

PREPRINT: EEPsA as a core ontology for energy efficiency and thermal comfort in buildings

Iker Esnaola-Gonzalez ^{a,*}, Jesús Bermúdez ^b, Izaskun Fernandez ^a and Aitor Arnaiz ^a

^a *TEKNIKER, Basque Research and Technology Alliance (BRTA), Iñaki Goenaga 5, 20600 Eibar, Spain*
E-mails: iker.esnaola@tekniker.es, izaskun.fernandez@tekniker.es, aitor.arnaiz@tekniker.es

^b E-mail: jesus.bermudez@ehu.eus

Abstract.

THIS IS A PREPRINT VERSION OF THE ARTICLE AVAILABLE AT: <http://doi.org/10.3233/AO-210245>

Achieving a comfortable thermal situation within buildings with an efficient use of energy remains still an open challenge for most buildings. In this regard, IoT (Internet of Things) and KDD (Knowledge Discovery in Databases) processes may be combined to solve these problems, even though data analysts may feel overwhelmed by heterogeneity and volume of the data to be considered. Data analysts could benefit from an application assistant that supports them throughout the KDD process and aids them discovering which are the most relevant variables for the matter at hand, or informing about relationships among relevant data. In this article, the EEPsA (Energy Efficiency Prediction Semantic Assistant) ontology which supports such an assistant is presented. This ontology is developed on top of three ODPs (Ontology Design Patterns) which address weaknesses of existing proposals to represent features of interest and their respective qualities, as well as observations and actuations, the sensors and actuators that generate them, and the procedures used. The ontology is designed so that its customization to address similar problems in different types of buildings can be approached methodically.

Keywords: Ontology, Ontology Design Patterns, Buildings, Energy Efficiency, Thermal Comfort

1. Introduction

In the early 2000s, Klepeis et al. (2001) estimated that people used to spend around 90% of their time indoors, and this is a situation that may still hold nowadays. Thence, feeling comfortable while staying indoors is a must. User comfort can be influenced by different aspects such as visual, acoustic or thermal conditions. According to the ANSI/ASHRAE Standard 55-2017¹, thermal comfort is defined as follows: “that condition of mind that expresses satisfaction with the thermal environment and is assessed by subjective evaluation”. Being a subjective perception, under the same thermal conditions people can experience different levels of comfort.

Although many times being an overlooked factor, extensive research has been conducted proving the impact of thermal comfort on humans. Haynes (2008) and Hedge and Gaygen (2010) showed the relation between indoor environment conditions and working efficiency or productivity, which have a direct effect on company revenues. Mulville et al. (2016) demonstrated that indoor environment conditions

* Corresponding author. E-mail: iker.esnaola@tekniker.es.

¹<https://www.ashrae.org/technical-resources/bookstore/standard-55-thermal-environmental-conditions-for-human-occupancy>

can have a significant impact on occupants comfort, morale, health and wellbeing in commercial office buildings. It is also proved by Parsons (2014) that having an uncomfortable thermal situation involves many risks including clinical diseases, health impairments, and reduced human performance and work capacity. Therefore, all these evidences reinforce the need of ensuring comfortable thermal conditions in buildings.

However, occupants' thermal comfort is not the only concern related to buildings. According to Abergel et al. (2017), the building sector consumes more than 35% of global energy and it is responsible for nearly 40% of energy-related CO₂ emissions in the EU. This is why, with views to meet the energy sustainability and minimize the climate change, this sector is addressed by the set of binding legislations agreed by the European Commission in the EU 2020 climate and energy package². Therefore, the efficient management of building energy is becoming the trend for the future generation of buildings.

Fulfilling occupants' comfort whilst reducing energy consumption is still an unsolved problem in most buildings. Furthermore, it is important to note that certain type of buildings have specific features which may further hinder this problem. For example, tertiary buildings normally contain spaces with big dimensions which are prone to have bigger thermal inertia (Verbeke and Audenaert, 2018). This results in longer periods of time to heat up or cool down spaces and consequently, they cannot be effectively climatized with rather simple solutions like thermostat-based reactive systems. Instead, heating or cooling systems need to be activated in a specific mode in advance, in order to ensure having a comfortable thermal condition in a given time. The expansion of the Internet of Things (IoT) and Knowledge Discovery in Databases (KDD) (Fayyad et al., 1996) techniques may allow to improve matters in this regard. On the one hand, according to Gubbi et al. (2013), the IoT facilitates the monitoring of real-world qualities and events thanks to physical things equipped with electronic components and ubiquitous intelligence that allow them to connect, interact and exchange data. This led to the massive amount of data available nowadays, which has the potential to enable new discoveries and improve decision making processes. However, this data tends to be diverse and heterogeneous. Devices from different vendors may represent data in different formats, and even when a common format is used, the internal data model schema typically varies. Moreover, data may come from disparate external sources (often referred to as exogenous data), which further aggravates the data heterogeneity situation. Furthermore, the great variety of technical data hinders the human comprehension with regards to assessing which data is relevant for the matter at hand. These circumstances definitely pose a challenge for data analysts in charge of a KDD process which enables the extraction of useful knowledge from raw data.

In this context, data analysts have to deal with data related to the building where energy efficiency and user thermal comfort is aimed at. This data may describe building structural element properties including materials, heat transfer coefficients, and orientation of their boundaries (e.g. a room located in the second floor of a building which has a skylight with 2 m² of surface. A door with a U-factor of 2.61 that is opened by swinging to the left, and connects the hall with the southern outside part of the building). Data analysts also need to take into account information about sensors and actuators deployed in the building, their location, features and certainly their measurements and actuations (e.g. a temperature sensor located in the meeting room 042 that measured 22°C on 13th February 2020 at 09:40. A blind actuator that lowered blinds of window 21 on 21st June 2019 at 15:30). Likewise, data about weather conditions and weather forecasts for the building location are relevant (e.g. a forecast for Madrid made by the Spanish meteorology agency on 12th May 2020 at 13:00 forecasting a relative humidity of 62% on 10th May

²https://ec.europa.eu/clima/policies/strategies/2020_en

2020 at 15:00. A weather report that described cloudy skies during the morning of 7th November 2019 in Amsterdam). Furthermore, there is other information to consider such as the space occupancy, work schedule or human related organization (e.g. the 30th January 2020 is a reduced working hours day. The occupancy value of the meeting room 07 at 12:00 is of 6 people). Under such circumstances where a deep energy efficiency, thermal comfort and building domain knowledge is required to efficiently handle all this information, having insufficient domain expertise could make data analysts feel overwhelmed. Consequently, they typically resort to a trial and error approach searching for variables and tasks that could be confidently used to make accurate predictions. Moreover, due to the plethora of possible combination of algorithms in each KDD phase, according to Bernstein et al. (2005), even expert data analysts may turn to the trial and error approach. This is definitely an undesirable approach and it would be much more profitable to count with a KDD process assistant supported by technologies that enable the management of data semantics, data interrelationships, and knowledge representation. This means that it is necessary the representation of features of interest and their respective qualities, as well as observations and actuations, the sensors and actuators that generate them, and the procedures used. Furthermore, observations and actuations have to be described with respect to their values, in addition to their spatial and temporal context. In this paper, the development of a core ontology that supports such a KDD assistant is described.

The rest of this article is structured as follows. Section 2 reviews ontologies related to the domain of discourse. Section 3 presents the development of the proposed core ontology. Section 4 shows the customization of the ontology to apply it in a real-world use case. Finally the conclusions of this work are shown in Section 5.

2. Related Work

Linking or mapping raw data to existing ontologies or vocabularies, allows a better representation of the data, structuring it and setting formal types, relations, properties and restrictions that hold among them. In addition, as stated by Noy (2004), it allows representing data coming from multiple sources in a unified way, thereby supporting data integration. Another benefit of the semantic annotation lies in the additional background knowledge about a domain that can be added to the dataset. This leads to the enrichment of the dataset, as well as enabling the application of indexing techniques, which are based on resource URIs and ensure the retrieval and navigation through related resources (Andrews et al., 2012). Last but not least, after a semantic annotation process, data is more domain-oriented than the original source and allows more application-independent solutions. Consequently, there is no need for the user to be aware of raw data's underlying structure.

Due to the aforementioned benefits, Yoon et al. (1999), Kopanas et al. (2002) and Pinto and Santos (2009) state that annotating data semantically can contribute improving the KDD process. A comprehensive comparative review of ontologies involved in conceptualizations of observations and actuations with a special focus of the building domain is described in Esnaola-Gonzalez et al. (2020). Next, a brief overview of the most relevant ontologies is presented.

The initial Semantic Sensor Network (SSNO) ontology³ presented by Compton et al. (2012) was developed by the W3C Semantic Sensor Networks Incubator Group (SSN-XG) and it proposed a conceptual schema for describing sensors, accuracy and capabilities of such sensors, their observations and

³<http://www.w3.org/ns/ssn/>

methods used for sensing. Concepts for operating and survival ranges were also included, as well as sensors' performance within those ranges. Finally, a structure for field deployment was defined to describe deployment lifetime and sensing purposes. The initial SSN ontology was aligned with DOLCE ultra-lite (DUL) ontology and built on top of the Stimulus-Sensor-Observation (SSO) ODP presented by Janowicz and Compton (2010), which describes the relationships between sensors, stimulus, and observations. The SSNO ontology has been reused by other domain ontologies including the IoT-O ontology⁴ presented by Seydoux et al. (2016), the IoT-Lite ontology⁵ presented by Bermudez-Edo et al. (2017) and the FIESTA-IoT ontology⁶ presented by Agarwal et al. (2016) to name a few. There are ontologies like the Actuation-Actuator-Effect (AAE) ODP⁷, which adapts the SSNO ontology's SSO ODP for actuators, by modelling the relationship between an actuator and the effect it has on its environment through actuations.

The W3C Spatial Data on the Web Working Group (SDWWG⁸) proposed an update of the SSNO ontology called SOSA/SSN ontology⁹ that became a W3C recommendation and was presented by Haller et al. (2018). This new ontology follows a horizontal and vertical modularization architecture by including a lightweight but self-contained core ontology called SOSA¹⁰ (Sensor, Observation, Sample, and Actuator) for its elementary classes and properties. Furthermore, the SOSA/SSN ontology's scope is not limited to observations, but it is extended to cover actuations and samplings. In line with the changes implemented in the SOSA/SSN ontology, SOSA drops the direct DUL alignment although it can still be optionally achieved via the SSN-DUL alignment module¹¹. Moreover, similar to the original SSO pattern, SOSA acts as a central building block for the new SOSA/SSN ontology but puts more emphasis on its lightweight expressivity and the ability to be used standalone. There are ontologies like the Semantic Smart Sensor Network (S3N) ontology¹² presented by Sagar et al. (2018) which extend the SOSA/SSN ontology.

The SEAS Ontology¹³ presented by Lefrançois (2017) is an ontology designed as a set of simple core ODPs (Ontology Design Patterns) that can be instantiated for multiple engineering related verticals. The SEAS ontology modules are developed based on the following three core modules: the SEAS Feature of Interest ontology¹⁴ which defines features of interest (*seas:FeatureOfInterest*) and their qualities (*seas:Property*), the SEAS Evaluation ontology¹⁵ describing evaluation of these qualities, and the SEAS System ontology¹⁶ representing virtually isolated systems connected with other systems. On top of these core modules, several vertical SEAS ontology modules are defined, which are dependent of a specific domain. The Procedure Execution (PEP) ontology¹⁷ defines procedure executors that implement procedure

⁴<https://www.irit.fr/recherches/MELODI/ontologies/IoT-O>

⁵<http://www.w3.org/Submission/iot-lite/>

⁶<http://ontology.fiesta-iot.eu/ontologyDocs/fiesta-iot/doc>

⁷<http://ontologydesignpatterns.org/wiki/Submissions:Actuation-Actuator-Effect>

⁸<https://www.w3.org/2015/spatial>

⁹<http://www.w3.org/ns/sosa/>

¹⁰<http://www.w3.org/ns/sosa/>

¹¹<https://www.w3.org/ns/ssn/dul>

¹²<https://github.com/s3n-ontology/s3n/blob/master/s3n.ttl>

¹³<https://w3id.org/seas/>

¹⁴<https://w3id.org/seas/FeatureOfInterestOntology>

¹⁵<https://w3id.org/seas/EvaluationOntology>

¹⁶<https://w3id.org/seas/SystemOntology>

¹⁷<https://w3id.org/pep/>

methods, and generate procedure execution activities. Furthermore, PEP defines an ODP as a generalization of SOSA's sensor-procedure-observation and actuator-procedure-actuation models. Additionally, the SEAS ontology offers a set of alignments to related ontologies including the SOSA/SSN.

The Observation ODP¹⁸ aims at representing observations of things, under a set of parameters. This set of parameters may include the place where the observation was made, the time when it was made, and any other feature concerning the specific thing being observed.

The IoT Application Profile (IoT-AP) ontology¹⁹ presented by Gangemi et al. (2017), is an ontology for representing and modelling the knowledge within the domain of the IoT. The ontology is designed re-using ODPs such as the aforementioned Observation and the time indexed situation²⁰. It focuses on observations, but it also covers sensors that generate those observations, their values and observation collections.

3. The EEPsA Ontology

Towards the incorporation of the Semantic Technologies in the assistant that supports data analysts through the KDD process, it is of utmost importance to rely on proper ontologies and vocabularies that codify the required knowledge and enables a proper annotation of the data. As a matter of fact, such an ontology or network of ontologies may be the cornerstone of this assistant.

This section describes the EEPsA (Energy Efficiency Prediction Semantic Assistant) ontology²¹ which focuses on energy efficiency and thermal comfort in buildings but it is aimed at being reusable and easily customizable for other use cases in similar domains.

3.1. Ontology Development Methodology

Ontologies must be carefully designed and implemented, as these tasks have a direct impact on their final quality. Therefore, the use of well-founded ontology development methodologies (e.g. On-To-Knowledge presented by Sure et al. (2004) and DILIGENT presented by (Pinto et al., 2004)) is advised. For the development of the EEPsA ontology, the NeOn Methodology proposed by Suárez-Figueroa et al. (2012) was followed mainly because it does not prescribe a rigid workflow but instead it suggests a variety of paths. The NeOn Methodology comprises 9 scenarios supporting different aspects of the ontology development process, and for the EEPsA ontology the following scenarios were applied:

- Scenario 1: From specification to implementation. In this scenario, the OSRD (Ontology Requirement Specification Document (Suárez-Figueroa and Gómez-Pérez, 2012)) was created, which collected the ontology purpose, its intended users, and the set of ontology requirements in the form of Competency Questions (CQ). For building and maintaining the ontology, the Protégé²² (Musen, 2015) tool version 5.1.0 was used and for managing the different versions of the ontology, the Git version-control system.

¹⁸<http://ontologydesignpatterns.org/wiki/Submissions:Observation>

¹⁹<http://stlab.istc.cnr.it/IoT-AP/IoT-AP.rdf>, not available at the moment of writing this dissertation.

²⁰<http://ontologydesignpatterns.org/wiki/Submissions:TimeIndexedSituation>

²¹<https://w3id.org/eeepsa>

²²<https://protege.stanford.edu/>

- Scenario 7: Reusing ontology design patterns. In this scenario, existing ODPs were reviewed (e.g. from the ODP repository [OntologyDesignPatterns.org](http://ontologydesignpatterns.org)²³). Since existing ODPs could not satisfy the requirements captured in the OSRD, a set of basic ODPs were developed. These ODPs were used as building blocks on top of whom the rest of the EEPsA ontology was developed.
- Scenario 3: Reusing ontological resources and Scenario 4: Reusing and re-engineering ontological resources. According to Simperl (2009) the reuse of ontological resources built by others that have already reached some degree of consensus is a good practice in ontology development processes. In this scenario, a set of ontologies were reviewed, assessed and compared to decide whether they were suitable for reuse. In some cases, these ontologies were re-engineered prior to their reuse.

The following competency questions summarize basic requirements for assisting data analysts through a KDD process for solving energy efficiency and thermal comfort problems in buildings. Therefore, the development of a core ontology that satisfies them is a prime task.

- CQ01: What are the qualities that influence a feature of interest?
- CQ02: What are the qualities that affect a given quality of a feature of interest?
- CQ03: Which feature of interest does a given quality belong to?
- CQ04: What are the observations/actuators performed by a given procedure?
- CQ05: What are the observations/actuators performed by a given sensor/actuator?
- CQ06: What are the procedures implemented by a given sensor/actuator?
- CQ07: What are the features of interest on a given observation/actuation?
- CQ08: What are the qualities sensed/actuated by a given observations/actuators?
- CQ09: What are the features of interest of a given sensor/actuator?
- CQ10: What are the qualities sensed/actuated by a given sensor/actuator?
- CQ11: Which is the value of an observation?
- CQ12: When was an actuation generated?
- CQ13: For what time interval or instant is valid an observation?
- CQ14: For what spatial location is valid an observation?

For each competency question CQ_n, a twin competency question CQ_nⁱ can be considered, which consists in rephrasing the question in the opposite direction. For instance, CQ01ⁱ would be defined as “What is the feature of interest influenced by a given quality?”. In terms of a SPARQL query, it means that the query variable is moved from the subject position to the object position, or the other way round, in the triple pattern.

3.2. Developing the EEPsA ontology on top of ODPs

In ontology development processes, recurrent design problems may arise. Indeed, these problems may happen during the ontology conceptualization activity, the ontology formalization activity, or during the ontology implementation activity. Gangemi and Presutti (2009) defines an ODP as a modelling solution to solve this kind of problems. Furthermore, according to Hitzler et al. (2016), ODPs should ideally be extensible but self-contained, minimize ontological commitments to foster reuse, address one or more explicit requirements (such as use cases or competency questions), be associatable to an ontology unit test, be the representation of a core notion in a domain of expertise, be alignable to other patterns,

²³<http://ontologydesignpatterns.org>

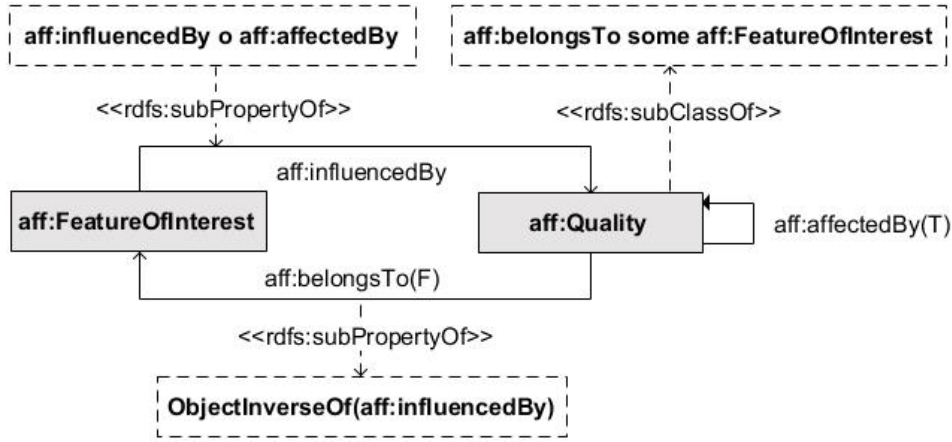


Fig. 1. The AffectedBy ODP. (F) represents functional and (T) transitive properties.

span more than one application area or domain, address a single invariant instead of targeting multiple reoccurring issues at the same time, follow established modelling best practices, and so forth.

Developing the EEPsA ontology on top of ODPs was found convenient due to the great flexibility provided by this modelling solution, which allows a proper segmentation of the intended conceptualization. In this case, the considered CQs were divided in three subsets: {CQ01, CQ02, CQ03}, {CQ04, CQ05, CQ06, CQ07, CQ08, CQ09, CQ10} and {CQ11, CQ12, CQ13, CQ14}. In order to solve each of those subsets, an ODP was defined. The proposed ODPs are inspired by existing ontologies and ODPs which address similar CQs but they do not fully satisfy those previously enumerated.

Even though these ODPs are motivated by energy efficiency and thermal comfort problems in buildings, they are designed to be applicable to similar problems in different use cases. Therefore, for each ODP a set of alignments or mappings are developed. These alignments target domain ontologies as well as upper-level ontologies, as setting mappings to a common upper ontology alleviates integration problems (Noy, 2004), helps to ensure clarity in modelling and avoids errors that have unintended reasoning implications (Cox, 2016). These alignments are kept in separate files and are available online in each ODP's documentation page. Furthermore, an instantiation example of each ODP is shown in Appendix A, along with SPARQL queries that solve some CQs.

Next, the three proposed ODPs are presented: the AffectedBy ODP²⁴, the EEP (Execution-Executor-Procedure) ODP²⁵ and the RC (Result-Context) ODP²⁶.

3.2.1. The AffectedBy ODP

Data analysts dealing with energy efficiency problems in buildings would benefit from a resource that supports the discovery of relevant variables that affect the environment of a given space or another feature of interest. Any of these variables will be represented as qualities of a feature of interest. Specifically, the competency questions CQ01, CQ02 and CQ03 must be considered. Therefore, the conceptualization must include classes representing features of interest (*aff:FeatureOfInterest*) and their qualities (*aff:Quality*).

²⁴<https://w3id.org/affectedBy>

²⁵<https://w3id.org/eep>

²⁶<https://w3id.org/rc>

The SOSA/SSN ontology contains a building block that may be useful for this matter. However, an inadequacy was spotted. The *ssn:Property* class is textually defined as “a quality of an entity. An aspect of an entity that is intrinsic to and cannot exist without the entity”. Furthermore, the *ssn:Property* class is linked to the *sosa:FeatureOfInterest* class with the *ssn:isPropertyOf* object property. Nevertheless, this object property is not functional, so the following triples can be found in a triple set annotated with SOSA/SSN terms:

```

:temperature rdf:type ssn:Property ;
               ssn:isPropertyOf :room03 .
:room03 rdf:type sosa:FeatureOfInterest .

:temperature ssn:isPropertyOf :room07 .
:room07 rdf:type sosa:FeatureOfInterest .

:room03 owl:differentFrom :room07 .

```

According to the aforementioned *ssn:Property*’s class textual definition, individual *:temperature* is intrinsic to and cannot exist without the existence of individual *:room03*. However, the triples shown contradict such definition because the individual *:temperature* is a quality of different entities (namely a quality of individual *:room03* and individual *:room07*).

A recent publication of Haller et al. (2018) about the SOSA/SSN ontology is aware of this possibility and explicitly expresses that “multiple observations across different features of interest or by different sensors or both can measure the same generic feature”. The publication also recognizes the choice to represent observable properties as inherent characteristics specific to a feature of interest. Therefore, the SOSA/SSN ontology allow different ways of modelling observable properties and it is expected that “communities and applications to develop their own approaches to building catalogues of observable properties and choosing appropriate levels of specificity”. However, the fact that different stakeholders adopt different modelling options may derive in interoperability problems.

This issue is tackled in the SEAS Feature of Interest ontology, where an ODP to describe features of interest and their qualities is defined. In this pattern, the *seas:isPropertyOf* object property links a *seas:Property* (which is equivalent to the class *ssn:Property*) to a *seas:FeatureOfInterest* (which is equivalent to the class *sosa:FeatureOfInterest*), and it is declared as subproperty of *ssn:isPropertyOf*. However, *seas:isPropertyOf* is functional. Therefore, it represents more faithfully the textual definition of *ssn:Property*.

The AffectedBy ODP defines the *aff:belongsTo* object property as functional to support the notion that a quality is intrinsic to the feature of interest to which it belongs. It is defined with *aff:Quality* as domain and *aff:FeatureOfInterest* as range, and it solves CQ03. Furthermore, the following axiom formalizes that every quality belongs to a feature of interest:

$$\text{aff:Quality} \sqsubseteq \exists \text{aff:belongsTo.aff:FeatureOfInterest} .$$

The SEAS Feature of Interest ontology also defines the *seas:derivesFrom* object property which links a *seas:Property* to another *seas:Property* it derives from. This object property is defined as a symmetric property. However, this constraint is unnecessary for the use case considered in this article and sometimes even inappropriate. For instance, the temperature of individual *:room03* may derive from the occupancy of the room, but the occupancy does not necessarily derive from the temperature of the room.

In order to tackle this specific issue and to solve CQ02, the *aff:affectedBy* object property is introduced. This property has class *aff:Quality* both as its domain and its range, and plays a slightly different role compared with *seas:derivesFrom*. In fact, *aff:affectedBy* is declared to be transitive.

In addition, the SEAS Feature of Interest ontology contains a textual comment that, although relevant, it is not materialized as an axiom. It is intended that:

$$seas:hasProperty \circ seas:derivesFrom \sqsubseteq seas:hasProperty .$$

The inconvenience of adding such a property chain axiom is that *seas:hasProperty* and its inverse become non-simple object properties and therefore they cannot be used in cardinality constraint expressions due to undecidability issues.

In order to solve CQ01, the object property *aff:influencedBy*²⁷ with *aff:FeatureOfInterest* as its domain and *aff:Quality* as its range is introduced, alongside with the next property chain axiom:

$$aff:influencedBy \circ aff:affectedBy \sqsubseteq aff:influencedBy .$$

In contrast to the aforementioned SEAS case, the selected set of axioms in the AffectedBy ODP do not cause any undecidability problem.

Finally, the property axiom representing that *aff:belongsTo* is subproperty of the inverse of *aff:influencedBy* is introduced in the AffectedBy ODP.

A diagram of the AffectedBy ODP is shown in Figure 1.

AffectedBy ODP Alignments. The AffectedBy ODP is aligned with the SOSA/SSN ontology and the SEAS Feature of Interest ontology. Furthermore, it is mapped with the upper-level DUL ontology. These alignments are kept in separate files and are available online in the AffectedBy ODP's documentation page <https://w3id.org/affectedBy>.

3.2.2. The EEP ODP

Another interesting information for data analysts working on energy efficiency and thermal comfort problems in buildings could be addressed by CQ04, CQ05, CQ06, CQ07, CQ08, CQ09 and CQ10. These CQs are the requirements considered for the EEP ODP.

It may be questionable why competency questions related to results of observations or actuations are disregarded in this ODP, specially because it is common to include this information as parameters of observations or actuations. However, there are some modelling alternatives such as the SEAS Evaluation ontology, where the qualification of the value of a *seas:Property* is preferred. Moreover, different conceptualizations of the result and their spatio-temporal context may be conceived depending on the application. This is the rationale behind designing a separate ODP (presented in the next subsection) to represent result-related matters. Such a design intends to improve the reusability of the proposal, allowing users to easily replace such ODP if they are not satisfied with its modelling decision.

The aforementioned CQs (CQ04 to CQ10) have been tackled by the SOSA/SSN ontology. However, a set of triples annotated with SOSA/SSN (for instance the set shown in Figure 2) cannot properly solve a question like CQ10ⁱ: which is the sensor that observes the temperature of :room07?

```
:sensor1 sosa:madeObservation :obs1;
```

²⁷In the previous version of the AffectedBy ODP published in Esnaola-Gonzalez et al. (2018b), this object property was named *aff:hasQuality*. However, it was renamed after *aff:influencedBy* to avoid misleading interpretations.

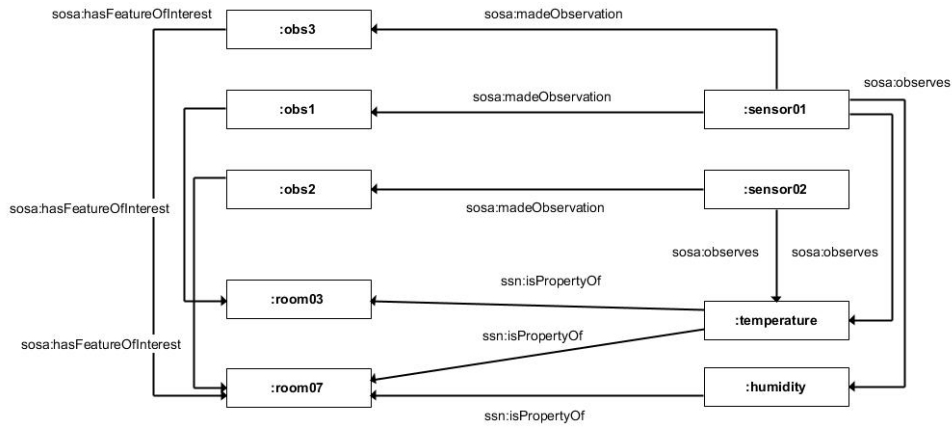


Fig. 2. A SOSA/SSN annotated set of triples.

```

sosa:observes :temperature .
:temperature ssn:isPropertyOf :room03 .
:obs1 sosa:hasFeatureOfInterest :room03 .

:sensor2 sosa:madeObservation :obs2 ;
sosa:observes :temperature .
:temperature ssn:isPropertyOf :room07 .
:obs2 sosa:hasFeatureOfInterest :room07 .

:sensor1 sosa:madeObservation :obs3 ;
sosa:observes :humidity .
:humidity ssn:isPropertyOf :room07 .
:obs3 sosa:hasFeatureOfInterest :room07 .

```

The rationale behind this issue is that there is no property directly linking sensors to features of interest, and moreover, composition of properties that link them through the *sosa:Observation* class are not sufficiently constrained.

The SEAS PEP ontology generalizes the core concepts of SOSA/SSN (i.e. Observation, Actuation, Sensor, Actuator, and Procedure). The proposed Execution-Executor-Procedure (EEP) ODP is an adaptation of the PEP ontology to fully satisfy the required competency questions, overcoming the indicated weaknesses about SOSA/SSN.

The EEP ODP imports the AffectedBy ODP alongside with its notion that a quality is intrinsic to the feature of interest it belongs to. Apart from the two classes imported from the AffectedBy ODP (i.e. *aff:FeatureOfInterest* and *aff:Quality*), the EEP ODP consists of three more classes: *eep:Execution*, *eep:Executor*, and *eep:Procedure* (see Figure 3). An individual of *eep:Execution* is an event upon a quality of a feature of interest, produced by an agent by performing a procedure. As for an individual of *eep:Executor*, it is an agent capable of performing tasks by following procedures. Lastly, an individual of *eep:Procedure* is a description of some actions to be executed by agents.

Note that individuals of class *eep:Execution* can be abstractly represented by a ternary relationship of its executor, the procedure used to produce the execution, and the quality of the feature of interest being considered. Accordingly, the class *eep:Execution* is the domain of the three functional object properties:

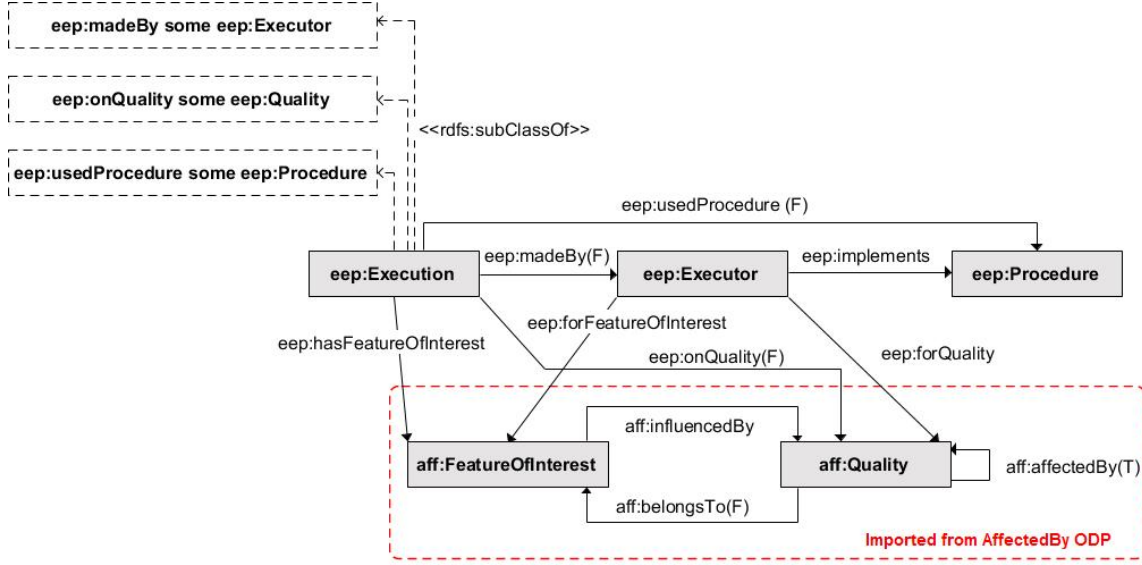


Fig. 3. The Execution-Executor-Procedure (EEP) ODP. (F) represents functional and (T) transitive properties.

eep:madeBy, *eep:usedProcedure*, and *eep:onQuality*. Moreover the following axioms are introduced:

$$\begin{aligned} eep:Execution &\sqsubseteq \exists eep:madeBy.eep:Executor, \\ eep:Execution &\sqsubseteq \exists eep:onQuality.eep:Quality, \text{ and} \\ eep:Execution &\sqsubseteq \exists eep:usedProcedure.eep:Procedure \end{aligned}$$

The object property *eep:madeBy* links an execution to the agent that performs the action; the object property *eep:usedProcedure* links an execution to the procedure that describes the task to be performed; and the object property *eep:onQuality* links an execution to the quality concerned by the execution. These three functional object properties jointly with the functional *aff:belongsTo* form the backbone of the EEP ODP.

The remaining object properties are: *eep:implements*, linking executors to procedures; *eep:hasFeatureOfInterest*, linking executions to features of interest; *eep:forQuality*, linking executors to qualities; and *eep:forFeatureOfInterest*, linking executors to features of interest. The values of all of them are inferred by the values of the four functional properties that form the backbone, due to the corresponding property chain axioms included in the EEP ODP:

$$\begin{aligned} eep:madeBy^{-1} \circ eep:usedProcedure &\sqsubseteq eep:implements, \\ eep:onQuality \circ eep:belongsTo &\sqsubseteq eep:hasFeatureOfInterest, \\ eep:madeBy^{-1} \circ eep:onQuality &\sqsubseteq eep:forQuality, \text{ and} \\ eep:forQuality \circ eep:belongsTo &\sqsubseteq eep:forFeatureOfInterest. \end{aligned}$$

EEP ODP Alignments. The EEP ODP is aligned with the SOSA/SSN ontology, the PEP ontology and PROV-O. Furthermore, it is mapped to the upper-level DUL ontology. These alignments are kept in separate files and are available online in the EEP ODP's documentation page <https://w3id.org/eep>.

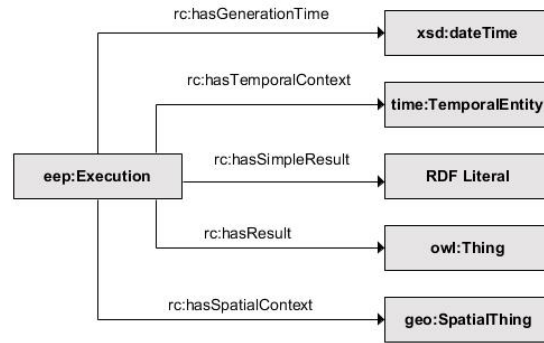


Fig. 4. The Result-Context (RC) ODP.

3.2.3. The RC ODP

Although the AffectedBy and EEP ODPs alleviate much of the data analysts' information needs, they may still require from data representing the results of the executions and their contexts. For example: which is the value of an observation? Or when was an actuation performed? This information may be collected answering the competency questions CQ11, CQ12, CQ13 and CQ14.

Every ontology or ontology network covering observations or actuations need to take into account the representation of these actions' results. For example, the SOSA/SSN ontology uses the *sosa:hasResult* object property, the IoT Application Profile (IoT-AP) ontology²⁸ published by Gangemi et al. (2017) uses *iotap:hasObservationValue* and om-lite uses *om-lite:result* object property. Values of these properties can be complex objects that usually include units of measurement, the measurement value, and some other optional parameters. However, sometimes a simple representation with a literal type value may suffice. In order to tackle these situations SOSA/SSN proposes the *sosa:hasSimpleResult* datatype property. Furthermore, properties representing results are typically associated to observations and actuations, even though there are alternative modelling options. For instance, in the SEAS ontology network, the SEAS Evaluation ontology associates *seas:value* and *seas:simpleValue* properties to the *seas:Property* class.

With respect to the proposed Result-Context (RC) ODP (shown in Figure 4), the representation of both complex and simple results is modelled with the object property *rc:hasResult* and the datatype property *rc:hasSimpleResult* respectively. This way, CQ11 is solved.

There are occasions in which parameters referring to temporal and spatial aspects may be necessary to qualify a result. Regarding the representation of temporal aspects, the SOSA/SSN ontology distinguishes between the time when the result of an observation, actuation, or sampling applies to the feature of interest (with the object property *sosa:phenomenonTime*) and the instant of time when such an observation, actuation or sampling was completed (with the datatype property *sosa:resultTime*). The phenomenon time is specified with an individual of OWL-Time ontology's *time:TemporalEntity* class as it may be either an instant, an interval of time, or even a temporal complex. Meanwhile, the result time describes an instant represented with *xsd:dateTime*. As for the SEAS Evaluation ontology, the temporal context is modelled with the *seas:hasTemporalContext* object property that links an evaluation with its temporal entity modelled as an individual of *time:TemporalEntity*. Furthermore, PROV-O also enables the representation of temporal context. Specifically, the *prov:generatedAtTime* datatype property allows representing the completion of production of a new entity, which would be similar to the *sosa:resultTime* datatype property.

²⁸<http://stlab.istc.cnr.it/IoT-AP/IoT-AP.rdf>, not available at the moment of writing this article.

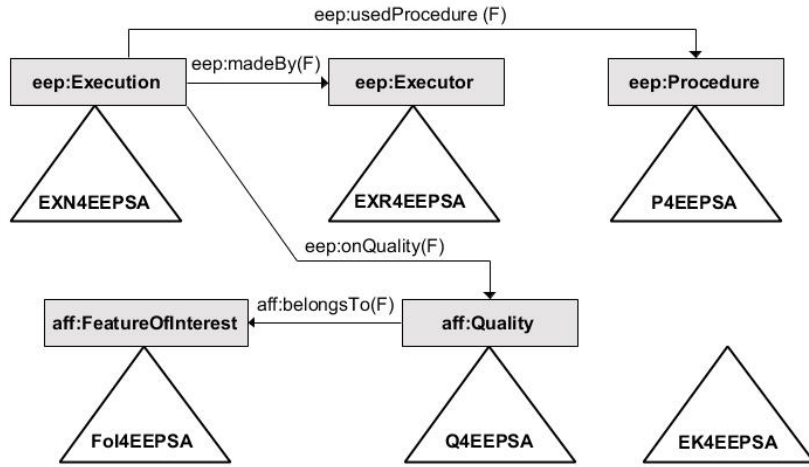


Fig. 5. Overview of the EEPsA ontology.

With respect to the RC ODP, it defines two properties: *rc:hasGenerationTime* which is equivalent to *sosa:resultTime*, and *rc:hasTemporalContext* which is equivalent to *sosa:PhenomenonTime*. These definitions solve CQ12 and CQ13 respectively.

When using the SOSA/SSN ontology, spatial aspects of an observation/actuation/sampling are expected to be associated with the feature of interest, the sensor/actuator/sampler or the platform on which they are mounted. However, the representation of this association is not covered by the ontology itself, and has to be made by deferring to external ontologies. By contrast, the SEAS Evaluation ontology leans towards a modelling option which is similar to the temporal aspect. Namely, it defines the *seas:hasSpatialContext* that links an evaluation to its spatial validity context represented as an individual of *geo:SpatialThing* class.

In the RC ODP, the *rc:hasSpatialContext* object property has been defined. It plays *seas:hasSpatialContext* property's same role, but it has *eep:Execution* class as domain, and *geo:SpatialThing* as range. This object property solves CQ14.

RC ODP Alignments. The RC ODP is aligned with the SOSA/SSN and PROV-O ontologies. These alignments are kept in separate files and are available online in the RC ODP's documentation page <https://w3id.org/rc>.

The RC ODP is designed as an horizontal extension of the EEP ODP. But, there are cases where data analysts may require from both ODPs so they need to be used jointly. For example:

- CQ15: Which is the temperature value of room 03 on 2018-11-20 at 16:00?

These three ODPs are the cornerstone of the EEPsA ontology. As a matter of fact, the classes defined by the AffectedBy and EEP ODPs act as stub classes, and for each of them an ontology module is developed. The EEPsA ontology is the addition of the following ontological resources: the three ODPs presented (AffectedBy, EEP and RC), five ontology modules specializing the stub classes defined by these ODPs (FoI4EEPSA, Q4EEPSA, P4EEPSA, EXR4EEPSA and EXN4EEPSA), and an ontology module containing expert knowledge (EK4EEPSA). Figure 5 shows an overview of the EEPsA ontology.

3.3. Ontology Modules

The modularization of ontologies consists in partitioning them into independent self-contained knowledge components. A modular approach brings benefits such as the flexibility for component reuse (Grau et al., 2008), the support for more efficient query answering (Stuckenschmidt and Klein, 2007), and the enhancement of components change and evolution (Ensan and Du, 2013).

When an already existing ontology is large and monolithic, it needs to be splitted up in order to benefit from the mentioned advantages. There are different techniques that perform ontology partitioning by dividing an ontology into a set of significant modules that together form the original ontology. However, according to d'Aquin et al. (2009) there is no universal way to modularize an ontology and the choice of a particular technique or approach should be guided by the requirements of the application or use case.

In order to avoid performing ontology modularization techniques in the future, modularization is advised to be implemented from an early ontology development stage. The EEPsa ontology is modularized by design. Next, the EEPsa ontology modules are presented.

3.3.1. FoI4EEPSA (Feature of Interest for EEPsa ontology module)

This ontology module covers the knowledge specializing the *aff:FeatureOfInterest* class for the EEPsa Ontology. In the context of this article, a feature of interest is understood as an abstraction of a real world phenomena (object, event, etc). A feature of interest is then described in terms of its qualities, which are qualifiable, quantifiable, observable or operable.

In particular, the FoI4EEPSA ontology module²⁹ tries to tackle CQs such as the following:

- CQ16: Which building does a given space belong to?
- CQ17: How many spaces does a building have?
- CQ18: In which storey is a given space located?

Different ontologies that cover the representation of the building domain were reviewed (a survey can be found in Esnaola-Gonzalez et al. (2020)), and finally BOT³⁰ was considered to be reused for basic building topology descriptions due to its conciseness, nice documentation, careful metadata with explanatory descriptions of the intended meanings of their terms and alignments to other domain ontologies. The Building Topology Ontology (BOT) presented by Rasmussen et al. (2017) is a minimal OWL DL ontology for covering core concepts of a building and for defining relationships between their subcomponents.

As for representing building elements, which are also an important part of the domain at hand, the FoI4EEPSA ontology module needs to solve the following CQs:

- CQ19: Which space does a given door belong to?
- CQ20: How many windows does a given space have?
- CQ21: Is a window adjacent to outdoors?

To this end, the Building Productt Ontology (PRODUCT³¹) proposed by the W3C LBD community group was considered. PRODUCT (which at the moment of writing this article is still under development) has a much wider coverage scope than the needed, so its importation would result in increasing EEPsa ontology's size with unnecessary concepts. The Building Element Ontology³², which

²⁹<https://w3id.org/eeepsa/foi4eeepsa>

³⁰<https://w3id.org/bot>

³¹<https://github.com/w3c-lbd-cg/product>

³²<https://pi.pauwel.be/voc/buildingelement>

has been proved to be valid semantically tagging or classifying *bot:Element* instances in *?*, was also considered. However, it has several concepts that may result too technical for the end users of the EEPsa ontology. Therefore, following the simplicity goal of the EEPsa ontology, importing PRODUCT of the Building Element Ontology was discarded. Instead, a set of building elements identified in the EEPsa ontology requirements were defined, such as doors (*foi4eepsa:Door*) and windows (*foi4eepsa:Window*). Furthermore, a class *foi4eepsa:ExternalBuildingElement* was defined to represent building elements that face outdoors. This representation mimics the approach followed by EEOnt, and allows the representation of doors and windows that open to the outdoor (via *foi4eepsa:ExternalDoor* and *foi4eepsa:ExternalWindow* classes), as well as external walls (*foi4eepsa:ExternalWall*). These new terms defined in FoI4EEPSA are mapped to the related PRODUCT ontology terms. PRODUCT is in turn aligned with the IFC4 Addendum 2 standard, making the FoI4EEPSA ontology module interoperable. Likewise, FoI4EEPSA terms are mapped to the Building Element Ontology.

Last but not least, information related to the building context is also an important aspect. Namely, FoI4EEPSA has to solve the following CQs:

- CQ22: Which is the intended use of the building?
- CQ23: When was the building built?
- CQ24: Which is the gross floor area of the building?

IFC presents a comprehensive collection of property sets (known as PSETs) for describing different aspects of building and building-related context. However, the conceptualization of these properties in ifcOWL as instances of classes (e.g. *ifc:IfcIdentifier* or *ifc:IfcLabel*) is counterintuitive to semantic web principles that would expect OWL properties to represent them. Therefore, inspired by the semantic transformations proposed by de Farias et al. (2015), FoI4EEPSA defines a re-engineering of the relevant properties contained in IFC PSET Building Common and IFC PSET Building collections. Namely, concepts such as *foi4eepsa:hasYearOfConstruction* are used to represent the construction year of a building, and *foi4eepsa:hasMarketCategory* to define the type of use of the building (e.g. residential or commercial).

3.3.2. Q4EEPSA (Quality for EEPsa ontology module)

This ontology module covers the knowledge specializing the *aff:Quality* class, which refers to qualities or aspects of a feature of interest that are intrinsic to and cannot exist without the feature of interest.

In particular, the Q4EEPSA ontology module³³ tries to tackle CQs such as the following:

- CQ25: Which are the actuatable qualities?
- CQ26: Which are the predictable qualities?
- CQ27: Which are the thermal comfort qualities?

In Q4EEPSA two categories of qualities are differentiated. On the one hand, observable qualities of a feature of interest defined by the class *q4eepsa:ObservableQuality*. Bearing in mind the conceptualization of observation proposed by the O&M model followed by the EEPsa ontology, this class comprises qualities that can be observed, estimated and even forecasted. On the other hand, qualities of a feature of interest that can be acted on, defined by the class *q4eepsa:ActuatableQuality*. Qualities that are relevant for the EEPsa's domain of discourse are classified at least in one of the aforementioned classes. Likewise, qualities that belong to these categories are also classified into orthogonal groups according to dimensions like their area of interest.

³³<https://w3id.org/eeepsa/q4eepsa>

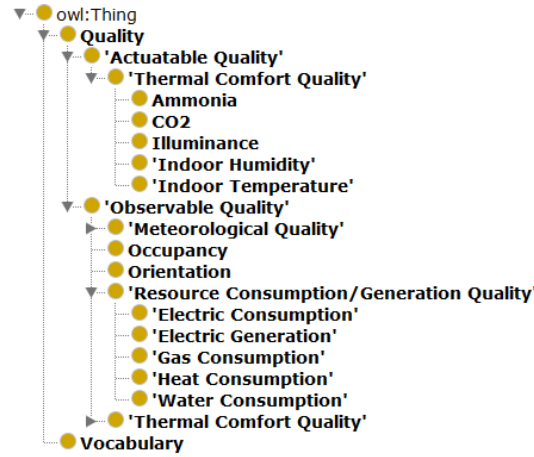


Fig. 6. Overview of the classes defined in Q4EEPSA (visualized in Protégé).

Meteorological qualities such as the solar radiation (*q4eeppsa:SolarRadiation*) or the cloud coverage (*q4eeppsa:CloudCover*), are defined as subclasses of *q4eeppsa:MeteorologicalQuality*, which are observable but not actuatable as defined with the following axiom:

$$q4eeppsa:MeteorologicalQuality \sqsubseteq q4eeppsa:ObservableQuality \sqcap \neg q4eeppsa:ActuableQuality .$$

Qualities related with the thermal comfort within a space such as indoor temperature (*q4eeppsa:IndoorTemperature*) and indoor humidity (*q4eeppsa:IndoorHumidity*) are represented as subclasses of *q4eeppsa:ThermalComfortQuality*. These qualities can be observed and acted on. Furthermore, qualities related to the resource consumption such as water consumption (*q4eeppsa:WaterConsumption*) or electric generation (*q4eeppsa:ElectricGeneration*), are also defined. These concepts are described as subclasses of *q4eeppsa:ResourceConsumptionQuality*, which is observable. However, even though it can be indirectly actuated on (for example with consumption restriction strategies), a consumption is not directly actuatable, so that it is not categorised as a subclass of *q4eeppsa:ActuableQuality*. Some of the mentioned classes are reengineered and reused from the M3-lite taxonomy because it contains a great set of well-organized quality classes.

The Q4EEPSA ontology module is aligned with related ontologies such as SAREF and the SEAS Generic Property ontology³⁴.

Figure 6 shows an overview of the main Q4EEPSA classes.

3.3.3. P4EEPSA (Procedure for EEPsa ontology module)

This ontology module covers the knowledge specializing the *eep:Procedure* class, which represents workflows, protocols, plans, algorithms, or computational methods specifying how to produce an event.

In particular, the P4EEPSA ontology module³⁵ tries to tackle CQs such as the following:

- CQ28: What are the actuating procedures?

³⁴<https://w3id.org/seas/GenericPropertyOntology>

³⁵<https://w3id.org/eeppsa/p4eeppsa>

- CQ29: What are the predictive procedures?
- CQ30: What are the imputation procedures?

P4EEPSA represents four types of procedures: actuating procedures (*p4eeppsa:ActuatingProecedure*) specifying how to act on an event; sensing procedures (*p4eeppsa:SensingProcedure*) specifying how to sense an event; imputation procedures (*p4eeppsa:ImputationProcedure*) specifying how to impute an event; and predictive procedures (*p4eeppsa:PredictiveProcedure*) specifying how to predict an event. Such a classification of procedures is a requirement of the data analyst assistant.

3.3.4. EXR4EEPSA (Executor for EEPsa ontology module)

This ontology module covers the knowledge specializing the *eeppsa:Executor* class, which represents agents that produce an event by implementing a procedure.

The EXR4EEPSA ontology module ³⁶ tries to tackle CQs such as the following:

- CQ31: Which type of sensor is a given sensor?
- CQ32: Is a given executor a window actuator?
- CQ33: Is a given executor a predictive model?

EXR4EEPSA concepts are categorised in four different classes: sensors, actuators, predictive models and imputation methods. *exr4eeppsa:Sensor* represents agents that implement a procedure to sense a change in a real world's quality. Following SOSA/SSN's conceptualization, a sensor is not necessarily a physical device, and it can also be virtual, or even a human being. Sensors are classified in two main classes: meters and environment sensors. On the one hand, the class *exr4eeppsa:UtilityMeter* defines a set of meters observing the water, heat, gas or electricity consumption, as well as meters for observing the energy generated (e.g. from photovoltaic panels). On the other hand, sensors observing environment conditions include anemometers (*exr4eeppsa:Anemometer*) for sensing wind speed and humidity sensors (*exr4eeppsa:HumiditySensor*). Furthermore, these environment sensors include the *exr4eeppsa:AirQualitySensor* subclass comprising agents sensing air pollution and gases in the surrounding area (e.g. *exr4eeppsa:CO2Sensor*).

exr4eeppsa:Actuator represents agents that implement a procedure to act on a real world quality. This concept is more general than *seas:Actuator*, *iot-lite:ActuatingDevice* or *sosa:Actuator* since, similarly to sensors, the agent does not necessarily need to be a device or a physical element. It can be for example a software that switches on or off a light bulb. This class includes a set of common actuators for an energy efficiency problem in tertiary buildings, such as door actuators (*exr4eeppsa:DoorActuator*) and window actuators (*exr4eeppsa:WindowActuator*).

EXR4EEPSA is not aimed at making an exhaustive representation of different types of sensors and actuators. Instead, it focuses on describing sensors and actuators that are recurrent to energy efficiency and thermal comfort problems in buildings. Furthermore, two additional high-level class of executors are defined in EXR4EEPSA. The first one is *exr:PredictiveModel*, representing agents that implement a predictive modelling procedure to forecast unknown or future outcomes. The second one, the class *exr:ImputationMethod*, describes agents that implement a procedure to compute an estimation of missing values.

Some of these classes are inspired by the M3-lite taxonomy. However, they are not reused because they do not represent the same sensors/actuators (e.g. M3-lite represents only physical sensors, while in the

³⁶<https://w3id.org/eeppsa/exr4eeppsa>

context of EXR4EEPSA sensors are not necessarily physical objects). Some other classes are reengineered and reused from the SEAS Smart Meter ontology³⁷. Furthermore, the EXR4EEPSA ontology module is aligned with these two related domain ontologies.

3.3.5. EXN4EEPSA (Execution for EEPSA ontology module)

This ontology module covers the knowledge specializing the *eep:Execution* class. This class represents events or actions made by an agent executing a task implemented by a procedure with respect to a quality of a feature of interest.

In particular, the EXN4EEPSA ontology module³⁸ tries to tackle CQs such as the following:

- CQ34: Is a given execution an actuation?
- CQ35: Is a given execution an observation?
- CQ36: What are the imputed observations?

To that end, this ontology module defines three main concepts: an observation (*exn4eepsa:Observation*), which is an execution made by an executor to estimate or calculate a quality of a feature of interest; an actuation (*exn4eepsa:Actuation*) which is an execution made by an executor to act upon a quality of a feature of interest; and a missing value (*exn4eepsa:MissingValue*), which happens when executions are empty or null in attributes where a value should have been recorded. Likewise, an observation can be predicted or forecasted (*exn4eepsa:Forecast*), obtained after using an imputation method (*exn4eepsa:-Imputation*), or it can even be an outlier (*exn4eepsa:Outlier*) when it does not conform to the expected behaviour. Furthermore, EXN4EEPSA also defines the class *exn4eepsa:CollectionOfExecutions*. This class represents a set of executions, such as a sequence of missing values, or the collection of observations forecasted by a predictive model. Furthermore, object properties *exn4eepsa:hasMember* and its inverse *exn4eepsa:isMemberOf* are defined to associate individuals of class *eep:Execution* that belong to a collection of executions, and viceversa.

Such a detailed hierarchy of concepts is motivated by the relevance these concepts may have in data analysis problems. Furthermore, the EXN4EEPSA ontology module is aligned with a set of domain ontologies such as the SOSA/SSN ontology, the SEAS Device ontology, SAREF and om-lite ontology. It is important to note that other ontologies such as SmartEnv and S3N can be indirectly aligned with EXN4EEPSA since they are based on the SOSA/SSN ontology.

3.3.6. EK4EEPSA (Expert Knowledge for EEPSA ontology module)

This ontology module covers the necessary expert knowledge to provide inferencing capabilities that can be exploited by the data analyst assistant. This module is defined under the supervision of experts in the domain at hand in order to capture task-based knowledge.

In particular, EK4EEPSA³⁹ tries to tackle CQs such as the following:

- CQ37: What is a naturally enlightened space?
- CQ38: Which types of spaces are in a building?
- CQ39: Which are the qualities affecting a bad insulated space's temperature?

On the one hand, EK4EEPSA defines a classification of types of spaces in buildings. These space definitions are based on their structural features, such as spaces in contact with outdoor environment (*ek4eepsa:AdjacentToOutdoorSpace*) or spaces located below the ground floor (*ek4eepsa:BelowGroundLevelSpace*).

³⁷<https://w3id.org/seas/SmartMeterOntology>

³⁸<https://w3id.org/eepsa/exn4eepsa>

³⁹<https://w3id.org/eepsa/ek4eepsa>

However, other space definitions such as the proposed by the HBC (Human Comfort in Building) ontology⁴⁰ published by Qiua et al. (2018) may be incorporated, where spaces are mainly characterized by the equipment contained or not within themselves (e.g. *hbc:SpaceWithHeater* or *hbc:SpaceWithoutHeater*). Note that in the scenario tackled in this article, it may be convenient to make heavy usage of axioms expressing sufficient conditions to infer the recognition of individuals in appropriate classes. That is, it may be suitable to use equivalent class axioms with appropriate right hand class expressions, rather than being dependent on explicit assertions only. For instance, the *ek4eepsa:AdjacentToOutdoorSpace* is defined as follows:

$$\begin{aligned} ek4eepsa:AdjacentToOutdoorSpace &\equiv \\ bot:Space &\sqcap \\ \exists bot:hasElement.foi4eepsa:ExternalBuildingElement \end{aligned}$$

On the other hand, for each space type, qualities that affect their indoor temperature are captured. Such a modelling relies on qualities represented in Q4EEPSA and the axioms defined in the AffectedBy ODP. It is worth noting that this is the only EEPsA ontology module that has dependencies with other EEPsA ontology modules. However, the data analyst assistant has a requisite that needs for the ability to ask for interrelationships of entities coming from any other modules. For instance, the temperature of an adjacent to outdoor space may be affected by qualities such as the indoor humidity, and the occupancy of the room, as represented in the following axioms:

$$\begin{aligned} ek4eepsa:AdjacentToOutdoorSpaceIndoorTemperature &\sqsubseteq \\ \exists aff:affectedBy.q4eepsa:IndoorHumidity &\sqcap \exists aff:affectedBy.q4eepsa:Occupancy \\ \sqcap \exists aff:affectedBy.q4eepsa:SolarRadiation &\sqcap \exists aff:affectedBy.q4eepsa:WindSpeed . \end{aligned}$$

This knowledge modelling can be exploited by application programs and to support data analysts in a proper manner. After knowing which is the type of space at hand, data analysts get to know which are the qualities that are relevant to solve the energy efficiency or thermal comfort problem.

At the moment of writing this article, the EK4EEPSA ontology module solves the presented CQs. However, being an ontology module containing expert knowledge, it is extendible as more requisites are demanded.

3.4. Documentation

According to Peroni et al. (2013), a good ontology documentation increases its understandability and potential usability, both by experts in semantics and by people who are not necessarily experts. The documentation of the EEPsA ontology and its ontology modules is generated with WIDOCO (a Wizard for DOCUMENTing Ontologies) developed by Garijo (2017) which creates a set of linked enriched HTML pages. These HTML pages are extended with hand-made sections such as the alignments to other ontologies or with ontology usage examples.

W3C's Data on the Web Best Practices (Calegari et al., 2017) states that providing metadata is a fundamental requirement that helps human users and computer applications to understand the data as well as other important aspects that describes a dataset. All the ontological resources presented in this

⁴⁰<https://w3id.org/ibp/hbc>

article were annotated following guidelines described in Garijo and Poveda-Villalón (2017) as it was considered the most complete guideline among the ones reviewed. As a matter of fact, for each EEPsa ontology module or ODP, both the ontology itself and the classes and properties are annotated with all the recommended terms and some optional terms.

Next, the canonical URIs for the different ontology modules documentation are shown. All URIs are provided by the <https://w3id.org> re-direction service.

- EEPsa ontology: <https://w3id.org/eepsa>
- AffectedBy ODP: <https://w3id.org/affectedBy>
- EEP ODP: <https://w3id.org/eeep>
- RC ODP: <https://w3id.org/rc>
- FoI4EEPSa ontology module: <https://w3id.org/eepsa/foi4eeepsa>
- Q4EEPSa ontology module: <https://w3id.org/eepsa/q4eeepsa>
- P4EEPSa ontology module: <https://w3id.org/eepsa/p4eeepsa>
- EXR4EEPSa ontology module: <https://w3id.org/eepsa/exr4eeepsa>
- EXN4EEPSa ontology module: <https://w3id.org/eepsa/exn4eeepsa>
- EK4EEPSa ontology module: <https://w3id.org/eepsa/ek4eeepsa>

With regards to the ODPs, they are also available in the ODP repository⁴¹, which collects and makes ODPs available on the web, allowing users to download, propose, and discuss them.

3.5. Evaluation

There are many evaluation metrics for assessing ontologies in existing literature such as Brank et al. (2005) and Obrst et al. (2007). Most of them focus on structural notions without taking into account the semantics, leading to incomparable measurement results (Vrandečić and Sure, 2007). And even though these are valid metrics, they may not be enough to determine the quality of an ontology. In order to avoid biased evaluations, next, the EEPsa ontology and the modules that comprise it are assessed from three perspectives: design correctness, structural metrics, and modularity quality.

3.5.1. Design Correctness Metrics

The design correctness is evaluated using OOPS! (Ontology Pitfall Scanner) developed in Poveda-Villalón et al. (2014), which detects some of the most common pitfalls appearing within ontology developments. OOPS! is available online⁴² and evaluates an ontology against a catalogue of 41 potential pitfalls classified into three levels according to their severity: minor, important and critical. This tool was used during the ontology modules development phase, contributing to an early detection of pitfalls, and complementing the manual review of the ontology's correctness. Table 1 summarizes the number of pitfalls detected in the EEPsa ontology and its components.

Overall, most ontology modules share the same minor pitfalls "P04: Creating unconnected elements" and "P08: Missing annotations". These pitfalls appear mainly in the stub classes that ontology modules extend (e.g. class *aff:FeatureOfInterest* for the case of the FoI4EEPSa ontology module) as well for the *voaf:Vocabulary* class used to describe the ontology itself. These concepts are adequately annotated and connected in their source ontology module so annotating them again would derive in having duplicated metadata when all ontology modules are imported by the EEPsa ontology. Therefore, these pitfalls are ignored.

⁴¹<http://ontologydesignpatterns.org>

⁴²<http://oops.linkeddata.es/>

Table 1
Summary of ontology design correctness evaluation by OOPS!

Ontology	Minor	Important	Critical
EEP	13	2	0
AffectedBy	4	1	0
RC	8	3	0
FoI4EEPSA	7	1	0
Q4EEPSA	4	1	0
P4EEPSA	4	1	0
EXR4EEPSA	4	1	0
EXN4EEPSA	5	1	0
EK4EEPSA	5	1	0

Table 2
Summary of ontology structural metrics by Protégé's Ontology Metrics tab (OP = Object Properties, DP = Datatype Properties, * = Imported axioms are not considered)

Ontology	Axioms	Class	OP	DP	Annotation	DL Expressivity
EEP(*)	80	6	8	0	40	ALERIF
AffectedBy	62	3	3	0	31	ALERIF+
RC	40	4	3	2	20	AL(D)
FoI4EEPSA(*)	128	17	0	5	64	AL(D)
Q4EEPSA	197	30	0	0	124	AL
P4EEPSA	40	6	0	0	16	AL
EXR4EEPSA	207	33	0	0	127	AL
EXN4EEPSA	72	9	2	0	36	ALI
EK4EEPSA	81	25	4	0	32	ALC

Regarding the important pitfalls, the "P10: Missing disjointness" is repeated in all the ontology modules and ODPs. This pitfall arises when an ontology lacks from disjointness axioms between classes or between properties that should be defined as disjoint. However, in the EEPsA ontology modules case, those suggested disjointness axioms are an inconvenient conceptualization constraint, so it was decided not to add those constraints.

3.5.2. Structural Metrics

Structural metrics by themselves may not be enough to assess the quality of an ontology or an ontology module, but they may still be relevant to describe an ontology. Protégé has an Ontology Metrics tab⁴³ that displays entity and axiom counts for the axioms in the active ontology. Table 2 summarizes the structural metrics for the different EEPsA ontology modules, ODPs and the EEPsA ontology itself.

Results show that ODPs are richer from a DL expressivity point of view. They define more constraints, while the rest of the ontology modules are more light weighted. As for the size, most EEPsA ontology modules are rather small (less than 17 classes). The only exception are the Q4EEPSA, EXR4EEPSA and EK4EEPSA ontology modules, which represent over 25 classes. The first two are in charge of representing qualities, sensors and actuators that are typical in problems addressed in the article, so it is understandable to contain a bigger number of classes. The latter, in turn, actually defines only 8 new

⁴³<http://protegeproject.github.io/protege/views/ontology-metrics>

classes. The rest of the classes are defined in other modules but are necessary to describe the expert knowledge contained in the module.

3.5.3. Modularity Quality

The EEPISA ontology's module quality is also assessed based on the guidelines proposed by Khan and Keet (2016). This work creates a comprehensive list of module evaluation metrics as well as a definition of 14 types of ontology modules. For each type of ontology module, it is described which metrics need to be measured and the expected values for a high quality ontology module. In the case of the EEPISA ontology, modules of type T1 (ODP modules: AffectedBy, EEP and RC) and T2 (Subject domain modules: FoI4EEPSA, Q4EEPSA, P4EEPSA, EXR4EEPSA, EXN4EEPSA and EK4EEPSA) are identified. The evaluation is performed with TOMM⁴⁴ (Tool for Ontology Module Metrics) and results are available online⁴⁵.

Regarding the ODPs, the guidelines suggest that a good quality module should have a small size compared to the original ontology size (i.e. relative size), a small cohesion (i.e. the extent to which entities in a module are related to each other), and be complete. The proposed three EEPISA ODPs satisfy the small relative size and cohesion requirements. However, EEP and RC are not logically complete, as they do not describe terms defined in other ontologies (e.g. *aff:affectedBy* object property in EEP and *eep:Execution* in RC) to avoid duplicated metadata in the final EEPISA ontology.

With regards to the rest of the ontology modules, which can be classified as of type "T2-subject domain modules", they are required to fulfil these criteria to be considered good quality modules: small cohesion, large encapsulation (i.e. "swappability" or ease to exchange a module for another without side effects), small coupling (i.e. the degree of interdependence of a module) and small redundancy (i.e. the duplication of axioms within a set of ontology modules). All the EEPISA ontology modules satisfy these criteria.

3.6. Ontology Customization by Module Replacement

Although the EEPISA ontology is aimed at supporting data analysts in energy efficiency and thermal comfort problems in buildings, it is designed to enable its customization to support data analysts in similar problems in different types of buildings. Being modularized by design, the EEPISA ontology is expected to be easily modified. Furthermore, as it has been demonstrated that the EEPISA ontology modules are loosely coupled and have a few dependencies between them, this ontology customization can be methodically approached.

The customization of the EEPISA ontology is recommended to be performed via ontology module replacement. That is, existing ontology modules should be replaced with other ontology modules, which can be new modules or extensions of existing ones. This way, the development of customized EEPISA ontologies is expected to be of bounded complexity. This ontology customization process is illustrated for a real-world poultry farm use case in Esnaola et al. (2019).

4. EEPISA Ontology In Use

The incorporation of the Semantic Technologies in the assistant that supports data analysts through the KDD process for energy efficiency and thermal comfort problems is presented in Esnaola-Gonzalez

⁴⁴<http://www.thezfiles.co.za/Modularity/TOMM.zip>

⁴⁵<https://github.com/iesnaola/eeipsa/tree/master/Evaluation/TOMM>

et al. (2018a). This assistant is based on the presented EEPsa ontology, and this section aims to show the role that the EEPsa ontology plays in this assistance. Namely, it proves that the proposed ontology is not a mere collection of classes and properties to semantically annotate data, but it is aimed at providing assistance to data analysts through the KDD process.

The capabilities of the EEPsa ontology are demonstrated in Tekniker's headquarters, a building located in Eibar (Spain) which hosts the activities of the technological centre. Within this building, the electronics laboratory (from now on referred to as ELE lab) has been targeted, where different electronic components, systems and equipment are designed, developed and tested. Due to the intermittent usage of the laboratory, ensuring occupants' thermal comfort while achieving an efficient use of energy is a currently unsolved problem. Towards finding a solution to this problem, a system that proposes the optimal HVAC (Heat, Ventilation and Air Conditioning) activation strategies is sought. Such a system could rely on the assistant that supports data analysts through the KDD processes, and consequently, the EEPsa Ontology.

The first step to exploit the EEPsa ontology capabilities consists in annotating the target space, its structural elements, deployed equipment and the rest of the relevant elements with adequate ontological terms. The building topology was represented using the FoI4EEPsa ontology module. The Tekniker building (*:tekniker*) was represented as an instance of *bot:Building*, its minus 2 storey (*:minus2Storey*) as an instance of *bot:Storey* and the ELE lab (*:eleLab*) located in such storey as an instance of *bot:Space*. Likewise, the representation of structural elements including the room's door (*:door01*) and the walls, were represented using the same FoI4EEPsa ontology module. As for the representation of the equipment deployed within the laboratory such as temperature sensors (*:tempSen01*) or submetering systems (*:plug01*), the EXR4EEPsa ontology module was used. Regarding the measurements registered by the different devices, the EXN4EEPsa ontology module was used, and more specifically, the *exn4eepsa:Observation* class. An excerpt of the simplified representation of the ELE lab is available below:

```
:tekniker rdf:type bot:Building;
          bot:hasStorey :minus2Storey .

:minus2Storey rdf:type bot:Storey;
              bot:hasSpace :eleLab .

:eleLab rdf:type bot:Space;
         bot:hasElement :door01;
         bot:hasElement :tempSen01;
         bot:hasElement :plug01 .

:door01 rdf:type foi4eepsa:Door .
:tempSen01 rdf:type exr4eepsa:TemperatureSensor .
:plug01 rdf:type exr4eepsa:SmartPlug .

:obs01 rdf:type exn4eepsa:Observation;
        eep:madeBy :tempSen01;
        eep:usedProcedure :sensingProcedure01;
        eep:onQuality :eleLabTemp .

:sensingProcedure01 rdf:type p4eepsa:SensingProcedure .
```

:eleLabTemp rdf:type q4eepsa:IndoorTemperature .

Once this semantic annotation process has been finished, a reasoner can be used to infer relevant implicit knowledge. In this case particular case, a Hermit⁴⁶ version 1.3.8 OWL 2 DL reasoner was used. Based on this data representation, the reasoner inferred that the ELE lab was an instance of class *ek4eepsa:BelowGroundLevelSpace*. As a matter of fact, this is how a below ground level space is defined in EK4EEPSA ontology module.

$$\text{ek4eepsa:BelowGroundLevelSpace} \equiv \text{bot:Space} \sqcap \exists \text{bot:hasStorey.foi4eepsa:UndergroundStorey} .$$

Furthermore, since ELE lab is inferred to be an instance of class *ek4eepsa:BelowGroundLevelSpace*, it is also inferred to be influenced by some below ground level space indoor temperature, due to the axioms:

$$\text{ek4eepsa:BelowGroundLevelSpace} \sqsubseteq \text{bot:Space} , \exists \text{aff:influencedBy.ek4eepsa:BelowGroundLevelSpaceTemperature} .$$

Consequently, due to the definition of *ek4eepsa:BelowGroundLevelSpaceTemperature*, it is inferred that the indoor temperature of ELE lab is affected by the atmospheric pressure (*q4eepsa:AtmosphericPressure*), indoor humidity (*q4eepsa:IndoorHumidity*) and the occupancy of the room (*q4eepsa:Occupancy*).

$$\begin{aligned} \text{ek4eepsa:BelowGroundLevelSpaceTemperature} &\sqsubseteq \\ \exists \text{aff:affectedBy.q4eepsa:AtmosphericPressure} &\sqcap \\ \exists \text{aff:affectedBy.q4eepsa:IndoorHumidity} &\sqcap \\ \exists \text{aff:affectedBy.q4eepsa:Occupancy} . \end{aligned}$$

Therefore, once the data analyst makes the semantic representation of the ELE laboratory, thanks to the expert knowledge captured in the EEPsa ontology and the execution of a reasoner, it is inferred that the ELE laboratory is below ground level space, and consequently, its temperature is affected by atmospheric pressure, indoor humidity and laboratory occupancy. This example is a clear demonstration on how the EEPsa ontology can support data analysts through a KDD process for solving energy efficiency and thermal comfort problems.

5. Conclusions

Following a well-known ontology development methodology, the requirements of the EEPsa ontology were collected first. Then, the backbone of the EEPsa ontology was discussed and defined as a combination of three ODPs which try to be minimal in the number of classes and properties offered but complete with respect to the considered CQs and including appropriate ontology axioms that allow proper inferences. Moreover, the careful design of property axioms overcome some weaknesses discovered in existing ontologies. On top of these ODPs, a set of ontology modules were developed, each of them specializing knowledge in the scope of the stub classes defined in the ODPs and reusing existing resources to the extent possible. Furthermore, in order to contribute to the interoperability of

⁴⁶<http://www.hermit-reasoner.com/>

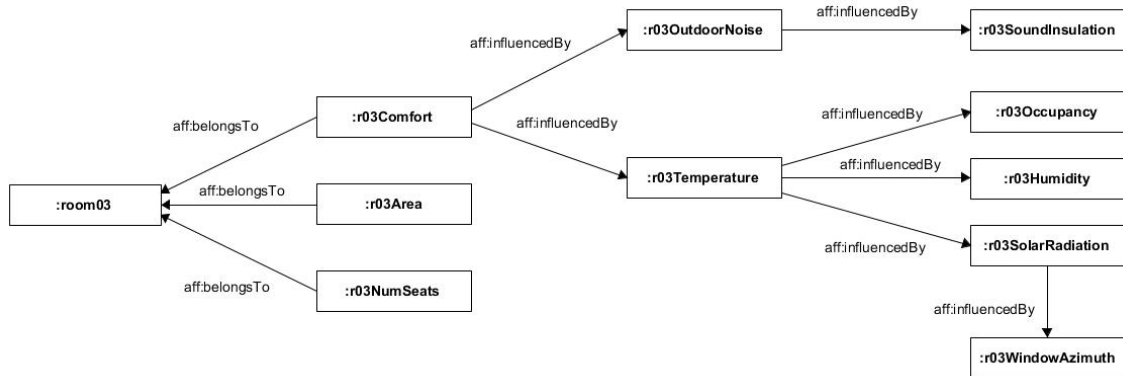


Fig. 7. Triples using the AffectedBy ODP vocabulary.

the solution, EEPsA ODPs and ontology modules are aligned with other related ontologies as well as upper-level ontologies. All these developments are properly documented, made available online, and evaluated from three different viewpoints. Moreover, thanks to the design of the EEPsA ontology and the high encapsulation of its modules, the customization of the ontology to address similar problems in different use cases is feasible.

The resulting EEPsA ontology is a core ontology which aims at supporting a data analyst assistant in energy efficiency and thermal comfort problems in buildings. This ontology has been instantiated in a real world laboratory, and it has been demonstrated that it facilitates the discovery of knowledge that can support data analysts.

Acknowledgements

This work is partly supported by the projects Internet of Food & Farms 2020 and RESPOND (integrated demand REsponse Solution towards energy POSitive Neighbourhoods) which have received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no. 731884 and no. 768619 respectively. This work is also supported by PRE-2016-1-0303, IT1041-16-GBV and FEDER/TIN2016-78011-C4-2-R.

This work was conducted using the Protégé resource, which is supported by grant GM10331601 from the National Institute of General Medical Sciences of the United States National Institutes of Health.

Appendix A. Application Examples of the ODPs

This appendix shows an example of the presented three ODPs.

A.1. AffectedBy ODP Example

Figure 7 shows a triple graph as an example for applying and answering some competency questions using the AffectedBy vocabulary.

With respect to this example, the following competency questions can be applied and answered:

- (CQ01): What are the properties that influence the feature of interest *:room03*?
 SELECT ?x
 WHERE { :room03 aff:influencedBy ?x. }
 Answer: *:r03Area, :r03NumSeats :r03Comfort, :r03Temperature, :r03OutdoorNoise, :r03Occupancy, :r03Humidity, :r03SolarRadiation, :r03SoundInsulation, :r03WindowAzimuth.*
 (After inferences provided by axioms $\text{aff:influencedBy} \circ \text{aff:affectedBy} \sqsubseteq \text{aff:influencedBy}$ and $\text{aff:belongsTo} \sqsubseteq \text{aff:influencedBy}^{-1}$).
- (CQ01ⁱ): Which is the feature of interest influenced by the property *:r03SolarRadiation*?
 SELECT ?x
 WHERE { ?x aff:influencedBy ?r03SolarRadiation. }
 Answer: *:room03.*
 (After inferences provided by the axioms $\text{aff:influencedBy} \circ \text{aff:affectedBy} \sqsubseteq \text{aff:influencedBy}$ and $\text{aff:belongsTo} \sqsubseteq \text{aff:influencedBy}^{-1}$).
- (CQ02): What are the properties that affect the property *:r03Temperature*?
 SELECT ?x
 WHERE { :r03Temperature aff:affectedBy ?x. }
 Answer: *:r03Occupancy, :r03Humidity, :r03SolarRadiation, :r03WindowAzimuth.*
 (After inferences provided by the transitivity of aff:affectedBy).
- (CQ03): Which feature of interest does the property *:r03Area* belongs to?
 SELECT ?x
 WHERE { :r03Area aff:belongsTo ?x. }
 Answer: *:room03.*

A.2. EEP ODP Example

Figure 8 shows an instantiation of the EEP ODP in a farm scenario where poultry are reared. In this case, a sensor *:sensor36* deployed in the farm individual *:farm* is in charge of measuring both farm's temperature and humidity (i.e. *:farmTemperature* and *:farmHumidity*). Furthermore, this sensor implements a monitoring procedure (*:monitoringProc*) to make two observations *:obs13* and *:obs14*.

With respect to this example, the following competency questions can be applied and answered:

- (CQ04): What are the executions performed by procedure *:monitoringProc*?
 SELECT ?x
 WHERE { ?x eep:usedProcedure :monitoringProc. }
 Answer: *:obs13, :obs14.*
- (CQ05): What are the observations performed by sensor *:sensor36*?
 SELECT ?x
 WHERE { ?x eep:madeBy :sensor36. }
 Answer: *:obs13, :obs14.*

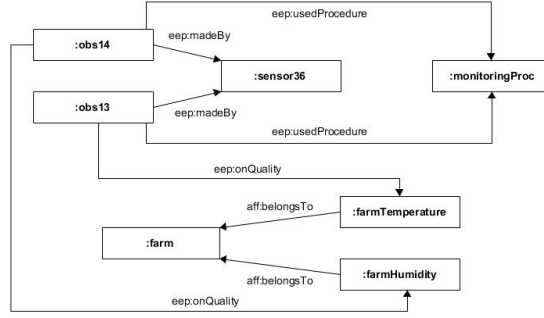


Fig. 8. Triples using the EEP ODP vocabulary.

- (CQ06): Which are the procedures implemented by the sensor `:sensor36`?
 SELECT ?x
 WHERE { :sensor36 eep:implements ?x. }
 Answer: `:monitoringProc`
 (After inferences provided by the axiom $eep:madeBy^{-1} \circ eep:usedProcedure \sqsubseteq eep:implements$).
- (CQ07ⁱ): What are the executions on the feature of interest `:farm`?
 SELECT ?x
 WHERE { ?x eep:hasFeatureOfInterest :farm. }
 Answer: `:obs13, :obs14`.
 (After inferences provided by the axiom $eep:onQuality \circ eep:belongsTo \sqsubseteq eep:hasFeatureOfInterest$).
- (CQ08): What are the qualities observed by the observation `:obs13`?
 SELECT ?x
 WHERE { :obs13 eep:onQuality ?x. }
 Answer: `:farmTemperature`.
- (CQ09ⁱ): What are the executors that observe/act on the feature of interest `:farm`?
 SELECT ?x
 WHERE { ?x eep:forFeatureOfInterest :farm. }
 Answer: `:sensor36`.
 (After inferences provided by the axioms $eep:forQuality \circ eep:belongsTo \sqsubseteq eep:forFeatureOfInterest$ and $eep:madeBy^{-1} \circ eep:onQuality \sqsubseteq eep:forQuality$).
- (CQ10): What are the qualities observed by sensor `:sensor36`?
 SELECT ?x
 WHERE { :sensor36 eep:forQuality ?x. }
 Answer: `:farmTemperature, :farmHumidity`.
 (After inferences provided by the axiom $eep:madeBy^{-1} \circ eep:onQuality \sqsubseteq eep:forQuality$).

A.3. RC ODP Example

The RC ODP is instantiated in a weather forecast report. In this case, an execution *:forecastReport* is generated on 2018-11-02 at 11:00 (with the data property *rc:hasGenerationTime*) and forecasts that there will be a temperature of 16°C (with the data property *rc:hasSimpleResult*) in Madrid (with the object property *rc:hasSpatialContext*) on 2018-11-03 at 16:00 (with the data property *rc:hasTemporalContext*). Figure 9 shows this instantiation example.

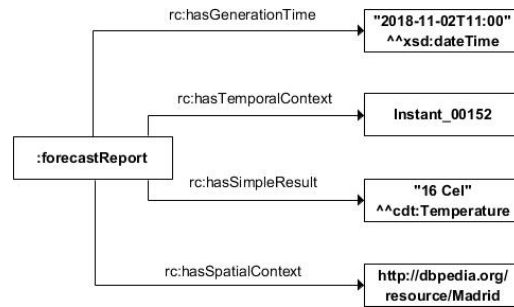


Fig. 9. Triples using the RC ODP vocabulary.

With respect to this example, the following competency questions can be applied and answered:

- (CQ11): Which is the simplified value of execution *:forecastReport*?
 SELECT ?x
 WHERE {*:forecastReport ec:hasSimpleResult ?x.*}
 Answer: "16 Cel"^^cdt:temperature.
- (CQ12): When is the execution *:forecastReport* generated?
 SELECT ?x
 WHERE {*:forecastReport ec:hasGenerationTime ?x.*}
 Answer: "2018-11-02T11:00:00"^^xsd:dateTime.
- (CQ13): For what time interval or instant is valid the execution *:forecastReport*?
 SELECT ?x
 WHERE {*:forecastReport ec:hasTemporalContext ?x.*}
 Answer: *:Instant_00152*.
- (CQ14): For what spatial location is valid the execution *:forecastReport*?
 SELECT ?x
 WHERE {*:forecastReport ec:hasSpatialContext ?x.*}
 Answer: *http://dbpedia.org/resource/Madrid*.

References

- Abergel, T., Dean, B., and Dulac, J. (2017). Towards a zero-emission, efficient, and resilient buildings and construction sector: Global status report. *UN Environment and International Energy Agency (2017)*.

- Agarwal, R., Fernandez, D. G., Elsaleh, T., Gyrard, A., Lanza, J., Sanchez, L., Georgantas, N., and Issarny, V. (2016). Unified
iot ontology to enable interoperability and federation of testbeds. In *3rd IEEE World Forum on Internet of Things*.
- Andrews, P., Zaihrayeu, I., and Pane, J. (2012). A classification of semantic annotation systems. *Semantic Web*, 3(3):223–248.
- Bermudez-Edo, M., Elsaleh, T., Barnaghi, P., and Taylor, K. (2017). Iot-lite: A lightweight semantic model for the internet of
things and its use with dynamic semantics. *Personal and Ubiquitous Computing*, 21(3):475–487.
- Bernstein, A., Provost, F., and Hill, S. (2005). Toward intelligent assistance for a data mining process: An ontology-based
approach for cost-sensitive classification. *IEEE Transactions on Knowledge and Data Engineering*, 17(4):503–518.
- Brank, J., Grobelnik, M., and Mladenović, D. (2005). A survey of ontology evaluation techniques. In *Proc. of 8th Int. multi-conf.
Information Society*, pages 166–169.
- Calegari, N., Burle, C., and Loscio, B. F. (2017). Data on the web best practices. W3C recommendation, W3C.
<https://www.w3.org/TR/2017/REC-dwbp-20170131/>.
- Compton, M., Barnaghi, P., Bermudez, L., García-Castro, R., Corcho, O., Cox, S., Graybeal, J., Hauswirth, M., Henson, C.,
and Herzog, A. (2012). The ssn ontology of the w3c semantic sensor network incubator group. *Web Semantics: Science,
Services and Agents on the World Wide Web*, 17:25–32.
- Cox, S. (2016). Ontology for observations and sampling features, with alignments to existing models. *Semantic Web*, 8(3):453–
470.
- d’Aquino, M., Schlicht, A., Stuckenschmidt, H., and Sabou, M. (2009). *Criteria and Evaluation for Ontology Modularization
Techniques*, pages 67–89. Springer Berlin Heidelberg, Berlin, Heidelberg.
- de Farias, T. M., Roxin, A., and Nicolle, C. (2015). Ifcwod, semantically adapting ifc model relations into owl properties.
Proceedings of the 32nd CIB W78 Conference on Information Technology in Construction.
- Ensan, F. and Du, W. (2013). A semantic metrics suite for evaluating modular ontologies. *Inf. Syst.*, 38(5):745–770.
- Esnaola, I., Fernandez, I., García, E., Ferreira, S., Gomez, M., Lázaro, I., and García, A. (2019). Towards animal welfare in
poultry farms through semantic technologies. In *IoT Connected World & Semantic Interoperability Workshop (IoT-CWSI)
2019*.
- Esnaola-Gonzalez, I., Bermúdez, J., Fernandez, I., and Arnaiz, A. (2018a). Semantic prediction assistant approach applied to
energy efficiency in tertiary buildings. *Semantic Web*, 9(6):735–762.
- Esnaola-Gonzalez, I., Bermúdez, J., Fernandez, I., and Arnaiz, A. (2018b). Two ontology design patterns toward energy
efficiency in buildings. In *Proceedings of the 9th Workshop on Ontology Design and Patterns (WOP 2018) co-located with
17th International Semantic Web Conference (ISWC 2018)*, volume 2195, pages 14–28. CEUR.
- Esnaola-Gonzalez, I., Bermúdez, J., Fernandez, I., and Arnaiz, A. (2020). Ontologies for observations and actuations in build-
ings: A survey. *Semantic Web*, Accepted - To be published.
- Fayyad, U., Piatetsky-Shapiro, G., and Smyth, P. (1996). From data mining to knowledge discovery in databases. *AI magazine*,
17(3):37.
- Gangemi, A., Lillo, R., Lodi, G., and Nuzzolese, A. G. (2017). A pattern-based ontology for the internet of things. *Proceedings
of the 8th Workshop on Ontology Design and Patterns (WOP 2017)*, 2043.
- Gangemi, A. and Presutti, V. (2009). *Ontology Design Patterns*, pages 221–243. Springer Berlin Heidelberg, Berlin, Heidelberg.
- Garijo, D. (2017). Widoco: A wizard for documenting ontologies. In d’Amato, C., Fernandez, M., Tamma, V., Lecue, F.,
Cudré-Mauroux, P., Sequeda, J., Lange, C., and Heflin, J., editors, *The Semantic Web – ISWC 2017*, pages 94–102, Cham.
Springer International Publishing.
- Garijo, D. and Poveda-Villalón, M. (2017). A checklist for complete vocabulary metadata. Technical report.
- Grau, B. C., Horrocks, I., Kazakov, Y., and Sattler, U. (2008). Modular reuse of ontologies: Theory and practice. *Journal of
Artificial Intelligence Research*, 31:273–318.
- Gubbi, J., Buyya, R., Marusic, S., and Palaniswami, M. (2013). Internet of things (iot): A vision, architectural elements, and
future directions. *Future generation computer systems*, 29(7):1645–1660.
- Haller, A., Janowicz, K., Cox, S., Lefrançois, M., Taylor, K., Phuoc, D. L., Lieberman, J., García-Castro, R., Atkinson, R.,
and Stadler, C. (2018). The modular ssn ontology: A joint w3c and ogc standard specifying the semantics of sensors,
observations, sampling, and actuation. *Semantic Web*, To be published.
- Haynes, B. P. (2008). The impact of office comfort on productivity. *Journal of Facilities Management*, 6(1):37–51.
- Hedge, A. and Gaygen, D. E. (2010). Indoor environment conditions and computer work in an office. *Hvac&R Research*,
16(2):123–138.
- Hitzler, P., Gangemi, A., and Janowicz, K. (2016). *Ontology Engineering with Ontology Design Patterns: Foundations and
Applications*, volume 25. IOS Press.
- Janowicz, K. and Compton, M. (2010). The stimulus-sensor-observation ontology design pattern and its integration into the
semantic sensor network ontology. volume 668.
- Khan, Z. C. and Keet, C. M. (2016). Dependencies between modularity metrics towards improved modules. In Blomqvist,
E., Ciancarini, P., Poggi, F., and Vitali, F., editors, *Knowledge Engineering and Knowledge Management*, pages 400–415,
Cham. Springer International Publishing.

- Klepeis, N. E., Nelson, W. C., Ott, W. R., Robinson, J. P., Tsang, A. M., Switzer, P., Behar, J. V., Hern, S. C., and Engelmann, W. H. (2001). The national human activity pattern survey (nhaps): a resource for assessing exposure to environmental pollutants. *Journal of Exposure Science and Environmental Epidemiology*, 11(3):231.
- Kopanas, I., Avouris, N. M., and Daskalaki, S. (2002). The role of domain knowledge in a large scale data mining project. In Vlahavas, I. P. and Spyropoulos, C. D., editors, *Methods and Applications of Artificial Intelligence*, pages 288–299, Berlin, Heidelberg. Springer Berlin Heidelberg.
- Lefrançois, M. (2017). Planned etsi saref extensions based on the w3c&ogc sosa/ssn-compatible seas ontology patterns. In *Proceedings of Workshop on Semantic Interoperability and Standardization in the IoT, SIS-IoT*.
- Mulville, M., Callaghan, N., and Isaac, D. (2016). The impact of the ambient environment and building configuration on occupant productivity in open-plan commercial offices. *Journal of Corporate Real Estate*, 18(3):180–193.
- Musen, M. A. (2015). The protégé project: a look back and a look forward. *AI matters*, 1(4):4–12.
- Noy, N. F. (2004). Semantic integration: a survey of ontology-based approaches. *ACM Sigmod Record*, 33(4):65–70.
- Obrst, L., Ceusters, W., Mani, I., Ray, S., and Smith, B. (2007). *The Evaluation of Ontologies*, pages 139–158. Springer US, Boston, MA.
- Parsons, K. (2014). *Human Thermal Environments: The Effects of Hot, Moderate, and Cold Environments on Human Health, Comfort, and Performance*. CRC Press, Inc., Boca Raton, FL, USA, 3 edition.
- Peroni, S., Shotton, D., and Vitali, F. (2013). Tools for the automatic generation of ontology documentation: A task-based evaluation. *Int. J. Semant. Web Inf. Syst.*, 9(1):21–44.
- Pinto, F. M. and Santos, M. F. (2009). Considering application domain ontologies for data mining. *WSEAS Trans. Info. Sci. and App.*, 6(9):1478–1492.
- Pinto, H. S., Staab, S., and Tempich, C. (2004). Diligent: Towards a fine-grained methodology for distributed, loosely-controlled and evolving engineering of ontologies. In *Proceedings of the 16th European Conference on Artificial Intelligence (ECAI)*, pages 393–397. IOS Press.
- Poveda-Villalón, M., Gómez-Pérez, A., and Suárez-Figueroa, M. C. (2014). Oops! (ontology pitfall scanner!): An on-line tool for ontology evaluation. *Int. J. Semant. Web Inf. Syst.*, 10(2):7–34.
- Qiua, H., Schneider, G., Kauppinen, T., Rudolph, S., and Steigerd, S. (2018). Reasoning on human experiences of indoor environments using semantic web technologies. In *Proceedings of the 35th International Symposium on Automation and Robotics in Construction (ISARC 2018)*, Berlin, Germany.
- Rasmussen, M. H., Pauwels, P., Hviiid, C. A., and Karlshøj, J. (2017). Proposing a central aec ontology that allows for domain specific extensions. In *Joint Conference on Computing in Construction*, volume 1, pages 237–244.
- Sagar, S., Lefrançois, M., Rebaï, I., Khemaja, M., Garlatti, S., Feki, J., and Médini, L. (2018). Modeling smart sensors on top of sosa/ssn and wot/d with the semantic smart sensor network (s3n) modular ontology. In *9th International Semantic Sensor Networks Workshop*.
- Seydoux, N., Drira, K., Hernandez, N., and Monteil, T. (2016). Iot-o, a core-domain iot ontology to represent connected devices networks. In *Knowledge Engineering and Knowledge Management: 20th International Conference, EKAW 2016, Bologna, Italy, November 19-23, 2016, Proceedings 20*, volume 10024, pages 561–576. Springer.
- Simperl, E. (2009). Reusing ontologies on the semantic web: A feasibility study. *Data & Knowledge Engineering*, 68(10):905–925.
- Stuckenschmidt, H. and Klein, M. (2007). Reasoning and change management in modular ontologies. *Data & Knowledge Engineering*, 63(2):200 – 223.
- Suárez-Figueroa, M. C. and Gómez-Pérez, A. (2012). *Ontology Requirements Specification*, pages 93–106. Springer Berlin Heidelberg, Berlin, Heidelberg.
- Suárez-Figueroa, M. C., Gómez-Pérez, A., and Fernández-López, M. (2012). *The NeOn Methodology for Ontology Engineering*, pages 9–34. Springer Berlin Heidelberg, Berlin, Heidelberg.
- Sure, Y., Staab, S., and Studer, R. (2004). *On-To-Knowledge Methodology (OTKM)*, pages 117–132. Springer Berlin Heidelberg, Berlin, Heidelberg.
- Verbeke, S. and Audenaert, A. (2018). Thermal inertia in buildings: A review of impacts across climate and building use. *Renewable and Sustainable Energy Reviews*, 82:2300 – 2318.
- Vrandečić, D. and Sure, Y. (2007). How to design better ontology metrics. In Franconi, E., Kifer, M., and May, W., editors, *The Semantic Web: Research and Applications*, pages 311–325, Berlin, Heidelberg. Springer Berlin Heidelberg.
- Yoon, S.-C., Henschen, L. J., Park, E. K., and Makki, S. (1999). Using domain knowledge in knowledge discovery. In *Proceedings of the Eighth International Conference on Information and Knowledge Management, CIKM '99*, pages 243–250, New York, NY, USA. ACM.