Reusing Domain Ontologies in Linked Building Data: the Case of Building Automation and Control

Walter TERKAJ [a,1], Georg Ferdinand SCHNEIDER [b] and Pieter PAUWELS [c]
[a] *Institute of Industrial Technologies and Automation (ITIA-CNR), Milan, Italy*
[b] *Fraunhofer Institute for Building Physics IBP, Nuremberg, Germany*
[c] *Ghent University, Ghent, Belgium*

Abstract. Linked data and semantic web technologies are gaining impact and importance in the Architecture, Engineering, Construction and Facility Management (AEC/FM) industry. Whereas we have seen a strong technological shift with the emergence of Building Information Modeling (BIM) tools, this second technological shift to the exchange and management of building data over the web might be even stronger than the first one. In order to make this a success, the AEC/FM industry will need strong and appropriate ontologies, as they will allow industry practitioners to structure their data in a commonly agreed format and exchange the data. Herein, we look at the ontologies that are emerging in the area of Building Automation and Control Systems (BACS). We propose a BACS ontology in strong alignment with existing ontologies and evaluate how it can be used for capturing automation and control systems of a building by modeling a use case.

Keywords. Linked Data, Semantic Web, Building Data, Building Automation Systems, Control Logic

1. Introduction

One of the major challenges in the engineering of complex cyber-physical systems is the interoperability between software applications that are used to engineer and manage these systems. Buildings constitute one of these systems and their engineering requires the interaction of a multitude of stakeholders over several stages of the life cycle [8]. Vast amounts of data are generated during this process. These data are typically stored across various formats and information silos being both in machine-readable formats (e.g. XML, STEP), or analog formats (e.g. drawings printed on paper). These data may include the design specification of the actual layout of a building, product data on the commissioned technical equipment, weather data [27] and sensor data obtained through the installation and operation of a Building Automation System (BAS) [23].

The ontology-based modeling approach and its related use of semantic web technologies seem to be a promising path towards addressing the prevalent heterogeneity of

[1] Corresponding Author: Institute of Industrial Technologies and Automation (ITIA-CNR), Milan, Italy; E-mail: walter.terkaj@itia.cnr.it

data in the Architecture, Engineering, Construction and Facility Management (AEC/FM) industry [5,30,15]. This is mostly undertaken by providing a common semantically well defined layer enabling seamless information exchange and linkage across domains. However, a plethora of ontologies is defined to provide conceptualizations of this domain (see Sect. 2). These ontologies can be used to represent data about buildings in a structured manner and help facilitating the information exchange between different stakeholders.

However, the existence of a multitude of ontologies that partly overlap in the scope of AEC/FM inhibits the wide-spread adoption of the technology throughout the industry. Several ontologies without standardization or consensus can only partially support the AEC/FM industry. Under the umbrella of the World Wide Web Consortium (W3C), the Linked Building Data Community Group (LBD) was initiated in 2016 to address this problem. This group aims at becoming a Working Group and thus effectively develop standard ontologies for the AEC/FM sector. When becoming a Working Group, the following mission will prevail, as listed in the editor draft of the charter [10]:

- to determine how building information can best be integrated with other data on the Web;
- to determine how machines and people can discover that different facts in different datasets relate to the same building, especially when *building* is expressed in different ways and levels of granularity;
- to identify and assess existing methods and tools and then create a set of best practices for their use;
- where desirable, to complete the standardization of informal technologies already in widespread use.

The community group aims at achieving this mission through a number of deliverables. This includes a central ontology (not an upper ontology) that allows to express the building topology of any building (Site - Building - Storey - Space - Element): the Building Topology Ontology (BOT) [20]. In addition, a PRODUCT ontology, a GEOM ontology, and a PSET ontology are being defined, allowing to represent product data, geometric representations, and properties, respectively. The set of four ontologies form the core of the work that is aligned with W3C recommendations for adjacent domains, e.g. the geospatial ontologies [25], SAREF [6], DogOnt [1], and the Semantic Sensor Network (SSN [4]). The group also aligns with the *W3C Data on the Web Best Practices* [9] for the adoption of Semantic Web Technologies (SWT) in the domain of building data.

One subgroup of the LBD Community Group focuses on the formal modeling of Building Automation and Control Systems (BACS). This partially reflects the need to integrate tools supporting the monitoring and automation of buildings being in the need of smart systems to automatically control technical equipment and improve building operation in terms of energy efficiency and indoor comfort.

As a first result of this work we present in this paper a modular ontology where we integrate information from Building Information Modelling (BIM) and BAS: notably building elements, sensors and actuators, devices of BAS and control logic. The work mainly reuses and aligns existing domain ontologies to comprise domain information in one common knowledge base for the control and automation of buildings.

In Sect. 2, we provide an overview on existing domain ontologies regarding smart appliances and BAS. Then in Sect. 3, we present the BACS Ontology. In Sect. 4, we apply the BACS ontology to model an application scenario of state-based room control in a fictional BAS. Finally, we draw the conclusions in Sect. 5.

2. Related Works

In this section we review existing ontology modeling approaches with a special focus on smart appliances and building automation systems. A comprehensive review on the usage of Semantic Web Technologies in the building domain is presented in [15], also including a discussion on ontologies in the BAS domain.

2.1. General Systems and Building Information Modelling (BIM)

The Industry Foundation Classes (IFC) standard[2] comprises a well accepted, open model for the information exchange when applying BIM in the AEC/FM-industry. The data model has already been converted to the Web Ontology Language (OWL) as the ifcOWL ontology [14]. Another set of ontologies including building information is available in the knowledge model for Smart Energy Aware Systems (SEAS) [13]. This includes ontology modules related to building automation and control, such as modules `DeviceOntology`, `OptimizationOntology`, and `FailableSystemOntology`.

An approach to align existing ontologies in the AEC domain is represented by the Building Topology Ontology (BOT) [20] that is part of the work conducted by the LBD Community Group. This ontology can be aligned with several of the mentioned ontologies, using some of the ontology alignment approaches proposed in [20]. In this scope, a BACS ontology is needed as well to model how appliances and devices can be used to automate and control the building and its components.

2.2. BACS and Energy-Related

Several works focused on the energy management of facilities [12], such as airports [30]. An approach for domotics intelligent devices (DogOnt) was proposed by [1]. The integration of buildings with grid and energy market information is tackled in the ThinkHome ontology [21]. An approach to integrate device descriptions on BAS devices, functional specification with adjacent domains of BIM was developed in the BASont ontology [16]. An approach to formalize semantic tags by means of ontology is described in the Haystack Tagging Ontology (HTO) [3]. Among the various ontologies related to BACS and smart appliances, the Smart Appliances REFerence (SAREF) ontology [6] unifies common accepted conceptualisations into one reference ontology; SAREF4BLDG [17] is its extension for the building domain.

2.3. Sensor Data and Control Logic

A well established ontology for the formal specification of sensor data is the W3C Semantic Sensor Network (SSN) ontology [4]. The SSN ontology has been recently updated [11] by including also the more general module SOSA (Sensor, Observation, Sample, and Actuator) making it possible to model key concepts also for the BACS domain, including sensor, actuator, observations, observable properties and results. The SSN ontology is included in the proposed BACS ontology (see Sect. 3). Within the reported approaches it may be observed that typically taxonomies are used to describe the actual control behaviour of a certain control logic in a BACS. A modelling effort to specify

[2] `http://www.buildingsmart-tech.org/ifc/IFC4/Add1/`

UML state machines, a well known modeling approach for state-based control logic, as an OWL ontology was presented by Dolog [7].

2.4. Summary

The reported ontologies form a comprehensive board of ontologies that can be reused to cover the BACS domain by spanning the description of building elements and equipment, sensors and actuators, BAS, and control logic. The ontology reuse, even if often applied in a limited fashion, represents a key best practice that was followed in this work.

3. The Building Automation and Control System Ontology

The proposed BACS ontology aims at supporting the modeling of the following information requirements:

- control behavior in a BAS (both discrete and continuous), including the sense-comprehend-actuate pattern in closed-loop control logic and the comprehend-actuate pattern in open-loop control logic [24];
- physical devices of Building Automation Systems (BAS) and their location in the building as well as affiliation to technical equipment;
- smart appliances;
- logical topology in a BAS.

The architecture of the BACS ontology exploits the ontology *reuse* and *modularity* principles [26]. The modular architecture consists of the following Terminological (Tbox) modules, as shown in Figure 1:

- `statistics`, defining basic concepts about probability distributions and descriptive statistics
- `fsm`, the ontology for Finite State Machine
- `sosa`, the Sensor, Observation, Sample, and Actuator (SOSA) Ontology
- `ssn`, the Semantic Sensor Network Ontology
- `expression`, a novel ontology formalizing algebraic and logical expressions
- `osph`, a novel ontology modeling object states and performance history
- `list`, ontology defining the set of entities used to describe the OWL list pattern
- `express`, ontology that maps the key concepts of EXPRESS language to OWL
- `ifcmr`, fragment of the ifcOWL ontology (version IFC4) generated from the EXPRESS sub-schema `IfcMeasureResource`
- `bot`, the Building Topology Ontology
- `bacs`, a novel domain ontology for building automation and control

All these modules are available on the web (see Table1). Such composition of modules was selected by carefully reviewing candidate ontologies (see Sect. 2) against the following criteria: (1) minimum overlap with each other, (2) possibly W3C standards (e.g. `ssn`) and (3) meeting the defined information requirements.

The ontology modules `fsm`, `sosa`, `ssn`, and `bot` were already mentioned in Sect. 2.

The group of modules `list`, `express`, and `ifcmr` supports the definition of quantity values. This ontology is a fragment of the ifcOWL ontology [28] generated from the IFC

sub-schema `IfcMeasureResource` according to the method presented in [28]. These modules could be replaced by other similar ontologies, such as the Ontology of Units of Measure (OM) [22] or the `QUDT` ontology [19].

The `statistics` module is a minimal ontology defining the basic classes and properties needed to model data related to descriptive statistics and probability distributions. As an alternative, the larger and more complex STATO ontology [2] could be adopted.

The three novel modules proposed in this work (i.e. `expression`, `osph`, `bacs`) are described in Sect. 3.1, whereas the overall integration and alignment is presented in Sect. 3.2. The prefixes of the namespaces are defined in Table 1.

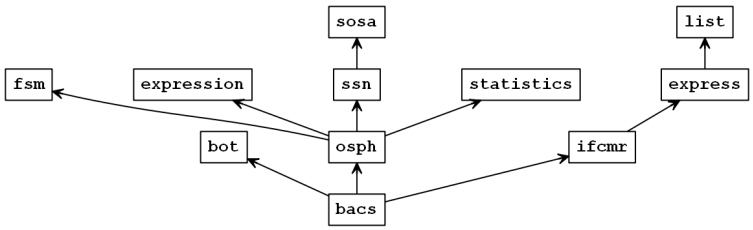

Figure 1. Modular architecture of the BACS Ontology. Arrows represent import relations.

Table 1. Namespaces and prefixes

Prefix	Value	Prefix	Value
bacs	http://www.ontoeng.com/bacs#	owl	http://www.w3.org/2002/07/owl#
bot	https://w3id.org/bot#	rdf	http://www.w3.org/1999/02/22-rdf-syntax-ns#
expression	http://www.ontoeng.com/expression#	rdfs	http://www.w3.org/2000/01/rdf-schema#
expr	https://w3id.org/express#	sosa	http://www.w3.org/ns/sosa/
fsm	http://www.learninglab.de/∼dolog/fsm/fsm.owl#	ssn	http://www.w3.org/ns/ssn/
ifcmr	http://www.ontoeng.com/IFC4_IfcMeasureResource#	statistics	http://www.ontoeng.com/statistics#
list	https://w3id.org/list#	xsd	http://www.w3.org/2001/XMLSchema#
osph	http://www.ontoeng.com/osph#		

3.1. Novel Ontology Modules

The `expression` ontology aims at formalizing algebraic and logical expressions. A generic expression can be decomposed into atomic, unary, and binary expressions. An atomic expression can be a constant or a variable. This ontology was developed to provide a simple formal definition of expressions that can be used also as conditions to be met before a transition is triggered (see Sect. 3.2). The classes and properties of the `expression` ontology are sketched in Figure 2.

The `osph` ontology is the evolution and generalization of an early proposal that was based on the ifcOWL ontology [29]. This ontology plays a key role because it models Object States and Performance History (OSPH), while integrating the `fsm`, `statistics`, `ssn`, and `expression` modules. The following classes are defined in the `osph` ontology:

- `osph:ObjectDefinition` is an abstract class whose definition resembles that of `IfcObjectDefinition` in the IFC standard.
- `osph:ObjectHistory` defines a history interval in the lifecycle of a generic object that is assigned via the object property `osph:isHistoryOf`. An interval can be de-

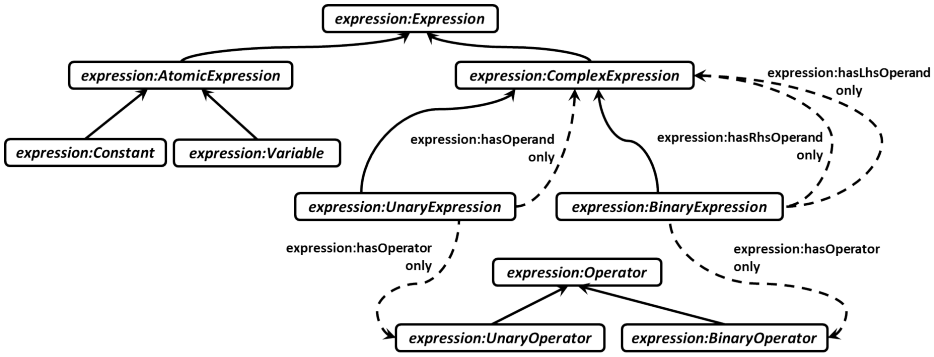

Figure 2. Classes and relations in the `expression` ontology. Dashed lines represent OWL restrictions, whereas solid lines represent subsumption relations.

composed into other intervals via the property `osph:isDecomposedByHistory`. A history interval can be characterized by a start and end time as `xsd:dateTime` via the properties `osph:hasIntervalStartTime` and `osph:hasIntervalEndTime`, respectively. A history interval can be related to individuals of `osph:StateFrequency` via the property `osph:hasStateFrequencies`.

- `osph:StateFrequency` describes the stay of an object in a specific state during a history interval.
- `osph:UnitOfMeasurement` is a class defining a generic unit of measurement.

The relations between these classes are shown in Figure 3, whereas the relations with classes defined in other modules are described in Sect. 3.2.

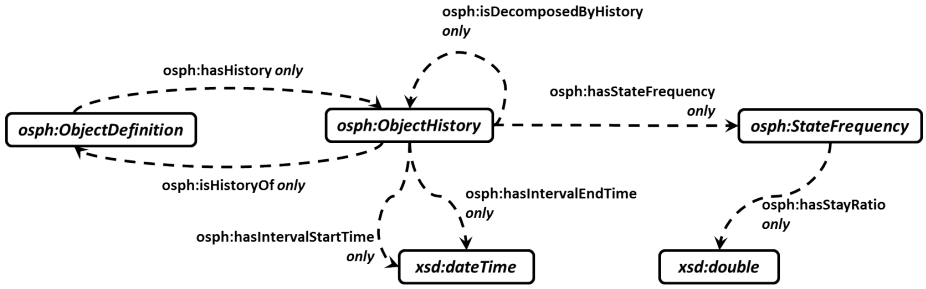

Figure 3. Classes and relations in the `osph` ontology, where dashed lines represent OWL restrictions.

The `bacs` module defines specializations of classes to directly support the instantiation of the use case (Sect. 4). Classes `bacs:LightSensor` and `bacs:TemperatureSensor` represent light and temperature sensors, respectively. The class `bacs:SpaceProperty` and its subclasses `bacs:SpaceIlluminanceProperty` and `bacs:SpaceTemperatureProperty` model specific properties of a space (e.g. a room). The class `bacs:SpaceObs` and its subclasses `bacs:SpaceIlluminanceObs` and `bacs:SpaceTemperatureObs` are designed to represent observations made in a space. In the future, all these classes will be re-assessed to check if they can be replaced by definitions imported from other dedicated domain ontologies.

3.2. Integration

The integration of the various ontology modules represents one of the main contributions in the proposed ontology architecture. Indeed, the novel modules `osph` and `bacs` mainly play the role of ontology mediator creating links between different domains, as it can be noticed in the diagram of Figure 1. The alignments between the various ontology modules are reported in Table 2 by specifying which module is the mediator (i.e. where the alignment is defined), which are the aligned modules, which are the involved classes and properties, and finally providing a description. A generic object (`osph:ObjectDefinition`) can be characterized by a state machine (`fsm:StateMachine`) and also by one or more histories (`osph:ObjectHistory`) that are able to capture the evolution of the object in terms of observations and state. An expression (`expression:Expression`) can be composed by constant values (`ifcmr:IfcValue`) and variables (`expression:Variable`) that are related to measurable properties (`ssn:Property`). The result of an observation (`sosa:Observation`) can be a descriptive statistics (`statistics:DescriptiveStatistics`) or a quantity value (`ifcmr:IfcValue`). The relevant alignments are also graphically represented in Figure 4.

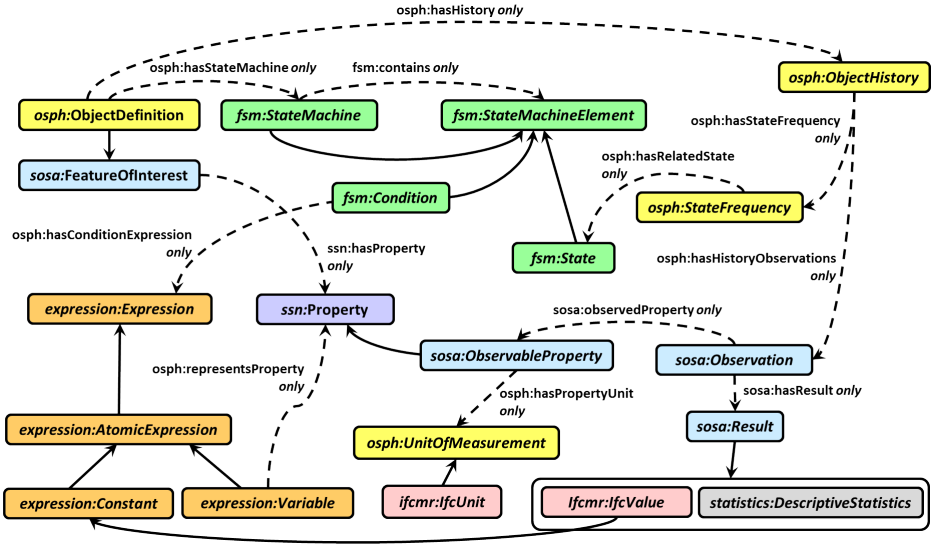

Figure 4. Excerpt of classes and relations in the BACS Ontology showing the key alignments. Dashed lines represent OWL restrictions, whereas solid lines represent subsumption relations.

4. Application Scenario

4.1. Description

The applicability of the proposed BACS ontology was tested against an application scenario that includes the instantiation of building elements, sensors and actuators, automation systems, and control logics. The scenario is motivated by the deployment of auto-

Table 2. Alignments between the modules of the BACS ontology.

Mediator	Modules	Classes and Properties	Description
osph	sosa,osph	osph:ObjectDefinition, sosa:FeatureOfInterest	osph:ObjectDefinition is defined as a subClassOf sosa:FeatureOfInterest.
osph	fsm,osph	osph:ObjectDefinition, fsm:StateMachine, osph:ObjectHistory; osph:hasStateMachine, osph:hasHistory	An individual of osph:ObjectDefinition can be linked with an individual of fsm:StateMachine via the property osph:hasStateMachine, and with individuals of osph:ObjectHistory via property osph:hasHistory.
osph	fsm,osph	osph:StateFrequency, fsm:State; osph:hasRelatedState	An individual of osph:StateFrequency is related with an individual of fsm:State via the property osph:hasRelatedState.
osph	sosa,osph	osph:ObjectHistory, sosa:Observation; osph:hasHistoryObservations	A history interval osph:ObjectHistory can be related to individuals of sosa:Observation via the property osph:hasHistoryObservations.
osph	expression, fsm,osph	fsm:Condition, expression:Expression; osph:hasConditionExpression	An individual of class fsm:Condition can be related to an individual of expression:Expression via the object property osph:hasConditionExpression.
osph	expression, ssn,osph	expression:Variable, ssn:Property; osph:representsProperty	An individual of class expression:Variable can be related to an individual of ssn:Property via the object property osph:representsProperty, i.e. a variable can be used to represent the property of a feature.
bacs	bot,osph	osph:ObjectDefinition, bot:Building, bot:Space, bot:Element, bot:Storey	bot:Building, bot:Space, bot:Element, bot:Storey are defined as subClassOf osph:ObjectDefinition.
bacs	bot,sosa	bot:Element, sosa:Sensor	sosa:Sensor is subClassOf bot:Element.
bacs	sosa,bacs	sosa:Sensor,bacs:LightSensor, bacs:TemperatureSensor	bacs:LightSensor and bacs:TemperatureSensor are subClassOf sosa:Sensor.
bacs	bacs,bot, osph	osph:ObjectHistory, bot:Space, bacs:SpaceObs, bacs:SpaceHistory; osph:hasHistoryObervations, osph:isDecomposedByHistory, osph:isHistoryOf	bacs:SpaceHistory is subClassOf osph:ObjectHistory and further specializes the restrictions characterizing osph:ObjectHistory by means of properties osph:hasHistoryObervations, osph:isDecomposedByHistory, osph:isHistoryOf and classes bacs:SpaceObs, bot:Space, bacs:SpaceHistory.
bacs	osph,ifcmr	osph:UnitOfMeasurement, ifcmr:IfcUnit	ifcmr:IfcUnit is subClassOf osph:UnitOfMeasurement.
bacs	sosa,bacs	sosa:ObservableProperty, bacs:SpaceProperty	bacs:SpaceProperty is subClassOf sosa:ObservableProperty.
bacs	sosa,bacs	sosa:Observation,bacs:SpaceObs	bacs:SpaceObs is subClassOf sosa:Observation.
bacs	statistics, ifcmr, sosa	statistics:DescriptiveStatistics, ifcmr:IfcValue, sosa:Result	sosa:Result is a subClassOf the union of ifcmr:IfcValue and statistics:DescriptiveStatistics, i.e. the result of an observation must be either a value or a statistics.
bacs	expression, ifcmr	expression:Constant, ifcmr:IfcValue	ifcmr:IfcValue is a subClassOf expression:Constant.

matic control in the room of a building. The focus is on control logic because it can significantly impact on the energy consumption and comfort conditions.

In the scenario, a room is equipped with a window, a controllable sunblind, a room air temperature sensor and an outdoor illuminance sensor. A finite state machine is designed to control the sunblind depending on the room air temperature and outdoor illuminance, as presented in [18]. The sunblind can be in one of the following states: noShade, nightShadeDeployed, dayShadeDeployed (Figure 5). The following observations have been made in the system:

- room temperature: 20°C at 2017-03-09T08:00:00; 24°C at 2017-03-09T10:00:00
- room illuminance: 90 lx at 2017-03-09T08:00:00; 200 lx at 2017-03-09T10:00:00
- sunblind state: noShade at 2017-03-09T09:30:00

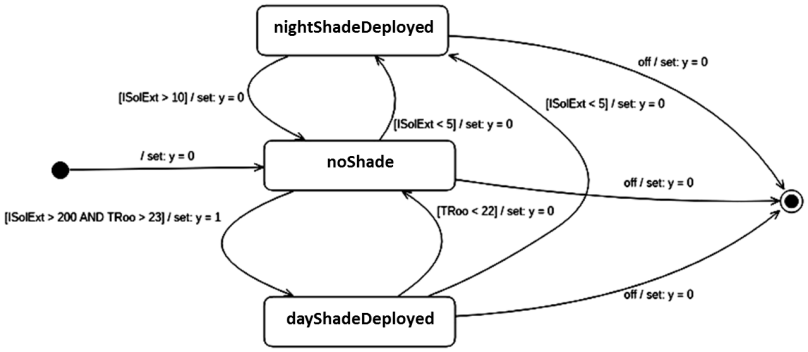

Figure 5. State Machine to control a sunblind in a room [18].

4.2. Instantiation

The Abox ontology module named `bacs_test`[3] imports the `bacs` module and instantiates the application scenario in terms of OWL individuals defined in its namespace. For sake of simplicity, the following fragments in turtle format present only a subset of definitions included in `bacs_test`.

Fragment 1 presents the definition of room, temperature sensor and sunblind. The room is associated with two properties: room temperature and room illuminance. Other definitions include building, building storey, window, and illuminance sensor.

Fragment 2 shows a part of the definition of the finite state machine modeling the control logic of the sunblind. As an example, the details of the transition from state `dayShadeDeployed` to `nightShadeDeployed` are reported. This transition can be triggered only if the guard condition is met, i.e. if the room illuminance is less then 5.0 lx.

Fragment 3 provides an example of how the history of the room can be characterized by observations of its properties thanks to the sensors. In addition, the history of the sunblind considers the evolution of the sunblind state.

Finally, a couple of SPARQL queries are presented to show how the contents of the ontology can be extracted to support business processes, while referring to the prefixes defined in Table 1.

```
:room a owl:NamedIndividual , bot:Space ;
    bot:containsElement :lightsensor , :sunblind , :tempsensor ;
    ssn:hasProperty :room_illuminance_prop , :room_temperature_prop ;
    bot:adjacentElement :window ; osph:hasHistory :room_history .
:tempsensor a owl:NamedIndividual , bacs:TemperatureSensor ;
    sosa:observes :room_temperature_prop .
:sunblind a owl:NamedIndividual , bot:Element ;
    osph:hasHistory :sb_history ; osph:hasStateMachine :sb_StateMachine .
:room_temperature_prop a owl:NamedIndividual , bacs:SpaceTemperatureProperty ;
    osph:hasPropertyUnit :temperature_unit .
:room_illuminance_prop a owl:NamedIndividual , bacs:SpaceIlluminanceProperty ;
    osph:hasPropertyUnit :illuminance_unit .
```

Fragment 1: Excerpt of space and elements instantiation

[3] `http://www.ontoeng.com/bacs_test`

```
:sb_StateMachine a owl:NamedIndividual , fsm:StateMachine ;
    fsm:contains :sb_InitialState, :sb_dayShadeDeployed, :sb_nightShadeDeployed,
        :sb_noShade, :sb_trDayNight, :sb_trDayNo, :sb_trInitNo, :sb_trNightNo,
        :sb_trNoDay, :sb_trNoNight, :sb_trDayNight_guard, :sb_trDayNo_guard,
        :sb_trNightNo_guard, :sb_trNoDay_guard, :sb_trNoNight_guard.
:sb_dayShadeDeployed a owl:NamedIndividual , fsm:Simple .
:sb_nightShadeDeployed a owl:NamedIndividual , fsm:Simple .
:sb_trDayNight a owl:NamedIndividual , fsm:Transition ;
    fsm:Source :sb_dayShadeDeployed ; fsm:Target :sb_nightShadeDeployed ;
    fsm:TransitionGuard :sb_trDayNight_guard .
:sb_trDayNight_guard a owl:NamedIndividual , fsm:Guard ;
    fsm:GuardCondition :sb_trDayNight_condition .
:sb_trDayNight_condition a owl:NamedIndividual , fsm:Condition ;
    osph:hasConditionExpression :sb_trDayNight_expr .
:sb_trDayNight_expr a owl:NamedIndividual , expression:BinaryExpression ;
    expression:hasLhsOperand :room_illuminance_var ;
    expression:hasOperator expression:LESSTHAN ;
    expression:hasRhsOperand :illumA .
:illumA a owl:NamedIndividual , ifcmr:IfcIlluminanceMeasure ;
    express:hasDouble "5.0"^^xsd:double .
```

Fragment 2: Excerpt of sunblind state machine instantiation

```
:room_history a owl:NamedIndividual , bacs:SpaceHistory ;
    osph:isDecomposedByHistory :room_history_int1 , :room_history_int2 ;
    osph:hasIntervalStartTime "2017-03-09T08:00:00"^^xsd:dateTime .
:room_history_int1 a owl:NamedIndividual , bacs:SpaceHistory ;
    osph:hasHistoryObservations :room_illum1 , :room_temp1 ;
    osph:hasIntervalStartTime "2017-03-09T08:00:00"^^xsd:dateTime .
:room_illum1 a owl:NamedIndividual , bacs:SpaceIlluminanceObs ;
    sosa:hasResult :illum1 ; sosa:madeBySensor :lightsensor ;
    sosa:observedProperty :room_illuminance_prop .
:illum1 a owl:NamedIndividual , ifcmr:IfcIlluminanceMeasure ;
    express:hasDouble "90.0"^^xsd:double .
:sb_history a owl:NamedIndividual , osph:ObjectHistory ;
    osph:hasStateFrequencies :statefreq1 ;
    osph:hasIntervalStartTime "2017-03-09T09:30:00"^^xsd:dateTime .
:statefreq1 a owl:NamedIndividual , osph:StateFrequency ;
    osph:hasRelatedState :noShade ; osph:hasStayRatio "1.0"^^xsd:double .
```

Fragment 3: Excerpt of room and sunblind history instantiation

The query in Fragment 4 gets the elements in the building and the associated state machine (if existing). The query in Fragment 5 explores the finite state machine of any element in a room and returns the state machine components (e.g. states, transitions, guards) and further details about transitions.

```
SELECT distinct ?building ?storey ?room ?elem ?statemach
WHERE {
  ?building rdf:type/rdfs:subClassOf* bot:Building.
  ?building bot:hasStorey ?storey. ?storey bot:hasSpace ?room .
  ?room bot:adjacentElement|bot:containsElement ?elem .
  OPTIONAL{ ?elem osph:hasStateMachine ?statemach .}
}
```

Fragment 4: SPARQL query to get the elements in the building

```
SELECT distinct ?statemach ?fsmelem ?class ?source ?target ?guard ?condition
WHERE {
  ?statemach rdf:type/rdfs:subClassOf* fsm:StateMachine .
  ?statemach fsm:contains/fsm:hasStateMachineElement* ?fsmelem .
  ?fsmelem rdf:type ?class . FILTER ( ?class != owl:NamedIndividual ) .
  OPTIONAL{
  ?class rdfs:subClassOf* fsm:Transition .
  ?fsmelem fsm:Source ?source . ?fsmelem fsm:Target ?target .
  OPTIONAL{
    ?fsmelem fsm:TransitionGuard ?guard . ?guard fsm:GuardCondition ?condition .}
}}
```

Fragment 5: SPARQL query to get the components of a state machine

5. Conclusions

This paper presented a modular ontology to model the domain of building control and automation, demonstrating its applicability in a test case. Reusing ontologies through integration and alignment is promising to tackle the building control and automation domain, but best practices and guidelines from ontology engineering will be further investigated. The proposed test case focuses on the building automation domain, but, as automation is relevant in many industries (process industry, manufacturing industry, etc.), the reuse of the ontology in other domains will be studied. Future works will address also:

- testing more complex cases requiring the interaction between smart objects;
- preparation of a library of general purpose SPARQL queries and update to support the use of the BACS ontology (e.g. extraction of object history, extraction of expressions, triggering and execution of a control action);
- integration with other ontologies specializing elements, sensors and actuators, and defining concepts related to geometry (e.g. placement, representation of objects).

Acknowledgments

This work has been partially funded by the Italian research project “Efficientamento dei processi di produzione e gestione integrata di utenze energivore con fonti rinnovabili e sistemi di accumulo mediante periferiche ICT in un contesto Smart District” within the “Piano Triennale 2015-17 della Ricerca di Sistema Elettrico Nazionale”.

References

[1] D. Bonino and F. Corno. DogOnt - Ontology Modeling for Intelligent Domotic Environments. *Lecture Notes in Computational Science*, 5318:790–803, 2008.

[2] O. Burke, A. Gonzalez-Beltran, and P. Rocca-Serra. STATistics Ontology, 2016. Available online: http://bioportal.bioontology.org/ontologies/STATO (Last accessed on 21 July 2017).

[3] V. Charpenay, S. Kabisch, D. Anicic, and H. Kosch. An ontology design pattern for iot device tagging systems. In *2015 5th International Conference on the Internet of Things (IOT)*, pages 138–145.

[4] M. Compton, P. Barnaghi, L. Bermudez, and et al. The SSN ontology of the W3C semantic sensor network incubator group. *Journal on Web Semantics*, 17:25 – 32, 2012.

[5] E. Curry, J. O'Donnell, E. Corry, S. Hasan, M. Keane, and S. O'Riain. Linking building data in the cloud: Integrating cross-domain building data using linked data. *Advanced Engineering Informatics*, 27(2):206–219, 2013.

[6] L. Daniele, F. den Hartog, and J. Roes. Created in Close Interaction with the Industry: The Smart Appliances REFerence (SAREF) Ontology. In R. Cuel and R. Young, editors, *Formal Ontologies Meet Industry*, volume 225, pages 100–112. Springer International Publishing, Cham, Switzerland, 2015.

[7] P. Dolog. Model-Driven Navigation Design for Semantic Web Applications with the UML-Guide. In *Proc. ICWE*, pages 75–86, 2004.

[8] 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, Hoboken NJ, USA, 2008.

[9] B. Farias Lóscio, C. Burle, and N. Calegari. Data on the Web Best Practices. https://www.w3.org/TR/dwbp/, 2017. Last accessed: 2017-07-14.

[10] L. C. Group. Building Data on the Web Working Group Charter. https://w3c-lbd-cg.github.io/lbd/charter/, 2017. Last accessed: 11 July 2017.

[11] A. Haller, K. Janowicz, S. Cox, D. L. Phuoc, K. Taylor, and M. Lefrançois. Semantic Sensor Network Ontology. https://www.w3.org/TR/2017/CR-vocab-ssn-20170711/, 2017. Last accessed: 2017-07-22.

[12] J. Kaiser and P. Stenzel. eeEmbedded D4.2: Energy System Information Model - ESIM. Technical report, *ee*Embedded Consortium, Brussels, Belgium, 2015.

[13] M. Lefrançois, J. Kalaoja, T. Ghariani, and A. Zimmermann. D2.2: The SEAS Knowledge Model. Technical report, ITEA2 12004 Smart Energy Aware Systems, Brussels, Belgium, 2017.

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

[15] P. Pauwels, S. Zhang, and Y.-C. Lee. Semantic web technologies in AEC industry: A literature overview. *Automation in Construction*, 73:145–165, 2017.

[16] J. Ploennigs, B. Hensel, H. Dibowski, and K. Kabitzsch. BASont - A modular, adaptive building automation system ontology. In *IECON 2012 - 38th Annual Conference of IEEE Industrial Electronics*, pages 4827–4833, Montreal, Canada.

[17] M. Poveda-Villalón and R. Garcia Castro. SAREF extension for building devices. https://w3id.org/def/saref4bldg#, 2017. Last accessed: 2017-07-11.

[18] C. Ptolemaeus, editor. *System Design, Modeling, and Simulation using Ptolemy II*. Ptolemy.org, 2014.

[19] qudt.org. QUDT. http://www.qudt.org/, 2017. Last accessed: 2017-07-22.

[20] M. H. Rasmussen, P. Pauwels, C. A. Hvid, and J. Karlshøj. Proposing a Central AEC Ontology that allows for Domain Specific Extensions. In *LC3 2017: Volume I Proceedings of the Joint Conference on Computing in Construction (JC3)*, pages 237–244, Heraklion, Greece, July 2017.

[21] C. Reinisch, M. J. Kofler, F. Iglesias, and W. Kastner. ThinkHome Energy Efficiency in Future Smart Homes. *EURASIP Journal on Embedded Systems*, (104617):1–18, 2011.

[22] H. Rijgersberg, M. van Assem, and J. Top. Ontology of units of measure and related concepts. *Semant. web*, 4(1):3–13, Jan. 2013.

[23] T. Sauter, S. Soucek, W. Kastner, and D. Dietrich. The Evolution of Factory and Building Automation. *IEEE Industrial Electronics Magazine*, 5(3):35–48, Sept. 2011.

[24] G. F. Schneider, P. Pauwels, and S. Steiger. Ontology-based modelling of control logic in building automation systems. *IEEE Transactions on Industrial Informatics*, (99):1–11, August 2017.

[25] SDWWG. Spatial Data on the Web Working Group Charter. https://www.w3.org/2015/spatial/charter, 2017. Last accessed: 11 July 2017.

[26] E. Simperl. Reusing ontologies on the semantic web: A feasibility study. *Data & Knowledge Engineering*, 68(10):905–925, 2009.

[27] P. Staroch. A weather ontology for predictive control in smart homes. Master's thesis, TU Vienna, Vienna, Austria, https://paul.staroch.name/thesis/thesis.pdf, 8 2013. Faculty for Computer Science.

[28] W. Terkaj and P. Pauwels. A method to generate a modular ifcOWL ontology. In *Proceedings of the 8th International Workshop on Formal Ontologies meet Industry*, 2017.

[29] W. Terkaj and M. Urgo. Ontology-based modeling of production systems for design and performance evaluation. In *2014 12th IEEE International Conference on Industrial Informatics (INDIN)*, pages 748–753, July 2014.

[30] N. M. Tomašević, M. Č. Batić, L. M. Blanes, M. M. Keane, and S. Vraneš. Ontology-based facility data model for energy management. *Advanced Engineering Informatics*, 29(4):971–984, 2015.