and the actual datapoint in the underlying automation system, the state value individual is annotated with the corresponding datapoint address. This datapoint address can be seen as a link which can be followed by a runtime implementation and the state value's data property is updated on physical value changes. In the following example, this concept is shown for the already introduced temperature sensor and the lighting `-EA02+DE+04+22` located in the same room. In order to demonstrate the mapping from two integration standards used in the building automation domain, this process is illustrated by means of the temperature sensor assumed to have an OBIX interface and the lighting being integrated via OPC UA. At the moment, there do not exist general mapping schemes from OBIX contracts and OPC UA object types to the OWL knowledge base representation because of the high diversity in automation components where each type needs to be treated individually. Nevertheless, for a number of further field devices (among others, a presence detector, an air quality meter and a sunblind actuator) a mapping from OBIX and OPC UA to OWL has been defined.

The following listing shows an OBIX object of a temperature sensor:

```
<obj name="—BT04+DE+04+22" href="/hvac/Sensors/
    TemperatureSensor04" displayName="RoomTemperature">
  <real name="value" href="value" val="24.23" unit="obix:
      units/celsius"/>
</obj>
```

It contains the name of the object, the destination of the resource (`href`) and a human readable display name. Besides the current measurement value, also the engineering unit is provided. The corresponding knowledge base individual in RDF/XML notation representing a state value (i.e., a datapoint object) looks like the following:

```
<owl:NamedIndividual rdf:about="&SeWoA;
    TemperatureStateValue4">
  <rdf:type rdf:resource="&SeWoA;TemperatureStateValue"/>
  <SeWoA:realStateValue rdf:datatype="&xsd;float">24.23f</
      SeWoA:realStateValue>
  <SeWoA:hasOBIXAddress rdf:datatype="&xsd;string">http
      ://128.130.56.51:8080/hvac/Sensors/
      TemperatureSensor04</SeWoA:hasOBIXAddress>
  <SeWoA:hasNativeUnit rdf:datatype="&xsd;unitEnum">Celsius
      </SeWoA:hasNativeUnit>
</owl:NamedIndividual>
```

Like shown in Figure 1 this `TemperatureState-Value4` individual is embedded in the context of another individual (`-BT04+DE+04+22`) standing for the physical temperature sensor. The state value individual shown in the above listing provides, besides type information, all the attributes exposed by the OBIX object. The present datapoint value is exposed by the `SeWoA:realStateValue` element. Via the `SeWoA:hasOBIXAddress` element containing the URL of the OBIX object, the link from to the knowledge base individual to the associated live OBIX datapoint is realised.

Like for OBIX, the OPC UA specification defines a standardised XML schema (XSD) for an XML format in which

OPC UA information models can be saved. Usually, OPC UA modelling tools export information models in XML files of this format. OPC UA servers can load such XML files and instantiate their address spaces which are then exposed to OPC UA clients accordingly. In the following, an excerpt of such an XML representation describing an OPC UA variable node is shown. Variable nodes are used in OPC UA to model datapoints. Besides a `BrowseName` attribute, which is set to the reference designation of the lighting `-EA02+DE+04+22`, there is also a human-readable localised DisplayName attribute reading `LightOnOffState`. The actual value is of type `Boolean` and currently set to `false`.

```
<Node i:type="VariableNode">
  <NodeId>
    <Identifier>ns=3;i=325433</Identifier>
   </NodeId>
  <NodeClass>Variable_2 </NodeClass>
  <BrowseName>
    <NamespaceIndex>5</NamespaceIndex>
    <Name>-EA02+DE+04+2</Name>
   </BrowseName>
  <DisplayName>
    <Locale>en</Locale>
    <Text>LightOnOffState </Text>
   </DisplayName>
...
  <Value>
    <Value>
      <Boolean>False </Boolean>
     </Value>
   </Value>
...
</Node>
```

The XML fragment describing the OWL individual resulting from the previous OPC UA variable node is shown in the following:

```
<owl:NamedIndividual rdf:about="&SeWoA;OnOffStateValue2">
  <rdf:type rdf:resource="&SeWoA;OnOffStateValue"/>
  <SeWoA:realStateValue rdf:datatype="&SeWoA;
        OnOffStateValueEnum">Off</SeWoA:realStateValue>
  <SeWoA:hasOPCUAAddress rdf:datatype="&SeWoA;OPCUAAddress
        ">ServerURL="opc://localhost:4840" NamespaceURI=www.
        auto.tuwien.ac.at IdentifierType=Numeric Identifier
        =325433</SeWoA:hasOPCUAAddress>
</owl:NamedIndividual>
```

Analogously to its temperature sensor datapoint sibling, this `OnOffStateValue2` datapoint individual is also referenced by a device individual (`-EA02+DE+04+22`) representing the actual lighting. Besides the type information and the actual state value, there is an XML element containing the address information necessary to access the associated OPC UA variable node. It can be seen as a container for a network-wide node address consisting of the server URL, and the OPC UA NodeId consisting of the `NamespaceURI`, the `IdentifierType` and the numeric `Identifier` itself.

IV. AUTOMATED KNOWLEDGE BASE INSTANTIATION

In order to achieve an efficient way of carrying out the previously described mapping, this information transformation process shall be automated. Since all the involved information encodings are based on XML, i.e., the gbXML, the OBIX and OPC UA objects as well as the resulting OWL individual, the natural technology choice for this purpose is an XSL transformation (XSLT) [21]. The two major tasks of such an XSLT are on the one hand to amalgamate information from a BIM and automation technology in order to embed devices and datapoints into a building topology, and on the other hand to include address information in the resulting knowledge base individuals in order to make them transparent for physical access to live process data. XSLT stylesheets have been implemented to transform OBIX objects representing a number of basic building automation components.

Figure 2 shows the overall concept and information flows including building information modelling, building automation systems configuration and runtime communication. Generally in this figure, information flow regarding configuration is illustrated by blue, curved arrows whereas the yellow, straight arrows represent runtime communication flow. The hierarchy of automation system components, also known as the automation pyramid, is illustrated in form of a large triangle. At the bottom of this hierarchy, there are on the one hand IoT devices and on the other hand in form of a second, inner triangle, automation components of legacy building automation technologies like KNX, Lonworks and BACnet. An integration server on top of these legacy systems provides unified, technology-independent access by means of OBIX or OPC UA. Contrary to that, IoT automation devices natively provide representational state transfer (REST) interfaces like OBIX. In the future, additional REST mappings might be established like for KNX [22]. The building knowledge base located on top of this hierarchy exchanges runtime information with the underlying systems via these OBIX or OPC UA interfaces, respectively. A SPARQL endpoint allows to apply semantic queries on the knowledge base. A semantic query which can be easily formulated in SPARQL is e.g. “Which rooms have a CO_2 level above a distinct threshold?” The answer would be a list of affected rooms including the CO_2 concentration.

The blue, curved arrows in the left part of Figure 2 show the information flow regarding static configuration. The central point is the XSLT following the transformation process described previously. Its output is transferred into the RDF triple store of the building knowledge base. The XSLT operates on models and instances level in order to provide not only an image of the actual building and its technical equipment to the knowledge base but also type information and type hierarchies which are essential for semantic reasoning on the knowledge base. Depending on which technology is in use, on model level OBIX, OPC UA and gbXML information models are taken as an input. The topology and other information regarding the actual building in consideration are taken from a gbXML file. OBIX or OPC UA XML files reflecting the automation systems configuration, i.e., the automation systems instances, are taken from the respective integration servers (OBIX or OPC UA) or native IoT devices. This is illustrated in Figure 2 by the blue, curved arrow originating in the automation pyramid where OBIX an OPC UA devices are located. The IEC81346 symbol overlaying the OBIX / OPC UA and the gbXML instances emphasises that both information sources

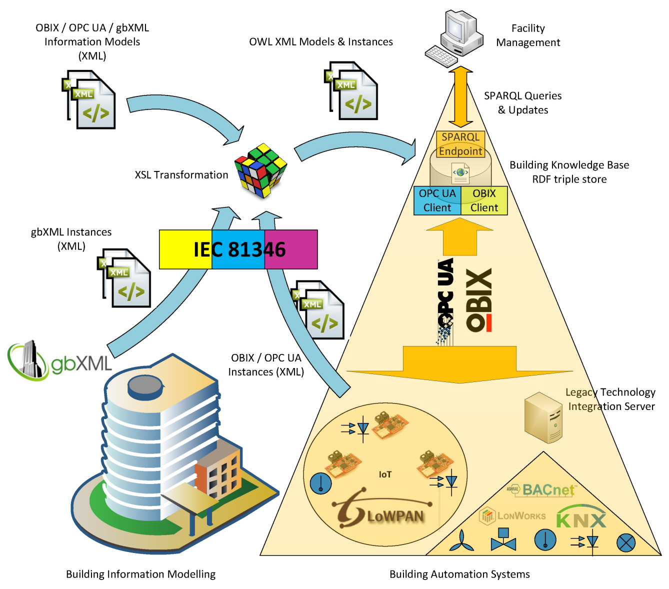

OBIX / OPC UA / gbXML Information Models (XML)
OWL XML Models & Instances
Facility Management
SPARQL Queries & Updates
SPARQL Endpoint
Building Knowledge Base RDF triple store
OPC UA Client
OBIX Client
XSL Transformation
gbXML Instances (XML)
IEC 81346
OPC UA
OBIX
gbXML
OBIX / OPC UA Instances (XML)
Legacy Technology Integration Server
IoT
LoWPAN
BACnet
LonWorks
KNX
Building Information Modelling
Building Automation Systems

Fig. 2. Information Flow and Prototype Architecture

must follow a common naming convention defined by this standard. Provided that this naming convention is taken into account, the XSLT is able to align the information gathered from building information modelling and automation systems engineering resulting in an embedding of the automation components in the building topology like shown in Figure 1.

V. Implementation of a Building Knowledge Base

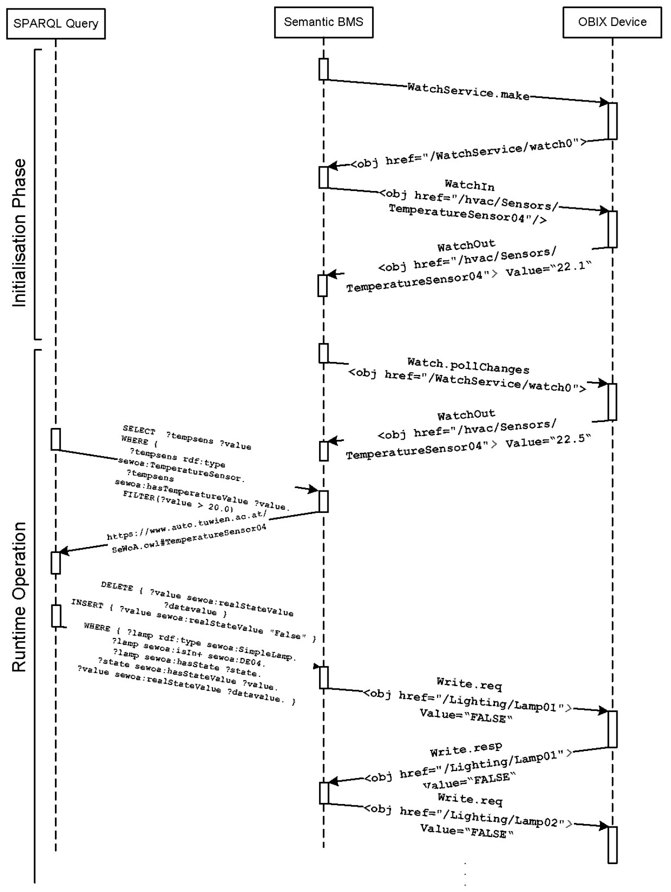
For a prototypical implementation of the knowledge base, the Java based open source Semantic Web framework Apache Jena [1] has been chosen as a basis. The prototypes architecture, like already mentioned, can be seen on top of the automation pyramid in Figure 2. As a test environment for this prototype, Zolertia [2] IoT devices as well as an OBIX server running the integration middleware IoTSys [3] has been used. The Apache Jena framework provides an RDF triple store, which is instantiated by means of the previously generated XML files, the output of the XSLT. It also comes with a SPARQL endpoint which allows executing semantic queries and updates on the RDF triple store. The RDF triple store holds a knowledge representation of the building topology and the automation systems configuration. Additionally to this static information, it shall also incorporate a live image of the underlying automation systems datapoints. This is achieved by an OPC UA as well as an OBIX client module which communicate with the triple store via an API. Via their network interfaces they establish connections to the underlying OBIX and OPC UA resources. More precisely, OPC UA and OBIX client modules register subscriptions on underlying datapoints corresponding to the knowledge base individuals instantiated in the triple store. The data properties in the RDF triple store and the underlying datapoints are mutually synchronised, either via the integration servers (OBIX and OPC UA) or by directly interacting with IoT-enabled devices. Figure 3 illustrates the interaction of the semantic BMS with an underlying OBIX resource, i.e., data synchronisation in upwards and downwards direction. The OBIX or OPC UA addresses necessary to access the correct resources are gained from the `hasOBIXAddress` or from the `hasOPCUAAddress` properties of the respective OWL individuals. After an initialisation phase, where the building knowledge base creates an OBIX watch object on the OBIX server, it registers all OBIX objects corresponding to OWL individuals for change-of-value observation by calling a `WatchIn` operation. In this example, this is only shown for the OBIX `TemperatureSensor04` object corresponding to the `-BT04+DE+04+22` OWL individual. The OBIX device acknowledges the `WatchIn` request by a `WatchOut` object containing the current value of the respective OBIX object. From now on, the building knowledge base continuously polls for value changes. If this is the case, the OBIX client gets informed and updates the value of the `hasNativeValue` data property of the datapoint individual via the API of the tripple store. Value buffering achieved this way is necessary for performance reasons with respect to SPARQL queries traversing a huge number of datapoint individuals for their value. Otherwise, if the current values would have to be requested from the OBIX and OPC UA resources when the query is issued, a considerable load in the underlying building automation network would be the result. This would in these cases lead to an unreasonable response time of the query.

In Figure 3, two exemplary SPARQL queries are shown. The first one requests for all temperature sensors with a reading greater than 20.0 degrees. The query returns the `-BT04+DE+04+22` individual and the buffered value of its associated data property. It is also possible to update states of the automation components by means of SPARQL queries which is shown in the following query in the same figure. The meaning of this query is to perform a central-off to all lights in the fourth floor `DE04`. Such a write access is realised using the SPARQL Update language by first removing the respective data properties by a DELETE statement followed by an INSERT statement which creates new data properties with the new `False` value. For every updated individual, the event listener of the RDF triple store API triggers the OBIX client to send write requests containing the new value to the regarding OBIX objects on the underlying resource.

VI. Conclusion and Outlook

The result of this work must be seen as a first step of using semantic web technologies for the automated integration of building automation systems. A workflow based on reference designation systems throughout the whole instrumentation and control engineering process as part of BIM still needs to be elaborated in detail. However, a concept of a holistic knowledge base encompassing building information, automation systems and actual process values was introduced. Moreover, a workflow of aligning information originating

[1] https://jena.apache.org/
[2] http://www.zolertia.com
[3] https://github.com/mjung85/iotsys


SPARQL Query
Semantic BMS
OBIX Device
Initialisation Phase
Runtime Operation
WatchService.make
<obj href="/WatchService/watch0">
WatchIn
<obj href="/hvac/Sensors/
TemperatureSensor04"/>
WatchOut
<obj href="/hvac/Sensors/
TemperatureSensor04"> Value="22.1"
Watch.pollChanges
<obj href="/WatchService/watch0">
WatchOut
<obj href="/hvac/Sensors/
TemperatureSensor04"> Value="22.5"
Write.req
<obj href="/Lighting/Lamp01">
Value="FALSE"
Write.resp
<obj href="/Lighting/Lamp01">
Value="FALSE"
Write.req
<obj href="/Lighting/Lamp02">
Value="FALSE"

```
SELECT  ?tempsens ?value
WHERE {
  ?tempsens rdf:type
sewoa:TemperatureSensor.
  ?tempsens
sewoa:hasTemperatureValue ?value.
  FILTER(?value > 20.0)
```

```
https://www.auto.tuwien.ac.at/
SeWoA.owl#TemperatureSensor04
```

```
DELETE { ?value sewoa:realStateValue
                 ?datavalue }
INSERT { ?value sewoa:realStateValue "False" }
WHERE { ?lamp rdf:type sewoa:SimpleLamp.
        ?lamp sewoa:isIn+ sewoa:DE04.
        ?lamp sewoa:hasState ?state.
      ?state sewoa:hasStateValue ?value.
   ?value sewoa:realStateValue ?datavalue. }
```

Fig. 3. Runtime Interaction of a Building Knowledge Base with an OBIX device

from different sources, namely topological building models (BIM) and automation systems OBIX or OPC UA interfaces using the IEC 81346 reference designation system has been sketched. This information transformation process has been realised by means of an XSLT which has been empirically evaluated for a limited set of entities. In order to show runtime interworking of this automatically instantiated knowledge base with actual automation components, an Apache Jena-based prototype equipped with OBIX and OPC UA client functionality has been implemented. An open source project named OpenKB4BMS[4] including both the transformation stylesheets and the knowledge base implementation has been published on GitHub.

Future work following this attempt should also consider the temporal aspect of runtime data by means of adding time stamps to the datapoint model. Furthermore, it is imaginable, as shown in [23], to represent time series of data within the knowledge base which would be valuable for use cases regarding diagnostics.

[4]https://github.com/afernbach/openKB4BMS

Acknowledgement

This work was funded by FFG (Austrian Research Promotion Agency) under the project "Kognitive Regelstrategieoptimierung zur Energieeffizienzsteigerung in Gebäuden KORE", FFG 848805.

References

[1] D. Bonino and F. Corno, *The Semantic Web - ISWC 2008: 7th International Semantic Web Conference, ISWC 2008, Karlsruhe, Germany, October 26-30, 2008. Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, ch. DogOnt - Ontology Modeling for Intelligent Domotic Environments, pp. 790–803.

[2] M. Kofler, *An ontology as shared vocabulary for distributed intelligence in smart homes*. Vienna, Austria: PhD Thesis, Vienna University of Technology, 2013.

[3] M. Grassi, M. Nucci, and F. Piazza, "Ontologies for smart homes and energy management: An implementation-driven survey," *2013 Workshop on Modeling and Simulation of Cyber-Physical Energy Systems, MSCPES 2013*, pp. 7–9, 2013.

[4] C. Eastman, P. Teicholz, R. Sacks, and K. Liston, *BIM-Handbook: A Guide to Building Information Modeling for Owners, Managers, Designers, Engineers and Contractors*. Wiley John + Sons, 2011.

[5] "Open Green Building XML Schema: a Building Information Modelling Soltution for Our Green World," 2016. [Online]. Available: www.gbXML.org

[6] "OPC UA Specification, Release 1.02," OPC Foundation, 2012.

[7] "oBIX 1.0 Committe Specification," OASIS, 2006.

[8] "Resource Description Framework (RDF)," W3C Recommendation, 2004. [Online]. Available: http://www.w3.org/TR/rdf-primer/

[9] "RDF Vocabulary Description Language 1.0 RDF Schema," W3C Recommendation, 2004. [Online]. Available: http://www.w3.org/TR/rdf-schema/

[10] "OWL2 Web Ontology Primer (Second Edition)," W3C Recommendation, 2012. [Online]. Available: http://www.w3.org/TR/2012/REC-owl2-primer20121211

[11] "RIF Core Dialect," W3C Recommendation, 2010. [Online]. Available: http://www.w3.org/TR/2010/REC-rif-core-20100622/

[12] "SPARQL Query Language for RDF," W3C Recommendation, 2008. [Online]. Available: http://www.w3.org/TR72008/REC-rdf-sparql-query-20080115

[13] "IEC 81346-1:2009 - Industrial systems, installations and equipment and industrial products Structuring principles and reference designations Part 1: Basic rules," 2009.

[14] "ISO/TS 81346-3:2012 - Industrial systems, installations and equipment and industrial products Structuring principles and reference designations Part 3: Application rules for a reference designation system," 2012.

[15] C. Bizer, T. Heath, and T. Berners-Lee, "Linked data: The story so far," *Semantic Services, Interoperability and Web Applications: Emerging Concepts*, pp. 205–227, 2009.

[16] M. Kofler, "An ontology as shared vocabulary for distributed intelligence in smart homes," Ph.D. dissertation, Vienna University of Technology, 2013.

[17] S. Villata, L. Costabello, N. Delaforge, and F. Gandon, "A Social Semantic Web Access Control Model," *Journal on Data Semantics*, vol. 2, no. 1, pp. 21–36, 2012.

[18] "KNX Specification," ISO/IEC 14543-3, 2014.

[19] "Open Data Communication in Building Automation, Controls and Building Management Control Network Protocol – Part 1-5," ISO 14908-1 - ISO 14908-4, 2008.

[20] "BACnet – A Data Communication Protocol for Building Automation and Control Networks," ANSI/ASHRAE 135, 2012.

[21] "XSL Transformations (XSLT)," W3C Recommendation, 1999. [Online]. Available: https://www.w3.org/TR/xslt

[22] J. Hennebert and G. Bovet, "Introducing the Web-of-Things in Building Automation : A Gateway for KNX installations," *10th international Conference on Informatics in Control, Automation and Robotics (ICINCO 2013)*, no. Icinco, 2013.

[23] C. A. Henson, H. Neuhaus, A. P. Sheth, K. Thirunarayan, and R. Buyya, "An ontological representation of time series observations on the semantic sensor web," 2009.