MONDRAGON
UNIBERTSITATEA

GOI ESKOLA
POLITEKNIKOA
FACULTY OF
ENGINEERING

# A METHODOLOGY FOR DESIGNING LAYERED ONTOLOGY STRUCTURES

Javier Cuenca Ariza

Computer and Electronics Departament
Faculty of Engineering
Mondragon Unibertsitatea

November 2019

MONDRAGON
UNIBERTSITATEA

GOI ESKOLA
POLITEKNIKOA

FACULTY OF
ENGINEERING

# A METHODOLOGY FOR DESIGNING LAYERED ONTOLOGY STRUCTURES

Javier Cuenca Ariza

Supervisor:

Felix Larrinaga Barrenechea
Computer and Electronics Departament
Faculty of Engineering
Mondragon Unibertsitatea

Co-Supervisor:

Edward Curry
Insight Centre For Data Analytics
National University of Ireland

*for obtaining the degree of Doctor under the doctoral program at Mondragon Unibertsitatea: Applied Engineering*

**Thesis Committee:**
Chair: Dr. Arkaitz Zubiaga Mendialdua (Queen Mary University of London)
Member: Dr. Izaskun Fernández González (IK4-Tekniker)
Member: Dr. Filipe De Carvalho Moutinho (NOVA University of Lisbon)
Member: Dr. María Poveda Villalón (Universidad Politécnica de Madrid)
Secretary: Dr. Alain Pérez Riaño (Mondragon Unibertsitatea)

November 2019

## Originality Statement

I declare that I am the sole author of this work. This is a true copy of the final document, including any revision which may have been ordered by my examiners. I understand that my work may be available to the public, either in the school library or in electronic format.

# Abstract

Semantic ontologies represent the knowledge from different domains, which is used as a knowledge base by intelligent agents. The creation of ontologies by different developers leads to heterogeneous ontologies, which hampers the interoperability between knowledge-based applications. This interoperability is achieved through global ontologies, which provide a common domain representation. Global ontologies must provide a balance of reusability-usability to minimise the ontology reuse effort in different applications.

To achieve this balance, ontology design methodologies focus on designing layered ontologies that classify into abstraction layers the domain knowledge relevant to many applications and the knowledge relevant to specific applications. During the design of the layered ontology structure, the domain knowledge classification is performed from scratch by domain experts and ontology engineers in collaboration with application stakeholders. Hence, the design of reusable and usable ontologies in complex domains takes a significant effort. Software Product Line (SPL) design techniques can be applied to facilitate the domain knowledge classification by analysing the knowledge similarities/differences of existing ontologies.

In this context, this thesis aims to define new methodological guidelines to design layered ontology structures that enable to classify the domain knowledge taking as reference existing ontologies, and to apply these guidelines to enable the development of reusable and usable ontologies in complex domains.

The MODDALS methodology guides the design of layered ontology structures for reusable and usable ontologies. It brings together SPL engineering techniques and ontology design techniques to enable the classification of the domain knowledge by exploiting the knowledge similarities/differences of existing ontologies. MODDALS eases the design of the layered ontology structure.

The MODDALS methodology was evaluated by applying it to design the layered structure of a reusable and usable global ontology for the energy domain. The designed layered structure was taken as reference to develop the ontology. The resulting ontology simplifies the ontology reuse process in different applications. In particular, it reduced the average ontology reuse time by 0.5 and 1.2 person-hours in in two different applications in comparison with a global energy ontology which does not follow a layered structure.

# Resumen

Las ontologías semánticas representan el conocimiento de diferentes dominios, utilizado como base de conocimiento por agentes inteligentes. Las ontologías son desarrolladas por diferentes ingenieros y son heterogéneas, afectando a la interoperabilidad entre aplicaciones. Esta interoperabilidad se logra mediante ontologías globales que proporcionan una representación común del dominio, las cuales deben proporcionar un balance de reusabilidad-usabilidad para minimizar el esfuerzo de reutilización en diferentes aplicaciones.

Para lograr este balance, las metodologías de diseño de ontologías proponen clasificar en capas de abstracción el conocimiento del dominio común a muchas aplicaciones y el que es relevante para aplicaciones específicas. Durante el diseño de la estructura de capas, el conocimiento se clasifica partiendo de cero por expertos del dominio e ingenieros de ontologías. Por lo tanto, el diseño de ontologías reusables y usables en dominios complejos requiere un gran esfuerzo. Las técnicas de diseño de líneas de producto de software pueden facilitar la clasificación del conocimiento analizando las similitudes/diferencias de conocimiento de ontologías existentes.

En este contexto, el objetivo de la tesis es crear una metodología de diseño de la estructura de capas para ontologías que permita clasificar el conocimiento tomando como referencia ontologías existentes, y aplicar esta metodología para poder desarrollar ontologías reusables y usables en dominios complejos.

La metodología MODDALS explica cómo diseñar estructuras de capas para ontologías reusables y usables. MODDALS adopta técnicas de diseño de líneas de producto en combinación con técnicas de diseño de ontologías para clasificar el conocimiento basándose en las similitudes/diferencias de ontologías existentes. Este enfoque facilita el diseño de la estructura de capas de la ontología.

La metodología MODDALS se ha evaluado aplicándola para diseñar la estructura de capas de una ontología global reusable y usable para el dominio de la energía. La estructura de capas diseñada se ha tomado como referencia para desarrollar la ontología. Con esta estructura, la ontología resultante simplifica la reutilización de ontologías en diferentes aplicaciones. En concreto, la ontología redujo el tiempo de reutilización en 0.5 y 1.2 personas-hora en dos aplicaciones respecto a una ontología global que no sigue una estructura por capas.

# Laburpena

Ontologia semantikoak datu domeinu ezberdinen ezagutza irudikatzen dute, agente adimendunek jakintza oinarri bezala erabiltzen dutena. Ontologiak ingeniari desberdinek garatzen dituzte eta heterogeneoak dira, aplikazioen arteko komunikazioa oztopatuz. Komunikazio hau ontologia globalen bidez lortzen da, domeinuaren errepresentazio komun bat ematen baitute. Ontologia globalek berrerabilgarritasun-erabilgarritasun oreka eman behar dute aplikazio desberdinetan berrerabiltzeko ahalegina murrizteko.

Horretarako, ontologia diseinu metodologiek aplikazio askok erabiltzen duten eta aplikazio zehatzetarako garrantzitsua den ezagutza abstrakzio geruzetan sailkatzea proposatzen dute. Geruza egituraren diseinuan zehar, domeinuko adituek eta ontologiako ingeniariek hutsetik sailkatzen dute jakintza, domeinu konplexuetan ontologia berrerabilgarriak eta erabilgarrien diseinu ahalegina areagotuz. Software produktu lerroak diseinatzeko erabiltzen diren teknikak jakintza sailkatzea erraztu ahal dute, ontologien ezagutza antzekotasunak edo desberdintasunak aztertuz.

Testuinguru honetan, honakoa da tesiaren helburua: ezagutza garatutako ontologien arabera sailkatzen duen ontologia berrerabilgarri eta erabilgarrien geruza egitura diseinatzeko metodologia bat garatzea; baita metodologia aplikatu ere, ontologia berrerabilgarri eta erabilgarriak domeinu konplexuetan garatu ahal izateko.

MODDALS metodologiak ontologia berrerabilgarri eta erabilgarrien abstrakzio geruzak nola diseinatu azaltzen du. MODDALS-ek software produktu lerro eta ontologia diseinu teknikak aplikatzen ditu ezagutza garatuta dauden ontologien antzekotasunen/desberdintasunen arabera sailkatzeko. Planteamendu honek geruza egitura diseinua errazten du.

MODDALS ebaluatu da energia domeinurako ontologia berrerabilgarri eta erabilgarri baten egitura diseinatzeko aplikatuz. Diseinatutako geruza egitura erreferentzia gisa hartu da ontologia gartzeko. Egitura onekin, garatutako ontologia berrerabiltzea errazten du aplikazio desberdinetan. Konkretuki, garatutako ontologiak berrerabilpen denbora 0.5 eta 1.2 pertsona-orduetan murriztu du bi aplikazioetan; geruza egitura jarraitzen ez duen ontologia batekin alderatuz.

# Agradecimientos - Acknowledgments

Me gustaría expresar mi gratitud a todos aquellos que me han ayudado durante todo el desarrollo de esta tesis.

En primer lugar, estoy muy agradecido al director, Félix Larrinaga, por darme la oportunidad de participar en este trabajo de investigación y por su ayuda, experiencia y conocimientos que han sido muy útiles para el desarrollo de la tesis. Su apoyo ha sido constante, dándome valiosos consejos y animándome en todo momento. Agradezco también a Mondragon Unibertsitatea por concederme la beca que ha hecho posible esta investigación.

I would also like to thank to the co-director of the thesis, Edward Curry, for his guidance during the thesis and giving me the opportunity to work with him and his team at the Insight Centre For Data Analytics (National University of Ireland). His feedback and rigor when reviewing our papers has been always valuable. I would also like to thank Insight Ph.D. students who helped me during my stay and participated in the experiments we designed to validate the solutions developed in this thesis.

Me gustaría agradecer también a todos los compañeros del departamento de informática y electrónica y del grupo de investigación de ingeniería de software y sistemas de Mondragon Unibertsitatea, investigadores de IK4 Tekniker (Izaskun e Iker), Jesús Bermúdez (Universidad Pública Vasca) y los miembros del Ontology Engineering Group (OEG) de la Universidad Politécnica de Madrid, por haber valorado mi tesis en sus diferentes etapas y haberme ofrecido su consejo durante la misma. También al resto de estudiantes de doctorado, ya que hemos compartido las experiencias que hemos tenido durante la tesis y nos hemos apoyado mutuamente.

A mis amigos, que han estado siempre conmigo y me ayudaron a desconectar en los momentos en los que lo necesitaba. Gracias por los buenos momentos que han hecho más llevadero este largo camino.

Por último, me gustaría dar las gracias a mis padres, Julio e Isabel, y a mi hermano Mikel, que me han apoyado en todas las decisiones que he tomado en mi vida, por sus consejos y por ayudarme de todas las maneras posibles.

# Contents

# List of Figures

# List of Tables

# Acronyms

**3GPP** 3rd Generation Partnership Project

**AI** Artificial Intelligence

**CQ** Competency Question

**CP** Content Ontology Design Pattern

**CVA** Commonality and Variability analysis

**CV ratio** Commonality Ratio

**DABGEO** Domain Analysis-Based Global Energy Ontology

**DER** Distributed Energy Resource

**DR** Demand Response

**DSO** Distribution System Operator

**ENR** Ontology edge node ratio

**EPAS** energy performance assessment system

**FODA** Feature-Oriented Domain Analysis

**GEPSS** Green Energy Provider Selection System

**HMI** Human Machine Interface

**HTML** HyperText Markup Language

**HTTP** Hypertext Transfer Protocol

**HVAC** heating, ventilation and air conditioning

**ICT** Information Communications Technology

**IOT** Internet of Things

**KA** Knowledge area

**KADD** Knowledge Area Description Document

**KA-Schema** Knowledge Area Schema

**MODDALS** Methodology for Ontology Design based in Domain Analysis and Layered Structure

**ODP** Ontology Design Pattern

**OEMA** Unified Ontology for Energy Management Applications

**OWL** Web Ontology Language

**RDF** Resource Description Framework

**RDFS** RDF Vocabulary Definition Language

**RQ** Research Question

**SOV** Size of vocabulary

**SPARQL** RDF Query Language

**SPL** Software Product Line

**TIP** Tree Impurity

**URI** Uniform Resource Identifier

**VE** Vocabulary expressiveness

**VPP** Virtual Power Plant

**W3C** World Wide Web Consortium

**WWW** World Wide Web

**XML** eXtensible Markup Language

# Part I

# Foundation and Context

Chapter 1

# Introduction

The advances made in the field of Information Communications Technology (ICT), in particular enabling technologies from Internet of Things (IOT), 5G, edge computing to Big Data, cloud computing, and Artificial Intelligence will make smart environments a reality [2]. Weiser et al. [166] defined a smart environment as "*a physical world that is richly and invisibly interwoven with sensors, actuators, displays, and computational elements, embedded seamlessly in the everyday objects of our lives, and connected through a continuous network*". In the past decade, smart environments have started to move from a research vision to concrete manifestations in real-world deployments. Some examples of emerging smart environments are Smart Grid, Smart Cities or Smart Enterprises [36].

As the IoT is enabling the deployment of lower-cost sensors, we see more broad adoption of IoT devices/sensors and gain more visibility (and data) into smart environments. This results in large quantities of high volume and high-velocity data from smart environments that belong to different and heterogeneous domains [41] (we consider a data domain as a set of related concepts that belong to a specific area of interest [75]). This data must be exchanged, related and processed by intelligent agents to create a data ecosystem: a socio-technical system enabling value to be extracted from data value chains supported by interacting organisations and individuals. By creating a data ecosystem, intelligent agents can extract knowledge from surroundings and make decisions based on the obtained knowledge to adapt the environment to inhabitants needs and respond to disruptive situations [38].

Smart environments are complex systems where data from heterogeneous domains must be collected, exchanged at high rates and in real-time. In addition, these data must be related to extract knowledge for intelligent decision-making. However, the ICT-based systems that gather and store data from different domains have traditionally operated in functional silos and rely on heterogeneous technologies [6]. These factors hinder the data exchange and knowledge extraction within smart environments and pose new interoperability challenges. In particular, new knowledge representation and exchange technologies are needed to enable intelligent decision-making within smart environments [60].

This challenge can be addressed through the use of knowledge graphs represented with ontology vocabularies [41]. In 2012 Google coined the term "Knowledge Graph" to refer to their use of information gathered from multiple sources to enrich their services, including search engine results. As defined by Paulheim [125] a "*knowledge graph (i) mainly describes real-world entities and their interrelations, organized in a graph, (ii) defines possible classes and relations of entities in a schema, (iii) allows for potentially interrelating arbitrary entities with each other and (iv) covers various topical domains*". As illustrated in Figure 1.1, a knowledge graph consists of a set of entities or concepts (i.e., persons, places) and a set of relations (i.e., hasName, knows, hasJob) used to relate and describe entities [89]. The combination of entities and relations enables to express facts about entities (i.e., *John knows Mary*, or *Mary hasGender Female*). More formally, each fact is known as statement or triple and has a Subject-Predicate-Object structure. The subject corresponds to an entity, the predicate corresponds to a relation and the object can be an entity related with the subject or a literal value that describes it. A set of statements conform a knowledge graph.

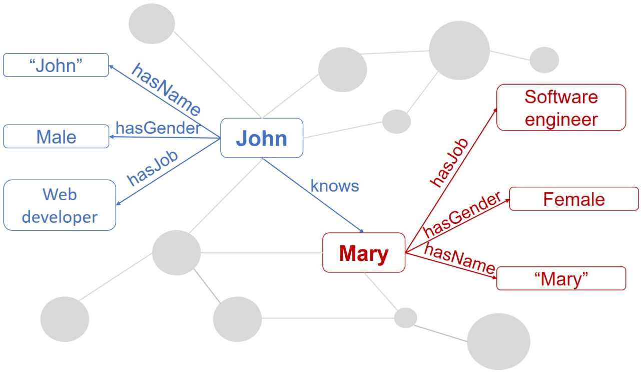


**Figure 1.1:** Example of a knowledge graph of Mary and John

With this structure, knowledge graphs enable to establish links and discover relations between data from heterogeneous domains in a flexible way, thus creating a knowledge network that eliminates data silos [125]. Knowledge graphs are used as knowledge bases by intelligent agents that perform complex analysis and decision-making about data from different domains [49].

Semantic Web technologies enable to create knowledge graphs and to exchange knowledge [94]. Over the last two decades, the Semantic Web has emerged as a major evolution of the current web [7]. The main purpose of such an extended web is to structure and give meaning to data published on the web, so that it can be processed and understood by personalized software agents in an attempt to automate many aspects of our lives. For that purpose, the Semantic Web encompasses a set of technologies that enable the creation of vocabularies. These vocabularies provide machine-readable data representation and link data from heterogeneous data domains. The Semantic Web vocabularies enable to integrate and discover relations between data from heterogeneous data domains in a flexible way, thus creating a knowledge network that eliminates data silos [7]. In addition, the Semantic Web provides standard mechanisms to access, exchange and process the knowledge represented with Semantic Web vocabularies for systems that rely on heterogeneous technologies [140]. With these technologies, the Semantic Web enables a better performance to intelligent agents and data analysis applications used for knowledge extraction and decision-making [75].

Ontologies are a basic pillar of the Semantic Web, since they are the formal representation of Semantic Web vocabularies [7]. They describe and represent a data domain as a set of concepts and relationships following a knowledge graph-based representation, in order to create a generic knowledge that can be shared across different software applications [69]. Ontologies also provide complex data representation mechanisms so that machines are capable of inferring knowledge from explicit facts. So far, ontologies are commonly developed to represent the knowledge from different domains, which is used as a knowledge base by intelligent agents in different fields, i.e., Artificial Intelligence (AI), Knowledge Engineering and Computer Science [150]. Ontologies are also used to create knowledge bases that enable intelligent decision-making in smart environments in order to adapt the environment to inhabitants and respond to disruptive situations [32, 36].

## 1.1 Research Context

The application of the Semantic Web in different domains has led to the development of a vast number of ontologies. These ontologies provide the vocabularies to represent the knowledge bases of applications with different purposes. Ontologies are developed by different engineers, who have different points of view when it comes to represent the knowledge of the same data domains. In some domains, the knowledge requirements of different applications overlap. Thus, the creation of ontologies by different developers leads to ontologies that represent the knowledge of the same data domains with different vocabularies. This domain representation diversity, known as *semantic heterogeneity*, leads to heterogeneous ontologies, and by extension, heterogeneous knowledge bases. Semantic heterogeneity leads to an interoperability problem that hampers the knowledge exchange between knowledge-based applications and hinders the full adoption of ontologies in real scenarios [104]. To enable knowledge sharing across different applications, an accessible, interoperable and reusable data model is needed [167].

To date, *global* or *shared* ontologies have been developed in different domains to overcome these interoperability issues, i.e., Soupa [19], Mobile ontology [162]. Global ontologies are ontologies that include common vocabularies to provide a common representation and a shared understanding of the domain [164, 21]. Ontologies are built reusing the knowledge from available ontologies to save ontology development time, what is known as *ontology reuse* [11]. Hence, the common knowledge of global ontologies can be reused to develop ontologies for different applications [19, 117]. This common knowledge representation overcomes the terminological differences of existing ontologies in the domain concerned, enabling the knowledge exchange and interoperability between knowledge bases and applications that use them [164, 21].

Ontologies are usually developed for specific applications, without taking into account their reuse in other applications [11]. However, there are cases where the ontology is developed to be reused in different applications to improve the interoperability between knowledge-based applications (what is the case of global ontologies) and/or to facilitate the ontology reuse across different applications [105]. An ontology reused in different applications in a given domain must be *reusable* [83, 112]. Ontology *reusability* was defined by [124] as "*the adaptation capability of an ontology to arbitrary application contexts*". A reusable ontology provides support to different applications in a given domain and must be easily adaptable. To provide reusability, the ontology must include the abstract domain knowledge reused by many applications. However, each application has individual knowledge requirements. If the ontology is too abstract, the effort of extending and customizing the knowledge to satisfy specific knowledge requirements would be high. Thus, ontology developers are less likely to reuse the ontology to develop ontologies for their applications [83, 112].

Considering this, the ontology must be also *usable*. Ontology usability deals with reducing "*the effort required to customize the ontology so that it can be used by humans or machines in a given application context*" [124]. A usable ontology minimises the ontology reuse effort when it is reused to develop ontologies for specific applications. To provide usability, the knowledge of the ontology must be as specific as possible to ease its customization to specific application requirements. Nevertheless, if the ontology represents the knowledge required by a specific application, the effort of adapting the ontology to applications with different knowledge requirements would be high [83, 112].

Bearing in mind that ontology reusability and usability are objectives in conflict, an ontology that supports different applications must achieve a balance of reusability-usability so that it can be reused in different applications with moderate effort [83, 112]. Achieving this balance is particularly important in complex domains, since in these domains an ontology reused by different applications will be a large-scale ontology [154]. The balance of reusability-usability is a key research area in the ontology engineering field and providing a solution to this problem would enable the use of large-scale ontologies in software industries. Hence, ontology designers need support to design reusable and usable ontologies [105].

## 1.2 Motivation and Problem

To date, layered ontologies have been the main ontology design approach to achieve a balance of reusability-usability, i.e., OntoCape [112]. An example of a layered ontology is shown in Figure 1.2. Layered ontologies classify into different abstraction layers the *common domain knowledge* (reused by most applications) and the *variant domain knowledge* (reused by specific application types). We consider an *application type* a family of applications that perform similar tasks or have similar objectives [83]. In addition, the knowledge of each layer is divided into small ontologies known as ontology modules [45]. Such a classification enables ontology developers to reuse only the necessary knowledge at the proper level of abstraction to develop ontologies that satisfy specific application requirements. Hence, the ontology reuse effort in different applications is reduced [111].

Previous works have proposed methodologies to design reusable and usable ontologies that follow a layered structure [146, 154, 112]. A detailed survey and gap analysis of these methodologies is presented in Chapter 3. These methodologies follow different paths to design and develop the ontologies but in all of them the layered structure of the ontology must be designed. The layered ontology structure is an informal model that includes the ontology layers and the knowledge they must include at a conceptual level (as the set of concepts and relations that they must include without going into implementation details) [111].

When it comes to design this structure, previous reusable and usable ontology design methodologies provide guidelines to define the ontology abstraction layers

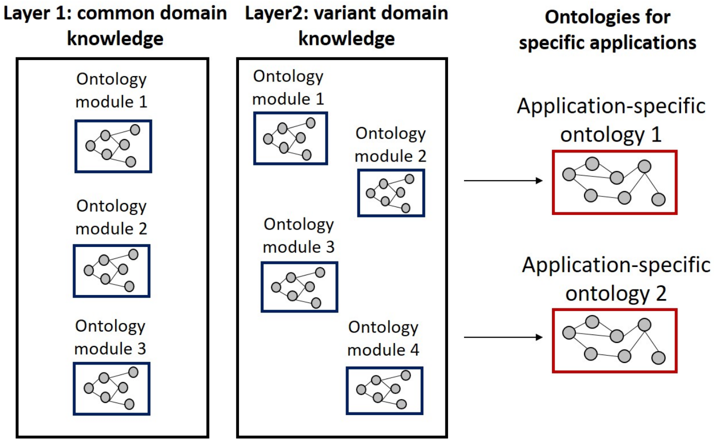

Layer 1: common domain knowledge
Layer2: variant domain knowledge
Ontologies for specific applications
Ontology module 1
Ontology module 2
Ontology module 3
Ontology module 1
Ontology module 2
Ontology module 3
Ontology module 4
Application-specific ontology 1
Application-specific ontology 2

**Figure 1.2:** Layered ontology example

and to classify the domain knowledge into different layers. In these methodologies, the classification of the domain knowledge is performed from scratch based on domain experts' and ontology engineers' expertise. They analyse in collaboration with stakeholders the theoretical framework and the knowledge requirements of the application types that will be supported by the layered ontology. Based on the gained expertise and the identified knowledge requirements, the ontology knowledge is classified into common and variant (and, by extension, into different layers). Hence, a significant effort is required to classify the ontology knowledge from scratch by applying existing reusable and usable ontology design methodologies. This effort hinders the development of reusable and usable ontologies that represent complex domains and support different applications.

In the software engineering field, the main approach to develop reusable and usable software are Software Product Lines (SPLs): software families that contain common reusable parts and variable parts that depend on specific customer needs to support mass customization [127]. For that purpose, software features for a set of applications are analysed and classified into *common features* (common to most applications) and *variant features* (only implemented by specific applications) [127, 4]. The software features of SPLs can be reused to develop new software minimizing the effort of adapting the reused software to specific requirements.

SPLs are quite similar in concept to layered ontologies. Both approaches deal with classifying software features (knowledge in the case of ontologies) into common and variant to facilitate the reuse of software and ontologies. However, unlike the design of layered ontology structures, the design of SPLs rarely starts from scratch [127]. The classification of software features is usually performed systematically taking as reference the software feature similarities and differences of existing applications and legacy systems [127, 86]. This approach makes the SPL design process easier and complements domain experts and software engineers expertise, thus minimizing their involvement and effort [127, 4].

After several decades of building Semantic Web applications in different domains, many developed ontologies are available [160]. Ontologies are usually developed to be reused and support certain application types. In domains with already developed ontologies, the analysis of existing applications applied to design SPLs can be replicated in the ontology engineering field to design the layered structure of reusable and usable ontologies. In particular, the similarities and differences of the knowledge represented by existing ontologies can be analysed to classify the domain knowledge into common and variant. This analysis would complement domain experts and ontology engineers' expertise and prevent them from classifying the domain knowledge from scratch.

As far as we know, previous reusable and usable ontology design methodologies do not take advantage of existing ontologies to save effort when designing the layered ontology structure (as SPL design approaches do). This effort reduction would ease the development of reusable and usable ontologies in complex domains.

The following subsection introduces a motivational scenario to highlight the importance of taking as reference existing ontologies to save effort when designing the layered ontology structure of reusable and usable ontologies.

### 1.2.1 Motivational Scenario: Semantic Representation of Energy Knowledge

One of the emerging smart environments where Semantic Web technologies have been applied is the Smart Grid. The Smart Grid aims to improve the energy management in terms of efficiency, sustainability and resilience through the integration of ICT-based technologies [54]. From the beginning of the current decade, energy ontologies have been developed to represent the knowledge from different energy domains, which is used as a knowledge base by specific energy management applications [92, 44, 41]. Semantically represented energy knowledge improves the performance of intelligent agents used for knowledge extraction and decision-making within knowledge-based energy management applications [39, 27, 56].

The representation of a significant number of energy domains is repeated across the existing energy ontologies, although they are reused by different applications [32]. This repetition evidences a convergence towards a common energy data representation model. However, the energy ontologies were developed by different engineers and they support applications with different energy management objectives. Thus, they represent the same energy domains with different level of detail and different vocabularies, leading to heterogeneous energy ontologies and knowledge bases.

The knowledge-based energy management applications are limited to pilot demonstrators deployed in specific scenarios, i.e., smart homes or buildings [32]. The energy management in real scenarios may require the knowledge exchange among applications that operate in different scenarios to perform an energy management at a greater scale. This knowledge exchange would be hampered by the heterogeneity of existing energy ontologies, which hinders their full adoption in real scenarios. A global ontology that provides a common knowledge representation of energy domains would enable interoperability between energy management applications that operate in different scenarios [32].

A global energy ontology should provide support to energy management applications that operate in different scenarios. Since the energy domains are complex, a global ontology will be a large-scale ontology and it should minimise the ontology customization effort when it is reused by specific energy management applications. Hence, the global energy ontology should provide a balance of reusability-usability.

Considering the benefits of the layered ontology design approach introduced in the previous section, a global energy ontology that follows a layered structure would reduce the ontology reuse effort in different applications. However, the energy domains are complex and a global energy ontology should provide support to a wide variety of energy management applications that operate in different environments such as smart homes, buildings or organizations. Hence, the application of existing reusable and usable ontology design methodologies to design the layered ontology structure would be a great effort for domain experts and ontology engineers. They would analyse requirements of each energy management application supported by the ontology in collaboration with application stakeholders to classify the ontology domain knowledge from scratch. This effort hinders the development of a global energy ontology that provides a balance of reusability-usability.

Considering these issues and the fact that there are already developed energy ontologies, these ontologies could be taken as reference to classify the domain knowledge to reduce the effort of designing the layered ontology structure.

Chapter 5 provides more detail about the motivational scenario described in this section, and discusses the reasons why the aforementioned guidelines are necessary to enable the development of a global energy ontology.

### 1.2.2 Challenges

The motivational scenario introduced in the previous subsection evidences the need to create a reusable and usable global ontology for the energy domain. Bearing in mind the complexity of the energy domains, the layered structure of the global energy ontology should be designed by taking as reference existing ontologies. Since previous reusable and usable ontology design methodologies do not take as reference existing ontologies to design the layered ontology structure, methodological guidelines that enable to apply this approach are needed. This challenge must be addressed to ease the development of the reusable and usable ontology not only for the energy domain, but for any other complex domain.

In summary, the main challenge evidenced in this section is the following:

**Challenge 1.** Define a methodology to design layered ontology structures of reusable and usable ontologies from an existing set of ontologies.

## 1.3 Thesis Objectives

The main objective of this thesis is *to define new methodological guidelines to design layered ontology structures that enable to classify the domain knowledge taking as reference existing ontologies, and to apply these guidelines to enable the development of reusable and usable ontologies in complex domains*.

To achieve this objective and to propose solutions to address the challenge identified in Section 1.2.2, a set of specific objectives have been defined.

The first objective of the thesis is the following:

**O1.** *Define a methodology to design layered ontology structures that enables the classification of the domain knowledge taking as reference existing ontologies*. To meet this objective, the main requirements that guide the construction of the proposed methodology are the following:

**R1.** The methodology is applied to define layered ontology structures. Hence, it should adopt the main activities and ontology design techniques applied by previous reusable and usable ontology design methodologies [112, 154, 146].

**R2.** As stated in Section 1.2, SPL design techniques enable to classify software features taking as reference the similarities and differences of existing applications. Therefore, the methodology should apply well-known SPL design techniques [127, 110] to classify systematically the domain knowledge into common and variant and into different abstraction layers taking as reference existing ontologies. In this way, the methodology would avoid the classification of the domain knowledge from scratch.

Considering the motivational scenario introduced in Section 1.2.2, the energy domain is a potential case study to validate the created methodology. In addition, the application of the methodology in this domain enables to address the challenge of creating a global energy ontology that provides a balance of reusability-usability. Therefore, the second and third objectives of the thesis are the following:

**O2.** *Validate the methodology by designing the layered structure of a global energy ontology.*

**O3.** *Develop a global energy ontology that provides a common energy domain representation and a balance of reusability-usability.* To meet this objective, the main requirements that guide the construction of the ontology are the following:

**R.1.** The ontology should unify the knowledge of the existing energy ontologies to provide a common knowledge representation of energy domains.

**R.2.** The ontology should classify the common and variant domain knowledge into different abstraction layers.

## 1.4 Research Hypotheses

The next hypotheses try to formulate the Research Questions (RQs) that, after giving a response and validation, will make accomplish the aforementioned objectives. The hypothesis H1 is related with objectives O1 and O2, while the hypothesis H2 is related with objective O3.

**H1.** Ontology and SPL design techniques applied in combination enable to classify the domain knowledge into different abstraction layers with high level of consensus taking as reference existing ontologies.

**H2.** A layered ontology that provides a common energy domain representation reduces the ontology reuse time and cost drivers in comparison to a global ontology without this structure.

## 1.5 Research Methodology

The methodology followed in the tesis consists of the following steps:

1. Survey of the state of the art to get a view of the landscape of the relevant research areas, and understand the achievements and limitations in the contributing areas of the thesis. The state of the art encompasses the methodologies applied to design reusable and usable ontologies, techniques applied to save design effort of reusable and usable software and ontologies developed for energy management applications.

2. Formulation of the hypothesis of the thesis and definition of the goals to be achieved.

3. Definition of a methodology to design layered ontology structures that enables to classify the domain knowledge taking as reference existing ontologies. This step includes the following tasks:

   a. Analysis and selection of the steps and ontology design principles applied by previous reusable and usable ontology design methodologies.

   b. Selection and adaptation to the ontology engineering field the SPL design techniques to be applied in the methodology.

   c. Definition of the methodology steps: detailed process, involved actors and inputs/outputs of each step.

4. Application and evaluation of the proposed methodology in the energy case study. This step includes the following tasks:

   a. Literature review of existing energy ontologies and the energy management applications they support. Selection of the relevant and available energy ontologies and classify them into application types.

   b. Application of the proposed methodology to design the layered structure of a global energy ontology by applying SPL and ontology design techniques in combination.

   c. Empirical evaluation of the proposed methodology in the energy case study to validate the methodology and the hypothesis H1.

5. Development of a reusable and usable global energy ontology. This step includes the following tasks:

   a. Unification of the knowledge of existing heterogeneous energy ontologies into a global energy ontology that provides a common representation of energy domains.

   b. Classification of the domain knowledge into different abstraction layers and ontology modules according to the layered ontology structure designed in Step 4.

6. Empirical evaluation of the global energy ontology in different energy management applications to validate the ontology and the hypothesis H2. This step includes the following tasks:

   a. Reuse of the global energy ontology in two different applications.

   b. Comparison of the ontology reuse effort with the effort of reusing the global energy ontology created in Step 5a.

   c. Analysis of the ontology reuse results.

## 1.6 Research Contributions

The main contribution of this thesis is the Methodology for Ontology Design based in Domain Analysis and Layered Structure (MODDALS). MODDALS addresses the challenge of defining a methodology to design the layered structure of reusable and usable ontologies that enables to classify the domain knowledge by taking as reference existing ontologies (see Section 1.2). Hence, MODDALS presents advances in the state of the art of the design of reusable and usable ontologies.

MODDALS was validated by applying it to design the layered structure of a reusable and usable global ontology for the energy domain: the Domain Analysis-Based Global Energy Ontology (DABGEO). DABGEO addresses the challenge of creating a global energy ontology that provides a balance of reusability-usability (see Section 1.2.1). By means of the application of MODDALS in the energy domain and the development of DABGEO, the thesis has also contributed to the advancement in the state of the art of the semantic representation of energy knowledge.

Below we summarize the main contributions of this thesis, the advances they present in the state of the art and the associated publications:

1. *The MODDALS methodology* (available at `https://innoweb.mondragon.edu/ontologies/MODDALS/index-en.html`). MODDALS guides domain experts and ontology engineers to design the layered structure of reusable and usable ontologies. The output of this process is an informal

model of the layered ontology with the ontology layers and the knowledge they include. To define the ontology structure, MODDALS applies the main activities and design principles from previous reusable and usable ontology design methodologies. In contrast to these methodologies, MODDALS applies SPL engineering techniques to systematically classify the domain knowledge into (1) common and variant and (2) different abstraction layers taking as reference already implemented ontologies. MODDALS was evaluated to determine whether it enables to classify the domain knowledge by taking as reference existing ontologies. Domain experts and ontology engineers applied MODDALS to design part of the layered structure of the DABGEO ontology. They were able to follow MODDALS steps to obtain similar knowledge classifications taking as reference existing ontologies. In particular, the degree of consensus when classifying the domain knowledge into different layers was 76% (we understand as degree of consensus the percentage of engineers who classified the knowledge into the same layer, as explained later in Chapter 6). This contribution was published in the *Applied Ontology Journal* [34] and at the *30th International Conference on Software Engineering and Knowledge Engineering* [31].

2. *The DABGEO ontology* (available at `http://www.purl.org/dabgeo`). DABGEO provides a common energy domain representation following a layered structure. In contrast with previous global energy ontologies, it classifies the energy domain knowledge into different abstraction layers according to its relevance for different applications. Therefore, DABGEO provides a balance between reusability and usability, while enabling interoperability between energy management applications that operate in different scenarios. To evaluate DABGEO in terms of balance of reusability and usability, it was reused by different ontology engineers to develop ontologies for two energy management applications and its reuse effort was compared with the effort of reusing OEMA. DABGEO could be adapted to each application with a moderate customization effort and reduced the average ontology reuse time by 0.4 and 1.1 person-hours in each application respectively. This contribution was published in the *Journal of Web Semantics* [33].

In addition to the aforementioned solutions, additional contribution of the thesis are:

3. *The Unified Ontology for Energy Management Applications (OEMA) ontology network* (available at `http://www.purl.org/oema`). OEMA provides a

common representation of energy domains to enable interoperability between different energy management applications. OEMA is the first version of the DABGEO ontology, since it is the result of the first iteration of DABGEO ontology development process. It constitutes a starting point for a widely accepted reusable and usable global energy ontology. This contribution was published at the *2nd International Workshop on Ontology Modularity, Contextuality, and Evolution* (co-located with *International Semantic Web Conference*) [30].

4. A comprehensive literature review of energy ontologies and knowledge-based energy management solutions. This literature review analyses the domains represented (and how they are represented) by existing energy ontologies and how they are used by energy management applications. In addition, the existing energy ontologies are classified according to the energy management application types they support. This contribution was published in a chapter of the book titled *Designing Cognitive Cities* (Springer) [32].

### 1.6.1 Publications

The works covered in this dissertation have been published or have been sent for review in different peer-reviewed international journals and conferences. We now list the scientific publications directly related to the work in this thesis:

**Journal papers:**

- Javier Cuenca, Félix Larrinaga, and Edward Curry. Moddals methodology for designing layered ontology structures. Applied Ontology, (Preprint):1–33, January 2020. DOI: 10.3233/AO-200225. Ranking Scopus: Q1.
- Javier Cuenca, Felix Larrinaga, and Edward Curry. Dabgeo: A reusable and usable global energy ontology for the energy domain. Journal of Web Semantics, In Press, February 2020. DOI: https://doi.org/10.1016/j.websem.2020.100550. Ranking Scopus: Q1.

**Book chapters:**

- Javier Cuenca, Felix Larrinaga, Luka Eciolaza, and Edward Curry. Towards cognitive cities in the energy domain. In Portmann E., Tabacchi M., Seising R., Habenstein A. (eds) Designing Cognitive Cities. Studies in Systems, Decision and Control, volume 176, pages 155–183. Springer, Cham, 2019. DOI: https://doi.org/10.1007/978-3-030-00317-3_7.

**Conference papers:**

- Javier Cuenca, Felix Larrinaga, and Edward Curry. A unified semantic ontology for energy management applications. In Bozzato L. et al. (eds) Joint Proceedings of the Web Stream Processing workshop (WSP 2017) and the 2nd International Workshop on Ontology Modularity, Contextuality, and Evolution (WOMoCoE 2017) co-located with 16th International Semantic Web Conference (Ranking GGS[1]: A+), volume 1936, pages 86–97. CEUR-WS, 2017.

- Javier Cuenca, Felix Larrinaga, and Edward Curry. Experiences on applying SPL engineering techniques to design a (re) usable ontology in the energy domain. In Proceedings of the International Conference on Software Engineering and Knowledge Engineering, SEKE, 2019, volume 2019-July, pages 606–611. KSI Research Inc. and Knowledge Systems I., 2019. DOI: 10.18293/SEKE2019-111. Ranking GGS: B.

## 1.7 Thesis Outline

The thesis is structured into three parts. The first part corresponds to the foundation and context of the thesis. This part includes the following chapters:

- *Chapter* 1 introduces the context and motivation of the thesis, the research objectives and hypotheses, the followed research methodology and the achieved contributions and publications.

- *Chapter* 2 provides the main background as well as terminology used during the rest of the document.

- *Chapter* 3 introduces the state of the art in the design of reusable and usable ontologies, and explains the main gaps of existing solutions and the challenges to be addressed. It also motivates the need to define a new methodology to design layered ontology structures to address these challenges.

The second part corresponds to the definition of the MODDALS methodology. This part includes the following chapter:

- *Chapter* 4 presents the MODDALS methodology and describes its main steps.

[1] http://gii-grin-scie-rating.scie.es/ratingSearch.jsf

The third part corresponds to the application and evaluation of MODDALS in the energy domain to design the layered structure of DABGEO ontology, as well as the development and evaluation of DABGEO. This part includes the following chapters:

- *Chapter* 5 introduces the state of the art in the semantic representation of energy knowledge, and explains the main gaps of existing solutions and the challenges to be addressed. It also motivates the need to apply MODDALS in the energy domain to address these challenges.
- In *Chapter* 6, the MODDALS methodology is applied to design the layered structure of DABGEO ontology. The application of MODDALS to design DABGEO layered structure is taken as reference to evaluate MODDALS.
- *Chapter* 7 presents the DABGEO ontology and explains its development process. In addition, DABGEO is evaluated by reusing it in two energy management applications.

The last part corresponds to the final remarks. This part includes the following chapter:

- *Chapter* 8 summarizes the main conclusions of the thesis and suggests future research directions.

Chapter 2

# Background

## 2.1 Introduction

This chapter introduces the background related to the thesis required to get familiarize with the rest of the document. In particular, the chapter introduces the main background on interoperability (Section 2.2), Semantic Web technologies (Section 2.3), semantic ontologies (Section 2.4), ontology engineering field (Section 2.5) and semantic heterogeneity (Section 2.6). Finally, Section 2.7 summarizes the main conclusions of the chapter.

## 2.2 Interoperability of Data Systems

In the context of data systems, interoperability is defined by 3rd Generation Partnership Project (3GPP) as *"the ability of two or more systems or components to exchange data and use information"* [158]. Computing systems are distributed and have a dynamic nature. In addition, these systems rely on heterogeneous technologies and use different information representations. These factors hamper the data exchange and data processing between different agents. Therefore, to achieve a full interoperability, the exchanged information not only must have a common syntactic base, but also a common structure and common semantics [122]. Considering this, there are four types/levels of interoperability [158]:

- *Technical interoperability*: enables machine-to machine communications through communication protocols and the hardware/software infrastructure required for those protocols to operate.

- *Syntactical interoperability*: provides a common syntax and encoding to the exchanged data, through data representation languages such as eXtensible Markup Language (XML) or HyperText Markup Language (HTML). While

XML is indeed a metalanguage widely used for data representation that is capable to emphasize usability and interoperability, HTML is only a representation format developed with a different purpose (implementing hypertext).

- *Semantic interoperability*: guarantees that there is a common meaning and understanding of the exchanged data.

- *Organisational interoperability*: enables the data exchanged between organizations that rely on different infrastructures and heterogeneous information systems. This level of interoperability requires a successful technical, syntactical and semantic interoperability.

## 2.3 Semantic Web

As stated in Chapter 1, the Semantic Web provides technologies to represent data as knowledge graphs and to exchange knowledge [94].

The Semantic Web (also known as Web 3.0, the Web of Data or the Linked Data Web) supposes a major evolution of the traditional World Wide Web (WWW) towards a Web annotated with explicit data. Semantic Web was defined in 2001 by Berners-Lee et al. [7] as *"an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation"*. Semantic Web adds metadata to the information available on the Web, creating vocabularies that describe additional information such as the content, meaning (semantics) and data relationships. Unlike the traditional WWW, which publishes data on the Web *"as raw dumps in formats such as CSV or XML, or marked up as HTML tables"* [8], the Semantic Web focuses on interlinking and describing the meaning of Web data.

The Semantic Web enables a better performance to intelligent agents and data analysis applications used for knowledge extraction and decision-making:

1. It enables the creation of vocabularies that provide machine-readable data representation.

2. The Semantic Web vocabularies enable establishing relations and linking data from heterogeneous domains, thus creating a knowledge network that eliminates data silos.

3. The Semantic Web also provides standard mechanisms to access, exchange and process machine-readable data represented by systems that rely on heterogeneous technologies. Therefore, semantic interoperability is enabled

and there is a common understanding of the exchanged data among different systems [158].

4. Thanks to the semantic descriptions, machines are capable of inferring knowledge from explicit facts.

### 2.3.1 Semantic Web Technologies and Standards

Semantic web encompasses a set of technologies and standards proposed by the World Wide Web Consortium (W3C) used to describe, link, exchange and process data on the Web in a standard and machine-readable way. These standards conform the Semantic Web technology stack (see Figure 2.1).

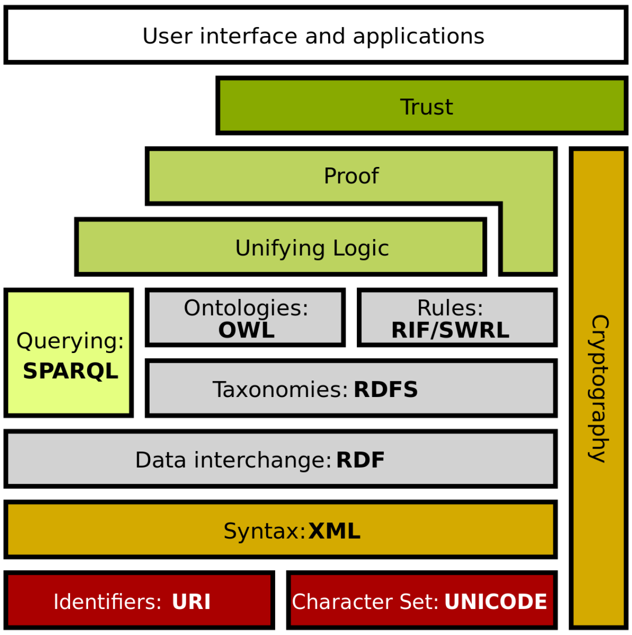


**Figure 2.1:** Semantic Web technology stack

- **Uniform Resource Identifiers (URIs)** (strings of characters used to unambiguously identify a particular resource) are used to identify uniquely web resources and the **Hypertext Transfer Protocol (HTTP)** protocol as data retrieval mechanism.

- The standard **Resource Description Framework (RDF)** language [91] is used to specify the data model of the Semantic Web. RDF represents data as a set of relations called statements or triples. Each RDF statement has a subject, a predicate and an object. The subject corresponds to an entity or "thing" such as a person or a place. The predicate is a property used to describe an entity, i.e., a person's name. Objects can be either entities that are subjects of other statements or RDF Literals that describe concrete data values such as string, integer or float values. An RDF statement has the following form:

Subject -> Predicate -> Object

A set of RDF triples constitutes an RDF graph (an example of an RDF graph is shown in Figure 2.2). The entities or properties of the graph (except literals) are identified by URIs. For instance, one of the best-known predicates defined by RDF is *rdf:type* [75], which is used to group entities together into more general entities.

There are different serializations to write RDF statements and graphs. The following are the best known:

- **RDF/XML**[1]: it uses the XML syntax to serialize RDF.
- **Turtle**[2]: it is a human-readable serialization of RDF.
- **N-Triples**[3]:it is a line-based RDF serialization easy to parse.

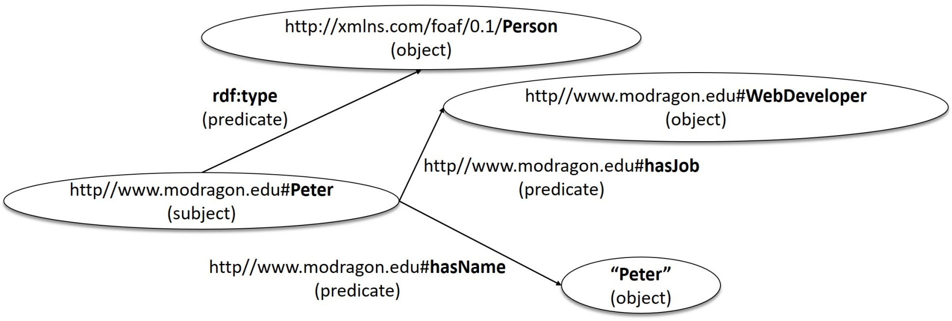

http://xmlns.com/foaf/0.1/Person
(object)
rdf:type
(predicate)
http//www.modragon.edu#WebDeveloper
(object)
http//www.modragon.edu#hasJob
(predicate)
http//www.modragon.edu#Peter
(subject)
http//www.modragon.edu#hasName
(predicate)
"Peter"
(object)

**Figure 2.2:** RDF graph example

- The **RDF Vocabulary Definition Language (RDFS)**[4] is an extension of the RDF. RDFS enables the definition of entity and property taxonomies, such as entity or property hierarchies or entity/property domains and ranges. RDFS provides additional vocabularies to describe and relate the data represented in RDF format, so that machines cannot interpret only the explicitly represented

[1] https://www.w3.org/TR/rdf-syntax-grammar/
[2] https://www.w3.org/TR/turtle/
[3] https://www.w3.org/TR/n-triples/
[4] https://www.w3.org/TR/rdf-schema/

data, but also to infer implicit information. Figure 2.3 shows an example of a basic inference based on the *rdfs:subClassOf* predicate provided by the RDFS language (we omit the full URIs to simplify the diagram). The *rdfs:subClassOf predicate* states that all instances that belong to an entity are instances of another. Considering the taxonomy shown in Figure 2.3, an intelligent agent can infer that all instances of the *Person* entity are instances of the *Living Being* entity. Therefore, the intelligent agent can also infer that *Peter*, apart from being a person, is a mammal and living being.

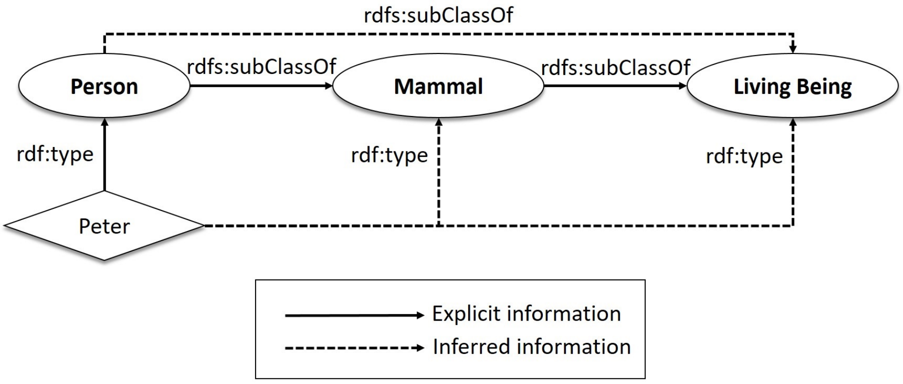


**Figure 2.3:** RDFS inference example

- The **Web Ontology Language (OWL)** [107] is used to represent complex knowledge about things for applications that need to process the content of information instead of just presenting it to humans. OWL enables establishing more complex relations and semantics than RDF and RDFS language to offer a higher expressiveness when representing web data. This expressiveness allows intelligent agents to infer new factors about semantically represented web data and improves the data query performance. In summary, OWL provides the basis for creating vocabularies used to describe web data with a high expressiveness. With this knowledge representation, intelligent agents can perform advanced data analysis and reasoning for knowledge extraction and decision-making. The current version of OWL is OWL-2[5], and it includes three profiles: OWL2 EL, QL or RL. These profiles add restrictions to the represented knowledge to provide different computational and/or implementation benefits.

- The **RDF Query Language (SPARQL)** [131] is the standard query language of the Semantic Web and specifically designed to query data represented in

[5] https://www.w3.org/TR/owl2-overview/

RDF, RDFS and OWL language across various systems. SPARQL can be used to express queries across data stored natively as RDF or in other data sources (i.e., XML documents, databases), since these data can be converted to RDF via middleware.

### 2.3.2 Linked Data

The set of best practises when applying the Semantic Web technologies is known as Linked Data [8]. Before explaining the Linked Data concept it is worth mentioning that Semantic Web and Linked Data are two different concepts (the latter deriving from the former).

Technically, Linked Data refers to data published on the Web in such a way that it is machine-readable, its meaning is explicitly defined, it is linked to other external datasets, and can in turn be linked to from external datasets. Bizer et al. [8] defined the four principles for publishing Linked Data:

- Use URIs as names for things.
- Use HTTP URIs so that people can look up those names.
- When someone looks up a URI, provide useful information using the standards.
- Include links to other URIs, so that they can discover more things.

The application of Semantic Web standards along with the aforementioned principles enables a quick and easy linking of data stored and represented with heterogeneous technologies, which makes the data exchange and reuse more efficient. However, Linked Data is not a replacement of current data sources (i.e., databases, data warehouses), but a complementary technology layer that enables data sharing and exchange among heterogeneous systems (see Figure 2.4). As shown in Figure 2.4, there are different mechanisms to publish data stored in different systems and formats.

### 2.3.3 Semantic Frameworks and Components

Semantic web frameworks are collections of tools and libraries used to manage semantically represented data [75]. The common functions that semantic frameworks provide are storing, querying and making inferences about RDF data. Semantic web frameworks are composed of three basic components:

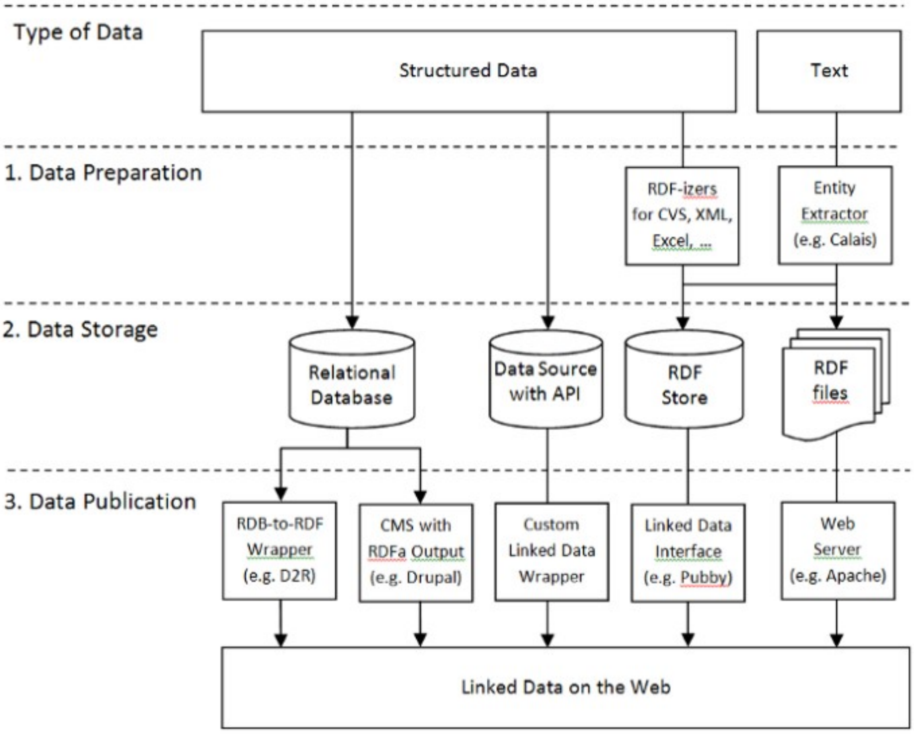

Type of Data
Structured Data
Text
1. Data Preparation
RDF-izers for CVS, XML, Excel, ...
Entity Extractor (e.g. Calais)
2. Data Storage
Relational Database
Data Source with API
RDF Store
RDF files
3. Data Publication
RDB-to-RDF Wrapper (e.g. D2R)
CMS with RDFa Output (e.g. Drupal)
Custom Linked Data Wrapper
Linked Data Interface (e.g. Pubby)
Web Server (e.g. Apache)
Linked Data on the Web

**Figure 2.4:** Linked Data Publishing Options and Workflows [74]

- *RDF repositories*: used to store web data in RDF language. All these statements together form a knowledge base. These semantic repositories are similar to database management systems.
- *Access components*: APIs (written in Java, Python, etc.) used to query, add or modify data of a RDF repository.
- *Inference components*: reasoning engines that apply interpretation of OWL semantics to the semantically represented data.

The most common semantic frameworks are Jena[6] and RDF4J[7] (formerly SESAME), which are written in Java [75]. Jena is an open source Semantic Web framework that supports RDF data storage, retrieval and analysis. RDF4J is another open source Semantic Web framework. It includes its own RDF repository, which can be deployed in an application server (i.e. Apache Tomcat) to allow RDF data access over http.

[6] https://jena.apache.org/tutorials/rdf_api.html

[7] http://rdf4j.org/

## 2.4 Semantic Ontologies

Ontologies are formal vocabularies usually written in OWL language and stored as documents on the Web. They describe and represent a data domain as a set of concepts and complex relationships between them [69]. For this reason, ontologies have been the foundation to represent data in the form of knowledge graphs [125].

Ontologies enable to create a generic knowledge that can be queried, processed and shared across different software applications [69]. Considering this, the knowledge represented with ontology vocabularies is used as a knowledge base by intelligent agents within smart environments.

The main ontology elements are classes, properties and instances (also known as individuals).

- *Ontology class*: classes represent a set of entities that have common features or are similar. Each class corresponds to a concept of the domain represented by the ontology. Within a domain, there can be abstract/top-level concepts and specific /low-level concepts that extend the knowledge of the abstract ones. When a concept c extends the concept c′ we can say that c′ is a subconcept of c. Similarly, in an ontology there are specific classes that are subclasses of more abstract classes, what is known as a *class hierarchy*.

- *Ontology instance*: an ontology instance or individual is an entity that belongs to a certain class.

- *Ontology property*: is a resource that is equivalent to predicates in statements (explained in Section 2.3.1). Properties are used to relate and describe both classes and individuals. Properties can either be datatype properties or object properties. Datatype properties relate classes and individuals with a literal value (i.e., string, number) that expresses a feature of that class/individual, i.e., person name, person age. Object properties relate classes and individuals with other classes and individuals.

Figure 2.5 shows a representation of one ontology that covers all concepts explained above. In this example, *Andrew* and *Peter* are individuals of the *Human* class, which in turn is a subclass of *Mammal*. Thus, we can consider the *Mammal* class as a top-level class and the *Human* class as a low-level class. The *knows* object property is used to specify that *Andrew* knows *Peter*. The *name* datatype property is used to relate persons with their names. Apart from explained elements, ontologies include restrictions on the relations between classes to express facts about classes that are always true (i.e., the maximum number of humans that Peter knows). These restrictions are called axioms and enable to express complex relations between concepts [75].

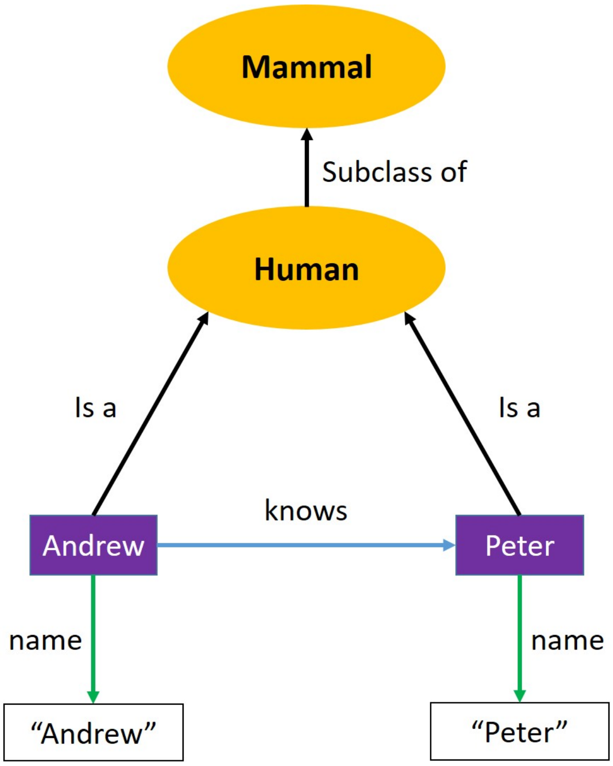


**Figure 2.5:** Ontology example

### 2.4.1 Modular Ontologies and Ontology Networks

An ontology can be developed as a whole or as a set of interconnected ontologies, what is known as *ontology network* [150] or *modular ontology*. Within ontology networks and modular ontologies, the knowledge is divided into ontology modules: small ontologies that contain the necessary elements (classes, properties, instances and axioms) for describing the knowledge of a particular topic of one of the domains concerned. The knowledge of the ontology modules can be related through properties. In addition, some modules may extend or require the knowledge of other modules. In these cases, the former modules include the knowledge of the latter. If a module A includes a module A', the knowledge specified in A' can be directly used (i.e., extend or refined) in module A. Since ontology modules contain less elements and relations than whole ontologies, the ontology complexity is reduced, thus enabling an easy ontology reuse and maintenance [45, 171].

Figure 2.6 shows an example of an ontology network. In this example there are three individual ontologies O1, O2 and O3. Each ontology represents one domain and includes domains specific concepts. O1 represents knowledge about people (i.e., age, gender), O2 represent knowledge about places (i.e., country, cities or districts) and O3 represents knowledge about buildings, i.e., home types or floors. In the example ontology, these ontology modules are interconnected through top-level relations that relate the knowledge they represent. In the example, three possible relations are specified: a person that lives in a house, a house that is built in a city and a person that was born in a city.

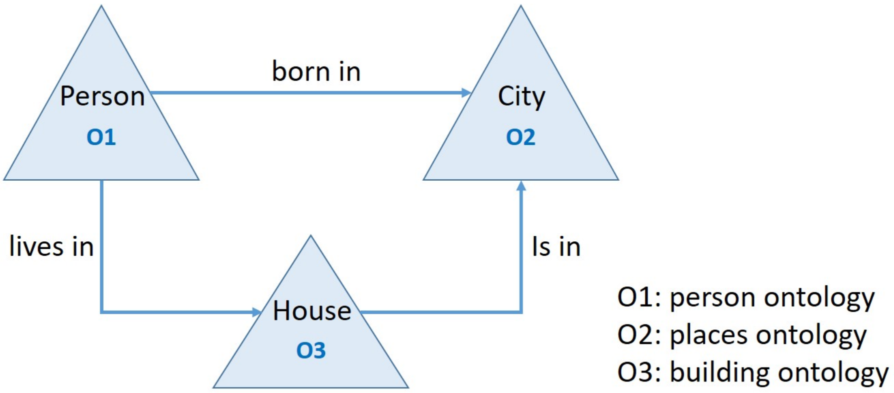


**Figure 2.6:** Ontology network example

## 2.5 Ontology Engineering

As defined by Zouaq and Nkambou [176] ontology engineering *"is a field that explores the methods and tools for handling the ontology lifecycle"*. One of the key research areas of the ontology engineering field is the definition and refinement of ontology development methodologies. The well-known ontology development methodologies are the following: METHONTOLOGY [58], On-to-knowledge [153], DILIGENT [126], NeOn [150] and SABiO [46]. These methodologies guide engineers to develop ontologies and handle each phase of the ontology life cycle.

The ontology development team usually involves domain experts, ontology engineers and stakeholders of knowledge-based applications [150]. Below we explain the main ontology development phases defined by the aforementioned methodologies (Fig. 2.7).

1. *Ontology requirement definition (also known as ontology initiation)*: the ontology main purpose, scope and requirements are defined at this stage. This phase

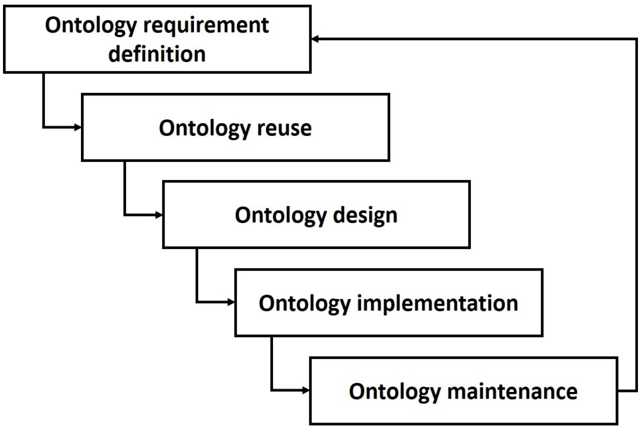


**Figure 2.7:** Ontology implementation phases

is usually conducted by domain experts and ontology engineers, which collaborate to establish the ontology requirements [150]. These requirements can be functional and non-functional. Functional requirements correspond to the knowledge the ontology must represent. This knowledge is defined taking into account the applications that will access the knowledge represented by the ontology. A well-known method to define ontology functional requirements are Competency Questions (CQs) [70]. The CQs are the queries that the ontologies should be able to answer when applications access to the knowledge they represent (i.e., *Where is a device placed?*, *What is the energy consumption of a device in a certain period of time?*). The CQs guide the rest of the ontology development process, since the ontology is designed and implemented to include the necessary elements (classes, properties and axioms) to answer these CQs. Ontology non-functional requirements are the ones not related to the ontology contents [150]. Some examples of non-functional requirements are the language in which the ontology must be written or the notation that the ontology elements must follow.

2. *Ontology reuse*: ontologies are usually built reusing the knowledge from available ontologies (or other non-ontological resources such as dictionaries, lexicons or existing standards) to save ontology development time [11]. In this phase, the existing ontologies are searched, analysed and compared to determine whether some of their elements can be reused to develop a new ontology. After analysing the existing ontologies, some of them and their elements are selected to be included in the developed ontology. Then, the selected ontologies or ontology parts can be included in the developed

ontology during its implementation. Ontologies can be reused as a whole or only specific parts can be reused to develop new ontologies.

Sometimes the knowledge of the reused ontology is reused as it is and it does not require any change. However, the knowledge requirements of the developed ontology are usually different from the ones of the reused ontologies. Therefore, the knowledge of the reused ontologies is customized/adapted and re-implemented to satisfy the knowledge requirements of the developed ontology [150]. The activities performed to adapt the reused knowledge to satisfy specific application requirements are known as ontology reengineering activities [150]. For instance, the reused ontology may contain more knowledge than the required by the developed ontology. If all this knowledge is included in the developed ontology, the applications that query the knowledge represented with the ontology vocabularies would process many concepts and relations and their computation performance would be hampered. Thus, the ontology elements from the reused ontologies not required by the application are removed/discarded [83], what is known as *ontology pruning* [150].

3. *Ontology design*: an informal and formal model of the ontology is defined. The informal model specifies the knowledge that the ontology must include in a conceptual level. That is, the concepts and relations that the ontology must include without going into implementation details. The formal model specifies how the knowledge must be represented with an ontology modelling language [111].

4. *Ontology implementation*: the ontology is implemented in an ontology language (i.e., OWL) following the defined design, so that a semantic application can use the knowledge it represents. The implementation of the ontology is performed through an ontology editor, which provides user-friendly interfaces to simplify ontology development process. *Protégé*[8] and *NeOn toolkit*[9] are well-known ontology editors. During this phase, the reused ontology knowledge is implemented and added to the ontology. In addition, new knowledge is implemented to complement the ontology knowledge to meet the developed ontology requirements. When an ontology editor is used, the ontology design and implementation phases are performed together [151].

[8] https://protege.stanford.edu/

[9] http://neon-toolkit.org/

5. *Ontology maintenance*: new versions of the ontology are carried out as new requirements are defined, i.e., to support more applications.

These phases can be conducted sequentially within a *waterfall life cycle* or the whole process can be repeated in different iterations within an *iterative-incremental life cycle* [150]. The sequential approach is adopted when the ontology development project is short or the requirements are closed. In the iterative approach, the high-level ontology requirements are specified and the ontology development process is divided into a set of iterations. In each iteration an ontology that addresses a set of the defined high-level requirements is developed. The output of each iteration is an implemented version of the developed ontology [152]. This approach is suitable when the ontology represents complex domains, the requirements are not clear at the beginning of the development process or the ontology development involves large groups of developers [150].

## 2.6 Semantic Heterogeneity

Ontologies are usually developed to support specific applications and must satisfy specific knowledge requirements. However, ontologies are developed by different engineers, who represent the knowledge of the same data domains with different vocabularies and level of detail [104, 142]. This knowledge representation diversity is called as semantic heterogeneity and hampers the interoperability between the knowledge-based applications and, by extension, the full adoption of ontologies in complex and real scenarios where data exchange between knowledge-based applications is a key requirement [104]. In addition, the reuse of heterogeneous ontologies leads to the creation of new heterogeneous ontologies [83, 117].

The semantic heterogeneity involves the use of different classes and properties to represent the same concepts and relationships, or the use of different class hierarchies and granularity to represent the same concepts, among other ontology mismatches [47, 119].

### 2.6.1 Global Ontologies

The main solution to overcome semantic heterogeneity is to create an ontology that provides a common representation of the knowledge represented by heterogeneous ontologies [104, 15]. These ontologies are known as *global* or *shared* ontologies, which include common vocabularies to provide a common knowledge representation and a shared understanding of a domain [164, 21]. Thus, the common knowledge of global ontologies can be reused to develop ontologies that fit specific application requirements [21, 83]. Although local ontologies of each application may not represent the same knowledge, the knowledge they have in common is represented with the same vocabularies. With this common knowledge representation, the applications can interoperate to exchange the knowledge represented with common semantic vocabularies within real and complex scenarios (Fig. 2.8).

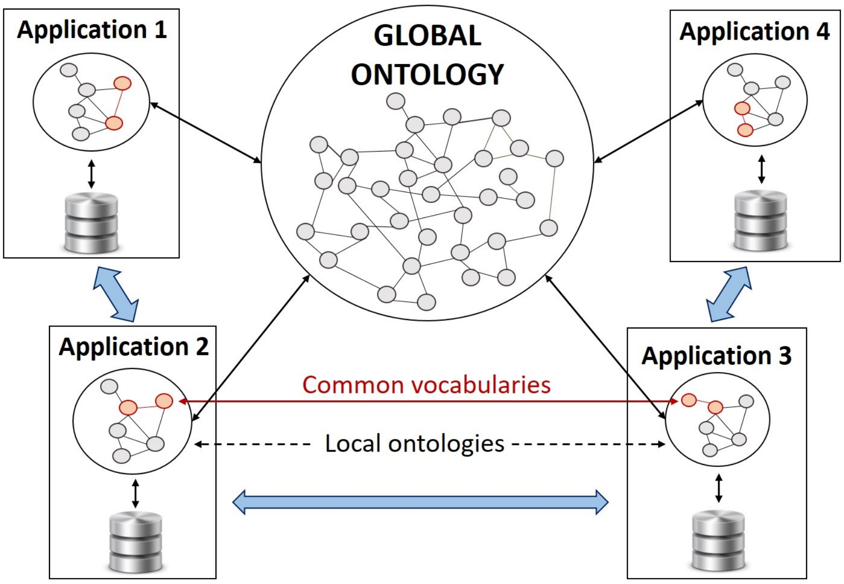


**Figure 2.8:** Knowledge exchange with a global ontology

To date, global ontologies have been developed in different domains to provide a common knowledge representation and to enable interoperability between knowledge-based applications. Some examples of global ontologies are the following: SOUPA ontology [19] (developed for the pervasive computing domain), OntoCape ontology [112] (developed for the chemical engineering domain), VSAO ontology [42] (developed for the anatomy domain), MUTO ontology [100] (developed for the web tagging domain), Mobile ontology [162] (for the mobile domain), IEEE Standard Ontology for Robotics and Automation [139] (developed for the robotics and automation domain) and a shared ontology that represents data of the building information model [117] (developed for the building information domain).

The key activities to create ontologies that provide a common knowledge representation to overcome the semantic heterogeneity are ontology matching and ontology merging [142, 104].

### 2.6.2 Ontology Matching

The ontology matching activity consists of identifying knowledge correspondences among the elements from different ontologies [142]. The ontology correspondences are of different kinds. For example, they can determine (1) that the elements from different ontologies (i.e., classes, properties) are equivalent (=), (2) that the elements of one ontology are more general ($\supseteq$) or less general ($\subseteq$) respecting to the element of another ontologies, or (3) that the elements from different ontologies are disjoint ($\perp$).

Figure 2.9 shows an example of different correspondence types between two ontologies (O1 and O2) that represents the knowledge about building devices. The equivalence correspondences show that the ontologies use different classes to represent the same domain concepts. For instance, they use the *Device* and *Equipment* classes to represent any device used in buildings. The more/less-general correspondences show that the ontologies include classes that can be superclasses or subclasses of the other ontology. For instance, O1 includes classes to represent water heating systems (*WaterHeatingSystem*) and electrical heaters (*ElectricalHeater*). These systems are types of heating systems, which are represented by the *HeatingSystem* class from O2. Finally, the disjoint correspondence shows that the *hasState* and *productionState* properties do not describe the same features about home devices. In this example, the *hasState* property refers to the operation state of the device (i.e., on/off state), while the *productionState* property refers to the production stages of the device manufacturing process (i.e., design, assembly).

The ontology matching activity enables to identify the overlapping knowledge among a set of ontologies. The overlapping knowledge is the knowledge that different ontologies have in common [119, 118]. By analysing the overlapping knowledge of a set of ontologies, ontology engineers can deduce whether ontologies represent heterogeneously the same domains. For example, in Figure 2.9 both ontologies O1 and O2 represent the knowledge about building devices, home appliances and heating system, which is the overlapping knowledge that these ontologies represent. The ontologies use different classes to represent *building device* and *home appliance* concepts. In addition, they use a different class hierarchy to represent the knowledge about heating systems. O1 provides a more granular class hierarchy, since it includes specific types of heating systems. Therefore, we can deduce that the ontologies represent heterogeneously the knowledge from the domain of building devices.

The correspondences between heterogeneous ontologies can be taken as reference to perform a common representation of the overlapping knowledge. Therefore, the ontology matching activity can be seen as the first step to overcome semantic

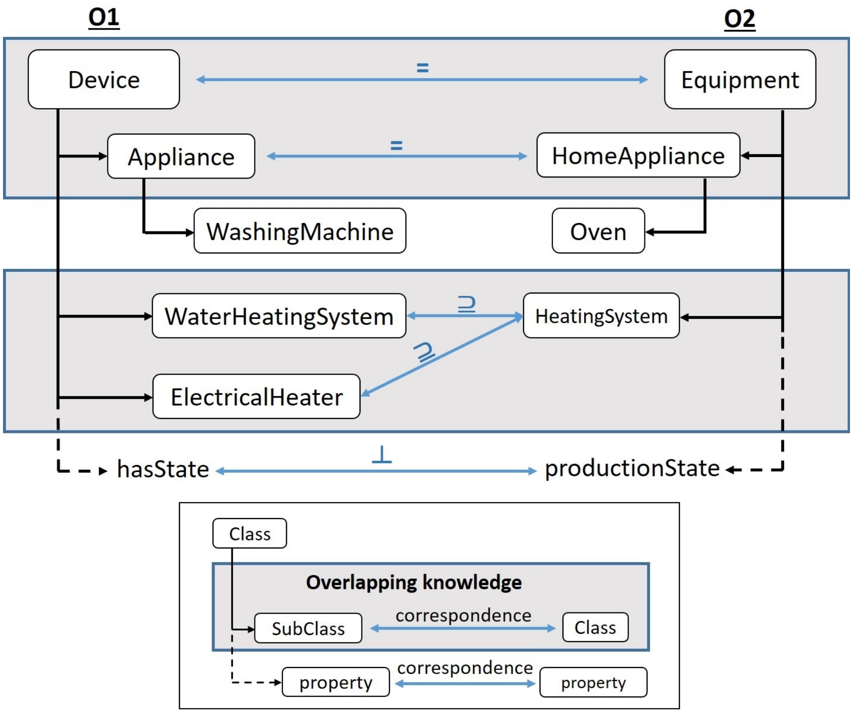


**Figure 2.9:** Ontology correspondence types

heterogeneity [142, 52].

To the best of our knowledge, most of ontology development methodologies do not consider the ontology matching activity in the ontology development process. Among the main ontology development methodologies (introduced in Section 2.5), NeOn includes ontology matching in its ontology development path. However, it does not provide guidelines to perform this activity. Instead, NeOn recommends to follow the guidelines defined by Euzenat and Le Duc [51] to perform the ontology matching activity. According to these guidelines, the ontology matching activity can be performed manually or semi-automatically by using ontology matching tools [142, 121]. Ontology matching tools identify correspondences between ontologies and usually rate each correspondence identified with the probability that the correspondence is correct or not. They apply different strategies to identify the correspondences [104]: computation of the distances between the strings of ontology concepts [16, 22, 23], similarity analysis of the ontology structure [119, 3, 35], the two previous strategies in combination [35, 99, 159, 29] or the use of external knowledge sources as background knowledge [23, 64, 85, 156]. In addition, these tools perform (semi) automatic actions to align or merge the ontologies based on the identified correspondences.

However, the ontology matching tools do not identify all the semantic correspondence between ontologies, and not all the correspondences they identified are correct [104]. Thus, the use of these tools require manual effort to analyse the identified correspondences and to decide how to integrate the ontologies. The development of new algorithms that provide a complete and precise ontology correspondence identification based on semantic similarities are key challenges to improve the performance of current ontology matching tools and to automatize the ontology matching activity [142, 121]. Currently, new automatic ontology matching tools that provide a precise ontology correspondence identification based on semantic-similarity algorithms are being proposed [104]. Nevertheless, to the best of our knowledge, these new tools are not yet widely adopted solutions.

Once the ontology correspondences are identified, the ontology merging activity is conducted [118, 21].

### 2.6.3 Ontology Merging

Ontology merging is the process of generating a single, coherent ontology from a set of heterogeneous ontologies related to the same subject or domain [21]. The merged ontology includes all the knowledge from all the source ontologies. In addition, it provides unique vocabularies and a unique structure to represent overlapping knowledge of source ontologies (taking as reference their knowledge correspondences) [47]. As an example, Figure 2.10 shows the result of merging the ontologies O1 and O2, whose knowledge correspondences where shown in Figure 2.9. The merged ontology includes all the knowledge represented by O1 and O2. In addition, the overlapping knowledge is represented using unique classes and a unique class hierarchy. In this example, the classes from O1 were selected to represent the concepts that both ontologies have in common.

As well as with the ontology matching activity, NeOn includes ontology merging in its ontology development path. The ontology merging activity can be conducted manually or semi-automatically. In particular, ontology matching tools can perform (semi) automatic actions to merge the ontologies based on the identified correspondences [104].

### 2.6.4 Ontology Alignment

An alternative approach to overcome semantic heterogeneity is *ontology alignment*. Ontology alignment consists of creating links between different ontologies according to their knowledge correspondences [21]. There are different techniques for linking

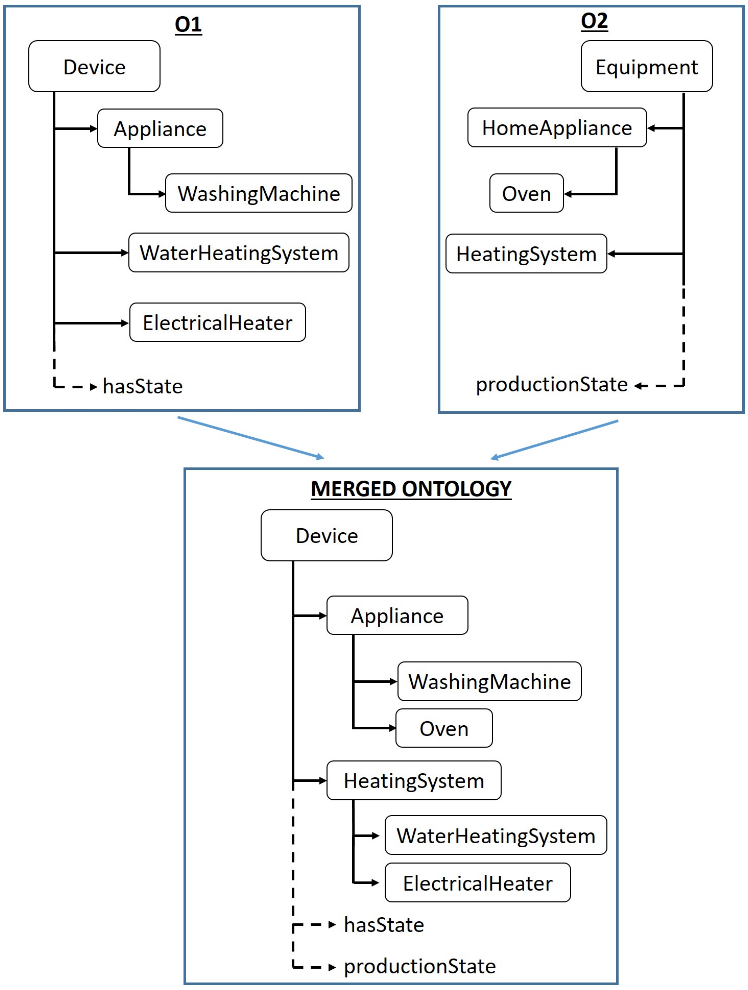


**Figure 2.10:** Complete merge

the knowledge from different ontologies [160]. To explain them we consider two ontologies, O1 and O2:

- **Import:** ontology import means that the elements of O2 are imported in O1 to complement the knowledge the latter represents, or vice versa.
- **Specialization:** some of the classes or properties of O2 are added as subclasses or subproperties of O1, or vice versa.
- **Generalization:** some of the classes or properties of O2 are added as superclasses or super-properties of O1, or vice versa.
- **Equivalence:** some of the classes or properties of O2 are added as equivalent classes or properties of O1 classes or properties, or vice versa.

- **Disjunction:** some of the classes or properties of O2 are added as disjoint classes or properties of O1 classes or properties, or vice versa.

The application of ontology alignment does not result into a global ontology that provides a common knowledge representation, but enables to link heterogeneous ontologies that represent the knowledge they have in common with different vocabularies. With these links, the data exchanged between knowledge-based applications can be translated through query mediators [52, 135] from one ontology representation to another according to the defined ontology links. Thus, legacy applications that rely on heterogeneous ontologies are interoperable. In addition, the knowledge from complementary domains is linked across ontologies. In addition, the knowledge from complementary domains is linked across ontologies, so that applications can link and query the data from different domains [21].

### 2.6.5 Discussion

The current trend to overcome semantic heterogeneity are global ontologies that provide a common domain representation.

To create a common knowledge representation, existing ontologies are merged. This approach is applied when the ontologies represent overlapping domains heterogeneously [21, 118]. However, the merged ontology does not provide interoperability between the legacy applications supported by heterogeneous ontologies.

An alternative approach to overcome semantic heterogeneity is ontology alignment. Ontology alignment provides interoperability between legacy knowledge-based applications that rely on existing heterogeneous ontologies. This approach is usually applied when the integrated ontologies represent complementary domains to enable applications to query cross-domain knowledge [21]. Nevertheless, the lack of common vocabularies and the reuse of the heterogeneous ontologies would lead to the creation of new heterogeneous ontologies developed for new knowledge-based applications. This would hamper the interoperability between new and legacy knowledge-based applications [83]. To achieve interoperability between these applications, new links should be defined between new and existing ontologies, what would be a time consuming process [117].

An intermediate approach to achieve interoperability between new and legacy applications is to merge the heterogeneous ontologies while maintaining the links between these ontologies according to their knowledge correspondences [47]. The common vocabularies from the merged ontology can be reused to develop new and interoperable ontologies. In addition, the links between a merged ontology and heterogeneous ontologies enable (1) to translate knowledge from the merged ontology vocabulary to the vocabulary any of the heterogeneous ontologies and (2) to translate the knowledge between the different heterogeneous ontologies. Hence, this intermediate approach enables interoperability between new knowledge-based applications that use the vocabularies of the merged ontology and legacy applications [28].

Figure 2.11 shows the result of applying this approach to merge ontologies O1 and O2 of Figure 2.9. The merged ontology preserves the unique knowledge representation of the merged ontology shown in Figure 2.10. In addition, the overlapping knowledge is linked with the knowledge from source ontologies.

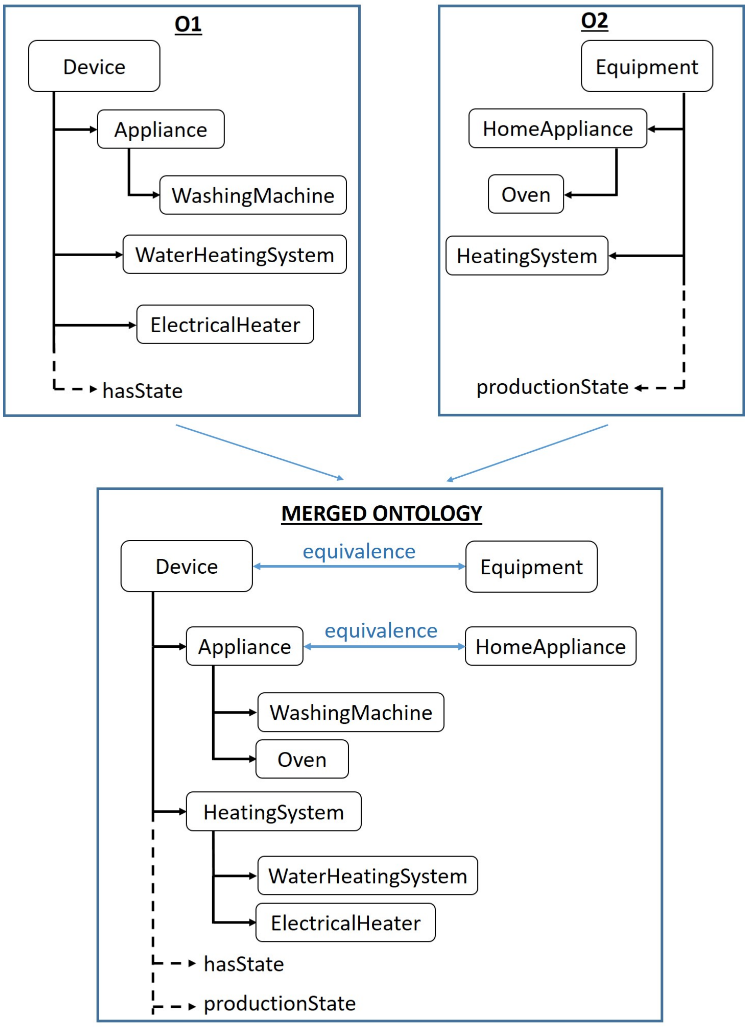


**Figure 2.11:** Intermediate approach between ontology merging and ontology alignment

## 2.7 Conclusions

In this chapter, relevant concepts in the interoperability, Semantic Web, semantic ontologies, ontology engineering and semantic heterogeneity have been introduced as the research background of the thesis. Regarding the interoperability, the main interoperability levels have been introduced. For the Semantic Web, the main standards (RDF, OWL and SPARQL) have been analysed, the Linked Data concept has been explained and semantic frameworks and components have been introduced. Then, semantic ontologies and their main elements have been introduced. Regarding the ontology engineering field, the main ontology development methodologies and the main phases of the ontology life-cycle have been introduced. Finally, the semantic heterogeneity problem has been described, global ontologies have been introduced as the main solution of this problem and the key activities to overcome semantic heterogeneity have been described.

Chapter 3

# Design of Reusable and Usable Ontologies: Literature Review

## 3.1 Introduction

This chapter presents the state of the art in the design of reusable and usable ontologies, the main contributing area of the thesis. Section 3.2 delves into the reusability-usability tradeoff problem. Section 3.3 provides an overview of the ontology design methods applied to improve the ontology reusability and usability. Section 3.4 introduces SPLs as the main reusable and usable software development approach. The section also provides an overview of SPL design methods, highlighting their main benefits. Section 3.5 provides a gap analysis of previous reusable and usable ontology design methods and defines the main challenges to be addressed. Section 3.5 also discusses how these methods can be applied in combination with SPL design techniques to overcome the identified gaps and highlights the need to define a new methodology to design layered ontology structures. Finally, Section 3.6 summarizes the identified gaps and challenges in the design of reusable and usable ontologies, as well as the proposed solution to address them.

## 3.2 Ontology Reusability-Usability Tradeoff Problem

Ontologies are usually developed for specific applications, without taking into account their reuse in other applications [11]. However, there are cases where the ontology is developed to be reused in different applications to improve the interoperability between knowledge-based applications (what is the case of global ontologies introduced in Chapter 2) and/or to facilitate the ontology reuse across different applications [105].

An ontology that supports different applications has to meet two major goals: it must be reusable and usable [83, 112]:

- *Reusability* was defined by the IEEE Standard Glossary of Software Engineering Terminology as "*the degree to which a software module or other work product can be used in more than one computing program or software system*" [25]. In particular, ontology reusability was defined by Pâslaru-Bontaş [124] as "*the adaptation capability of an ontology to arbitrary application contexts*". The goal of reusability is that the ontology can be adapted to a large number of applications.
- On the other hand, *usability* denotes "*the degree to which the software component is useful for a specific task or application*" [112]. By definition, an ontology is never ready for reuse, it must to be customized to be a knowledge base that satisfies the knowledge requirements of a specific application [112]. Thus, ontology usability, in particular, deals with reducing "*the effort required to customize the ontology so that it can be used by humans or machines in a given application context*" [124].

To achieve an acceptable level of reusability, the ontology must include abstract domain knowledge so that it can be reused by different applications. However, apart from sharing information with other applications, each application has individual objectives and requires a specific knowledge representation. If the ontology contains too abstract knowledge, the effort of extending and customizing the knowledge to meet specific application requirements would be high and ontology developers are less likely to reuse the ontology [112]. Hence, the ontology must be also usable to minimise the ontology customization effort when it is reused. Nevertheless, if the ontology represents the knowledge required by a specific application, the effort of reusing the ontology in that application would be reduced, but the effort of reusing the ontology in applications with other requirements would be high [112]. The ontology developers should add to the ontology the knowledge required by the new application. In addition, they should remove or discard the unnecessary knowledge that would hamper and scale down the computational processes of the application [83].

Both ontology reusability and usability are objectives "*in natural conflict*" [112]. An ontology that supports different applications must achieve a balance between reusability and usability so that it can be reused in different applications with moderate effort [83, 112]. This challenge is known as the *ontology reusability-usability tradeoff problem* [90]. Achieving a balance of reusability-usability is particularly important in complex domains. In these domains, an ontology that supports different applications will be a large-scale ontology that covers many concepts. Large-scale ontologies are mostly created to provide a common model of the domain that will be reused to develop ontologies for specific applications [45]. Therefore, the reusability-usability tradeoff problem constitutes a key research problem in the ontology engineering field. Providing a solution to this problem would enable the use of large-scale ontologies in software industries [105].

## 3.3 Reusable and Usable Ontology Design Methods

The following subsections provide an overview of the main methods applied to enable the design and development of reusable and usable ontologies. Firstly, Ontology Design Patterns (ODPs) are introduced as the main methods applied to design reusable ontologies. Then, the ontology classification frameworks and layered ontologies are introduced as the main design approaches to achieve a balance of reusability-usability. Finally, reusable and usable ontology design methodologies that apply these methods are analysed.

### 3.3.1 Ontology Design Patterns

During the last decade, the ontology engineering research community has focused on the application of ODPs to improve ontology reusability [61]. ODPs are modelling solutions to solve recurrent ontology design problems. Among different ODP types, the research community has focused particularly on Content Ontology Design Patterns (CPs) [61, 130, 78], small ontologies that can be used as building blocks in the ontology design to simplify the ontology development process. CPs provide a flexible structure and easy understanding of the reused knowledge to facilitate the ontology reuse by extending or combining CPs [61, 10].

Many CPs have been developed to be used as ontology building blocks in different domains. The most relevant CPs are published in the ODP repository run by the Association for Ontology Design & Patterns[1]. In addition, methodologies for designing ontologies based on the reuse of ODPs have been proposed. This is the case of the eXtreme Design methodology proposed by Blomqvist et al. [10]. The eXtreme methodology is an agile and interactive methodology that guides ontology developers to design and build ontologies through an incremental reuse of ODPs.

[1] http://ontologydesignpatterns.org/wiki/Main_Page

### 3.3.2 Ontology Classification Frameworks and Layered Ontologies

In contrast to the ontology reusability methods introduced in previous subsection, *layered ontologies* are the main ontology design approach applied to achieve a balance of reusability-usability. Layered ontologies are based ontology classification frameworks that classify the knowledge into different ontologies according to their level of generality/specificity [65].

Guarino [71] presented the first ontology classification framework, which distinguishes between the following ontologies:

- *Upper or top-level ontologies*: they represent general and domain independent knowledge and concepts (i.e., object, state) that can be reused in different domains.
- *Domain ontologies*: they extend the knowledge of the upper ontologies, since they represent the knowledge of a particular domain. Some domain ontologies represent only domain top-level knowledge, whereas other domain ontologies include domain-specific knowledge. Thus, some domain ontologies can extend the knowledge of other domain ontologies.
- *Task ontologies*: they extend the knowledge of domain ontologies and represent the knowledge related to generic tasks or activities. These ontologies are reused by applications that perform similar tasks.
- *Application ontologies*: they are the ontologies that include the most specific knowledge, since they represent the knowledge reused by certain applications.

This classification was refined later by the ontology classification presented by Gómez-Pérez et al. [65] in 2004, who introduced *domain-task ontologies*. These ontologies represent the domain knowledge reused by certain application types within a specific domain and they are located between domain and application ontologies. Finally, Scherp et al. [138] introduced the *core ontologies*, which are located between upper ontologies and domain ontologies. Core ontologies represent the domain independent knowledge that can be reused by any application in a specific domain. Figure 3.1 presents the result of these frameworks and extensions, providing a general overview of which are the ontology types depending on the generally/specificity level they represent, what knowledge they extend and their degree of reusability and usability.

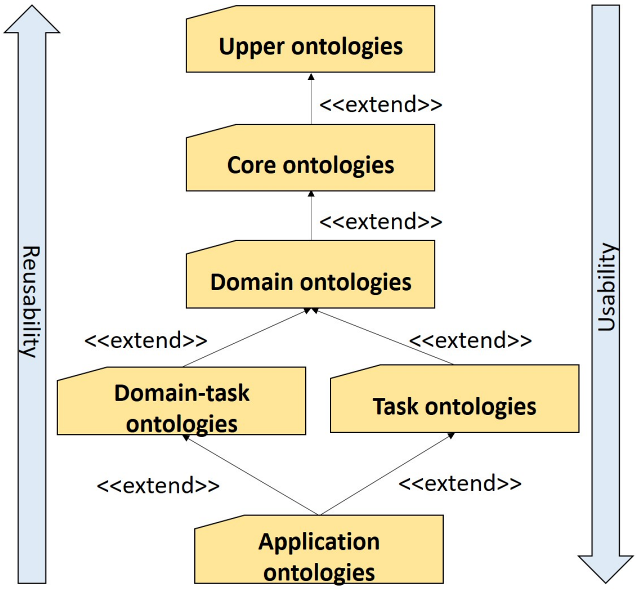


**Figure 3.1:** Ontology classification framework

Layered ontologies include multiple ontologies from the ontology classification framework shown in Figure 3.1. Layered ontologies classify represented domain knowledge in different abstraction layers according to knowledge generality/specificity level. They classify in different abstraction layers the domain knowledge reused by most applications (*common domain knowledge*) and the domain knowledge reused by specific application types (*variant domain knowledge*) [112, 154]. In general, the common domain knowledge corresponds to domain top-level concepts and abstract knowledge, while the variant domain knowledge is more specific. However, there might be specific knowledge reused by many applications that is considered as common and abstract knowledge reused by few applications considered as variant [154]. The knowledge of each layer is classified into ontology modules that contain necessary elements for describing a certain topic of the domains represented.

An example of the structure of a layered ontology is shown in Figure 3.2. Top-level layers include upper and core ontologies to represent general and domain independent knowledge. Low-level layers include domain and domain task-ontologies to represent the common and variant domain knowledge respectively. The layers that represent the common domain knowledge are at a higher level than those that represent variant domain knowledge. In general, the lower the layer is, the more specific concepts and properties it includes and the more axioms it includes. However, as explained in the previous paragraph, some abstract knowledge can be considered as variant while some of the domain-specific knowledge can be con-

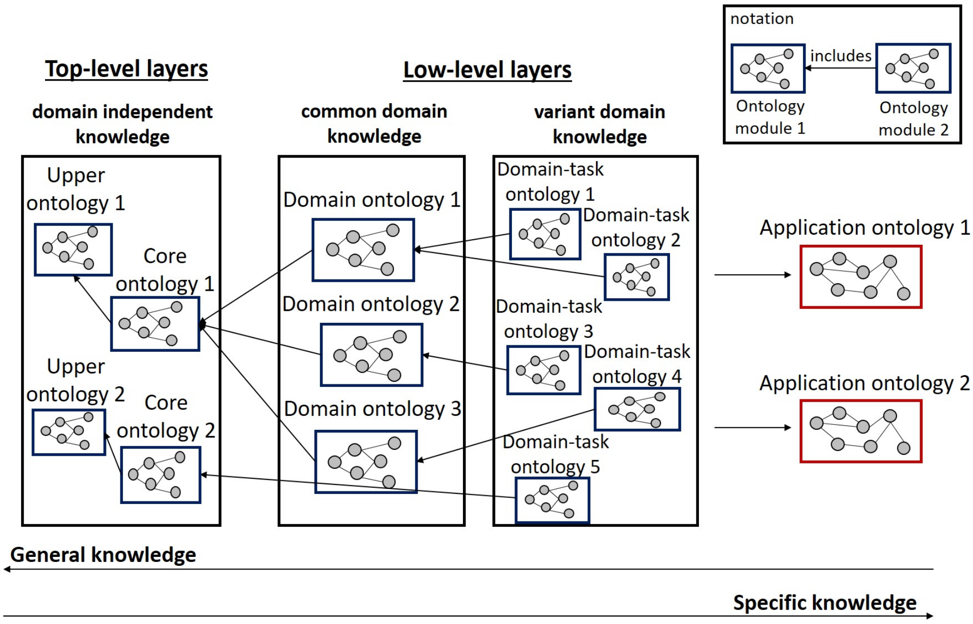


**Figure 3.2:** Example of the structure of a layered ontology

sidered as common. Hence, the layers that represent common domain knowledge may include some domain-specific knowledge and the layers that represent variant domain knowledge may include some abstract domain knowledge [154].

Within this layered structure, some ontology modules extend and include the knowledge of other modules. These modules are reused, adapted and combined by ontology engineers to develop application ontologies that fit application-specific requirements. This ontology structure provides the following benefits [111]:

- *Understandability and adaptability*: the ontology modules represent closely related topics and there are concise. Hence, they are easy to understand. In addition, the modules are independent as they are only relate with the modules whose knowledge they extend or depend on. Therefore, the ontology modules can be reused, combined and adapted to develop application ontologies without affecting other parts of the ontology.

- *Selection of domain knowledge at the proper level of abstraction*: with the layered structure, ontology developers can analyse and select at the proper level of generality and abstraction the necessary knowledge of each domain to develop application ontologies.

- *Maintainability*: layered ontologies provide easy ontology maintenance. On

> the one hand, to provide support to new applications, new knowledge can be added to the ontology as new ontology modules without modifying others. In addition, the ontology developer must only analyse a part of the ontology when adding new knowledge. On the other hand, the general and top-level domain knowledge is kept separate from the knowledge of low-level layers. Hence, when extending the ontology, its structure is likely to remain unchanged.

Considering these benefits, this ontology structure minimises the ontology reuse effort in different applications.

### 3.3.3 Reusable and Usable Ontology Design Methodologies

To date, several reusable and usable ontology design methodologies have been defined. These methodologies deal with designing and developing ontologies that base their structure on the ontology classification frameworks and the layered ontology architectures introduced in Section 3.3.2. The methodologies follow different paths to design and develop the ontologies but in all of them, the layered structure of the ontology must be designed. In this process, the ontology layers and the knowledge they must include is defined. This structure corresponds to an informal model of the ontology that includes the number of layers and the knowledge they include at a conceptual level [111]. Below, we describe these methodologies paying attention to (1) the layered ontology structure they propose and (2) how they classify the domain knowledge into different layers.

#### 3.3.3.1 Spyns et al. [146]

Description

Spyns et al. [146] presented the DOGMA methodology. This methodology is based on the DOGMA approach [83], which defines how to develop ontologies that separate common domain knowledge from the one reused by certain application types.

Proposed Layered Ontology Structure

According to the DOGMA approach, the ontology domain knowledge is classified into two layers: the *ontology base layer* and the *commitment layer*. The former includes the domain knowledge relevant to most application contexts and the later includes the variant knowledge reused by certain kind of applications.

### Domain Knowledge Classification

The DOGMA methodology follows three phases to design ontologies that follow DOGMA approach principles.

1. **Preparation and scoping:** in this phase, the ontology requirements and scope are defined. This phase is carried out conducting sequentially the following tasks: (1) definition of user requirements; (2) definition of purpose; (3) identification of domain experts; (4) compilation of knowledge resources; and (5) scoping of knowledge resources.

2. **Domain conceptualization:** this is the core phase of the DOGMA methodology. In this phase, the common domain knowledge included in the *ontology base layer* is defined as a set of RDF/OWL statements called *lexons*. This phase involves the following tasks:

   - *Knowledge discovery*: ontology knowledge is gathered semi-automatically from existing texts. The obtained terms are selected and grouped through AI algorithms, such as statistical methods, neural networks or natural language processing.
   - *Knowledge elicitation*: domain experts gather the obtained knowledge and spot a domain conceptualization based on their expertise.
   - *Knowledge negotiation*: ontology engineers elaborate a template with a high-level view of the domain. Domain experts refine these templates as considered valid for their respective organizations.
   - *Knowledge breakdown*: the defined knowledge is decomposed into a hierarchical structure. To perform this classification, the methodology recommends verbalizing elementary sentences and lexon engineering, which aims to transform the verbalized facts into language independent formal statements.

3. **Application specification:** in this phase, the knowledge of the ontology base layer is extended and constraints are set to define the knowledge reused by certain application types. This knowledge is defined by analysing the requirements of the applications supported by the ontology. The knowledge reused by certain application types is represented in the commitment layer as a set of explicit statements derived from ontology base lexons and rules applied over these lexons.

#### 3.3.3.2 Thakker et al. [154]

Description

Thakker et al. [154] set out a methodology to design layered ontologies that represent ill-defined and complex domains.

Proposed Layered Ontology Structure

The methodology proposes three layers to classify the ontology knowledge:

- *Upper ontology layer*: includes upper ontologies that represent general and domain independent knowledge.
- *High-level reusable domain layer*: this layer includes the domain knowledge common to all the application contexts/use cases supported by the ontology.
- *Case specific layer*: this layer includes the domain knowledge reused only by certain application types.

Domain Knowledge Classification

The methodology proposed by Thakker et al. [154] does not provide detailed guidelines to classify the domain knowledge. Domain experts and ontology engineers define and classify the domain knowledge into common and variant based on the theoretical framework and the analysis of the applications supported by the ontology. Then, the knowledge is classified into different layers.

#### 3.3.3.3 Morbach et al. [112]

Description

Morbach et al. [112] developed the OntoCape ontology, a highly reusable and usable ontology for the chemical process engineering domain. Morbach et al. [111] detail the OntoCape ontology design and implementation methodology and process. The authors mainly focus on the on the layered ontology structure and the ontology design principles followed to design and develop the ontology.

Proposed Layered Ontology Structure

OntoCape is a large-scale ontology that includes 62 ontology modules. The modules are classified into four abstraction layers:

- *Meta-layer*: includes ontologies that structure of the rest of the ontology and establishes the design principles that must follow lower level layers and the ontology extensions.
- *Upper layer*: includes upper ontologies that represent general and domain independent knowledge shared by the represented domains.

The next two layers classify the domain knowledge:

- *Conceptual layer*: includes the domain knowledge relevant to most application contexts.
- *Application-oriented layer*: includes the domain knowledge not common but still relevant to several application contexts. This layer can be seen as an intermediate layer between the common and variant domain knowledge.

The knowledge of these layers is classified into ontology modules, which were defined taking as reference the following ontology design requirements:

- *Coherence*: the ontology does not contain contradictory knowledge.
- *Conciseness*: the ontology must not contain redundant knowledge and it must represent only the knowledge required by the supported applications.
- *Intelligibility*: the ontology must be easy to understand for the ontology engineers that reuse the ontology knowledge.
- *Adaptability*: the ontology must enable an easy customization to satisfy different application requirements and an easy extension to provide support to new applications.
- *Minimal ontological commitment*: the common domain knowledge of the ontology should contain minimum ontology axiomatizations to support its reuse across different applications.
- *Efficiency*: the ontology knowledge contains only the necessary axiomatizations required by supported applications, so that intelligent agents perform an efficient knowledge reasoning.

To meet these requirements, the ontology modules of each layer were defined according to the main ontology modularization principles defined by Stuckenschmidt and Klein [149]. These principles establish that the ontology modules are loosely coupled and self-contained, that is, they must be linked and depend as little as possible on other modules to facilitate their understanding, reuse and maintenance.

### Domain Knowledge Classification

In the methodology applied to design OntoCape layered ontology structure, domain experts analyse the requirements of the applications supported by the ontology and define from scratch the CQs that the ontology should answer to meet those requirements. Then, ontology engineers define the concepts and properties that the ontology must include to answer the defined CQs. This conceptualization is refined with the feedback of application stakeholders.

The reviewed reusable and usable ontology design methodologies are summarized in Table 3.1. The table includes the layered structure proposed by each methodology and how the knowledge is classified into each layer.

| **Methodology** | **Proposed layered ontology structure** | **Domain knowledge classification** |
|---|---|---|
| Spyns et al. [146] | *Ontology base layer:* common domain knowledge<br>*Commitment layer:* variant domain knowledge | Analysis of supported applications and textual resources by domain experts and ontology engineers |
| Thakker et al. [154] | *Upper ontology layer:* general knowledge<br>*High-level reusable domain layer:* common domain knowledge<br>*Case specific layer:* variant domain knowledge | Analysis of theoretical framework and existing applications by domain experts and ontology engineers |
| Morbach et al. [112] | *Meta layer:* general knowledge<br>*Upper layer:* general knowledge<br>*Conceptual layer:* common domain knowledge<br>*Application-oriented layer:* variant domain knowledge | Analysis of supported application requirements by domain experts with application stakeholders |

**Table 3.1:** Summary of reviewed reusable and usable ontology design methodologies

## 3.4 Reusable and Usable Software Design Methods

In software engineering field, the main approach to design and develop reusable and usable software are SPLs. In the following subsections, we introduce the SPL approach and the main methods to design SPLs.

### 3.4.1 Software Product Line Engineering

SPL engineering has been studied as a recognized discipline for years [80]. SPL engineering was defined by Pohl et al. as "*a paradigm to develop software applications (software-intensive systems and software products) using platforms and mass customisation*". This field studies the development and life-cycle management of SPLs [127].

Clements and Northrop [24] defined SPLs as "*set of software-intensive systems sharing a common, managed set of features that satisfy the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way*". In SPLs, instead of developing one application to satisfy customer specific needs, a software family that contains common reusable parts and variable parts that depend on those needs is developed. The key characteristic of SPLs is that they classify software features into *common features*; common to a set of applications and *variant features*; only implemented by specific applications. With this structure, SPLs enable a systematic software reuse when developing similar software products that have specific needs, thus maximizing the software variability, that is, the flexibility of a software application to be adapted to specific needs [127]. Considering this, SPLs enable the development of large-scale software products to support mass customization.

The software feature classification to design SPLs is performed through a process called *domain analysis* [4] or *domain requirements engineering* [127]. The domain analysis is the first step of the SPL development process [127, 4] and is applied to identify the similarities and differences of software features in a set of applications.

### 3.4.2 Domain Analysis

Below we describe in detail the domain analysis process and the main methods and techniques proposed to perform it.

The domain analysis process encompasses two main activities [127, 4]:

1. *Commonality and Variability analysis (CVA)*: is the process of classifying the software features into common and variant and the core activity of the domain analysis process [127]. The software features relevant to most of applications

are classified as common software features, while the software features relevant only to specific applications are classified as variant software features.

2. *Feature modelling*: is the process of expressing the variability of the SPL in terms of common and variant features according to the CVA results. This activity is conducted to document the software features. This documentation is used as input for conducting the subsequent activities of the SPL development process [4].

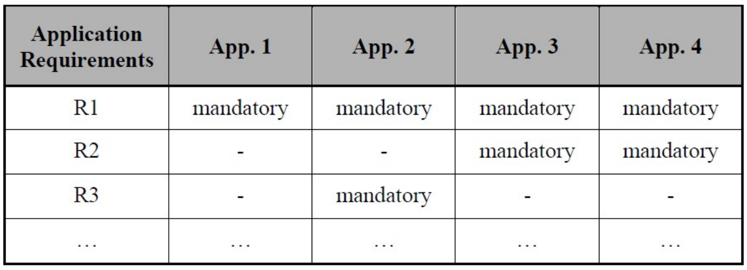
Below we describe the main techniques applied to perform the domain analysis process. We focus on the CVA activity, since it is the core activity of the domain analysis and the one where the knowledge is classified.

Most of the domain analysis methods that guide on how to conduct a domain analysis to classy software features are based on the Feature-Oriented Domain Analysis (FODA) variability model [86, 20]. This model establishes the main principles and the main steps of the SPL domain analysis process. Among the domain analysis methods based on the FODA variability model, we highlight the well-known methods proposed by Pohl et al. [127] and Apel et al. [4]. Apart from introducing the SPL main principles, they explain how to conduct the main activities of the domain analysis process. Each method emphasizes one activity of the domain analysis process.

Pohl et al. [127] enumerate the main activities of the domain analysis process and introduce the main techniques applied to classify the software features through a CVA. According to Pohl et al. [127], the following techniques can be applied to perform a CVA:

- *Application-requirements matrix*: it classifies software features into common and variant depending on how many applications require them. The software requirements mandatory for almost all applications are classified as common software features, while the rest are classified as variant software features. In the sample application-requirements matrix presented in Figure 3.3, the requirement 'R1' is mandatory for all applications. Therefore, it is considered as a candidate to be a common software feature in the SPL. The rest of requirements are only mandatory to specific applications, so they are considered as variant software features.

- *Priority-based variability and checklist-based variability analysis*: these techniques classify software features into common and variant depending on stakeholders' priorities. Each software application requirement is rated with a priority



| Application Requirements | App. 1 | App. 2 | App. 3 | App. 4 |
|---|---|---|---|---|
| R1 | mandatory | mandatory | mandatory | mandatory |
| R2 | - | - | mandatory | mandatory |
| R3 | - | mandatory | - | - |
| … | … | … | … | … |

**Figure 3.3:** Structure of an application requirements matrix for four applications [127]

level by different stakeholders. The application requirements considered as basic by most of stakeholders are considered as common software features, while the rest of requirements are considered as variant software features.

The domain analysis process can take as input different knowledge sources to classify the software features. These knowledge sources include stakeholders, existing products, failure reports and the applications that already exist in the market (they can be either own products or competitors' products) [127].

In SPL design, the domain analysis rarely starts from scratch, since "*product line engineering requires sophisticated domain experience*" [127]. The domain analysis is usually performed systematically taking as reference the existing applications and legacy systems developed in the domain where the SPL is applied [127, 4]. Specifically, the existing applications are analysed to extract their requirements and functionalities [169, 110, 73]. Then the similarities and differences of the software features implemented by existing applications are identified (i.e. through code similarity techniques [169]). Depending on how many applications implement them, the software features are classified into common and variant. In particular the software features implemented by all or most of applications are considered as common, while the rest are considered as variant [127]. It is worth mentioning that the software feature classification is not 100% dependent on existing applications. Domain experts define the common software features that should be common from a strategic point of view (i.e., they are basic requirements for every application supported by the SPL or they will become common in near future), regardless off they are implemented or not by all existing applications [127].

The software features of legacy applications have been defined by domain experts and software engineers after analysing the application requirements [55]. Therefore, the classification of software features based on existing applications provide three main benefits:

1. It does not require a broad knowledge of the domain and the applications

supported by the SPL [127].

2. It avoids the classification of the SPL software features from scratch. Domain experts and software engineers do not need to analyse the requirements of the applications supported by the SPL to define the software features of the SPL and to classify them into common and variant. Hence, the effort of the domain experts and software engineers in tasks such as meeting with application stakeholders or requirement analysis is reduced [4, 55].

3. The confidence in the SPL increases, since there is a confidence in the legacy applications analysed in the domain analysis [55, 86].

Considering these benefits, the SPL design effort is reduced. Hence, existing applications are the main knowledge source applied to conduct the domain analysis when designing SPLs [127, 86].

## 3.5 Gap Analysis and Discussion

Considering the ontology reusability and usability methods analysed in Section 3.3, the research community effort has mainly focused on the ontology reusability aspect through ODPs (introduced in Section 3.3.1). Only few reusable and usable ontology design methods have been proposed. These methods are focused on designing layered ontologies that classify the domain knowledge into different abstraction levels.

All the reviewed reusable and usable ontology design methodologies include guidelines to design the layered ontology structure. In particular, the following activities are conducted: (1) define the ontology abstraction layers and the kind of knowledge they will include (common or variant), (2) define the ontology knowledge, (3) classify the domain knowledge into common and variant and into different layers and (4) structure the knowledge in each layer.

The classification of the domain knowledge into different layers is performed from scratch based on domain experts' and ontology engineers' expertise. They analyse in collaboration with the application stakeholders the theoretical framework, as well as the operational aspects and the knowledge requirements of the application types that will be supported by the ontology. Then, domain experts and ontology engineers collaborate to define the ontology knowledge and classify it common and variant (and, by extension, into different abstraction layers) based on the gained expertise.

Layered ontologies are similar in structure to SPLs. In both approaches, software features are classified into common and variant to support large-scale software reuse. The SPL design process rarely starts from scratch, since the high involvement of domain experts and software engineers involves several risks when classifying the software features during the SPLs design process. On the one hand, experts with a sophisticate domain experience not only in the domain concerned but also on the applications supported by the SPL are required [127]. On the other hand, domain experts and software engineers must (1) analyse the requirements of each application and (2) have interviews with application stakeholders to define and classify the software features of the SPL from scratch. These constraints make it difficult to design SPLs and increase design effort [55].

These issues also arise when designing the layered structure of reusable and usable ontologies. Ontologies are being developed in increasingly complex domains (i.e., energy, chemical engineering) [154]. Hence, a significant effort is required to classify the ontology domain knowledge from scratch by applying existing reusable and usable ontology design methodologies. This effort hinders the development of reusable and usable ontologies in complex domains.

As stated in Section 3.4.2, SPLs are usually designed taking as reference existing applications and legacy systems to classify systematically the software features into common and variant. Specifically, a domain analysis of existing applications is conducted to identify the similarities and differences of their software features. Depending on how many applications implement them , the software features are classified into those common to a set of applications and those only implemented by specific applications. This approach complements the expertise of domain experts and software engineers, thus minimising their involvement and effort when designing the SPL.

As stated in Chapter 1, there are domains where ontologies have been developed to support different application types. In domains with already developed ontologies, the domain analysis of existing applications applied to design SPLs can be replicated in the ontology engineering field to design the layered structure of reusable and usable ontologies. In particular, the similarities and differences of the knowledge represented by existing ontologies can be taken as reference to classify the domain knowledge depending on how many ontologies represent it. Considering the benefits that the analysis of existing applications brings to the SPL design process (see Section 3.4.2), the analysis of existing ontologies would (1) complement the domain experts' and ontology engineers' expertise and (2) prevent them from classifying the domain knowledge from scratch.

As far as we know, previous reusable and usable ontology design methodologies do not take advantage of existing ontologies to save effort when designing the layered ontology structure (as SPL design approaches do). This effort reduction is a key enabler of the development of reusable and usable ontologies that represent complex domains. Therefore, there is the need to define a methodology to design the layered structure of reusable and usable ontologies that enables to classify the domain knowledge by taking as reference existing ontologies. Since the methodology is applied to define layered ontology structures, it should cover the main activities to design layered ontology structures applied by previous reusable and usable methodologies. In addition, the methodology should apply SPL design techniques to classify the common and variant domain knowledge according to a domain analysis of existing ontologies. In this way, the methodology would cover the main gap of previous reusable and usable ontology design methodologies.

## 3.6 Conclusions

In this chapter, we have introduced the balance between reusability and usability as the major goal of ontologies reused in different applications, which is the case of global ontologies. Then, we have highlighted the main limitations and challenges in the design of reusable and usable ontologies. In addition, we have set the bases of the solutions proposed to address the identified challenges.

The main reusable and usable ontology and SPL design methods have been analysed. Layered ontologies are the main ontology design approach applied to achieve a balance between reusability-usability. They classify into different abstraction layers the common and variant domain knowledge. This approach enables ontology developers to select at the proper level of abstraction the required knowledge to develop application ontologies. To date, a set of reusable and usable ontology design methodologies have been proposed to design layered ontology structures. These methodologies guide to define the ontology abstraction layers and to classify the domain knowledge into different layers. The classification of the domain knowledge into different layers is performed from scratch based on domain experts' and ontology engineers' expertise. Therefore, the application of these methodologies requires a high ontology design effort, which hinders the development of reusable and usable ontologies in complex domains.

Therefore, the application of these methodologies requires a high ontology design effort, which hinders the development of reusable and usable ontologies in complex domains. Considering how software features are classified when designing

SPLs, SPL design techniques can facilitate the domain knowledge classification by taking as reference existing ontologies. Hence, **there is the need to define a methodology to design the layered structure of reusable and usable ontologies that enables to classify the domain knowledge by taking as reference existing ontologies.**

# Part II

# Definition of MODDALS Methodology

Chapter 4

# MODDALS: A Methodology for Ontology Design based on Domain Analysis and Layered Structure

## 4.1 Introduction

As stated in Chapter 3, the application of current reusable and usable ontology design methodologies requires a high ontology design effort, since of the domain knowledge into different layers is performed from scratch based on domain experts' and ontology engineers' expertise. Therefore, there is the need to define a methodology to design the layered structure of reusable and usable ontologies that enables to classify the domain knowledge by taking as reference existing ontologies.

This chapter presents the MODDALS methodology. MODDALS is an open methodology and ontology developers can find online[1] detailed guidelines to apply it.

MODDALS guides domain experts and ontology engineers to design the layered structure of reusable and usable ontologies. The output of this process is an informal model with the ontology layers and the knowledge they include at a conceptual level. To define the layered ontology structure, MODDALS applies the main activities and design principles from previous reusable and usable ontology design methodologies [112, 154, 146].

In contrast to these methodologies, MODDALS applies SPL engineering techniques to systematically classify the domain knowledge (1) into common and variant and (2) at different abstraction layers taking as reference existing ontologies. The knowledge of the ontologies developed for specific application types is usually

[1] https://innoweb.mondragon.edu/ontologies/MODDALS/index-en.html

defined through the collaboration between domain experts and application stakeholders, who translate their knowledge into the ontology [150]. In MODDALS, the knowledge from those ontologies is exploited by domain experts and ontology engineers to classify the domain knowledge when designing the layered structure. Therefore, they do not need to analyse the knowledge requirements of different applications to classify the ontology domain knowledge from scratch, facilitating the design of the layered ontology structure.

This chapter addresses the first thesis objective (O1) defined in Chapter 1: *define a methodology to design layered ontology structures that enables the classification of the domain knowledge taking as reference existing ontologies.*

The chapter is organized as follows. In Section 4.2, MODDALS is compared and positioned respecting to previous ontology design methods. Section 4.3 explains the steps in MODDALS. Section 4.4 summarizes the main conclusions of the chapter.

## 4.2 Related Work

This section compares the MODDALS methodology with the reusable and usable ontology design methods reviewed in Chapter 3 to highlight its main contributions. In addition, MODDALS is positioned with respect to well-known ontology development to explain (1) which ontology development methodology should be adopted when applying MODDALS and (2) which phases of the ontology development process are covered by MODDALS. Finally, we summarize the position of MODDALS with respect to previous methodologies and we indicate when it should and should not be applied.

### 4.2.1 Reusable and Usable Ontology Design Methods

As explained in Section 3.3.1, ODPs have been researched as the main solution for designing reusable ontologies [61, 78]. When designing and developing layered ontologies, ODPs are applied to represent the domain independent knowledge of the upper-level layers [111]. In contrast to ODPs, MODDALS is focused on designing the layered ontology structure to represent only the domain knowledge of the ontology. Therefore, it is applicable to design the low-level layers of the layered ontology structure. The knowledge represented by ODPs is more abstract and would be located in upper layers within a layered ontology network. Hence, MODDALS is complementary to ODPs.

On the other hand, in the last decade several reusable and usable design methodologies that involve the design of layered ontology structures have been proposed

[112, 154, 146] (these methods were reviewed in Section 3.3.3). In contrast to these methods, MODDALS provides guidelines to classify the domain knowledge based on a domain analysis of existing ontologies applying SPL engineering techniques.

MODDALS also has common aspects with previous reusable and usable ontology design methodologies. MODDALS applies the main activities and ontology design principles applied by these methodologies. Therefore, the purpose of MODDALS is not to substitute these methodologies to improve the domain knowledge classification. It offers an alternative method to classify the domain knowledge into different layers.

### 4.2.2 Ontology Development Methodologies

As explained in Section 2.5, to date a set of well-known ontology development methodologies have been defined, i.e., METHONTOLOGY [58], On-to-knowledge [153], DILIGENT [126], NeOn [150] and SABiO [46]. With the exception of NeOn and SABiO, all these methodologies guide to develop ontologies from scratch and do not consider the ontology reuse aspect [150]. NeOn defined different paths to reuse ontologies and to the best of our knowledge, is the methodology that provides more detailed guidelines when reusing ontologies. As well as NeOn, SABiO also supports ontology reuse, with the difference that SABiO is thought to develop both domain and operational ontologies (machine-readable implementation version of ontologies, to guarantee desirable computational properties) [46].

NeOn defines a set of flexible scenarios to develop ontologies and ontology networks. These scenarios correspond to the methods (i.e., reuse, reuse and merge) that can be applied to reuse existing ontologies and non-ontological resources. Each scenario focuses in a particular ontology reuse method. NeOn scenarios are listed below, as well as the ontology reuse methods they encompass (for more details about each scenario we refer the reader to [150]):

- Scenario 2: Reusing and reengineering non-ontological resources
- Scenario 3: Reusing ontological resources
- Scenario 4: Reusing and reengineering ontological resources
- Scenario 5: Reusing and merging ontological resources
- Scenario 6: Reusing, merging and reengineering ontological resources
- Scenario 7: Reusing ODPs

- Scenario 8: Restructuring ontological resources
- Scenario 9: Localizing ontological resources

These scenarios applied in combination with the Scenario 1, which includes the key steps (i.e., ontology requirement definition, ontology design, and ontology implementation) of the ontology development process.

Figure 4.1 summarizes the ontology development phases proposed by NeOn methodology based on the aforementioned scenarios. If existing ontologies are reused to develop the ontology, the ontology reuse phase is included. Depending on the selected ontology reuse scenario(s), the ontology reuse phase can be conducted in different ways and it may be followed by the ontology merging and reengineering phases. The ontology development process can be performed sequentially or can be divided into different iterations, in each of which one or more scenarios are applied to meet a set of the ontology requirements [152].

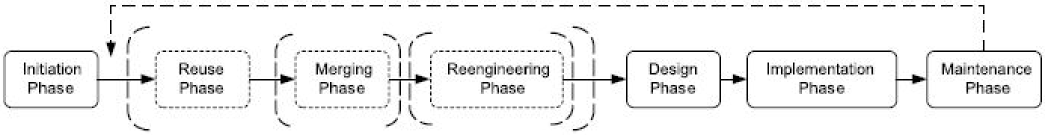


**Figure 4.1:** Ontology network life-cycle models proposed in the NeOn Methodology framework [152].

Since MODDALS takes as reference the knowledge of existing ontologies to design the layered ontology structure, the knowledge of the designed structure will include the knowledge from these ontologies. Existing ontologies are analysed to classify this knowledge into different layers. Hence, within the ontology life-cycle, MODDALS covers part of the *ontology reuse phase*. Since NeOn supports ontology reuse, MODDALS aligns with NeOn and can be applied as a set of additional steps that expand this methodology. In particular, it can be considered as a new scenario for reusing ontologies: organization of the various existing ontologies into an overall layered ontology structure.

In addition, in MODDALS the knowledge that the ontology must represent is defined: the knowledge from existing ontologies. In contrast, in NeOn the knowledge of the ontology is defined from scratch as the functional requirements of the ontology during the *ontology initiation phase*. Therefore, MODDALS also covers part of this phase.

Considering the ontology development phases covered by MODDALS, it should be applied right after the ontology initiation phase and before the ontology reuse phase of NeOn (Fig. 4.2). In addition, MODDALS has influence on these phases, performs part of them in a different way from NeOn. During the ontology initiation phase, the ontology purpose, scope and non-functional requirements should be defined. After this phase, MODDALS should be applied to (1) search for existing ontologies in the domain concerned, (2) define the ontology knowledge and (3) define the layered ontology structure. Then, in the ontology reuse phase, the knowledge from existing ontologies should be reused so that the developed ontology represents the defined knowledge according to the defined layered structure. In particular, MODDALS is compatible with NeOn reuse scenarios 3, 4, 5, 6, 7 and 8 since they are the ones that involve the reuse of existing ontologies and ODPs. The application of one scenario or another depends on (1) the strategy selected to reuse the elements of existing ontologies and (2) the adequacy of the reused ontologies (i.e., reuse effort, understandability, naming conventions) to meet the knowledge requirements of the developed ontology.

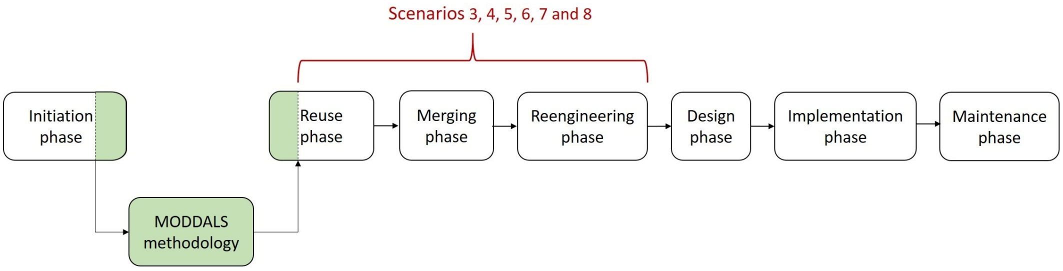


**Figure 4.2:** Application of MODDALS within NeOn methodology phases.

### 4.2.3 Position and Usage of MODDALS

Table 4.1 summarizes the related work analysed in this section as well as the position of MODDALS with respect to the analysed solutions.

Bearing in mind the features of MODDALS and its position with respect to the previous works, it should be applied when the following conditions are met:

1. The developed ontology must provide a balance of reusability-usability, since it is developed to be reused by different applications in a given domain.

2. There are already developed ontologies that support different applications in the domain.

3. The domain where the developed ontology is applied is complex.

| Related work | Position of MODDALS |
|---|---|
| *Reusable and usable ontology design methods*: ODPs [61], methodologies proposed by Thakker et al. [154], Spyns et al. [146] and OntoCAPE ontology [112]. | - Complementary to ODPs.<br>- Guidelines to classify the domain knowledge taking as reference existing ontologies.<br>- Adoption of the activities and ontology design principles applied by previous reusable and usable ontology design methodologies. |
| *Ontology development methodologies*: METHONTOLOGY [58], On-to-knowledge [153], DILIGENT [126], NeOn [150] and SABiO [46] | - It can be applied as a set of additional steps of NeOn and proposes a new ontology reuse scenario.<br>- It has influence on the ontology initiation and reuse phases. |

**Table 4.1:** Summary of the work related to MODDALS

4. The developed ontology represents domain knowledge.

Otherwise, it should not be applied in the following cases:

1. The ontology is developed for a specific application.
2. There are no ontologies developed in the domain concerned.
3. The domain where the developed ontology is applied is not complex.
4. The developed ontology represents domain independent knowledge.

## 4.3 MODDALS Methodology

This section explains the steps in MODDALS, which were defined bearing in mind the requirements defined in Chapter 1:

**R1.** The methodology is applied to define layered ontology structures. Hence, it should adopt the main activities and ontology design techniques applied by previous reusable and usable ontology design methodologies [112, 154, 146].

**R2.** SPL design techniques enable to classify software features taking as reference the similarities and differences of existing applications. Therefore, the methodology should apply well-known SPL design techniques [127, 110] to classify systematically the domain knowledge into common and variant and into different abstraction layers taking as reference existing ontologies. In this way, the methodology would avoid the classification of the domain knowledge from scratch.

Considering these requirements, MODDALS takes as input previous reusable and usable ontology design methodologies and well-known SPL engineering techniques. MODDALS steps have been defined bearing in mind the main activities applied by previous reusable and usable ontology design approaches: (1) definition of the ontology abstraction layers and the kind of knowledge they will include (common or variant), (2) definition of the ontology knowledge, (3) classification of the domain knowledge into common and variant and into different layers and (4) structuring of knowledge in each layer. These activities were adapted bearing in mind that MODDALS classifies the domain knowledge taking as reference existing ontologies and applying SPL engineering techniques.

Based on the aforementioned activities, MODDALS encompasses four main steps. These steps involve the collaboration between domain experts and ontology engineers and are conducted sequentially. In addition, MODDALS takes as reference already developed ontologies to classify the domain knowledge into different abstraction layers. Therefore, before applying the MODDALS steps, a preliminary step is required: analysis and classification of existing ontologies. Once the exiting ontologies have been selected and analysed, the methodology itself is implemented (Fig. 4.3).

### 4.3.1 Preliminary Step: Analysis and Classification of Existing Ontologies

In this step, domain experts classify the existing ontologies into the application types they support.

Firstly, a state of the art of the existing ontologies developed for specific applications is conducted in the domain concerned. The main objectives of the ontologies and the applications they support are analysed. The domain experts group the applications that perform similar tasks into application types. In the case that the applications do not perform similar tasks, each specific application is considered as an application type.

Then, the available ontologies that support analysed applications are selected (assuming that they have been designed and developed in collaboration with domain experts). The ontologies should be as documented as possible, since their knowledge

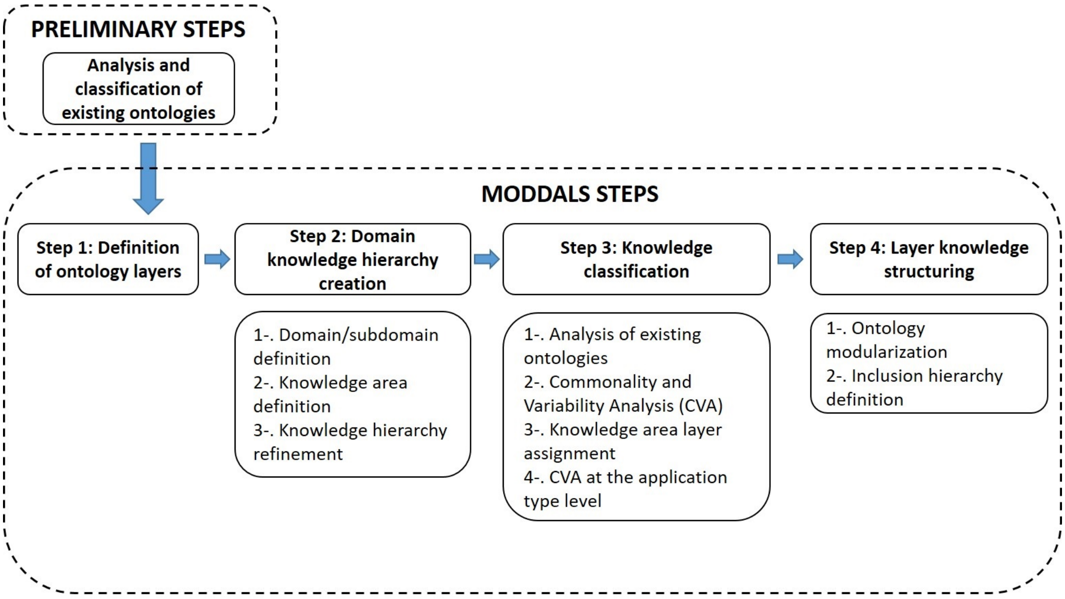

PRELIMINARY STEPS
Analysis and classification of existing ontologies
MODDALS STEPS
Step 1: Definition of ontology layers
Step 2: Domain knowledge hierarchy creation
Step 3: Knowledge classification
Step 4: Layer knowledge structuring
1-. Domain/subdomain definition
2-. Knowledge area definition
3-. Knowledge hierarchy refinement
1-. Analysis of existing ontologies
2-. Commonality and Variability Analysis (CVA)
3-. Knowledge area layer assignment
4-. CVA at the application type level
1-. Ontology modularization
2-. Inclusion hierarchy definition

**Figure 4.3:** MODDALS methodology steps

is the input to classify the knowledge in the designed layered structure. The selected ontologies are classified according to the application type they support.

It is worth mentioning that if there are only a few ontologies already developed in the domain or these ontologies are reused only by a few application types, the domain analysis will not be representative enough to classify the domain knowledge, as well as occurs when designing SPLs [86]. Therefore, MODDALS is not applicable in these cases. To define the minimum sample of ontologies to apply the methodology the FODA model (introduced in Section 3.4.2) is taken as reference, since it establishes the main principles and the main steps of the SPL domain analysis process. According to the FODA model [86], a domain analysis must take as input at least three applications (as divergent in functionality as possible). Therefore, we consider ontologies that provide support to at least three application types must be already developed within the domain where MODDALS is applied as a minimum sample to apply the methodology. If these conditions are not met, one of the reusable and usable ontology design methods introduced in Section 3.3.3 should be applied to design the ontology structure.

To the best of our knowledge in SPL engineering there is not a maximum limit of legacy applications to be included in the domain analysis [127]. Hence, in MODDALS the ontology design team must decide how many application types will be supported by the layered ontology and include in the domain analysis the ontologies already developed to support the selected application types.

The outcome of this step is a classification of existing ontologies according to the application types where they are reused, which is taken as input by the rest of MODDALS steps.

### 4.3.2 Step 1: Definition of Ontology Layers

In the first step, domain experts define the ontology layers that classify the domain knowledge and the kind of knowledge they include.

The layered structure proposed by MODDALS has been defined taking as reference the layers proposed by the previous reusable and usable ontology design methodologies. In addition, the defined layers must be compatible and comply with the knowledge classification method proposed in MODDALS: a domain analysis of existing ontologies by applying SPL engineering techniques.

When it comes to represent the domain knowledge, all the reusable and usable ontology design methodologies reviewed in Section 3.3.3 propose (1) a layer that includes the common domain knowledge reused by all application types covered by the ontology and (2) a layer that includes the variant domain knowledge reused by specific application types. A set of application types in a given domain will have knowledge in common, while each application will require specific knowledge [146]. Hence, the aforementioned layers are mandatory in a layered structure. These layers are compatible with the knowledge classification method applied on MODDALS, since the domain analysis classifies the software features (in this case knowledge) into the ones common to all applications and those that are implemented by specific applications [127].

In SPL design, there is no a middle ground when classifying the software features, since they are usually implemented by most of applications or specific applications [110]. However, in MODDALS we apply the domain analysis to classify knowledge instead of software features. Depending on the knowledge similarities and differences of existing ontologies, there might be knowledge that is not common but still reusable across a set of application types. Therefore, the ontology must include an intermediate layer. For that purpose, the OntoCape ontology [112] adds a layer that contains the domain knowledge not common but still relevant to several application types.

Considering these aspects, we propose in MODDALS a layered-structure that combines the layers proposed by previous approaches and contains three layers (Fig. 4.4). These layers constitute a template where the ontology knowledge is classified in the next steps. Previous reusable and usable ontology design methods do not follow a pre-established standard to name the layers. Hence, we have defined the name of the layers based on the kind of knowledge they include (common knowledge, variant knowledge still common to more than one application type, variant knowledge only reused by specific application types).

- The *common-domain layer* includes domain ontologies that represent the top-level knowledge of each domain. The domain ontologies of this layer also represent the common domain knowledge. The knowledge in this layer is extended by the knowledge in the next two layers, which is more specific.
- The *variant-domain layer* includes domain ontologies that represent the variant domain knowledge still common to more than one application type.
- The *domain-task layer* includes domain-task ontologies that represent the variant domain knowledge reused by specific application types. The ontology modules of this layer are classified according to the application type where they are reused. Thus, the structure of this layer can vary depending on the number of application types supported by the layered ontology. MODDALS classifies the domain knowledge taking as reference existing ontologies. Hence, only the application types supported by existing ontologies are considered to define the ontology structure of this layer. Possible future application types are not taken into account since "*a complete domain theory is lacking in almost any complex (engineering) domain*" [111].

  In some domains, a set of applications that belong to an application type can be grouped into a more specific application type, since they have specific objectives in common and perform highly related tasks [31]. In these cases, the *domain-task layer* is divided into two sublayers. The sublayers separate the knowledge reused only by a specific application type from the knowledge still relevant for more specific application types encompassed by the general application type. For instance, let us consider that the *application type 1* encompasses the *application type 1.1* and the *application type 1.2*. The knowledge reused by both *application type 1.1* and *application type 1.2* could be relevant for any other application type encompassed by the *application type 1*. This knowledge is placed in the *scenario sublayer*. In contrast, the knowledge reused only by the *application type 1.1* is only relevant for that application type. This knowledge should be placed in the *application type sublayer*. The domain experts can also name each sublayer using the terms in the domain concerned to facilitate the distinction between the two sublayers (as done in Chapter 6, where MODDALS is applied in the energy domain).

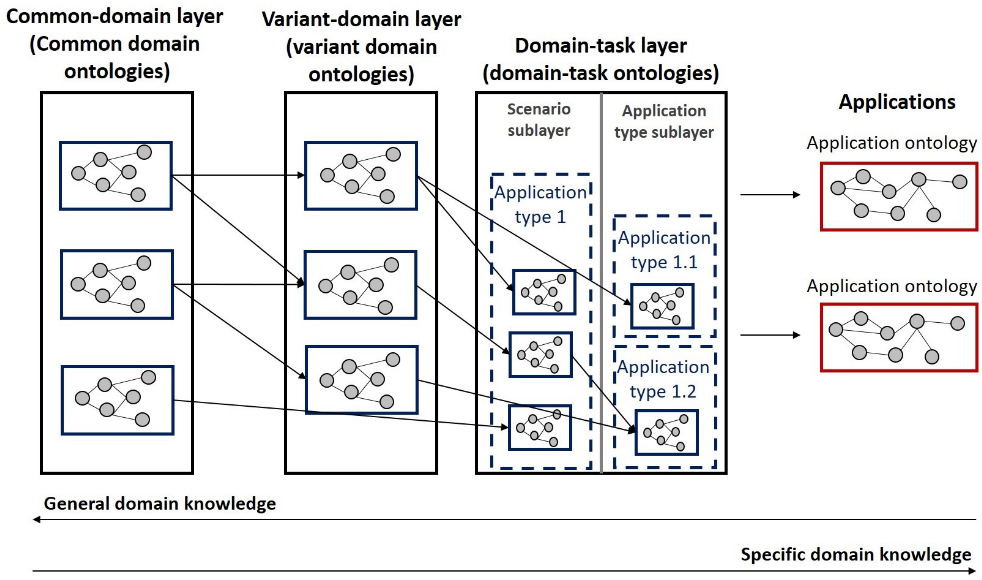

Common-domain layer (Common domain ontologies)
Variant-domain layer (variant domain ontologies)
Domain-task layer (domain-task ontologies)
Scenario sublayer
Application type sublayer
Application type 1
Application type 1.1
Application type 1.2
Applications
Application ontology 1
Application ontology 2
General domain knowledge
Specific domain knowledge

**Figure 4.4:** Ontology structure proposed by MODDALS

The outcome of this step is a high-level structure of the ontology with the layers described above.

### 4.3.3 Step 2: Domain Knowledge Hierarchy Creation

In the second step, both domain experts and ontology engineers collaborate to define the ontology knowledge.

In previous reusable and usable ontology design methodologies, the knowledge of the layered ontology is defined at a conceptual level. In addition, the knowledge is divided into different abstraction levels and knowledge pieces. This knowledge decomposition enables (1) the separation of abstract knowledge that is likely to be reused in most of applications from the specific knowledge and (2) the classification of the defined knowledge pieces into different abstraction levels [146].

In addition, in previous methodologies, the knowledge of the ontology is defined from scratch. However, MODDALS classifies the ontology domain knowledge based on a domain analysis of existing ontologies. Hence, in this methodology the layered ontology must include the knowledge represented by existing ontologies.

Considering these aspects, in this step the knowledge from existing ontologies is abstracted, divided and organized into a knowledge hierarchy that classifies it into different abstraction levels. The knowledge hierarchy proposed by MODDALS includes three main elements (Fig. 4.5):

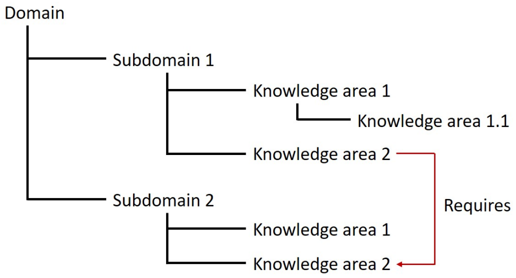


**Figure 4.5:** Domain knowledge hierarchy example

- *Domains:* the domains represented by the ontology are located in the first level of the hierarchy.
- *Subdomains*: extensive domains are divided into subdomains that cover the knowledge of an important part of the domain. Subdomains are located in the second level of the knowledge hierarchy.
- *Knowledge areas (KAs)*: in the third level of the knowledge hierarchy, consider a KA as a potential module of the layered ontology that encompasses the knowledge of a specific topic of a subdomain. The KAs are the knowledge pieces that are classified into different layers. Each KA can be divided into "child" sub-KAs that represent more specific knowledge. Therefore, we can say that a sub-KA extends the knowledge of a specific KA. In addition, some KAs, may represent specific knowledge by combining the knowledge from other KAs. In these cases, the former KAs require the knowledge from the latter. These relations are also reflected in the knowledge hierarchy.

Within this hierarchical structure, the KAs of the upper levels include abstract domain knowledge, while the KAs of low levels include more specific domain knowledge. Hence, the knowledge hierarchy enables to abstract and divide the knowledge from existing ontologies, so that the defined KAs can be classified in the next steps into the layers defined in Step 1.

Before explaining how the knowledge hierarchy is created, it is important to distinguish the knowledge it includes from the knowledge of existing ontologies. The knowledge hierarchy includes the knowledge of existing ontologies at the conceptual level, as a set of concepts and relations. On the contrary, ontologies include this knowledge implemented through classes, properties and axioms used to represent the concepts and relations.

To define the hierarchy, the domain experts and ontology engineers collaborate to perform a manual analysis of the elements of existing ontologies in an ontology editor to identify the domains they represent and to divide them into KAs.

This step includes three activities that are conducted sequentially.

1. **Domain/subdomain definition:** in this activity, domain experts and ontology engineers analyse the knowledge represented by exiting ontologies to identify the domains they represent. The top-level concepts of each domain are also defined by domain experts. If the domains are too extensive, they are divided into subdomains according to domain experts' criteria.

2. **Knowledge area definition:** in this activity, ontology engineers (in collaboration with domain experts) analyse existing ontologies to divide the knowledge of the defined subdomains into KAs.

   Ontology partitioning and module extraction algorithms/tools [45, 134, 68] are well-known methods to extract semi-automatically and divide knowledge from ontologies [45]. However, existing ontologies are developed by different engineers and with different objectives, so they are heterogeneous. Thus, to the best of our knowledge, the application of existing ontology partition and module extraction algorithms/tools in different ontologies would lead to different ontology module classifications. The same knowledge extracted from different ontologies may be included into different modules and linked with different knowledge. These issues would lead to an inconsistent knowledge hierarchy. Hence, a more abstract method to define KAs is required in MODDALS. To avoid these issues, the CQs answered by existing ontologies can be taken as reference to divide the knowledge they represent into KAs.

   To answer each CQ, the ontology must include the necessary ontology elements (classes, properties and axioms) that represent certain concepts and relations. Thus, CQs are a natural guide for splitting ontologies into small knowledge fragments [136]. By identifying the CQs each ontology answers, the concepts and relations needed to answer them can also be identified and considered as a whole to define a KA. Hence, this method enables the abstraction and division of knowledge from different ontologies regardless of their heterogeneous knowledge representation. The CQs defined to develop ontologies are not always available [136]. Therefore, in MODDALS ontology engineers perform a manual analysis of ontology elements to identify the CQs they answered by existing ontologies (it can be considered as a reverse engineering process) and divide the knowledge into KAs. This strategy is also followed in when designing SPL taking as reference exiting applications [73, 13], since "*legacy systems rarely have an accurate functional specification*" [73]. In particular, as explained in Section 3.4.2, the requirements and functionalities are extracted from the existing applications before analysing their similarities and differences.

The *knowledge area definition* activity includes two sub-activities.

**2.1. Class hierarchy-based KA definition:** some ontology class hierarchies are self-descriptive enough to answer a set of CQs. Hence, the class hierarchies of existing ontologies are analysed to identify the first CQs. For instance, a class hierarchy that contains the *Device* class with more specific devices (i.e., *Appliance* or *Sensor* classes) as subclasses can answer the following CQ: *what type of devices are there?* Thus, the KA corresponding to this CQ could be defined. This KA would encompass the device concept. Considering this, the first KAs of the knowledge hierarchy are defined based on some class hierarchies of existing ontologies. These KAs are named as the subject of the CQ they answer. In the previous example, the subject of the CQ was devices, so the KA should be called *devices*.

Each level of the class hierarchies is considered as sub-KA of the previous level. For instance, if the *Appliance* class of the previous example includes subclasses to represent more specific appliances (i.e., white goods) the *appliances KA* should be defined. This KA would encompass the *appliance* concept and would be a sub-KA of the *devices KA*. If the class hierarchy includes many class levels, the last levels (which often include very specific classes [53]) can be considered as a whole to define a KA to avoid an unmanageable number of KA levels in the class hierarchy.

The existing ontologies may represent the same concepts with different class hierarchy structures. Therefore, in these cases a common class hierarchy of these concepts must be defined before defining the KAs. In these cases, the class hierarchy that describes each concept with the highest granularity is selected among existing ontologies and is populated with classes from other ontologies.

**2.2. Ontology elements relation-based KA definition:** the rest of CQs are answered through the relations of a set of ontology elements. Hence, the classes of the existing ontologies and the relations between classes through properties (and the axioms applied on them) are analysed to identify the remaining CQs. All the concepts and relations represented by the ontology elements that answer each CQ can conform a KA.

The CQs that cover similar topics are grouped by domain experts to create new KAs, which encompass all the knowledge required to answer these CQs. Each of these KAs is named by joining the key words of the CQs it encompasses. For example, let us consider that the analysed ontologies contain the *hasName*, *hasModel* and *hasSerialNumber* properties to describe certain features of Devices to answer the following CQs: *What is the name of a device?*, *What is the model of a device?* and *What is the serial number of a device?*. These CQs describe the information of the device related with the manufacturer, so they can be grouped into the *device manufacturer data KA*. This KA encompasses the concepts and relations that answer the aforementioned CQs.

By grouping CQs, some KAs may include unnecessary knowledge for certain applications. However, if we define one KA for each identified CQ, the knowledge hierarchy would contain an unmanageable number of KAs. Thus, the layered ontology would contain an unmanageable number of modules [136]. We must assume that "*an ontology is never ready for use, but must always be adapted and refined to a knowledge base for the envisioned application*" [112]. Therefore, the CQs are grouped according to domain experts' criteria and the desired KA classification granularity.

3. **Knowledge hierarchy refinement:** in this activity, domain experts classify each KA into one domain/sub-domain and one level of the knowledge hierarchy, according to the knowledge that the KA represents or extends. In addition, they define the dependencies between KAs. If two KAs require the knowledge of each other, they are joined into a single one to avoid circularity and an inconsistent knowledge hierarchy.

Finally domain experts, provide a complete description of each KA, explaining the knowledge it encompasses and when the KA should be considered as represented.

The *domain knowledge hierarchy creation* step has two outcomes:

1. *The Knowledge Area Schema (KA-Schema)*: is the schema that contains the ontology knowledge hierarchy (with a similar structure as the schema shown

in Figure 4.5). Since it has a tree structure, the KA-Schema can be created with any tool used to create tree diagrams [84] or mind maps [18].

2. *The Knowledge Area Description Document (KADD)*: this document includes the list of the KAs, the description about the knowledge they encompass (with the concepts and relations it should include) and the list of CQs they encompass. The KADD should be written following the template shown in Table 4.2, which describes the sample *device manufacturer data KA*.

| Knolwedge area | Description/Competency Questions |
|---|---|
| Device manufacturer data | • **Description:** this knowledge area encompasses all the knowledge used to represent the device features related with the manufacturer (i.e., brand, model, serial number). It does not encompass device features related with operational aspects (i.e., power, height).<br>• **Competency Questions:**<br>1. What is the name of a device?<br>2. What is the model of a device?<br>3. What is the serial number of a device? |

**Table 4.2:** Template for the KADD

### 4.3.4 Step 3: Knowledge Classification

In the third step, ontology engineers classify the KAs defined in Step 2 (see Section 4.3.3) into each abstraction layer. This step takes as input the KA-Schema and KADD defined in Step 2.

A domain analysis of existing ontologies is performed by applying well-known SPL engineering techniques to classify the knowledge, since it is one of the core requirements of MODDALS. We defined this step based on the well-known domain analysis techniques and guidelines proposed by Pohl et al. [127] and Moon et al. [110] (introduced in Section 3.4.2),which were adapted to be applied in the ontology engineering field.

In SPL design, domain experts define based on their expertise the common software features that will be relevant for the domain from a strategical point of view before conducting the domain analysis [127]. Considering this, in MODDALS the same approach is applied. Before conducting the domain analysis, domain experts analyse the defined KAs to identify the ones that include abstract concepts and relations relevant to the domain and future applications. These KAs are defined as common from a strategical point of view and directly included in the *common-domain layer* regardless of its presence in existing ontologies. If the classification of these KAs depended only on their presence in existing ontologies, they might be classified in low-level layers although being relevant for the domain. Hence, as well as in the SPL design process, in MODDALS the domain experts have influence in the knowledge classification and it is not 100% dependent on existing applications.

The rest of KAs are classified according to the domain analyses of existing ontologies. This step includes five activities, which are conducted sequentially.

1. **Analysis of existing ontologies:** existing ontologies are analysed by ontology engineers to see whether they represent the KAs defined in Step 2. It is worth mentioning that this analysis has a different purpose and is more exhaustive than the one conducted in Step 2. In Step 2 the ontologies are analysed to identify and divide the knowledge they represent into KAs. In this step the ontologies are analysed to identify how many of them represent the defined KAs.

   We consider that an ontology represents a KA if it includes the necessary elements (classes/statements/axioms) to answer at least one of the CQs encompassed by the KA concerned. A related point to consider is that if a "child" KA is represented by the ontology, the "parent" KA that represents more abstract knowledge is considered represented. This rule avoids the placement of a parent KA is in a lower level than a child KA.

   Most of the ontology analysis is performed manually by the ontology engineer by examining in the ontology editor for the elements that represent the data encompassed by each KA. To identify faster the ontology elements that represent the knowledge of the KA, the ontology engineer can use the tools available in the editor (i.e., search engines) to find the key words of the KA and its description/CQs in the ontology elements.

   In addition to ontology engineers, domain experts also take part on this activity. They can assist ontology engineers with additional explanations and clarifications about the defined KAs. This collaboration helps ontology engineers to understand better the knowledge encompassed by a KA when it is not clear whether the KA is represented by an ontology.

2. **Commonality and Variability Analysis:** ontology engineers conduct a CVA of existing ontologies to determine whether the KAs of each subdomain are common to application types. As explained in Section 3.4.2, there are two

types of techniques to perform a CVA: the application requirements-matrix and the priority-based variability and checklist based variability analysis. We selected the *application-requirements matrix* to apply it in MODDALS among the aforementioned CVA techniques. This technique was selected because in MODDALS a CVA is applied to determine if the KAs are common to application types based on their presence or not in existing ontologies. These ontologies already include the knowledge defined by domain experts and the application stakeholders. The priority-based variability and checklist based variability analysis would involve defining a great part of the common and variant knowledge from scratch and doing meetings with stakeholders to establish their priorities.

To define this step, we took as reference the application-requirements matrix-based CVA conducted by Moon et al. [110], since it explains how to the apply application-requirements matrix technique in the design of SPLs through an application example. Since in MODDALS the CVA is conducted to classify the domain knowledge, we defined a new term for the matrix: the *application-knowledge matrix*. An example of the application-knowledge matrix template we propose in MODDALS is shown in Table 4.3. The left column contains the KAs of a specific subdomain (i.e., knowledge area 1, knowledge area 1.1). The top rows list different application types and the ontologies (i.e., ontology 1 (O1), ontology 2 (O2)) according to the application type they support. The matrix indicates if an ontology represents a KA ('X') or not ('-'). With this information, the ontology engineer deduces which application types reuse each KA. We consider that an application type reuses a KA if the KA is represented by at least one ontology that provides support to the application type.

To determine whether a KA is common or variant, their Commonality Ratio (CV ratio) is taken as a reference [110]. In this case, the CV ratio is the ratio of the number of application types that reuse a specific KA to the total number of application types. For instance, in Table 4.3 the *knowledge area 1* is reused by all application types, so it has a CV ratio of 100%. To the best of our knowledge, there is no systematic method to determine the exact threshold value of the CV ratio to distinguish the common and variant software features in SPL design. The CVAs conducted in the SPL engineering field [110, 13, 115] consider as common features the ones that are present in most of applications.

Considering the strategy followed in SPL design, in MODDALS the ontology engineer determines CV ratio threshold depending on the number of the

| | Application type 1 | | | Application type 2 | | Application type 3 | Application type 4 | | |
|---|---|---|---|---|---|---|---|---|---|
| **Ontology** / **Knowledge areas** | O1 | O2 | O3 | O4 | O5 | O6 | O7 | O8 | **Commonality Ratio** |
| Knowledge area 1 | X | - | - | X | - | X | X | X | 100% |
| Knowledge area 1.1 | X | X | - | X | X | X | X | X | 100% |
| Knowledge area 1.2 | X | X | - | - | X | - | - | - | 50% |
| Knowledge area 2 | X | X | X | X | - | - | X | - | 75% |
| Knowledge area 3 | - | - | - | X | X | - | - | - | 25% |
| Knowledge area 4 | - | - | - | - | - | - | X | - | 25% |

**Table 4.3:** Example of an application-knowledge matrix

application types included in the domain analysis. In the example, there are four application types. Therefore, we can consider that if at least three of them reuse the KA, the KA is reused by most of applications and can be considered as common. In particular, 75% can be as threshold value of the CV ratio to distinguish between common and variant KAs. The common KAs are the ones that equal or exceed the threshold value of the CV ratio, while the rest of KAs are considered variant.

3. **Knowledge area layer assignment:** ontology engineers place the KAs in different layers according to the CVA results. Common KAs are placed in the *common-domain layer*. Variant KAs reused by more than one application type are assigned to the variant-domain layer. Variant KAs reused only by one application type are placed in the *domain-task layer*. In addition, the KAs of this layer are classified according to the application type that reuse it.

4. **CVA at the application type level:** if the *domain-task layer* includes two sublayers to represent the knowledge of general and specific application types, another CVA at the application type level is required. Ontology engineers conduct this CVA to determine if KAs of this layer are relevant to the general application type or only to the specific application type. The KAs reused by more than one specific application types are likely to be reused by more future specific application types. Thus, these KAs are considered relevant to the general application type and they are placed in the *scenario sublayer*. The KAs reused only by a specific application type are assigned to the *application type sublayer*. The CVA at the application type level is applied to check if KAs are reused by one or more specific application types, so the CV ratio is not

taken as a reference. According to the results of the example CVA (Table 4.3), *knowledge area 3* and *knowledge area 4* are only reused by application type 2. If we consider that this application type encompasses more specific application types (*application type 2.1*, *application type 2.2* and *application type 2.3*) a CVA at the application type level is conducted (Table 4.4). According to the CVA results, *knowledge area 3* is placed in in the *scenario sublayer* and *knowledge area 4* is placed in the *application type sublayer*.

| | **Application type 2** | | |
|---|---|---|---|
| | **Application type 2.1** | **Application type 2.2** | **Application type 2.3** |
| Knowledge area 3 | X | X | - |
| Knowledge area 4 | - | - | X |

**Table 4.4:** CVA at application type level

The outcome of the *domain analysis* step is a list of the KAs of each layer/sublayer. To write this list we propose the template shown in Table 4.5. We have filled in this template according to the results of the sample CVAs shown in Tables 4.3 and 4.4.

| **Common-domain layer** | **Variant-domain layer** | **Domain-task layer** |
|---|---|---|
| Knowledge area 1<br>Knowledge area 1.1<br>Knowledge area 2 | Knowledge area 1.2 | **Scenario sublayer** |
| | | Knowledge area 3 |
| | | **Application type sublayer** |
| | | Knowledge area 4 |

**Table 4.5:** Template of the KAs classification into different layers

### 4.3.5 Step 4: Layer Knowledge Structuring

The last step is to define how the knowledge of each layer defined in step 1 is structured. This step is conducted by ontology engineers and takes as input the knowledge hierarchy defined in Step 2 (see Section 4.3.3) and the KA classification obtained in Step 3 (see Section 4.3.4).

The ontologies that follow the structure designed with MODDALS will correspond to layered ontologies reused by different applications. Hence, the knowledge of the layers must be structured to facilitate ontology reuse, as well as the inclusion of new knowledge to support new applications. To meet these requirements, previous reusable and usable ontology approaches [112, 154] structure the knowledge of each layer into ontology modules and define the high-level relations between them when designing the layered ontology structure. In addition, they apply the main principles of ontology modularization: loosely coupling and self-containment. These principles establish that an ontology module must depend as little as possible on other modules to ease their understanding, reuse and maintenance [45, 149].

Considering these principles, in this step the KAs of each layer are structured into ontology modules. The step includes two activities, which are conducted sequentially.

1. **Ontology modularization:** the ontology engineers classify the KAs of the ontology into different modules, which are defined in the following cases:

   - An ontology module is defined to include the top-level concepts of each domain and placed in the *common-domain layer*. The ontology module takes its name from the domain or the top-level concept (if the module includes only one concept). In this way, we abstract the knowledge that is extended by the rest of ontology modules.
   - An ontology module is defined for each KA (the module encompasses the knowledge of the KA), and placed in one ontology layer/sublayer according to the domain analysis results. There are two special cases where further classification is required. (1) The KAs of the *common-domain layer* are likely to be reused in most ontologies derived from the layered ontology. Hence, the KAs of each subdomain that belong to the *domain-task layer* are grouped into a single module that represents the subdomain common domain knowledge. (2) The ontology modules of the domain-task layer are classified according to the application type where the KA is reused.

2. **Inclusion hierarchy definition:** the ontology engineers organise previously defined ontology modules into an inclusion hierarchy that establishes the high-level relations between the ontology modules. Each ontology module must include only the modules whose knowledge extends or requires. These relations define how the modules will be linked during the ontology implementation.

   The ontology modules that represent the common knowledge of each subdomain will extend the domain top-level concepts. Therefore, these modules must include the modules that represent the domain top-level concepts. The rest of relations between modules are defined taking as reference the relations between KAs in the knowledge hierarchy defined in Step 2. By following these guidelines, only the ontology modules that represent closely related topics are related and their relations are limited. This ontology module independence will enable an easier reuse of individual modules when constructing application ontologies and the customization of particular modules without affecting other modules when reusing and extending the ontology [111].

As summary and example of this step, Figure 4.6 shows how the KAs of each are structured into ontology modules.

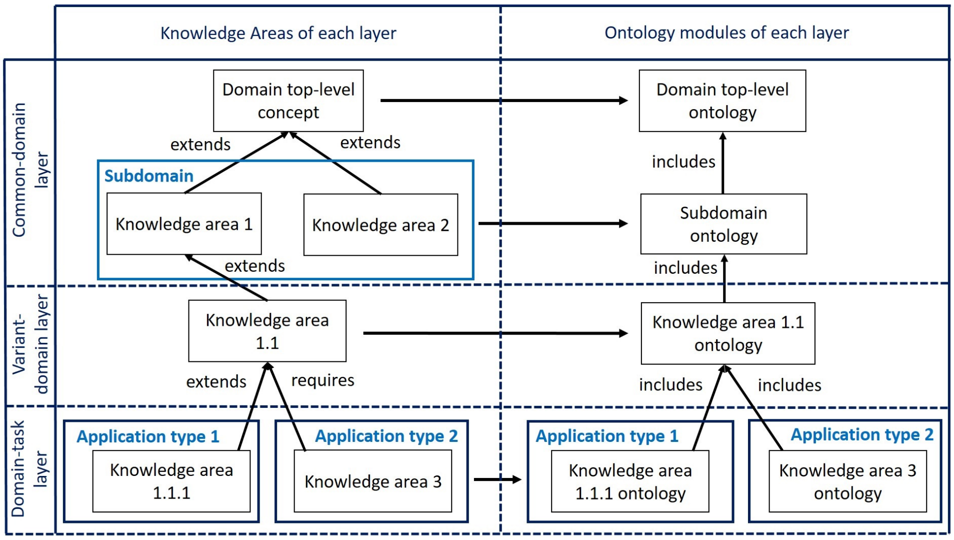


**Figure 4.6:** Ontology modularization and inclusion hierarchy definition

The outcome of this step is the informal model that includes the list of ontology modules of each layer and the high-level relations between the ontology modules. To write this list we propose the template shown in Table 4.6. We have filled in this template according to the sample ontology module classification shown in Figure 4.6 It is worth mentioning that the part of the template corresponding to the *domain-task layer* changes depending on the sublayers of the ontology and the application types it supports.

The informal model is complemented with the descriptions of the knowledge of each module at a conceptual level. These descriptions are taken from the descriptions of KAs made in Step 2.

| | | |
|---|---|---|
| **Common-domain layer** | | • Domain top-level ontology<br>• Subdomain ontology (*includes*: domain top-level ontology) |
| **Variant-domain layer** | | • Knowledge area 1.1 ontology (*includes*: subdomain ontology) |
| **Domain-task layer** | **Application type 1** | • Knowledge area 1.1.1 ontology (*includes*: knowledge area 1.1 ontology) |
| | **Application type 2** | • Knowledge area 3 ontology (*includes*: knowledge area 1.1 ontology) |

**Table 4.6:** Template of the ontology modules classification

## 4.4 Conclusions

In this chapter, we presented the MODDALS methodology. It guides domain experts and ontology engineers to design the layered structure of reusable and usable ontologies. The output of this process is an informal model with the ontology layers and the knowledge they include at a conceptual level.

MODDALS is the result of combining the best practices of the ontology engineering and SPL engineering fields. MODDALS adopts the main activities and ontology design principles applied by previous reusable and usable methodologies to define the layered ontology structure. In contrast to these methodologies, SPL engineering techniques are applied to classify the domain knowledge into defined layers according to a domain analysis of existing ontologies. This approach complements domain experts' and ontology engineers' expertise and prevents them from classifying the domain knowledge from scratch, facilitating the design of the layered ontology structure.

Based on the aforementioned techniques, MODDALS encompasses four sequential steps: (1) definition of ontology layers, (2) domain knowledge hierarchy creation, (3) knowledge classification and (4) layer knowledge structuring. In the chapter, we have described in detail the purpose, the activities, the actors involved and the inputs and outputs of each step.

# Part III

# Application in the Energy Domain

Chapter 5

# Semantic Representation of Energy Knowledge: Literature Review

## 5.1 Introduction

As stated in Chapter 1, the semantic representation of energy knowledge is a motivational scenario to highlight the need for a new reusable and usable ontology design methodology and the second contributing area of the thesis. This chapter presents the state of the art in this contributing area and motivates the need to apply MODDALS in the energy domain to address the challenges in the semantic representation of energy knowledge.

The chapter is organized as follows. Section 5.2 introduces the future Smart Grid vision. Section 5.3 explains the role of the Semantic Web in the achievement of this vision. Section 5.4 provides an overview about the energy management solutions based on Semantic Web technologies. Section 5.5 provides a gap analysis of energy ontologies and knowledge-based energy management applications, and defines the main challenges to be addressed. Taking as reference the identified gaps and challenges, Section 5.6 justifies the need for a new reusable and usable ontology design methodology in the energy domain. Finally, Section 5.7 summarizes the conclusions of the chapter.

## 5.2 Smart Grid

The Smart grid is envisioned as the next generation power grid. In contrast to the current grid, the Smart Grid integrates ICT to the existing power grid "*to create a widely distributed automated energy delivery network*" [54]. ICTs enable a two-way communication network between energy stakeholders such as residential customers, commercial and industrial customers, energy service providers, energy suppliers, Distribution System Operators (DSOs), Energy Market and energy transmission companies [120]. This network will improve demand side management and the knowledge that energy stakeholders have about energy usage. Figure 5.1 provides an overview of the Smart Grid showing the integration of different Distributed Energy Resources (DERs) and energy stakeholders.

Furthermore, by adding ICT to the current energy grid, a scalable and reliable integration of DERs such as renewable energy sources (i.e., photovoltaics, wind turbines), energy storage systems (i.e., batteries, capacitors) and electric vehicles can be performed. This will lead to new scenarios such as microgrids or Virtual Power Plants (VPPs). Both microgrids and VPPs are networks that will replace conventional power plants and will improve current grid efficiency and flexibility by integrating distributed generation, energy storage systems and loads [54, 157].

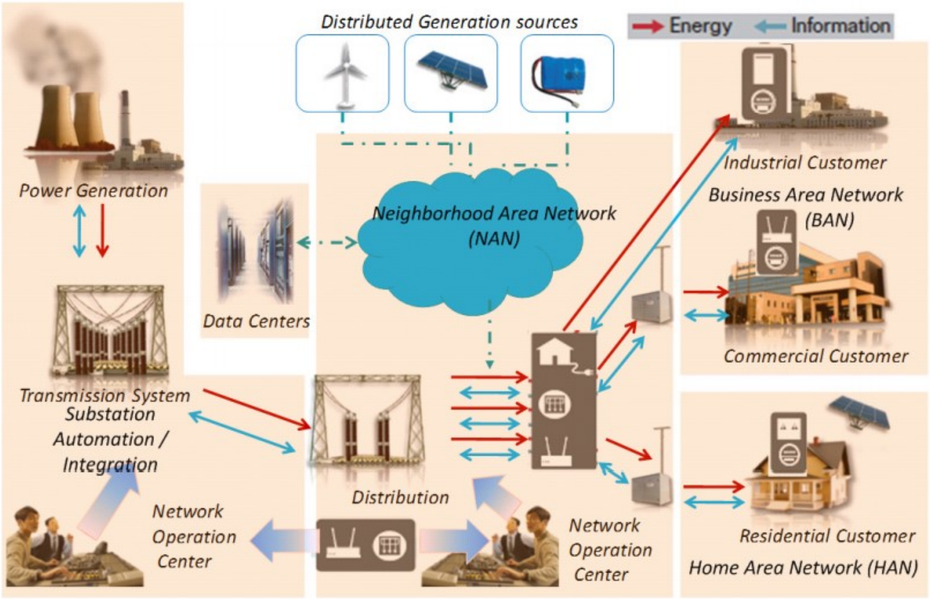


**Figure 5.1:** Smart Grid Communication Infrastructures [168]

Both advanced metering infrastructures and DERs integration require future Smart Grid applications such as home and building energy management systems, energy Demand Response (DR) applications, power outage management systems, advanced power distribution management and asset management [72]. One of the best-known Smart Grid applications is DR. According to Gungor et al. [72], the objective of DR is the "*control of the energy demand and loads during critical peak situations to achieve a balance between electrical energy supply and demand*". Through the deployment of these applications, Smart Grid aims to improve current infrastructures energy efficiency, sustainability and resilience [32].

- Energy efficiency deals with optimizing the use of both non-renewable and renewable energy sources through smart systems.

- In terms of energy sustainability, the objective is to provide citizens (i.e., energy consumers, energy auditors, building designers) a complete assessment of the energy performance of different infrastructures (i.e., homes, public buildings, organizations), and suggest actions to change their energy management behavioural patterns for economic, social and ecological purposes.

- Finally, energy resilience is about using ICT in collaboration with humans in order to prevent, avoid and react to power outages caused by power peak periods or natural disasters.

## 5.3 Semantic Web Role in the Energy Domain

The objective of using ICT-based solutions in the energy domain is to improve energy resources management and to integrate buildings and infrastructures (i.e., smart homes, public buildings, organization facilities) in the future Smart Grid. As stated in Section 5.2, Smart Grid aims to improve current infrastructures energy efficiency sustainability and resilience through ICT-based energy management solutions.

Current ICT-based solutions are mainly focused on optimizing infrastructure limited resources (i.e., energy, water) [60], and are being widely adopted in infrastructures [40]. In the energy domain, these solutions correspond to energy efficiency systems focused on optimizing the use of both non-renewable and renewable energy sources. These systems are made up of metering infrastructures (i.e., sensor networks) [54], communication technologies (i.e., ZigBee, 6LoWPAN) [103], data storage repositories (relational database management systems, NoSQL databases) and optimization techniques [81].

However, energy sustainability and resilience solutions require new data representation and exchange technologies [32]. Energy sustainability solutions include tools for infrastructure data exchange and analysis, i.e., Big Data [60]. They also include intuitive and user-centered display and social media tools for human-machine interaction. All these tools together conform *learning systems* [32, 60]. Energy resilience solutions add dynamic systems that learn, detect and react to real-time environmental changes in collaboration with humans. This collaboration is enabled by new soft computing methods (i.e., natural language processing, pattern recognition algorithms) that provide a human-computer automatic interaction. All these technologies together conform *cognitive systems* [32, 60].

Both learning and cognitive systems must learn from different environments/infrastructures in order to assist actors in changing their energy behavioural patterns and adapting to disruptive changes in collaboration with humans. In other words, learning and cognitive systems are required to exchange, extract knowledge and make decisions about different domains for large volumes of energy data collected at high rates and in most cases in real time [60]. According to Corrado et al. [26], energy data can be classified into the following categories:

1. **Energy performance data:** it includes energy consumption systems data (i.e., appliances, heating, ventilation and air conditioning (HVAC) systems), energy production and storage systems data (i.e., renewable energy sources, batteries), sensors/actuators data (i.e., energy consumption measurement sensors), device operation data (i.e., device states, device working modes), energy usage quantities (i.e., energy consumption, renewable energy production) and energy performance indicators (i.e., $CO_2$ emissions, energy cost).

2. **Energy-related and contextual data:** it includes building and infrastructure structural data (i.e., building construction, building geometry), infrastructure operation data (i.e., environmental conditions), geographical data (i.e., latitude and longitude, height above sea level), weather data (i.e., temperature, humidity, precipitation), environmental data (air pollutants of the urban area such as nitrogen dioxide, ozone), socio-economic data (i.e., population income and poverty, economic activity, population density) and energy actors (i.e., individuals, organizations) and the roles they have in the energy market (i.e., energy consumer, energy producer).

The ICT-based systems that gather and store energy data from different energy domains have traditionally operated in functional silos and rely on heterogeneous technologies. These factors hinder the data exchange and knowledge extraction for decision-making within Smart Grid sustainability and resilience systems and pose the following interoperability challenges [113]:

1. There is the need to create a model or representation of energy data from different domains [60].

2. This data must be represented and exchanged in a standard and machine-readable way [113].

3. Common vocabularies for human-machine interaction are needed [60].

Hence, new knowledge representation and exchange technologies are needed to enable an intelligent decision-making within the Smart Grid. Semantic Web provides the necessary technologies for addressing these challenges. Learning and cognitive systems will benefit from Semantic Web technologies in several ways:

1. Semantic Web enables the creation of vocabularies that provide machine and human-readable data representation.
2. Semantic Web vocabularies enable establishing relations and linking data from heterogeneous domains, thus creating a knowledge network that eliminates data silos.
3. The Semantic Web also provides standard mechanisms to access, exchange and process machine-readable data represented by systems that rely on heterogeneous technologies. Therefore, semantic interoperability is enabled, and there is a common understanding of the exchanged data among different systems
4. Thanks to semantics, machines are capable of inferring knowledge from explicit facts.

All benefits together result in a better performance of learning and cognitive systems that include intelligent agents and data analysis applications used for knowledge extraction and decision-making. Semantically represented knowledge improves the performance of intelligent agents and data analysis applications used for knowledge extraction and decision-making within energy management applications [41, 27, 56]. Therefore, the Semantic Web can be considered as the base of ICT-based solutions focused on improving energy sustainability and resilience. The Semantic Web is an intermediate layer between energy data and learning and cognitive systems (see Figure 5.2) [32].

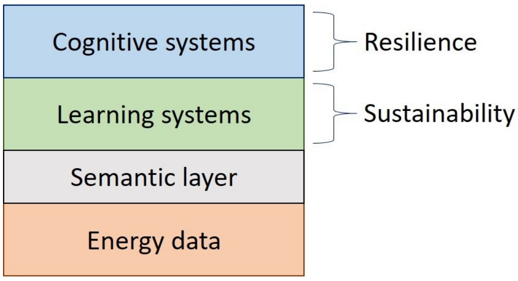


**Figure 5.2:** Semantic web as the base of energy management solutions

## 5.4 Overview of Energy Management Solutions based on Semantic Web Technologies

From the beginning of the current decade, Semantic Web technologies have been applied for developing energy ontologies that represent the knowledge from different energy domains. This knowledge is used as knowledge base by specific Smart Grid sustainability and resilience systems that rely on learning and cognitive technologies.

In this section, we show the state of the art of energy ontologies and knowledge-based energy sustainability and resilience management solutions, as well as their main purpose. An extended version of the presented state of the art can be found at [32]

### 5.4.1 Energy Ontologies

Below an overview of the existing energy ontologies, the knowledge they represent and their purpose is provided.

On the one hand, Kofler et al. [92] and Daniele et al. [44] presented ontologies that represent the knowledge about Smart Home energy performance.

Kofler et al. [92] presented the ThinkHome ontology, which was developed within the ThinkHome project[1]. This ontology represents, in a machine-readable way, home energy consumption, production and energy-related contextual data. The ontology is made up by several ontologies that represent the knowledge of different domains.

- *Building ontology*: it represents building architecture data, i.e., layout, spaces data.
- *User information ontology*: it represents user comfort preferences and user schedules.
- *Processes ontology*: it represents home system processes and user activity data.
- *Exterior ontology*: it represents weather and climate conditions.
- *Energy and resource parameter ontology*: it represents home equipment or devices (i.e., home appliances, energy measurement sensors) data, home energy demand and supply data and energy providers and energy tariffs data.

[1] http://www.eui.eu/Projects/THINK/Home.aspx

The ThinkHome ontology is expected to represent the knowledge bases of multi-agent smart home energy management systems. Furthermore, the authors suggest how represented data can be used and combined with a multi-agent system in order **to improve energy efficiency at future smart homes.** The proposed use cases are:

- Select energy providers depending on produced energy type or energy tariffs, i.e., consume only energy produced by renewable energy sources or select a provider that has an excess of energy and sells it cheaper.
- Disconnect unnecessary equipment according to occupancy or customer behaviour patterns, i.e., disconnect from the electricity the grid entertainment equipment such as the TV when user is unlikely to return more to the living room.

Daniele et al. [44] presented the SAREF ontology [43], and its later version: SAREF4EE. The SAREF4EE ontology represents the knowledge of the domains related with the energy performance and flexibility of home energy devices. Specifically, the SAREF4EE ontology represents the knowledge about: home appliances, sensors and actuators data (i.e., device manufacturer, device state, device function, energy flexibility), building spaces (i.e., rooms), home energy production and consumption data, associated costs, energy performance data time intervals, home weather conditions and home occupancy data. The objective of the SAREF4EE ontology is **to improve interoperability among electrical appliances of different manufacturers,** allowing them to be connected with customer energy management systems used for Smart Grid DR optimization strategies.

Fernbach et al. [59] and Tomašević et al. [155] presented ontologies to provide interoperability between energy management systems for different applications.

Fernbach et al. [59] present an ontology that represents the knowledge about building features and building automation system, which monitor and control automatically HVAC systems of indoor environments [87]. The ontology is presented as a first step of using Semantic Web technologies for the **automated integration of building automation systems developed by different manufacturers.**

Tomašević et al. [155] presented a facility ontology that represents the knowledge about facility devices. This knowledge includes device technical characteristics (i.e., vendor specific data) and data about the location of these systems in the facility. The facility ontology was developed as common information model **to provide interoperability between supervision and control systems from different vendors focused on improving facilities energy management.**

Blomqvist and Thollander [9] published as Linked Data energy efficiency improvements, energy saving recommendations and energy measures taken from previous energy audits within the DEFRAM project[2]. These data are represented with a vocabularies of an ontology developed by the authors. The ontology includes the knowledge about energy audits and measures of industrial organizations and recommendations for improving energy management given after previous audits. It also represents the knowledge about investment cost of applying such recommendations, achieved energy saves and organizations' additional information such as organization location or organization facility size. The final purpose is to use the published linked data as a knowledge base for future ICT-based solutions **to help organizations for saving energy based on energy audits performed over similar organizations, to facilitate researchers and policy makers comparing and analysing data from different audits and to facilitate third parties' applications that use energy audits data.**

The OntoMG ontology [137] provides support to microgrid energy management systems, since it represents the knowledge about the domains related with microgrids energy performance. The ontology encompasses renewable and non-renewable generators, storage equipment, electrically connected loads and their properties, which include mobility, economical, operational and ecological aspects. The purpose of this ontology is to be used by computational and optimization techniques aiming **to achieve different microgrid objectives such as minimising transmission losses, generating good power quality or minimisation of greenhouse effect gases.**

Hippolyte et al. [76, 77] and Gillani et al. [62] presented energy ontologies to support energy management applications deployed in Smart Grid wider areas.

Hippolyte et al. [76] presented the MAS2TERING ontology, which was developed within the MAS2TERING European project[3]. This ontology represents Smart Grid prosumers' (energy stakeholders that produce and consume energy) energy data. It links concepts of data representation standards used in different energy domains. These concepts are the following: home area networks (smart appliances, power profiles, renewable energy generation, smart meters and smart user interfaces), energy DR concepts (i.e., market context, dynamic pricing and event descriptions) and Smart Grid stakeholders' and their roles and responsibilities within both the energy supply value chain and the flexibility value chain. The MAS2TERING ontology is aimed **to facilitate the representation the data of different Smart Grid**

[2] https://www.ida.liu.se/~evabl45/defram.en.shtml

[3] http://www.mas2tering.eu/

**domains and provide interoperability among different Smart Grid agents and stakeholders.** The authors' final purpose is to use this ontology as a base for Smart Grid multi-agent systems for an energy market coordination process for improving energy flexibility among energy prosumers and DSOs.

Hippolyte et al. [77] presented the EE-DISTRICT ontology, which provides a common energy knowledge representation for district energy management applications. EE-DISTRICT represents knowledge about district energy consumption systems (i.e., district heating, district cooling), district energy suppliers and operators, and district facilities owners and managers. The purpose of EE-DISTRICT ontology is **to unify the knowledge access to district energy management software applications.**

Gillani et al. [62] presented the ProSGv3 ontology, which represents the energy data of prosumer oriented Smart Grid scenarios. The ProSGv3 ontology encompasses the knowledge about infrastructure data (type of operation, time and geographical location, and power critical premises), electrical appliances data (consumption and temporal data, power consumption rating and operational patterns), electrical generation systems data, power storage systems data (type, produced power, charge and discharge efficiency), weather report data, events (i.e., electrical appliance events, weather events, storage events and generator events), energy production and consumption services contractual information and connectivity relationships between producers and consumers. The ontology is aimed to be complemented with an inductive reasoning layer. This layer will contain applications for detecting appliance energy consumption patterns, as well as energy production and energy producer performance patterns (i.e. efficiency, impact to the environment). The objective is **to improve Smart Grid DR and sustainability by predicting Smart Grid energy consumption and production**.

Finally, the smart city ontology catalogue[4] includes a set of ontologies that represent the knowledge about smart city energy performance. The Mirabel ontology [161] represents the knowledge about energy actors' (i.e., home end-users) and energy flexibility for specific devices (i.e., home appliances). The purpose is **to connect energy management systems developed by different energy stakeholders to handle supply and demand of energy.** The The Leeds City Council Energy Consumption Ontology (LCC ontology) [132] represents the knowledge about building energy consumption. This ontology has been developed **to publish energy consumption data about cities' infrastructures as a knowledge base.**

[4] http://smartcity.linkeddata.es/

### 5.4.2 Knowledge-based Energy Sustainability Solutions

Energy sustainability systems [37, 79, 17, 116, 128, 1, 56, 170, 148] use different data analysis techniques and display tools over semantically represented energy data models. These systems provide citizens a holistic view of infrastructures energy performance and suggest actions for changing their energy management behavioural patterns. Once the knowledge-based energy sustainability systems are widely adopted in current infrastructures, home energy consumers and both public and private organizations will see their energy bills slashed. They also will be able to choose between a wide variety of energy vendors depending on their energy tariffs. Public and private organizations will also be benefited, since they will perform a more efficiency management of their energy consumption sources (i.e., facilities, business travel) with both economic and ecological purposes [37].

Below an overview of the developed knowledge-based energy sustainability solutions is presented.

Curry et al. [37], Hu et al. [79], Niknam and Karshenas [116], and Pont et al. [128] presented energy management systems that assess citizens about urban infrastructure energy performance.

Curry et al. [37] presented an enterprise energy observatory system. The aim of this system is **to improve enterprise energy management at different levels from both economic and ecological perspectives.** The enterprise energy observatory system includes data analysis and display applications that provide an enterprise energy performance view at organizational, function and individual level:

- *Organizational level:* executives can view the real-time consumption of energy across all enterprises domains such as Information Technology facilities or travel.
- *Function level:* the system provides a fine-grained understanding of what business activities are responsible for IT energy usage, and can enable IT to bill appropriately.
- *Individual level:* it gives an employee real-time energy consumption data on their IT facilities or travels.

The system includes also internal applications (i.e., a complex event processing engine, data search and query engines) that ease the knowledge extraction of enterprise Linked Data by energy analysis applications. All system applications are underpinned by energy related data from different enterprise domains that have been published as Linked Data [39].

Hu et al. [79] developed a building energy performance assessment system (EPAS) within the SuperB project[5]. This system **shows the performance gap between predicted and measured building energy performance data.** The EPAS includes tools that measure, analyse and show building or particular zones energy performance data. The energy performance data are expressed as energy metrics such as energy use intensity, energy cost or normalised atmospheric emissions. These metrics are compared with building predicted energy performance data. A building energy performance simulation model makes these predictions. Data used by the EPAS analysis and display tools is represented with the vocabularies of an ontology [27] that contains and links/fuses building data of different domains.

Niknam and Karshenas [116], and Pont et al. [128] also presented building EPASs, but in this case these systems are focused on the design stage.

The EPAS developed by Niknam and Karshenas [116] **shows building designers the building energy performance corresponding to a building specific design.** The objective is to optimize the building design for a better energy performance. Specifically, a prototype of the EPAS was developed to predict through a heating cost calculation algorithm the building heating cost based on the data about the design and simulated environmental conditions. The EPAS is underpinned by four ontologies that represent the knowledge about building properties, mechanical equipment specifications, historical weather information of building geographic location and energy cost information.

Pont et al. [128] presented a web decision support and optimization platform for building designers. The purpose of the web platform is to make buildings energy performance-oriented designs within the SEMERGY project[6]. This platform **shows building designers' suggestions about different building components alternatives according to user preferences and technical constraints for optimizing heating demand, environmental impact and investment cost.** These suggestions are made by a reasoning interface that makes inferences from semantically represented data about building design and simulated environmental conditions.

The SEMANCO[7] integrated platform was developed within the SEMANCO project[8]. This platform shows the energy related data about cities to different actors. The aim of this platform is **to provide a complete view of a city's energy performance in order to help different city actors (i.e. energy policy makers, building**

[5] http://cordis.europa.eu/project/rcn/187015_en.html

[6] http://www.semergy.net/

[7] http://www.semanco-project.eu/index_htm_files/SEMANCO_D5.4_20131028.pdf

[8] http://semanco-project.eu/

**designers, citizen) to make informed decisions for reducing cities carbon emissions.** The platform includes visualization tools that display energy data and analysis tools that perform different analysis tasks (i.e., make energy performance predictions, classify buildings according to their consumption or carbon emissions) over cities energy data at different scales (building, neighbourhood, municipality or region). The integrated platform is underpinned by the SEMANCO ontology, which captures energy efficiency concepts of urban areas [26]. In particular, the SEMANCO ontology represents the knowledge about building energy consumption data, associated energy performance indicators (i.e., energy savings, energy costs) and timestamps, consumed energy sources, building features, building equipment features and services. The ontology also represents external factors that hinder the energy performance such as weather conditions, building geographical location, demographic, environmental and socio-economic data.

The solutions presented by Burel et al. [17], Fensel et al. [56], Yuce and Rezgui [170], and Stavropoulos et al. [148] apart from offering energy assessment, are focused on **offering citizens suggestions for improving urban infrastructures energy performance.**

Burel et al. [17] presented the EnergyUse collaborative web platform. The purpose of this platform is **to raise a home end users' climate change awareness.** The platform collects home appliances energy consumption data from smart plugs and allows end users viewing and comparing the actual energy consumption of various appliances. Users can also share energy consumption values with other users and create open discussions about energy saving tips. Discussions are described and classified by tags defined by users. These tags correspond to energy appliances and topics related with the discussed energy saving tips. The EnergyUse platform includes tools that analyse and extract concepts from discussions created. These tools link extracted concepts with appliance and environmental terms included in external semantic repositories in order to create new tags and descriptions for discussions. The purpose of these additional tags and descriptions is to improve user navigation experience among discussions. Finally, the EnergyUse platform also exports appliance consumption and community generated energy tips as linked data to be used by third parties, such as other users or websites. The EnergyUse platform is supported by the EnergyUse ontology. The EnergyUse ontology represents the knowledge of the domains related with the home energy consumption. Specifically, it represents the knowledge about user profiles that use the platform, home appliances and HVAC systems data, home sensors and actuators data, home appliances energy consumption measures and energy tips discussion data [17].

Fensel et al. [56] presented a home energy management platform developed within SESAME and SESAME-S[9] projects. The aim of this platform is **to help home users making better decisions in order to reduce their energy consumption.** The platform allows users defining energy saving policies and it generates its own energy saving policies through an ontology reasoning engine. Specifically, this ontology reasoning engine generates schedules and rules for turning on and off home devices based on tariff plans and desired indoor environmental conditions. Energy saving policies are presented through different user interfaces aimed to stimulate and facilitate users to use energy more responsibly. Home energy data are semantically represented with the vocabularies of several ontologies. These ontologies represent the knowledge about home automation devices, metering equipment, and energy types and tariffs.

Yuce and Rezgui [170] presented a building energy management system that **assists users to save energy** developed within the KnoholEM Project[10]. This system is underpinned by a semantic knowledge database that contains building information and devices metering data. These data are used by an artificial neural network that learns building consumption patterns, and a genetic algorithm-based optimization tool that generates optimized energy saving rules taking into account learned energy consumption patterns and different objectives (including comfort) and constraints. These rules are presented to facility managers as energy saving suggestions through a graphical user interface.

Stavropoulos et al. [148] developed a building energy management system that combines energy assessment, energy advice and building automation was developed. This system **monitors building energy performance and shows this information to allow users taking actions to increment energy savings.** Intelligent agents within the system also devise short-term and long-term energy saving policies automatically generated and enforced. Furthermore, the system is also designed to receive energy providers' instructions in future Smart Grids. This system is supported by the BOnSAI ontology [147], which represents the knowledge about building energy performance. In particular, it represents the knowledge about building appliances and sensor/actuators data, building structure data, user location and energy and environmental condition measures.

[9] http://sesame-s.ftw.at.
[10] http://www.knoholem.eu/page.jsp?id=2

### 5.4.3 Knowledge-based Energy Resilience Solutions

Energy resilience systems [174, 63, 141, 102, 172] are focused on improving Smart Grid DR by detecting disruptive situations (i.e., power peak periods) over semantically represented data. These systems react to disruptive situations in collaboration with humans.

Below an overview of the developed knowledge-based energy resilience solutions is presented.

Zhou et al. [174] and Gillani et al. [63] presented complex event processing engines applied in the energy domain. Complex event processing deals with detecting "*real-time situations represented as event patterns*" Zhou et al. [174].

Zhou et al. [174] developed a complex event processing engine [175] developed within the Los Angeles Smart Grid Demonstration Project[11]. The purpose is **to enable dynamic DR applications that detect power peak situations and perform actions to improve DR.** The complex event processing engine is supported by a Smart Grid semantic information model [173] made up of different ontologies in order to represent different energy data domains (i.e., electrical equipment, infrastructure information, weather information).

The complex event processing engine developed by Gillani et al. [63] is oriented **to ease energy transaction between energy producers, consumers and prosumers within the Smart Grid.** The complex event processing engine is supported by the ontology that represents energy data of prosumer oriented Smart Grids presented by Gillani et al. [62].

Shi et al. [141] and Maffei et al. [102] presented energy management solutions to improve microgrids resilience.

Shi et al. [141] presented **a microgrid energy management and control system that combines both sustainability and resilience actions**. On the one hand, the microgrid energy management system includes a Human Machine Interface (HMI) for microgrid monitoring and control. Apart from that, the system includes a microgrid scheduling algorithm and a microgrid DR optimization algorithm. The DR optimization algorithm adapts microgrid demand to real-time energy prices. The energy-scheduling algorithm schedules microgrid DERs and loads with both economic and ecological optimization purposes. Both algorithms use semantically represented data that includes microgrid devices information, weather forecast information, DR signals received from the utility and energy market information.

[11] https://www.smartgrid.gov/project/los_angeles_department_water_and_power_smart_grid_regional_demonstration.html

Maffei et al. [102] presented a semantic-middleware for multi-objective energy management in microgrids. This system **is focused on reducing the microgrid operating costs, as well as line losses** based on the microgrid forecast energy demand and renewable generation. The system is underpinned by an ontology that represents and links data about microgrid loads and devices, loads energy performance and load control and metering systems.

Finally, Zhang et al. [172] presented an energy management platform for VPPs. VPPs are groups of DERs and controllable loads that act as single energy stakeholders within the Smart Grid. Within VPPs, energy prosumers sell their surplus energy during energy curtailment or energy consumption peak load periods. The energy management platform adapts VPPs energy production and consumption to peak loads that occur both either in the VPP or the Smart Grid. The energy management platform includes algorithms used **to face energy peak load periods in Smart Grid and VPP in a distributed manner.** The selection of the strategy is based on energy generation sources and loads, respective energy generation and consumption forecasting performed by machine learning algorithms, i.e., Dynamic Bayesian Networks. All the information used by the platform to manage VPPs energy DR is semantically represented by an ontology, which includes knowledge about building features and energy performance.

## 5.5 Gap Analysis and Discussion

According to Curry et al. [40], the development ICT-based solutions for infrastructures is divided into two cycles. The first cycle corresponds to a research phase that includes experimental design and pilot deployment. The second cycle is focused on citywide deployments of ICT solutions to drive mass market adoption.

This cyclic approach can be also extended to the energy management solutions presented in the previous section. There is a lot of literature about knowledge-based systems focused on improving energy sustainability. All these systems are limited to pilot demonstrators that in some cases were implemented in specific scenarios, such as buildings [79], organization facilities [37] or smart homes [56]. There is less literature about knowledge-based systems focused on improving Smart Grid resilience. These solutions are still in the experimental design [174] or correspond to implementations in pilot demonstrators such as pilot microgrids [141, 102] or pilot VPPs [172]. Therefore, we can state that knowledge-based energy sustainability and resilience solutions are still in early stages towards their deployment in real scenarios and mass-market adoption [40].

Each knowledge-based energy management application uses a specific ontology to represent its knowledge base because each application has individual knowledge requirements to manage different types of infrastructures with different purposes, i.e., home energy saving, energy DR improvement. Nevertheless, the representation of a significant number of energy domains is repeated across the energy ontologies. Table 5.1 shows the energy domain represented by some of the ontologies reviewed in Section 5.4. For example, all these energy ontologies represent the knowledge about energy consumption systems or DERs.

| **Ontology**<br>**Energy domains** | **ThinkHome ontology**[12] | **SAREF4EE ontology**[13] | **BOnSAI ontology**[14] | **EnergyUse ontology**[15] | **ProSGV3 ontology**[16] |
|---|---|---|---|---|---|
| Energy consumption systems data | X | X | X | X | X |
| Distributed energy sources data | X | X | X | X | X |
| Metering/actuation equipment data | X | X | X | - | X |
| Device operation data | X | X | X | X | X |
| Infrastructure structural data | X | X | X | X | X |
| Infrastructure operation data | - | - | - | - | X |
| Energy usage data | X | X | X | X | X |
| Energy key performance indicators | - | - | - | - | - |
| Geographical data | X | - | X | X | X |
| Weather/climate data | - | - | - | - | X |
| Environmental data | - | - | - | - | X |
| Individual actors data | X | - | X | X | - |
| Energy market roles data | X | - | X | X | X |

**Table 5.1:** Energy data domains represented by energy ontologies

[12] https://www.auto.tuwien.ac.at/downloads/thinkhome/ontology/

[13] http://ontology.tno.nl/saref4ee/

[14] http://lpis.csd.auth.gr/ontologies/bonsai/BOnSAI.owl

[15] http://eelst.cs.unibo.it/apps/LODE/source?url=http://socsem.open.ac.uk/ontologies/eu

[16] http://data-satin.telecom-st-etienne.fr/ontologies/smartgrids/proSGV3/ProSG.html

Since they are developed by different engineers and with different purposes, the energy ontologies represent the same energy domains with different level of detail and with different vocabularies. Table 5.2 shows how some of the existing energy ontologies represent each energy domain with different levels of detail. The level of detail was determined by performing a high-level analysis of the knowledge representation of each ontology. The completeness of an ontology is mainly determined by checking whether (1) it includes enough classes and properties to represent the required knowledge and (2) the ontology class hierarchies are granular enough to represent the required knowledge [163]. Therefore, we analysed the number of classes/properties and the granularity level of class hierarchies applied by the ontology to determine the level of detail with which the ontology represents each domain. To the best of our knowledge, there is not a widely accepted boundary of number of classes/properties and granularity level to determine if an ontology represents or not with a high level of detail a certain domain. Thus, we have adopted our own boundaries based on how each energy ontology represents each energy domain. We have classified the level of detail with which an ontology represents a certain energy domain in three categories: High, Medium and Low.

We consider that an ontology represents an energy domain with a high level of detail if (1) it represents 70% of identified classes and properties in the domain concerned and (2) it extends the top-level classes with class hierarchies of at least four levels. We consider that an ontology represents an energy domain with a low level of detail if (1) it represents at most 20% of identified classes and properties in the domain concerned and (2) it extends the top-level classes with class hierarchies of at most two levels. We consider that the rest of the ontologies represent the domain with a medium level of detail.

The diversity of level of detail in the representation of energy domains is more evident in Figure 5.3, which shows which concepts that represent energy consumption systems are included by each ontology. Although most of ontologies coincide in a set of concepts, each of them introduce new concepts. In addition, the ontologies represent the same knowledge with different terminology and vocabularies. For instance, the ThinkHome ontology represents home appliances through the *Appliance* class, while other ontologies employ other classes such as *HomeAppliance* or *ElectricalAppliance*. The same thing occurs with the ontology properties and restrictions applied to represent each domain.

| Ontology / Energy domains | ThinkHome ontology[17] | SAREF4EE ontology[18] | BOnSAI ontology[19] | EnergyUse ontology[20] | ProSGV3 ontology[21] |
|---|---|---|---|---|---|
| Energy consumption systems data | High | Low | Medium | Low | Medium |
| Distributed energy sources data | Medium | Low | Low | Medium | Medium |
| Metering/actuation equipment data | High | Medium | Medium | - | Medium |
| Device operation data | Medium | High | Low | Low | Low |
| Infrastructure structural data | High | Low | Medium | Low | Medium |
| Infrastructure operation data | - | - | - | - | Low |
| Energy usage data | High | Low | Low | Low | High |
| Energy key performance indicators | - | - | - | - | - |
| Geographical data | High | - | Low | Low | Medium |
| Weather/climate data | - | - | - | - | Low |
| Environmental data | - | - | - | - | Medium |
| Individual actors data | Medium | - | Low | Low | - |
| Energy market roles data | Medium | - | Low | Low | Medium |

**Table 5.2:** Level of detail of energy domain representation (H=High/M=Medium/L=Low)

Therefore, the energy ontologies present a domain representation diversity, that is, a semantic heterogeneity. This heterogeneity hinders the deployment of knowledge-based sustainability and resilience solutions in real scenarios. Current knowledge-based energy management solutions are limited to pilot demonstrators in specific infrastructures. However, the adoption of these solutions in real scenarios will probably require the knowledge exchange among applications that operate in different infrastructures (i.e., smart homes, organizations) to perform a large-scale energy management. Since knowledge-based energy management applications

[17] https://www.auto.tuwien.ac.at/downloads/thinkhome/ontology/

[18] http://ontology.tno.nl/saref4ee/

[19] http://lpis.csd.auth.gr/ontologies/bonsai/BOnSAI.owl

[20] http://eelst.cs.unibo.it/apps/LODE/source?url=http://socsem.open.ac.uk/ontologies/eu

[21] http://data-satin.telecom-st-etienne.fr/ontologies/smartgrids/proSGV3/ProSG.html

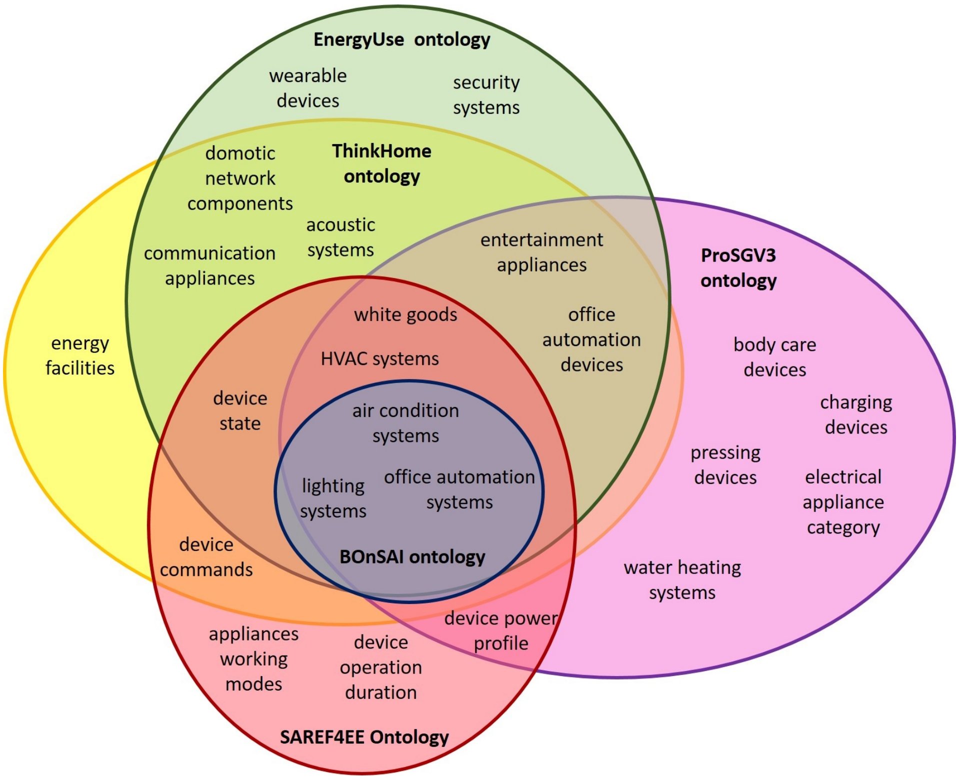


**Figure 5.3:** Level of detail of energy consumption systems data representation

rely on heterogeneous knowledge bases, the semantic interoperability among these applications, and by extension the knowledge exchange, is hampered.

As stated in Chapter 2, global ontologies are the main approach to overcome semantic heterogeneity. Hence, there is the need of creating a global ontology that provides a common and unique representation of the energy domains heterogeneously represented by existing energy ontologies. In that way, the knowledge of a global energy ontology could be reused as a base to develop interoperable ontologies that represent the knowledge required by different energy management applications. Although these ontologies may not represent the same knowledge, the knowledge they have in common would be represented with the same vocabularies.

Figure 5.4 provides a motivating scenario to highlight the need for a global ontology. Let us consider three scenarios; a smart home, a green building that is self-sufficient in solar energy (i.e., owned by an organization) and a solar power plant connected through a renewable electrical grid, forming a microgrid. Each scenario includes an energy management system:

1. A smart home energy management system that controls the home loads

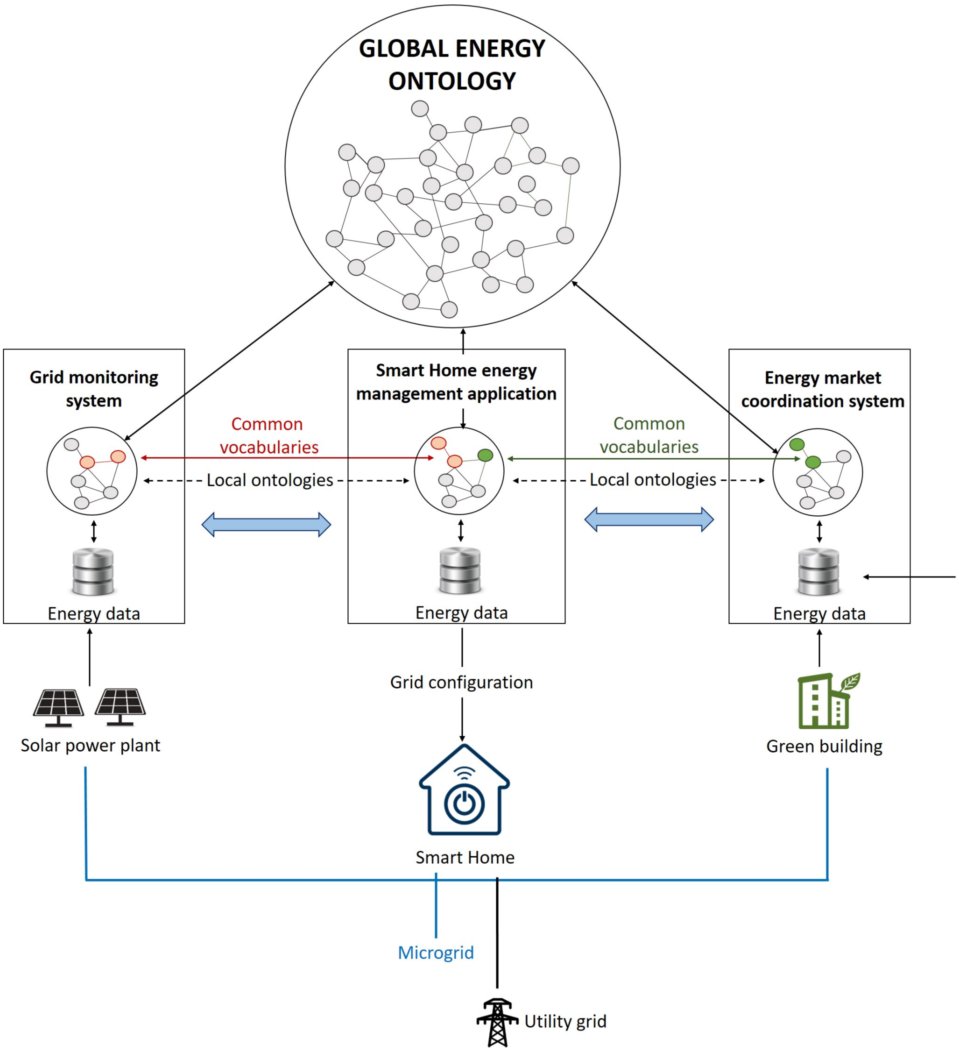


**Figure 5.4:** Motivational scenario to highlight the need for a global energy ontology

which do not need permanent supply (i.e., entertainment equipment, small appliances) to save energy based on real-time home occupancy data, home occupancy patterns and device usage profiles.

2. A building energy assessment system that compares the forecasted energy consumption and generation of a green building self-sufficient in solar energy to prevent power outages, as well as monitoring in real time the stored energy.
3. A solar power plant monitoring system that predicts the energy generation of the software plant (based on solar panel features and weather forecast) and compares it with the real generation to detect possible power loses in the power plant.

These systems are underpinned by knowledge bases represented with the vocabularies from different energy ontologies. These knowledge bases enable knowledge extraction and decision making from energy data from heterogeneous domains to improve the energy sustainability and resilience in specific scenarios. However, considering the features of the presented microgrid, an energy management at a greater scale can be performed to improve the energy sustainability and resilience among these infrastructures. For instance, the smart home energy management system can be configured so that the home uses the renewable energy from the microgrid while (1) the power plant has no power loses and it is expected to continue to produce energy in the short term (2) the green building has surplus energy and is expected to continue to produce energy in the short term. Otherwise, the smart home must use energy from the utility grid. This new scenario requires the exchange of knowledge about real time energy performance (energy generation forecast, energy storage and energy loses) between the smart home energy management system and the green building and power plant energy management systems. If the knowledge about energy performance is represented with different ontology vocabularies in the knowledge base of each system, the exchange of this knowledge will be hampered. A global energy ontology would provide a common representation of energy performance knowledge. This knowledge representation can be reused as a base to develop ontologies used to represent the knowledge required by each energy management system. Although the knowledge bases of each system may not represent the same knowledge, the knowledge they have in common would be represented with the same vocabularies. Thus, interoperability between the energy management systems would be enabled.

The energy domains are complex. The existing energy ontologies represent complex and extensive domains. For example, the data of energy devices encompasses data about device features (i.e., device type), device operation data (i.e., device state, device working mode, device location) or device energy performance (i.e., energy production/consumption). Therefore, a global energy ontology will be a large-scale ontology. A global energy ontology should provide support to a wide variety of energy management applications, so it should be reusable. Thus, it should be reusable and it should include abstract domain knowledge reused by many applications. Apart from sharing information with other energy management applications, each energy management application has individual objectives and requires a specific knowledge representation. If the global ontology is too abstract, the effort of adapting and customizing the knowledge to satisfy specific application requirements would be high. Thus, ontology developers are less likely to reuse the global energy ontology.

Considering this, a global energy ontology should also minimise the ontology customization effort when it is be reused by specific energy management applications, so it should be usable. Nevertheless, if the ontology represents the knowledge required by a specific energy management application, the effort of reusing the ontology in applications with different knowledge requirements would be high. Therefore, a global energy ontology should provide a balance of reusability-usability. Considering the benefits of the layered ontology design approach (see Section 3.3.2), its application to design a global energy ontology would reduce the ontology reuse effort in different applications.

In summary, the main objectives of a global energy ontology are the following:

1. To provide a common energy domain representation.

2. To provide a balance of reusability-usability.

However, as stated in Chapter 2 a common domain representation does not enable interoperability between new and legacy applications. Therefore, the global energy ontology should be linked with existing energy ontologies to enable interoperability between new and legacy knowledge-based energy management applications.

## 5.6 Need for a New Reusable and Usable Ontology Design Methodology in the Energy Domain

Since the energy domains are complex, the application of current reusable and usable ontology design methodologies (reviewed in Section 3.3.3) to design the layered structure of a global energy ontology that follows a layered structure would require a great effort. In particular, the definition of the ontology common and variant domain knowledge from scratch would require a high involvement and effort to domain experts and ontology engineers. Thus, MODDALS is a key enabler of the development a reusable and usable global energy ontology. Considering the semantic-based energy management solutions reviewed in Section 5.4, there are already developed many energy ontologies to support energy management applications that operate in different scenarios. Hence, the necessary conditions are met to apply MODDALS.

## 5.7 Conclusions

In this chapter, we have highlighted the main limitations and challenges in the application of the Semantic Web in the energy domain. In addition, we have set the bases of the solutions proposed to address the identified challenges.

The energy ontologies and knowledge-based energy sustainability and resilience applications have been analysed. This analysis has revealed that existing energy ontologies apply heterogeneous vocabularies to represent the same energy domains, which hampers the interoperability between energy management applications that operate in different scenarios. This problem hinders the deployment of knowledge-based sustainability and resilience solutions in real scenarios. Therefore, **there is the need to create a global ontology that provides a common energy domain representation.** Energy domains are complex and a global energy ontology should provide support to a wide variety of energy management applications. Thus, **a global ontology should provide a balance of reusability-usability.**

Since the energy domains are complex, the application of current reusable and usable ontology design methodologies to design the layered structure of the global energy ontology would require a high effort. Hence, **MODDALS is a key enabler of the development a reusable and usable global energy ontology.**

Chapter 6

# Application and Evaluation of MODDALS in the Energy Domain

## 6.1 Introduction

In Chapter 5, we analysed the current knowledge-based energy management solutions. We concluded that the main challenge in the semantic representation of energy knowledge is the development of a reusable and usable global energy ontology. Since the energy domains are complex, MODDALS is a key enabler of the development of a reusable and usable ontologies. Therefore, the energy domain is a potential case study to apply and evaluate MODDALS.

This chapter focuses on the application and the evaluation of the MODDALS methodology in the energy domain. MODDALS was applied to design the layered structure of the DABGEO ontology [1]. DABGEO is a reusable and usable global ontology for the energy domain that follows a layered structure (DABGEO structure and development process are described in detail Chapter 7). The application of MODDALS to design DABGEO layered structure was taken as reference to evaluate empirically the methodology. This evaluation was performed to determine whether MODDALS enables to classify the domain knowledge by taking as reference existing ontologies and to validate the first hypothesis (H1) of the thesis: *ontology and SPL design techniques applied in combination enable to classify the domain knowledge into different abstraction layers with high level of consensus taking as reference existing ontologies*.

This chapter addresses the second thesis objective (O2) defined in Chapter 1: *validate the methodology by designing the layered structure of a global energy ontology*.

The chapter is organized as follows. Section 6.2 shows how MODDALS was

[1] http://www.purl.org/dabgeo

applied to design the layered ontology structure of DABGEO. Section 6.3 presents an empirical evaluation of MODDALS. Section 6.4 summarizes the main conclusions extracted from the application of MODDALS and its evaluation results.

## 6.2 Application of MODDALS to Design DABGEO Layered Structure

The next subsections show how MODDALS steps (presented in Chapter 4) were conducted to design DABGEO structure. DABGEO design and development team included people with expertise in energy domain data representation and ontology engineers. The DABGEO ontology is quite extensive (it includes 97 ontology modules), so the next subsections explain through examples how certain parts of the layered ontology structure was designed. The same process was followed to design the rest of the layered ontology structure.

### 6.2.1 Preliminary Step: Analysis and Classification of Existing Energy Ontologies

Domain experts classified the existing energy ontologies according to the energy management application types they support. This step corresponds to the *preliminary step (analysis and classification of existing ontologies)* of MODDALS presented in Section 4.3.1.

First, a state of the art of the existing energy ontologies and the energy sustainability and resilience applications they support was conducted. The conducted state of the art was presented in the previous Chapter (Section 5.4). Domain experts analysed the objectives of the identified energy ontologies and energy management solutions. The applications with similar objectives and tasks were grouped into application types.

The existing energy ontologies provide support to a wide variety of Smart Grid energy management applications. The reviewed applications are deployed in specific Smart Grid scenarios (infrastructures of the Smart Grid such as smart homes or buildings). Therefore, the Smart Grid energy management applications can be classified into the different types depending on the Smart Grid scenarios where they are deployed. We define these application types as *Smart Grid scenarios* [32]:

- **Smart home energy management applications:** they are focused on controlling and monitoring home device energy operation. With this information, these applications provide home users a complete energy performance assessment and give them advice to reduce the home energy consumption and its ecological energy impact.

- **Building/district/city energy management applications**: they are focused on giving a complete energy performance assessment about the energy usage and the main energy performance indicators of buildings and districts.
- **Organization energy management applications**: they are focused on providing a holistic view of organization energy performance assessment and suggesting energy reduction measures.
- **Microgrid energy management applications**: they are focused on improving the efficiency and flexibility of microgrids, which aim to improve the current grid efficiency and flexibility by integrating distributed energy generation, energy storage systems and loads.
- **Smart Grid DR management applications**: they are focused on managing the energy consumption of infrastructures in response to the current energy supply conditions.

Each Smart Grid scenario encompasses more specific application types, since inside each scenario there are applications that have common objectives [32]. For instance, within smart home energy management, there are applications focused on home energy assessment and device control (i.e., [92]), home energy saving advice (i.e., [17, 56]) and home appliances DR management (i.e., [44]). Below we enumerate the specific application types encompassed by each Smart Grid scenario:

- **Smart home energy management applications:** home energy assessment and device control applications, home energy saving advice applications and home appliances DR management applications.
- **Building/district/city energy management applications:** building automation systems integration applications, city energy performance assessment applications and building energy saving advice applications.
- **Organization energy management applications:** organization energy saving advice applications and organization energy assessment applications.
- **Microgrid energy management applications:** include microgrid multi-objective energy management applications.

- **Smart Grid DR management applications:** Smart Grid energy market coordination process applications and data driven DR applications.

Table 6.1, shows in which of the aforementioned application types were classified the reviewed energy management solutions.

Finally, among the reviewed solutions, the available ontologies were selected and classified according to the application type they support depending on the applications were they are reused. As a result of this selection process, Table 6.2 shows the selected energy ontologies and Table 6.3 shows how they were classified into energy management application types. It is worth mentioning that there are not available energy ontologies for all the application types enumerated in Table 6.1, i.e., the ontologies developed to provide support to microgrid energy management applications. Hence, these application types were not included in the final ontology classification shown in Table 6.3.

| **Smart Grid scenario** | **Application type** | **Energy management solutions** |
|---|---|---|
| Smart home energy management | Home Energy assessment and device control | Kofler et al. [92] |
| | Home energy saving advice | Burel et al. [17]<br>Fensel et al. [56] |
| | Home appliances Demand Response management | Daniele et al. [44] |
| Building/district/city energy management | Building automation systems integration | Fernbach et al. [59]<br>Tomasevic et al. [155] |
| | Building energy performance assessment | Hu et al. [79]<br>Niknam and Karshenas [116]<br>Pont et al. [128] |
| | City energy performance assessment applications | Corrado et al. [26]<br>Hippolyte et al. [77] |
| | Building energy saving advice | Yuce and Rezgui [170]<br>Stavropoulos et al. [148] |
| Organization energy management | Organization energy saving advice | Blomqvist and Thollander [9] |
| | Organization energy saving advice | Curry et al. [37] |
| Microgrid energy management | Microgrid multi-objective energy management | Salameh et al. [137]<br>Shi et al. [141]<br>Maffei et al. [102] |
| Smart Grid Demand Response management | Smart Grid energy market coordination process applications | Hippolyte et al. [76]<br>Zhang et al. [172]<br>Gillani et al. [63] |
| | Data driven Demand Response applications | Gillani et al. [62]<br>Zhou et al. [174] |

**Table 6.1:** Classification of knowledge-based energy management solutions into application types

| Name | Main purpose | RDF/XML URI reference |
|---|---|---|
| ThinkHome ontology | Representation of smart home energy data | https://www.auto.tuwien.ac.at/ downloads/thinkhome/ontology/ |
| DEFRAM project ontology | Representation of organization energy audits and investment cost of energy saving actions | http://www.ida.liu.se/projects/semtech/ schemas/energy/2013/09/efficiency.owl |
| SAREF4EE ontology | To improve interoperability among electrical appliances of different manufacturers | http://ontology.tno.nl/saref4ee/ |
| BOnSAI ontology | Representation of building energy performance | http://lpis.csd.auth.gr/ontologies/ bonsai/BOnSAI.owl |
| EnergyUse ontology | Representation of home energy consumption data and discussions between home users | http://eelst.cs.unibo.it/apps/LODE/source? url=http://socsem.open.ac.uk/ontologies/eu |
| ProSGV3 ontology | Representation of prosumer oriented Smart Grid data | http://data-satin.telecom-st-etienne.fr/ontologies/ smartgrids/proSGV3/ProSG.html |
| LCC ontology | Representation of building energy consumption data | http://smartcity.linkeddata.es/lcc/lcc-dataset.ttl |
| Mirabel ontology | Representation of devices energy flexibility and user preferences | https://sites.google.com/site/smartappliancesproject/ ontologies/mirabel.ttl |
| DERI Linked Dataspace | Representation of organization energy performance data | http://vocab.deri.ie/ |
| SEMANCO ontology | Representation of urban area energy performance | http://semanco-tools.eu/ontology-releases/eu/ semanco/ontology/SEMANCO/SEMANCO.owl |

**Table 6.2:** Selected energy ontologies

| **Smart Grid scenario** | **Application type** | **Ontology** |
|---|---|---|
| Smart home energy management | Home energy assessment and device control | ThinkHome ontology |
| | Home energy saving advice | EnergyUse ontology |
| | Home appliances Demand Response management | SAREF4EE ontology<br>Mirabel ontology |
| Building/district/city energy management | Building automation systems integration | LCC ontology |
| | City energy performance assessment | SEMANCO ontology |
| | Building energy saving advice | BonSAI ontology |
| Organization energy management | Organization energy saving advice | DEFRAM project ontology |
| | Organization energy assessment | DERI Linked dataspace |
| Smart Grid Demand Response management | Data driven Demand Response | ProSGv3 ontology |

**Table 6.3:** Classification of energy ontologies into energy management application types

### 6.2.2 Step 1: Definition of DABGEO Ontology Layers

The layered ontology structure was defined for DABGEO. This step corresponds to the *definition of ontology layers step* (Step 1) of MODDALS presented in Section 4.3.2. The layered ontology structure was defined by the domain experts taking as reference the ontology classification obtained in the preliminary step and the layered ontology structure proposed in Step 1 of MODDALS.

The layered structure defined for DABGEO includes three layers (Fig. 6.1). The *common-domain layer* represents the top-level knowledge of energy domains and the knowledge common to Smart Grid scenarios. Variant domain knowledge still common to more than one Smart Grid scenario is included in the *variant-domain layer*. The *domain-task layer* includes the knowledge reused in specific Smart Grid scenarios. As explained in Section 6.2.1, the energy management application types that correspond to each Smart Grid scenario can be classified into more specific application types. Hence, the *domain-task layer* is divided into two sublayers: the *scenario sublayer* and the *application type* sublayer. The former represents the knowledge relevant to a certain Smart Grid scenario and the later represents the knowledge reused only by certain energy management application types of a Smart Grid scenario. The domain experts named each sublayer to facilitate the distinction between both sublayers. These layers and the rest of DABGEO structure are described in more detail in Chapter 7.

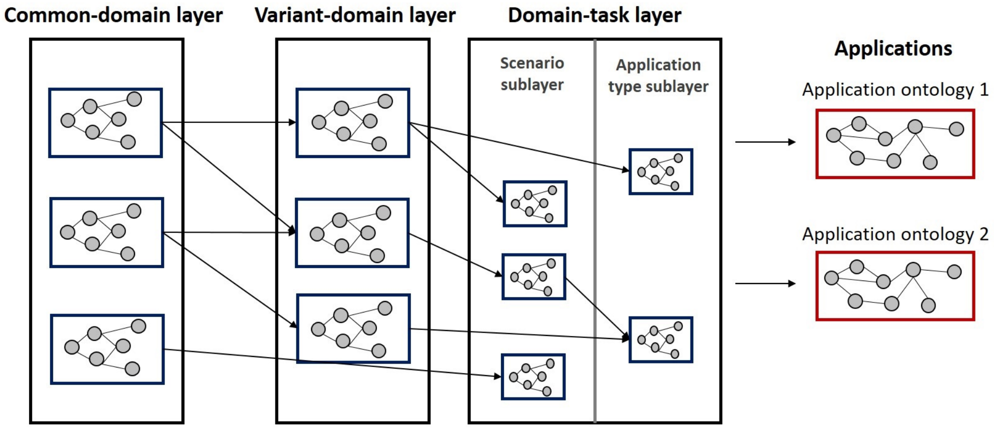


**Figure 6.1:** DABGEO ontology layers

### 6.2.3 Step 2: DABGEO Knowledge Hierarchy Creation

In this step, the domain knowledge hierarchy of DABGEO was defined.

In this step, the domain knowledge hierarchy of DABGEO was defined. This step corresponds to the *domain knowledge hierarchy creation step* (Step 2) of MODDALS presented in Section 4.3.3.

Figure 6.2 shows part of the knowledge hierarchy of DABGEO, corresponding to the *energy equipment domain*. It is worth mentioning that the whole hierarchy includes 5 domains, 15 subdomains and 106 KAs.

Since the DABGEO domain knowledge was classified based on a domain analysis of existing energy ontologies, the knowledge hierarchy includes the knowledge represented by existing energy ontologies. The domain experts and ontology engineers collaborated to perform a manual analysis of ontology elements in Protégé to identify the domains they represent and to divide them into KAs.

Below we describe how Step 2 activities were conducted to define the part of the knowledge hierarchy shown in Figure 6.2.

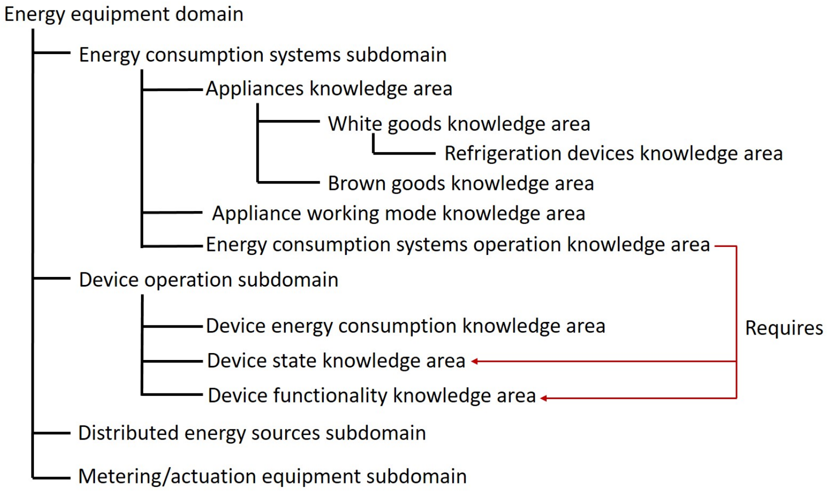

Energy equipment domain
Energy consumption systems subdomain
Appliances knowledge area
White goods knowledge area
Refrigeration devices knowledge area
Brown goods knowledge area
Appliance working mode knowledge area
Energy consumption systems operation knowledge area
Device operation subdomain
Device energy consumption knowledge area
Device state knowledge area
Device functionality knowledge area
Requires
Distributed energy sources subdomain
Metering/actuation equipment subdomain

**Figure 6.2:** Part of DABGEO knowledge hierarchy for the energy equipment domain

1. **Domain/subdomain definition:** in this activity, domain experts and ontology engineers analysed the knowledge represented by existing energy ontologies to identify the domains they represent:
   - *Energy equipment domain*: the features and operation data about energy consumption production and storage devices.
   - *Infrastructure domain*: data on structural features and environmental conditions of infrastructures such as homes or buildings.
   - *Energy performance domain*: data on energy performance values and indicators such as energy consumption or production.

- *Energy external factors domain*: data on factors that may hinder the energy performance such as weather or environmental conditions.
- *Smart Grid stakeholders domain*: data on the actors that participate in the energy market such as energy consumers and producers.

In addition, the root concepts of each domain were defined by domain experts. For instance, *device* was defined as the root concept of the *energy equipment domain* because this concept is extended by the rest of the data (device types, device operation data) included in the domain.

These domains were divided into subdomains by domain experts because they are extensive. For instance, many concepts are needed to describe the whole *energy equipment domain*, since this domain encompasses data about many device types and their operational aspects. This domain was divided into four subdomains (see Figure 6.2):

- *Energy consumption systems subdomain*: data about energy consumption devices such as home appliances, building automation equipment, industrial equipment, HVAC systems.
- *Device operation data subdomain*: device operational aspects such as device power profile, device state, device energy performance or device flexibility operations.
- *Distributed energy sources subdomain*: renewable and non-renewable energy sources data (i.e., photovoltaics, electric vehicles, fossil fuels, home power plants) and ESSs such as batteries, capacitors or energy carriers.
- *Metering/actuation equipment subdomain*: energy, weather and environmental conditions measurement sensors (i.e., humidity sensor, CO2 sensor, boiler controller) and equipment actuators, i.e., lighting control, pump controller.

In Chapter 7, where DABGEO structure is described, we enumerate the subdomains into which the rest of domains represented by DABGEO were divided.

2. **Knowledge area definition:** in this activity, the ontology engineers (in collaboration with domain experts) analysed the existing energy ontologies to identify the CQs they answered. The CQs were taken as reference to divide

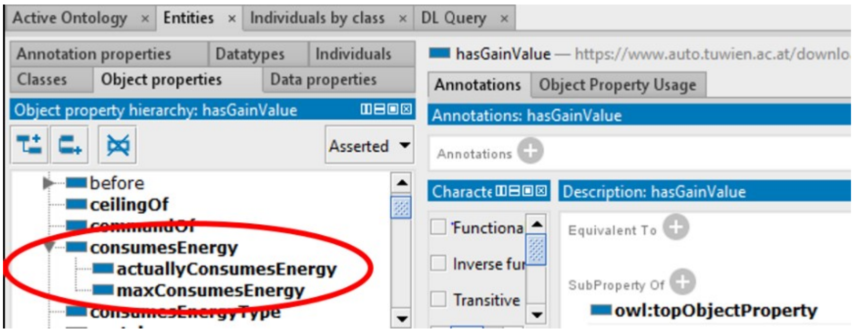
the knowledge of existing energy ontologies into KAs. In total, 10 energy ontologies were analysed, which are listed in Table 6.2.

Below we explain how the sub-activities of the knowledge area definition and classification activity were conducted to define some sample KAs within the *energy consumption systems* and *device operation* subdomains.

**2.1. Class hierarchy-based KA definition:** firstly, the class hierarchies of the energy ontologies were analysed by ontology engineers to identify the CQs. Regarding energy consumption systems data, the energy ontologies represent the *Appliance class* and more specific appliances as subclasses of this class. Therefore, one of the CQs answered by the class hierarchies is *What type of appliances are there?* Hence, the *appliance KA* was defined, which encompasses the *appliance* concept. Among the analysed ontologies, ThinkHome is the one that classifies appliances with more granularity. Thus, the class hierarchy of this ontology was taken as reference to define the appliance KA and its sub-KAs. ThinkHome classifies the Appliance class into subclasses that represent specific appliance types such as *Brown goods* and *White goods*, which, in turn, encompass subclasses that represent specific white and brown good types. The class hierarchy was populated with specific classes from other ontologies such as classes that represent specific white goods (i.e., Refrigeration devices). Each of these classes were defined as KAs (see Figure 6.2). In addition, each KA of each class was defined as a sub-KA of the corresponding superclass.

Regarding the device operation data, the existing energy ontologies answer the following CQs: *What are the device functionality types, What are the device state types?*. Hence, the *device functionality* and *device state* KAs were defined.

**2.2. Ontology elements relation-based KA definition:** the remaining KAs were defined after identifying the CQs answered by a set of interrelated elements of existing energy ontologies. As an example, Figures 6.3 and 6.4 show a set of ontology elements of ThinkHome and EnergyUse ontologies respectively within a Protégé screenshot. As marked (in red) in Figure 6.3, the ThinkHome ontology includes the *consumesEnergy*, *actuallyConsumesEnergy* and *maxConsumesEnergy* properties. These properties describe the energy consumption, actual energy consumption and maximum energy consumption of a certain device respectively. Hence, the ThinkHome ontology answers the following CQs: *What is the energy consumption of a device?*, *How much energy is a device consuming?* and *What is the maximum energy consumption of a device?* On the other hand, as shown in Figure 6.4, the EnergyUse ontology includes the *hasConsumption* property to answer the *What is the energy consumption of a device?* CQ. All these CQs describe energy consumption of devices, so they were grouped by the domain experts into the *device energy consumption KA* (which also includes CQs answered by other energy ontologies). This KA encompasses the knowledge that answers the aforementioned CQs. In the same way, the energy consumption systems operation and appliance working mode KAs were defined. These KAs encompass the knowledge about operational aspects of specific energy consumption systems and appliance working modes respectively.

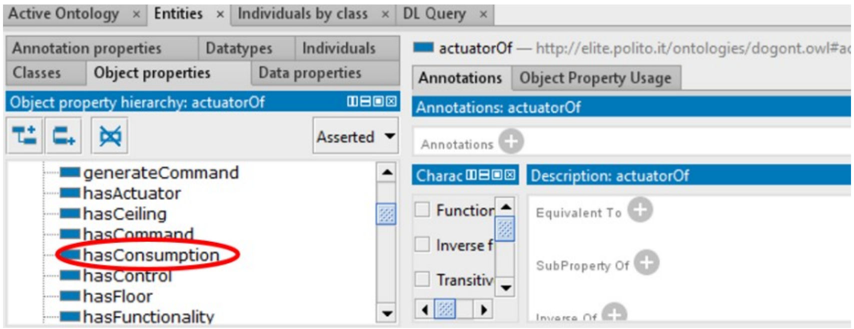


before
ceilingOf
commandOf
consumesEnergy
actuallyConsumesEnergy
maxConsumesEnergy
consumesEnergyType

hasGainValue — https://www.auto.tuwien.ac.at/downlo

owl:topObjectProperty

**Figure 6.3:** Ontology elements of the ThinkHome ontology



generateCommand
hasActuator
hasCeiling
hasCommand
hasConsumption
hasControl
hasFloor
hasFunctionality

actuatorOf — http://elite.polito.it/ontologies/dogont.owl#ac

**Figure 6.4:** Ontology elements of the EnergyUse ontology

3. **Knowledge hierarchy refinement:** in this activity, the KAs were placed into a knowledge hierarchy level according to the knowledge they represent and extend, thus completing the knowledge hierarchy. Figure 6.2 shows in

which subdomain and hierarchy level was placed each KA introduced in previous examples. In addition, the KA dependencies were also defined. For instance, the *energy consumption systems operation* KA describes specific states and functionalities of energy consumption systems and encompasses CQs such as *What is the minimum number of states an air condition system has?* and *Do ventilating systems have any notification functionality?*. Therefore, this KA requires the knowledge of *device state* and *device functionality* KAs, which include knowledge about possible device states and functionalities respectively. Finally, the domain experts provided a complete description of each KA and the knowledge/CQs it encompasses in the KADD document. As an example, Table 6.4 shows the KADD document that includes some the KAs shown in Figure 6.2.

### 6.2.4 Step 3: DABGEO Knowledge Classification

A domain analysis of existing energy ontologies was conducted by the ontology engineers to classify the defined KAs into each layer. This step corresponds to the *knowledge classification step* (Step 3) of MODDALS presented in Section 4.3.4.

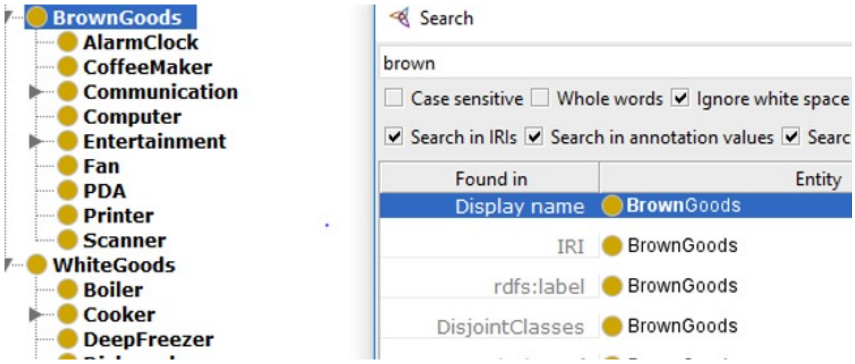
Firstly, the domain experts identified the KAs that represent relevant knowledge for the domain and future applications. These KAs were considered as common and placed in the *common-domain layer* regardless of their presentence on existing ontologies.

Then, the rest of the activities of MODDALS knowledge classification step were conducted.

1. **Analysis of existing ontologies:** existing energy ontologies were manually analysed with Protégé to determine if they represented the KAs of energy domains. Specifically, tools available in this editor were used to find the KA key words (extracted from the KA description provided by the domain expert) in the ontology elements. If the ontology contained necessary elements or statements to answer the CQs encompassed by the KA, the KA was considered as represented by the ontology. As an example, Figure 6.5 shows a screenshot of a set of ThinkHome ontology classes that represent specific brown goods (i.e., alarm clock, entertainment equipment). Therefore, the ontology answers the CQ *what types of brown goods are there?*, which is encompassed by the *brown goods KA*. Taking this into account, we considered that the ThinkHome ontology represents this KA.

| Knowledge area | Competency Questions/description |
|---|---|
| Appliances | • **Description:** it represents data about different types of (home) appliances (white goods, brown goods). We consider that this knowledge area is represented by an ontology if data about any appliance is represented or of there is a class that explicitly represents ‘Appliances’. This knowledge area encompasses the following sub-knowledge areas: brown goods and white goods.<br>• **Competency Questions:**<br>1. What types of appliances/electrical appliances are there? |
| Brown goods | • **Description:** it represents data about any small appliance such as coffee makers, office and entertainment equipment or multimedia devices. We consider that this knowledge area is represented by an ontology if any of these devices are represented or of there is a class that explicitly represents ‘brown goods’. This knowledge area encompasses the following sub-knowledge areas: IT equipment and entertainment equipment.<br>• **Competency Questions:**<br>1. What types of brown goods are there? |
| White goods | • **Description:** it represents data about any large electrical goods used domestically such as refrigerators and washing machines, typically white in colour. We consider that this knowledge area is represented by an ontology if any of these devices are represented or of there is a class that explicitly represents ‘white goods’. This knowledge area encompasses the following sub-knowledge areas: cooking devices, cleaning devices and refrigeration devices.<br>• **Competency Questions:**<br>1. What types of white goods are there? |
| Device energy consumption | • **Description:** it represents data about devices energy consumption, i.e., the consumption amount in a certain period of time.<br>• **Competency Questions:**<br>1. How much energy is consumed by a certain device?<br>2. How much energy is consumed by an energy consumer facility at a certain point in time?<br>3. How much energy is maximally consumed by a certain energy consumer facility in a specific state?<br>4. What is the energy consumption summary of a certain appliance?<br>5. What is the energy consumption statistic of a certain energy consumption summary?<br>6. What is the typical energy/power consumption of a certain device?<br>7. What types of electrical appliance consumption categories are there?<br>8. What is the consumption class of a certain electrical appliance? |

**Table 6.4:** KADD of the energy equipment domain



brown

☐ Case sensitive ☐ Whole words ☑ Ignore white space

☑ Search in IRIs ☑ Search in annotation values ☑ Searc

| Found in | Entity |
|---|---|
| Display name | BrownGoods |
| IRI | BrownGoods |
| rdfs:label | BrownGoods |
| DisjointClasses | BrownGoods |

**Figure 6.5:** Representation of the brown goods KA by ThinkHome ontology

The *brown goods KA* is an intuitive example that requires only the analysis of certain classes to determine whether the KA is represented. However, other KAs required a more exhaustive analysis, since they were represented by more specific classes and relations. Taking as an example the device energy consumption KA (described at the end of Section 6.2.3), only certain properties were applied to relate device operational aspects with specific energy consumption systems. Hence, a more exhaustive analysis of energy ontologies was performed to see whether they represent this KA.

2. **Commonality and Variability Analysis:** a CVA was conducted to classify the KAs of each energy subdomain. An application-knowledge matrix of each energy subdomain was created to determine which Smart Grid scenarios reuse each subdomain KA, taking as reference the representation of these KAs by existing energy ontologies. As an example, Table 6.5 shows the application-knowledge matrix of some KAs of the *energy consumption systems subdomain* (included in the knowledge hierarchy of Figure 6.1). The left column includes the KAs, while the top row includes the Smart Grid scenarios and the ontologies that provide support to the applications deployed in these scenarios. To simplify the table, we omitted several ontologies.

   There are currently four Smart Grid scenarios for which ontologies were developed [32]. We considered as common the KAs reused in most of scenario: at least three. Hence, 75% was used as the threshold value to classify the KAs as common or variant depending on their CV ratio.

3. **Knowledge area layer assignment:** the KAs were classified into different layers according to the CVA results. For instance, the *appliances*, *brown goods* and *white goods* KAs were classified into the *common-domain layer*, since their CV ratio was equal of above 75%. The *refrigeration devices KA* was placed in

| | **Smart Grid scenarios** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | **Smart home energy management** | | | | **Building/district/city energy management** | **Organisation energy management applications** | | **Smart Grid DR management** | |
| | ThinkHome ontology | EnergyUse ontology | SAREF4EE ontology | Mirabel ontology | SEMANCO ontology | DEFRAM project ontology | DERI Linked dataspace | ProSGV3 ontology | Commonality ratio |
| Appliances | X | X | X | X | X | X | X | X | 100% |
| Brown goods | X | X | - | - | - | - | X | X | 75% |
| White goods | X | X | X | - | X | - | - | X | 75% |
| Refrigeration devices | X | X | - | - | - | - | - | X | 50% |
| Energy consumption systems operation | X | X | - | - | - | - | - | - | 25% |
| Appliance working mode | X | - | - | - | - | - | - | - | 25% |

**Table 6.5:** Application-knowledge matrix of the energy consumption systems subdomain

the *variant-domain layer*, since it was common to more than one Smart Grid scenario although its CV ratio was below 75%.

4. **CVA at the application type level:** the KAs reused only one Smart Grid scenario were classified into the sublayers of the *domain-task layer* according to the CVA at the application type level. Following the sample CVA shown in Table 6.5, the *energy consumption systems operation* and the *appliance working mode* KAs were included in this domain analysis, since they were only represented by ontologies from smart home energy management applications. This low representation is because these KAs encompass the knowledge that answers very specific CQs that only ontologies reused in smart home energy management applications must answer. The domain analysis at the application type level for these KAs is shown in Table 6.6. The *energy consumption systems operation KA* was reused by more than one smart home energy management application type (home energy assessment and home energy saving advice applications), so it was placed in the *scenario sublayer*. The *appliance working mode KA* was reused only by one smart home energy management application type (home appliances DR management), so it was placed in the *application type sublayer*.

| | **Smart home energy management** | | | |
|---|---|---|---|---|
| | **Home energy assessment** | **Home energy saving advice** | **Home appliances DR management** | |
| Ontologies<br>Knowledge areas | ThinkHome ontology | EnergyUse ontology | SAREF4EE ontology | Mirabel ontology |
| Energy consumption systems operation | X | X | - | - |
| Appliance working mode | - | - | X | - |

**Table 6.6:** CVA at application level of energy consumption systems subdomain

Finally, ontology engineers wrote the list of the KAs of each layer/sublayer following the template proposed in Table 4.5. As an example, Table 6.7 shows the classification of the sample KAs included in the CVAs of Table 6.5 and Table 6.6.

| Common-domain layer | Variant-domain layer | Domain-task layer |
|---|---|---|
| Appliances<br>Brown goods<br>White goods | Refrigeration devices | **Scenario sublayer** |
| | | Energy consumptions systems operation |
| | | **Application type sublayer** |
| | | Appliance working mode |

**Table 6.7:** Classification of energy KAs

### 6.2.5 Step 4: Structuring of DABGEO Layer Knowledge

Finally, the knowledge of each layer was structured into ontology modules by the ontology engineers, thus completing the design of DABGEO layered ontology structure. This step corresponds to the *layer knowledge structuring step* (Step 4) of MODDALS presented in Section 4.3.5.

Ontology engineers conducted the activities of this step (ontology modularization and inclusion hierarchy definition) to structure the knowledge of each layer into ontology modules. Then, they wrote informal model of DABGEO that includes the list of ontology modules of each layer and the high-level relations between modules following the template proposed in Table 4.6.

As an example, Table 6.8 shows part of the informal model of DABGEO corresponding to the *energy consumption systems subdomain*. In particular, it includes the ontology modules the represent the knowledge of the KAs included in the sample domain analysis shown in Section 6.2.4.

Below we detail how the activities of this step were carried out to create the sample informal model of DABGEO shown in Table 6.8.

1. **Ontology modularization:** in Step 2, device was defined as the top-level concept of the *energy equipment domain* and, by extension of the *energy consumption systems subdomain* (see Section 6.2.3). Hence, the *Device ontology module* was defined, which represents the *Device* top-level concept and device main properties, i.e., device name. In addition, all the common KAs (i.e. *appliances*, *white goods* KAs) of this subdomain were grouped into the *energy consumption systems ontology module*, which includes all the knowledge they encompass. Both ontology modules are placed in the *common-domain layer* (see Table 6.8). In the same way, the common KAs o the rest of subdomains were grouped into an ontology module. Then, one ontology module was defined for each variant KA (i.e., *refrigeration devices ontology*), and these modules were classified into lower-level layers according to the results of Step 3 (see Section 6.2.4). Within the *scenario* and *application type* sublayers, the ontology modules were classified depending on the Smart Grid scenario

or the specific energy management application type where the KAs they represent are reused (see Table 6.8).

2. **Inclusion hierarchy definition:** the defined ontology modules were organised into an inclusion hierarchy that establishes the high-level relations between the ontology modules. The inclusion hierarchy was defined based on the knowledge that the ontology modules extend or require (taking as reference the knowledge hierarchy defined in Step 2). For instance, the Device ontology is included by the *energy consumption systems ontology*, which in turn is included by a set of ontology modules from lower-level layers. Additionally, some modules from the *energy consumption systems subdomain* (*energy consumption systems operation ontology*) include modules from other subdomains (*device state ontology* and *device functionality ontology*), since they require that knowledge (see Table 6.8).

| | | | |
|---|---|---|---|
| **Common-domain layer** | | | • Device ontology<br>• Energy consumption systems ontology (*includes*: device ontology) |
| **Variant-domain layer** | | | • Refrigeration devices ontology (*includes*: energy consumption systems ontology) |
| **Domain-task layer** | **Scenario sublayer** | **Smart home energy management** | • Energy consumption systems operation ontology (*includes*: energy consumption systems ontology, device state ontology and device functionality ontology) |
| | **Application type sublayer** | **Home appliances Demand Response management** | • Appliance working mode ontology (*includes*: energy consumption systems ontology) |

**Table 6.8:** Ontology modular structure of energy consumption systems subdomain

## 6.3 Evaluation of MODDALS

As stated by De Hoog [48], "*it is extremely difficult to judge the value of a methodology in an objective way*". It is unlikely that anyone will be willing to pay twice for building or designing the same extended ontology using different approaches. Hence, the evaluation of previous ontology development and design methodologies consisted on showing the experiences of applying the methodology in one or more use cases [150, 88]. Considering this, we report in this section how we performed a first evaluation of the MODDALS methodology.

The main objective of MODDALS is to enable to classify the domain knowledge by taking as reference existing ontologies. Hence, the evaluation has focused on determining if MODDALS enables this classification. To demonstrate this aspect, we checked whether MODDALS steps can be correctly followed by different domain experts and ontology engineers. We consider that MODDALS steps can be followed correctly if different domain experts and ontology engineers are able to obtain similar knowledge classifications performing a domain analysis of existing ontologies. Therefore, the evaluation of MODDALS has focused on answering the following RQ:

- Can MODDALS be applied by different domain experts and ontology engineers with similar knowledge classification results?

To answer this question, MODDALS was applied by different energy domain experts and ontology engineers to design part of the layered structure of DABGEO. A group of two domain experts and ontology engineers conducted Steps 1 and 2, while the ontology engineers (eight in total) conducted Steps 3 and 4 with the collaboration of the experts. Each ontology engineer performed Steps 3 and 4 individually in a blind process. However, they could contact the domain experts for any clarification or additional explanation about the defined KAs. The knowledge classifications obtained by each engineer are analysed to check if they are similar in Section 6.3.1.

Regarding the background knowledge of the MODDALS evaluation participants, the domain experts had experience at energy data representation. In particular, they participated in projects related with interoperability within energy management applications. In addition, they had previously developed ontologies. Regarding the eight engineers who conducted Steps 3 and 4, they are specialized in software engineering and data analysis. In addition, 62% of engineers had developed an ontology before conducting the experiment.

Finally, to get the experiences of the domain experts and ontology engineers on applying MODDALS, we performed a survey, which is a well-known method for evaluating methodologies [123, 152]. The survey includes a questionnaire that the participants in the MODDALS evaluation answered to (1) identify MODDALS main benefits and drawbacks, (2) identify future lines of research to improve the methodology and (3) determine whether it is ready to be applied in other domains apart from the Energy. In Section 6.3.2, we show the responses to the questionnaire.

### 6.3.1 MODDALS Application Results

In this section, we first show the energy knowledge classification obtained by different ontology engineers after applying MODDALS to design part of DABGEO layered structure. To compare the knowledge classifications and analyse whether they are similar, we analysed the number of modules defined by each engineer in each layer. However, although the number of modules is the same, they may contain different knowledge. Hence, the *degree of consensus* with which the ontology engineers classified the KAs into different layers was also analysed. The degree of consensus of a KA is the percentage of ontology engineers who classified the KA into the same layer.

Figure 6.6 shows how many modules were defined by each engineer in each layer of DABGEO. Figure 6.6 also shows the number of modules of the *domain-task layer* that were classified into each energy management application type. It is worth mentioning that the *domain-task layer* did not include any sublayer, since the designed part of the ontology structure was limited to support three application types: home energy saving advice, home appliances DR management and Smart Grid DR management applications.

In general, the number of modules defined by each ontology engineer was similar in all layers. This similarity is due to the high degree of consensus with which the ontology engineers classified the KAs into different layers. Within the conducted evaluation, the average degree of consensus of all the KAs classified by the ontology engineers was 76%. It is worth mentioning that from the sixth ontology engineer that applied MODDALS onwards, the average degree of consensus remained stable in 76%. Considering these results, the domain experts and different ontology engineers were able to follow MODDALS steps to obtain a similar ontology design.

Most of the KAs (specifically 80%) whose degree of consensus was above the average (76%) were classified into the *common-domain* and *variant-domain* layers. As an example, some of these KAs, as well as their degree of consensus and the layer/application type where these KAs were placed, are shown in Figure 6.7. There-

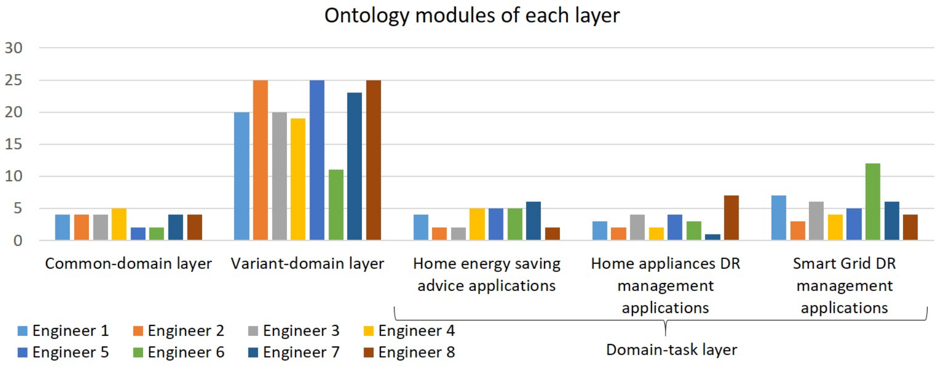


**Figure 6.6:** Ontology modules of each layer

fore, we can conclude that there was a high consensus when separating the common domain knowledge from the variant knowledge reused by specific application types.

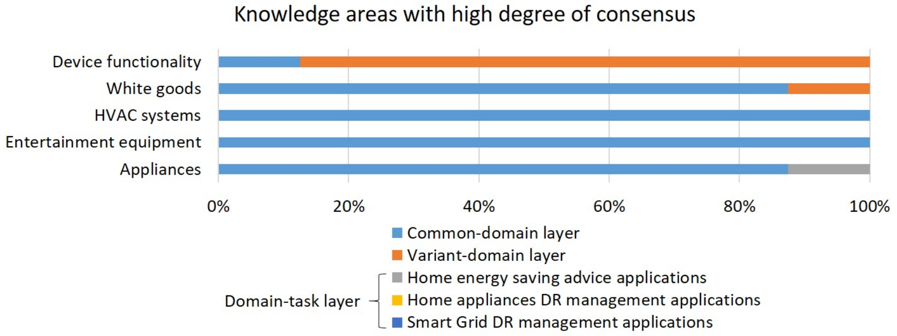


**Figure 6.7:** KAs with high degree of consensus

Although ontology engineers could contact the domain experts for any clarification about the knowledge the KAs encompass, each ontology engineer had their own interpretation about the knowledge represented by existing ontologies. Thus, the degree of consensus of some KAs was lower (some examples are shown Figure 6.8). This aspect constitutes one of the drawbacks of MODDALS, as we discuss later in Section 6.3.2. A significant part (62%) of the KAs with low degree of consensus are child KAs of KAs whose degree of consensus is above the average (76%). Therefore, most of the differences in the classification of knowledge occurred in KAs that represent very specific knowledge, without affecting the rest of the classification.

Considering these results, domain experts and ontology engineers could follow MODDALS steps to obtain similar knowledge classifications. This classification was performed based on a domain analysis of existing ontologies, which complemented domain experts and ontology engineers' experience. Therefore, **we can state that MODDALS can be applied by different domain experts and ontology engineers**

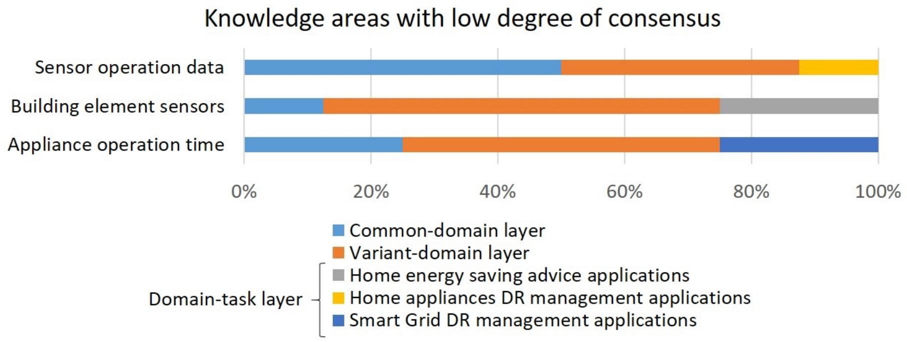


**Figure 6.8:** KAs with low degree of consensus

**with similar knowledge classification results, enabling to classify the domain knowledge by taking as reference existing ontologies.**

### 6.3.2 MODDALS Feedback

This section explains the responses of MODDALS evaluation participants to the questionnaire we provided for feedback on the methodology. The questionnaire included the following questions:

- What are the positive aspects of MODDALS?
- What are the disadvantages of MODDALS?
- What are your suggestions for improving MODDALS?
- Would you apply the MODDALS methodology again and recommend it to other developers to design the layered structure of ontologies in other domains (Yes/No)?

We received 8 responses from participants involved in MODDALS evaluation. According to the survey respondents, the main benefits of MODDALS are the following:

1. Due to the domain analysis of existing ontologies, MODDALS provides a detailed classification of the knowledge reused by specific application types, while keeping separate the knowledge relevant to many applications. Some of the comments of survey respondents about this benefit were: "*Common-domain layer starts with very general ideas and then it goes to more specific concepts in the next layers*"; "*It gives clear steps for determining which knowledge areas are common to existent ontologies and which knowledge areas are specific to certain*

*ontologies/applications"*; *"designed ontologies are likely to provide a balance between reusability and usability"*; *"It is very useful to compare different ontologies and identify which aspects are common on them"*.

2. MODDALS is easy to follow and provides clear and mechanical steps. Some of the comments of survey respondents about this benefit were: *"It gives clear steps (mostly mechanical)"*; *"It is a simple process"*; *"easy approach"*.

3. MODDALS provides a method to improve the reuse of already developed knowledge to enable the development of interoperable ontologies. Some of the comments of survey respondents about this benefit were: *"It seems a good method for refactoring already available ontologies without discarding what it has been applied in the domain and enhancing interoperability"*.

On the other hand, the following are the main disadvantages of MODDALS according to the survey respondents:

1. Although it prevents domain experts and ontology engineers from designing the ontology structure from scratch, MODDALS still requires a significant manual ontology analysis effort to check if each KA is represented by existing ontologies. In particular, MODDALS evaluation participants checked whether 47 KAs (and the CQs they encompass) were represented in each ontology. Some of the comments of survey respondents about this disadvantage were: *"It requires much time to perform the domain analysis of existing ontologies"* [sic]; *"identifying if the knowledge area is represented in the ontology is not always straight forward for the ontology engineer"*; *"time-consuming process"*

2. The classification of some KAs was mainly subject to ontology engineers' interpretation of the KA description provided by domain experts and the analysed knowledge from existing ontologies. On the one hand, some of the KA descriptions were open to multiple interpretations. In addition, a manual analysis of ontology engineers may not be sufficient to detect whether certain KAs are represented, since part of the ontology knowledge may be implicit. Therefore, part of the domain knowledge classification is quite subjective, which may influence the final design of the ontology. Some of the comments of survey respondents about this disadvantage were: *"it depends on how well the knowledge area is described by the domain expert and how well documented is the ontology"*; *"step three of the methodology might create a bit of ambiguity"*; *"implied relationships might exist in an ontology, and the end result might not have taken this into account"*; *"analysing the ontologies can be subjective if the sub models are*

*not well defined*". This disadvantage is clearly reflected in the results shown in Figure 6.7 (Section 6.3.1).

3. Although it enables the design of maintainable ontology structures, MODDALS does not provide guidelines to extend the ontology structure and reclassify the knowledge when new ontologies and applications arise. Some of the comments of survey respondents about this disadvantage were: "*MODDALS guidelines are limited to design the first version of the ontology*".

In relation to these disadvantages, the questionnaire respondents provided the following suggestions for improving MODDALS:

1. The main improvement aspect is to define automatic tools to conduct the knowledge classification step to reduce the effort of applying the methodology and to detect all the implicit ontology knowledge.
2. Make domain experts a more active part in Step 3 to reduce different interpretations of the analysed ontology knowledge when classifying them into different layers.
3. Step 3 should define more detailed guidelines that explain how to decide whether a KA is represented by an ontology to minimise possible interpretations of ontology engineers when classifying KAs into different layers.

Finally, 80% of respondents would recommend the application of MODDALS to design the layered structure of reusable and usable ontologies in other domains. Therefore, we consider that MODDALS has the potential to be applied in more domains apart from the Energy.

## 6.4 Conclusions

In this chapter, the MODDALS methodology was applied by an ontology design team that included domain experts and ontology engineers to design the layered structure of DABGEO, a global ontology for the energy domain. In that way, we illustrated how this methodology is applied in a real use case.

MODDALS was evaluated to determine whether it enables to classify the domain knowledge by taking as reference existing ontologies. Domain experts and different ontology engineers designed part of DABGEO layered ontology structure by applying MODDALS. They were able to follow MODDALS steps to obtain similar ontology designs by performing a domain analysis of existing ontologies (the degree of consensus when classifying the domain knowledge was 76%). Hence, we can state that MODDALS enables to classify the domain knowledge by taking as reference existing ontologies.

Taking into account MODDALS evaluation results, we have validated the methodology and demonstrated first hypothesis (H1) of the thesis: *ontology and SPL design techniques applied in combination enable to classify the domain knowledge into different abstraction layers with high level of consensus taking as reference existing ontologies*.

According to MODDALS evaluation participants, its main advantages are: (1) it provides a detailed domain knowledge classification; (2) it is easy to follow and (3) improves the reuse of existing knowledge to develop interoperable ontologies. By contrast, the main disadvantages of the methodology are: (1) the knowledge classification step is time consuming due to the manual ontology analysis effort required, (2) part of the domain knowledge classification is quite subjective, which may influence the final design of the ontology and (3) it does not provide guidelines for ontology maintenance. Hence, MODDALS is still a first step towards a widely accepted methodology to design layered ontology structures for reusable and usable ontologies in complex domains.

Finally, most of MODDALS evaluation participants recommend applying MODDALS to design the layered structure of ontologies in other complex domains apart from the Energy. Thus, we consider that MODDALS has the potential to be applied in other complex domains apart from the Energy.

Chapter 7

# DABGEO: the Domain Analysis-Based Global Energy Ontology

## 7.1 Introduction

As stated Chapter 5, energy ontologies developed for specific applications apply heterogeneous vocabularies to represent the same energy domains, which hampers the interoperability between energy management applications that operate in different scenarios. Therefore, there is the need to create a global ontology that provides a common energy domain representation. Since the energy domains are complex and the global ontology should support to different applications, it should provide a balance of reusability-usability.

The application of MODDALS in Chapter 6 enabled the development of DABGEO ontology, which is presented and described in this chapter. DABGEO (current version 1.0) is a reusable and usable global ontology for the energy domain. It can be reused by ontology engineers to develop ontologies for specific energy management applications. DABGEO is a large-scale ontology that includes 97 modules. DABGEO modules are published and can be downloaded at DABGEO Home Page: http://www.purl.org/dabgeo. DABGEO is licensed under the Creative Commons Attribution 4.0[1].

DABGEO provides a common representation of the energy domains represented heterogeneously by the available energy ontologies developed for specific applications (reviewed in Section 5.4). This common knowledge representation enables the creation of interoperable knowledge bases for energy management applications. In

[1] https://creativecommons.org/licenses/by/4.0/

addition, it includes links between the vocabularies of the existing energy ontologies to enable interoperability between new and legacy knowledge-based energy management applications.

In contrast with previous global energy ontologies, in DABGEO the common domain knowledge and variant domain knowledge are separated and classified into different abstraction layers. The knowledge of each abstraction layer is divided into small ontology modules that represent the knowledge of specific topics of the domains represented. This layered and modularized structure provides the following benefits when the ontology is reused:

1. It simplifies the ontology understanding and the ontology customization process to adapt it to specific application requirements [111].
2. Ontology developers can select at the proper level of abstraction only the necessary knowledge to develop applications that satisfy specific application requirements [111].
3. Ontology developers can focus on analysing and reusing only the knowledge reused by similar applications to the one to be developed.

These benefits result in a reduction of the ontology reuse effort in different energy management applications. Therefore, DABGEO provides a balance between reusability and usability to facilitate the reuse of knowledge to develop interoperable knowledge bases for different knowledge-based energy management applications.

DABGEO was evaluated by measuring its reuse effort in in two energy management applications. In addition, the reuse effort of DABGEO was compared with the effort of reusing the OEMA ontology network. OEMA is a global energy ontology developed in the first iteration of DABGEO. The purpose of this evaluation is to evaluate the DABGEO balance of reusability-usability, as well to demonstrate the second hypothesis of the thesis (H2): *a layered ontology that provides a common energy domain representation reduces the ontology reuse time and cost drivers in comparison to a global ontology without this structure*.

This chapter addresses the third thesis objective (O3) defined in Chapter 1: *develop a global energy ontology that provides a common energy domain representation and a balance of reusability-usability*.

The rest of the chapter is structured as follows. In Section 7.2, DABGEO is positioned with respect to the existing energy ontologies to highlight its main contributions. Section 7.3 describes DABGEO content, structure and its main benefits.

Section 7.4 explains DABGEO development process. Section 7.5 presents an empirical evaluation for DABGEO conducted to demonstrate its balance of reusability-usability. Section 7.6 discusses the ontology evaluation results. Finally, Section 7.7 summarizes the main conclusions of the chapter.

## 7.2 Related Work

In this section, DABGEO is positioned with respect to the current energy ontologies.

There are many ontologies developed to represent the knowledge about energy domains [32]. DABGEO provides a common representation of the domains represented heterogeneously by the energy ontologies reused in specific energy management applications. Hence, DABGEO integrates and reuses the knowledge from these ontologies. Considering this, DABGEO is first positioned with respect to the available energy ontologies reused in specific energy management applications (which were reviewed in Section 5.4). On the other hand, the overview includes previously developed global energy ontologies to highlight the main contribution of DABGEO.

### 7.2.1 Energy Domain Ontologies

The energy ontologies reviewed in Section 5.4 are reused by specific energy management applications that operate in different Smart Grid scenarios. These ontologies enable energy management applications to extract knowledge for intelligent decision making. However, their heterogeneity hinders the interoperability between knowledge-based energy management applications. In contrast to these ontologies, DABGEO is intended as a more general-purpose ontology. It provides a common representation of the energy domains represented by these ontologies. Hence, DABGEO enables to create interoperable knowledge bases for knowledge-based energy management applications that operate in different scenarios, thus enabling knowledge exchange between these applications.

### 7.2.2 Global Energy Ontologies

In recent years, several global energy ontologies have also been developed to be reused to develop ontologies for different energy management applications and to provide interoperability between these applications. The authors developed in Section 7.4.1 the OEMA ontology network [30] as the first iteration of DABGEO (as explained in Section 7.4). OEMA represents in a unified way the energy domains represented by energy ontologies developed for specific Smart Grid scenarios.

Lefrançois [96] presented the SEAS ontology. SEAS is a modular ontology that represents different energy domains to enable interoperability between smart systems that manage the operation of the future energy grid.

On the one hand, the OEMA ontology network (freely available at http://www.purl.org/oema) represents in a unified way the energy domains represented by energy ontologies developed for specific Smart Grid scenarios [30]. OEMA is the first version of DABGEO, since developed as the first iteration of DABGEO development process (the development process of OEMA is described in detail in Section 7.4). The OEMA ontology network represents the same energy domains as DABGEO, since it takes as reference and reuses the knowledge of the same energy ontologies to provide a common representation of energy domains. OEMA is divided into several ontologies that represent one energy domain each. Thus, the common and variant knowledge of each energy domain is represented in a single ontology. Considering this structure, OEMA puts emphasis on being detailed and complete, even at the cost of being less reusable and usable. Since the energy domains are complex, OEMA ontologies are too. This complexity hampers ontology reuse [171], since ontology developers must extract the knowledge needed by each application from these complex ontologies each time they develop an application ontology. The ontology developers spend significant time adding new knowledge, modifying the ontology knowledge, and extracting the necessary knowledge to satisfy the application knowledge requirements. In contrast to OEMA, DABGEO separates the common and variant domain knowledge into abstraction layers and divides this knowledge into ontology modules. DABGEO enables ontology developer to reuse only the necessary knowledge when developing application ontologies. Hence, DABGEO can be seen as an evolution of the OEMA ontology network.

On the other hand, Lefrançois [96] presented the SEAS ontology. SEAS is a modular ontology that represents different energy domains to enable interoperability between smart systems that manage the operation of the future energy grid. On the other hand, SEAS represents the abstract domain knowledge reused by many applications (i.e., it includes concepts such as Device or Observation), thus enabling ontology reuse in different applications. Therefore, depending on the application where it is reused SEAS may require a significant effort to extend its knowledge to satisfy specific knowledge requirements. Apart from including abstract domain knowledge, DABGEO includes specific knowledge reused only by certain application types (i.e. knowledge about devices used in smart home energy management applications). Hence, in contrast to SEAS, DABGEO enables ontology developers to reuse both abstract knowledge and specific knowledge that is closer to application requirements.

Table 7.1 summarizes the benefits and drawbacks of the ontologies reviewed in this section, as well as the main contributions of DABGEO with respect to the reviewed ontologies.

| **Related Work** | **Benefits and drawbacks** | **Differential aspects of DABGEO** |
|---|---|---|
| Energy domain ontologies: ThinkHome [92], Saref4EE [44], EnergyUse [17], Deri Linked Dataspace [39], BonSAI [147], OntoMG [155], DEFRAM project ontology [9], MAS2TERING [76], EE-DISTRICT [77], ProSGv3 [62], SEMANCO [26], Mirabel [161] and LCC [132]. | - *Benefits:* they enable energy management applications to extract knowledge for intelligent decision making.<br>- *Drawbacks:* their heterogeneity hinders the interoperability between knowledge-based energy management applications | Common representation of energy domains to enable interoperability between energy management applications that operate in different scenarios. |
| OEMA ontology network [30] | - *Benefits:* provides a common energy domain representation.<br>- *Drawbacks:* OEMA ontologies are complex and extensive. | Classification of the common energy knowledge and variant energy knowledge into different abstraction layers to provide a balance of ontology reusability-usability. |
| SEAS ontology [96] | - *Benefits:* represents different energy domains to enable interoperability between smart systems.<br>- *Drawbacks:* it only includes abstract domain knowledge. | |

**Table 7.1:** Summary of the work related to DABGEO ontology

## 7.3 DABGEO Ontology Overview

This section describes the content and structure of DABGEO, as well as the main benefits of the ontology.

### 7.3.1 DABGEO Ontology Content and Structure

This section describes the content and structure of DABGEO. DABGEO provides a common knowledge representation of the energy data domains represented by the existing energy ontologies (these domains were identified during the design of DABGEO layered structure in Chapter 6, Section 6.2.3).

In total, DABGEO includes 97 modules, which were implemented in OWL-2 DL[2] with Protégé (version 5.1.0). Concepts, relations, and attributes were modelled as classes, object properties and data properties, respectively. Axioms were represented

[2] https://www.w3.org/TR/owl2-overview/

in Protégé using diverse OWL restrictions (i.e., cardinality restrictions, object property restriction or datatype restrictions). Since DABGEO is a large-scale ontology, describing the main classes and properties of each module would make it difficult to understand DABGEO structure. Therefore, this section offers a high-level description of DABGEO content and structure without going into detail in each module. The specification of each ontology module, which describes the main classes and properties of the ontology module and the requirements it meets, can be found at DABGEO home page[3].

Within DABGEO, the represented energy domains are divided into subdomains that cover the knowledge of important parts of the domain. Figure 7.1 provides and overview of the DABGEO high-level structure, enumerating the subdomains in which the represented energy domains are divided.

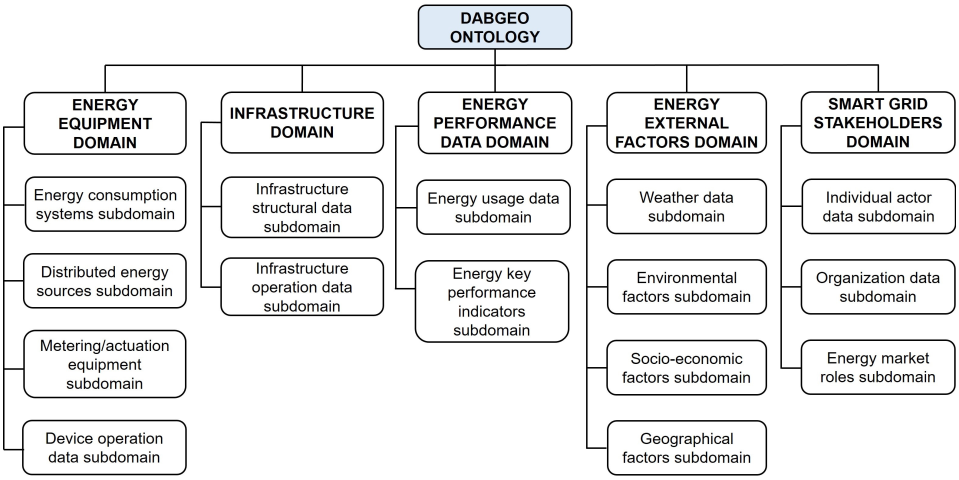

DABGEO ONTOLOGY
ENERGY EQUIPMENT DOMAIN
Energy consumption systems subdomain
Distributed energy sources subdomain
Metering/actuation equipment subdomain
Device operation data subdomain
INFRASTRUCTURE DOMAIN
Infrastructure structural data subdomain
Infrastructure operation data subdomain
ENERGY PERFORMANCE DATA DOMAIN
Energy usage data subdomain
Energy key performance indicators subdomain
ENERGY EXTERNAL FACTORS DOMAIN
Weather data subdomain
Environmental factors subdomain
Socio-economic factors subdomain
Geographical factors subdomain
SMART GRID STAKEHOLDERS DOMAIN
Individual actor data subdomain
Organization data subdomain
Energy market roles subdomain

**Figure 7.1:** DABGEO ontology high-level structure

These subdomains are described below:

1. **Energy equipment domain:** this domain encompasses the following subdomains:

    - *Energy consumption systems subdomain*: data about energy consumption devices such as home appliances, building automation equipment, industrial equipment, HVAC systems.
    - *Distributed energy sources subdomain*: renewable and non-renewable energy sources data (i.e., photovoltaics, electric vehicles, fossil fuels, home power plants) and energy storage systems such as batteries, capacitors or energy carriers.

[3] http://www.purl.org/dabgeo

   - *Metering/actuation equipment subdomain*: energy, weather and environmental conditions measurement sensors (i.e., humidity sensor, CO2 sensor, boiler controller) and equipment actuators, i.e., lighting control, pump controller.
   - *Device operation data subdomain*: device operational aspects such as device power profile, device state, device energy performance or device flexibility operations.

   The main classes of DABGEO used to represent the main concepts of the aforementioned subdomains are the following: ***Device*** (used to represent different types of devices such as energy consumption systems), ***EnergyConsumptionSystem*** (used to represent energy consumption devices), ***EnergyGenerator*** (used to represent energy generation devices such as solar panels) and ***MeteringActuation*** (used to represent sensors and actuators). The knowledge about these concepts is extended by classes, properties and axioms used to represent the knowledge about energy equipment features and operational aspects such as device energy consumption or device power profile.

2. **Infrastructure domain:** this domain encompasses the following subdomains:

   - *Infrastructure structural data subdomain*: different types of infrastructures and buildings (i.e., homes, industrial infrastructures, campuses, microgrids, power plants), building/infrastructure features (i.e., surface, material) and geometrical aspects such as rooms or floors.
   - *Infrastructure operational data subdomain*: infrastructure internal and external environmental conditions, i.e., room temperature.

   The main class of DABGEO used to represent the knowledge about the aforementioned subdomains is the ***Infrastructure*** class. This class is used to represent different types of infrastructures. The knowledge about these concepts is extended by classes, properties and axioms used to represent the knowledge about building/infrastructure features, geometrical details and internal and external environmental conditions.

3. **Energy performance domain:** this domain encompasses the following subdomains:

   - *Energy usage data subdomain*: energy production, consumption and storage values measured by sensors. Measurements' time intervals and units.

- *Energy key performance indicators subdomain*: energy performance indicators such as energy cost or energy gain.

The main class of DABGEO used to represent the knowledge about the aforementioned subdomains is the ***EnergyParameter*** class. This class is used to represent energy performance values such production, consumption and storage values and energy key performance indicators. The knowledge about these concepts is extended by classes, properties and axioms used to represent specific energy performance values and indicators the values and time intervals of these parameters.

4. **Energy external factors domain:** this domain encompasses the following subdomains:

   - *Geographical data subdomain*: infrastructures location data, i.e., latitude and longitude, state, city, district.
   - *Weather/climate data subdomain*: weather phenomenon (i.e., humidity, precipitation), weather conditions (i.e., cloud, sun) and weather forecast.
   - *Environmental data subdomain*: air pollutants of a place (i.e., nitrogen dioxide, ozone), and air pollutants indicators such as pollutant limit value or pollutant target value.
   - *Socio-economic data subdomain*: population demography data (i.e., population density, population percentage of children), population (i.e., population main origin, population average income) and individual (i.e., person education level, home language) social and economic data and household economic data, i.e., household pricing).

   The main classes of DABGEO used to represent the main concepts of the aforementioned subdomains are the following: ***WeatherPhenomenon*** (used to represent weather conditions), SocioEconomicFactor (used to represent the basic overall social and economic data pertaining to the population) and EnvironmentalFactor (used to represent the principal air pollutants in the urban area).

5. **Smart Grid stakeholders domain:** this domain encompasses the following subdomains:

   - *Individual actor data subdomain*: individual energy users that include home users, building occupants or employees.

- *Organization data subdomain*: organizations such as corporations or educational institutions.
- *Energy market roles data subdomain*: the roles that Smart Grid stakeholders have in the energy market, i.e., energy consumers, energy suppliers, DSOs.

The main classes of DABGEO used to represent the main concepts of the aforementioned subdomains are the following: ***Actor*** (used to represent actors that participate in the usage process such as home users, building occupants and organizations) and ***EnergyMarketRole*** (used to represent roles that energy actors have in the energy market). The knowledge about these concepts is extended by classes, properties and axioms used to represent the knowledge about actor preferences of on energy devices (i.e., minimum/maximum price that the user is willing to pay for energy production/consumption), organization internal structure (i.e., organization members and business processes) or the energy type provided by the energy providers (i.e., electric energy, thermal energy).

Below, the detailed DABGEO structure is explained using as an example the *energy equipment domain*. We refer to DABGEO home page[4] for further information about the representation of the rest of domains. Figure 7.2 provides a detailed overview of DABGEO structure concerning the energy equipment domain. The knowledge of subdomains is divided into ontology modules that represent the knowledge of a particular topic of the subdomain (to simplify the understanding of Figure 7.2, we have omitted a couple of modules and module relationships). The ontology modules of DABGEO are classified into three abstraction layers. In the next subsections we describe the kind of knowledge included in each layer. It is worth mentioning that the list of modules included by each layer can be found at DABGEO home page.

#### 7.3.1.1 Common-domain Layer

The *common-domain layer* includes the domain knowledge common to all Smart Grid scenarios. For instance, the *Device ontology module* represents the *Device* concept and device main properties (i.e., device name). As another example, the *energy consumption systems ontology module* represents the knowledge about energy consumption system types, i.e., knowledge about appliances or HVAC systems. This module

[4] `http://www.purl.org/dabgeo`

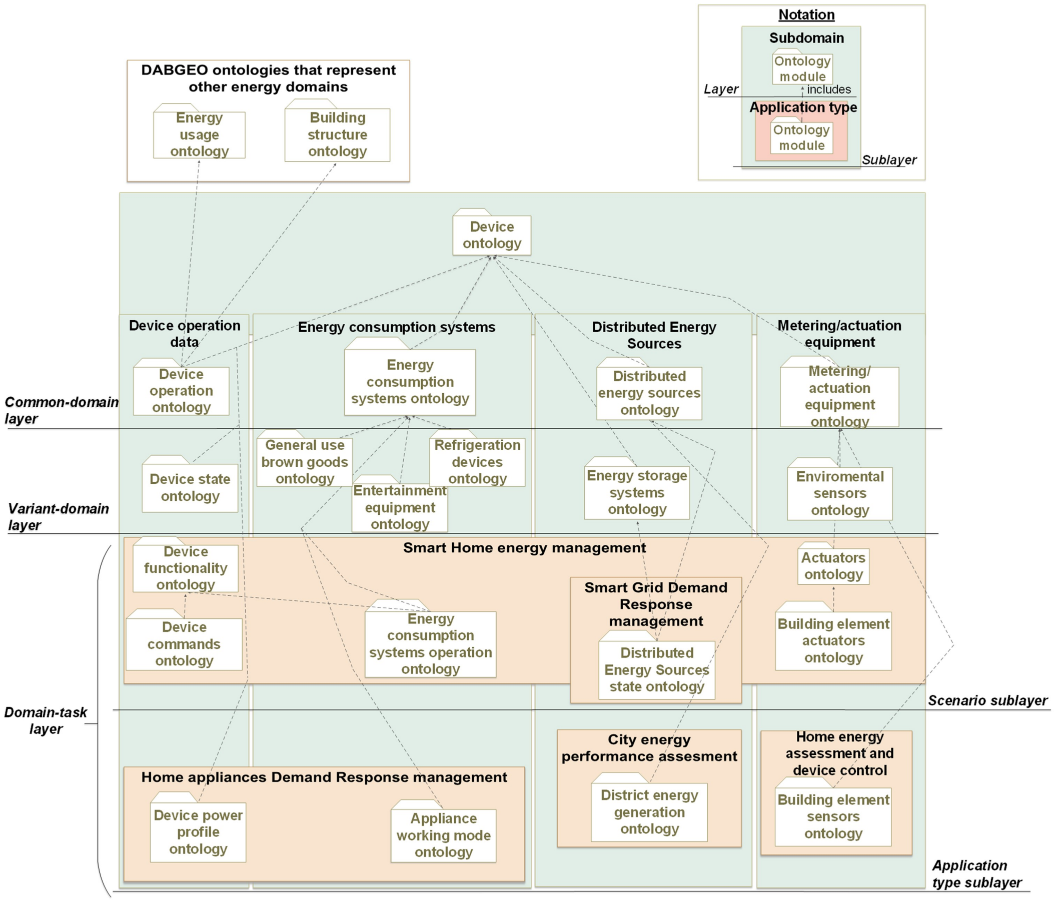

Notation
Subdomain
Ontology module
Layer
includes
Application type
Ontology module
Sublayer
DABGEO ontologies that represent other energy domains
Energy usage ontology
Building structure ontology
Device ontology
Device operation data
Energy consumption systems
Distributed Energy Sources
Metering/actuation equipment
Device operation ontology
Energy consumption systems ontology
Distributed energy sources ontology
Metering/ actuation equipment ontology
Common-domain layer
Device state ontology
General use brown goods ontology
Refrigeration devices ontology
Entertainment equipment ontology
Energy storage systems ontology
Enviromental sensors ontology
Variant-domain layer
Smart Home energy management
Device functionality ontology
Device commands ontology
Energy consumption systems operation ontology
Smart Grid Demand Response management
Distributed Energy Sources state ontology
Actuators ontology
Building element actuators ontology
Scenario sublayer
Domain-task layer
Home appliances Demand Response management
Device power profile ontology
Appliance working mode ontology
City energy performance assesment
District energy generation ontology
Home energy assessment and device control
Building element sensors ontology
Application type sublayer

**Figure 7.2:** DABGEO ontology structure (energy equipment domain)

extends the knowledge about devices, so it includes the knowledge of the *Device ontology module*.

### 7.3.1.2 Variant-domain Layer

The *variant-domain layer layer* includes the variant domain knowledge reused only in several Smart Grid scenarios. Since the knowledge of this layer is relevant for fewer applications, the ontology modules include more specific knowledge. Thus, they extend and include the knowledge of the common-domain layer ontology modules. For instance, in the *energy consumption systems subdomain* these ontology modules represent the knowledge about specific appliances or HVAC systems. Two examples of these ontology modules are the *general use brown goods ontology* (which represents the knowledge about specific brown goods such as body care devices) or *air conditioning systems ontology* (which represents the knowledge about specific air conditioning systems such as space cooling systems).

#### 7.3.1.3 Domain-task Layer

The *domain-task layer* includes the domain knowledge reused in specific Smart Grid scenarios. As explained in Section 6.2.1, the energy management application types that correspond to each Smart Grid scenario can be classified into more specific application types. Therefore, the *domain-task layer* is divided into two sublayers: the *scenario sublayer* and the *application type sublayer*. These sublayers separate the knowledge reused only by a specific application type from the knowledge still relevant for all the specific application types encompassed by the Smart Grid scenario. These sublayers classify the ontology modules according to the Smart Grid scenario or application type that reuse them:

- The *scenario sublayer* represents the knowledge relevant to a certain Smart Grid scenario. For example, the knowledge about *device commands* or *device functionality* (represented by the homonymous modules) is only relevant for smart home energy management applications. As another example, knowledge about *district energy generation systems* is only relevant to different types of building/district/city energy management applications.

- The *application type sublayer* represents the knowledge reused only by certain energy management application types for a specific Smart Grid scenario. For example, within smart home energy management applications, only home appliances DR management applications reuse the knowledge about appliance operation such as *device power profile* or *appliance working modes*.

Finally, it is worth mentioning that the ontology modules that belong to one of the domains/subdomains represented by DABGEO may include the knowledge of DABGEO modules that represent other energy domains/subdomains. For example, the *device operation ontology module* links the knowledge about devices with device operation aspects such as device energy consumption and the device location in certain building spaces. The knowledge about these aspects is represented by the *energy usage* and *building structure* ontology modules of DABGEO. These modules belong to other energy domains represented by DABGEO (the *energy performance* and *infrastructure* domains respectively). Hence, the device operation ontology module includes the knowledge of these modules (see Figure 7.2).

### 7.3.2 DABGEO Main Benefits

On the one hand, the common energy domain representation of DABGEO provides the following benefits:

1. *Development of interoperable ontologies*: the common knowledge of DABGEO enables the creation of interoperable ontologies that follow a common energy domain representation and, by extension, the creation of interoperable knowledge bases for energy management applications.

2. *Backward interoperability*: DABGEO links the equivalent knowledge from heterogeneous energy ontologies. Hence, it enables interoperability between new and legacy energy management applications supported by heterogeneous ontologies.

On the other hand, the layered and modularized structure followed by DABGEO, provides the following benefits when reusing the ontology:

1. *Understandability and adaptability* [111, 45]: the knowledge of DABGEO is divided into small modules. These modules represent closely related topics and they are concise, since they include only concepts and relations to represent the knowledge of the concerned topic. Hence, the modules have a low complexity and are easy to understand. In addition, the modules are independent as they only relate with the modules whose knowledge they extend or depend on. Therefore, the ontology modules can be reused, combined and adapted to develop application ontologies without affecting other parts of the ontology.

2. *Selection of domain knowledge at the proper level of abstraction* [111]: with the layered structure, ontology developers can analyse and select at the proper level of generality and abstraction only the necessary knowledge of each domain to develop application ontologies. Depending on the application developed, ontology developers can just use modules that include abstract knowledge or modules that include both abstract and specific knowledge. For example, a home energy management application and a district energy management application may require different specific knowledge of the same domains and thus may reuse different modules from the domain-task layer. In contrast, these applications may share the knowledge from upper layers.

3. *Selective knowledge analysis and reuse*: DABGEO classifies the specific variant domain knowledge according to the application types that reuse it. This feature enables ontology developers to only focus on analysing and reusing the modules that contain the knowledge reused by similar applications to the one they must develop. For example, let us consider an ontology developer who reuses DABGEO to develop an application ontology for an application that manages the home appliances energy consumption to adjust it to energy tariffs. Considering the goal of the application, it can be considered as a smart home energy management application and more specifically, as a home appliances DR management application. Therefore, the ontology developer needs to only analyse the specific domain knowledge of the ontology modules classified into the smart home energy management and home appliances DR management application types of the domain-task layer.

4. *Ontology maintainability* [111]: the structure of DABGEO simplifies ontology maintenance. To provide support to new energy management applications, new knowledge can be added to the ontology as new ontology modules. DABGEO abstracts the top-level and most relevant knowledge from the specific domain knowledge. In this way, the new ontology modules will extend specific modules of the ontology without modifying the rest of modules and the ontology structure.

## 7.4 DABGEO Development Process

As stated in Chapter 6, DABGEO development team included people with expertise in energy domain data representation and ontology engineers. The development process of DABGEO was guided by the requirements defined in Chapter 1:

1. The ontology should unify the knowledge of the existing energy ontologies to provide a common knowledge representation of energy domains.

2. The ontology should classify the common and variant domain knowledge into different abstraction layers.

Bearing in these requirements, the MODDALS methodology was applied to design the layered structure of DABGEO in Chapter 6. Since MODDALS can be applied as a set of additional steps of the NeOn methodology (see Section 4.2.1), NeOn was applied to develop DABGEO.

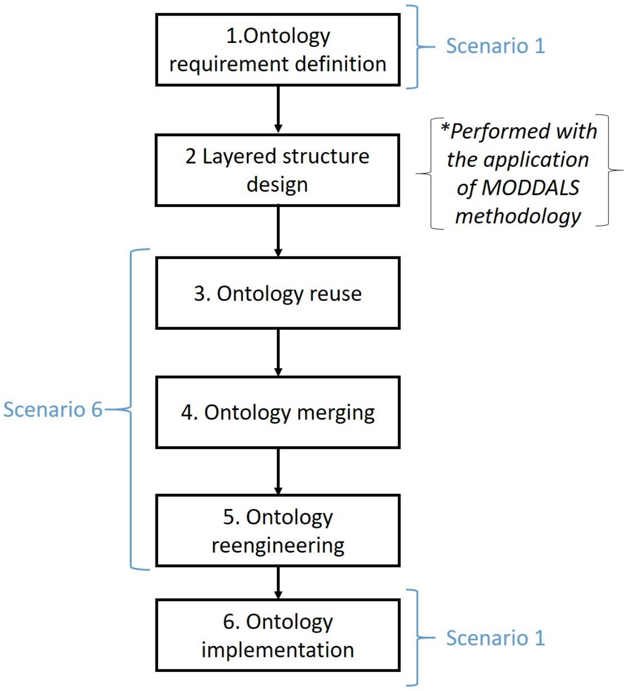

1.Ontology requirement definition
Scenario 1
2 Layered structure design
*Performed with the application of MODDALS methodology
3. Ontology reuse
Scenario 6
4. Ontology merging
5. Ontology reengineering
6. Ontology implementation
Scenario 1

**Figure 7.3:** DABGEO development process

In particular, the application of MODDALS in in Chapter 6 was only a phase of the whole DABGEO development process. MODDALS was applied as an ontology reuse scenario within the NeOn methodology to design the layered structure of DABGEO. As stated in Section 5.5, a global energy ontology must provide a common representation of energy domains. In addition it should link the vocabularies from existing ontologies to enable interoperability between new and legacy energy management applications. Considering this, the knowledge of existing ontologies was merged to provide a common knowledge representation and links were established between the overlapping knowledge represented heterogeneously by existing ontologies. Hence, the Scenario 6 of NeOn was applied after MODDALS to reuse, merge and reengineer the knowledge of existing ontologies. In addition the key ontology development steps from NeOn Scenario 1 (ontology requirement definition, ontology implementation) were conducted. By applying these scenarios in combination, the steps shown in Figure 7.3 were defined to develop DABGEO.

1. *Ontology requirement definition*: DABGEO purpose and scope is defined by domain experts and ontology engineers. The main goal of DABGEO is to provide a common knowledge representation of energy domains represented by heterogeneous ontologies, while keeping moderate the ontology reuse

effort. In addition, the high-level requirements of the ontology are defined: common knowledge representation and abstraction layering.

2. *Layered structure design*: the layered structure of DABGEO is designed by applying the steps proposed by MODDALS (see Section 6.2).

3. *Ontology reuse*: the elements from existing energy ontologies are selected for reuse by ontology engineers. In addition, the knowledge correspondences between energy ontologies are identified. The identified correspondences are taken as reference to link and merge the ontologies.

4. *Ontology merging*: the knowledge of existing energy ontologies is merged by ontology engineers according to the knowledge correspondences identified in the previous steps. The result of this process is a single ontology that provides a common representation of energy domains.

5. *ontology reengineering*: the merged knowledge is reengineered to (1) complete the integration of the knowledge of existing ontologies and (2) to facilitate its reuse during the ontology implementation phase.

6. *Ontology implementation*: the merged knowledge from existing energy ontologies is reused and implemented according to the layered structure. The syntax and logical consistency of the ontology are evaluated. Finally, the ontology is published online along with its documentation.

According to the defied ontology development process, the merged ontology is implemented by reusing the knowledge of existing energy ontologies. Therefore, the merged ontology obtained at the end of Step 5 can be seen a first implemented version of DABGEO that provides a common representation of the knowledge represented by heterogeneous energy ontologies. On the other hand, the DABGEO implementation step (Step 6) encompasses the reuse and implementation of the knowledge of the merged ontology according to the layered structure obtained by applying MODDALS. The realization of this step results in the final implementation of DABGEO.

Considering the two versions of DABGEO that are obtained, we identified two ontology development iterations in the overall DABGEO development process: the development of the merged ontology and the development of the final layered and modular ontology. Since the energy ontologies represent complex and extensive energy domains, both processes require a great effort. Therefore, DABGEO development process is complex. Dividing the DABGEO development process into the identified iterations enables to evaluate the merged ontology to detect possible implementation issues (i.e., logical inconsistencies, syntax errors) before classifying the knowledge into different layers. Thus, the complexity of the ontology development process is reduced [150].

Moreover, the reuse effort of the merged ontology can be compared with the ontology reuse effort of the final version of DABGEO. This comparison can be taken as reference to demonstrate that the layered structure of DABGEO reduces the ontology reuse effort in different applications.

Therefore, the DABGEO development process (Fig. 7.3) was divided into two iterations, which encompass the development processes of the merged ontology and DABGEO respectively (Fig. 7.4). These iterations take as input DABGEO high-level requirements and layered structure (defined in the first two steps of DABGEO development process).

1. *First iteration*: in this iteration the first requirement of DABGEO (R1) is addressed: it must provide a common energy domain representation. This iteration encompasses the steps 3, 4 and 5 of DABGEO development process. The knowledge of existing ontologies was reused and merged into an ontology that provides a common representation of the energy domains. In addition, after implementing the knowledge of the ontology, the reused knowledge was reengineered to (1) complete the integration of the knowledge of existing ontologies and (2) to facilitate the reuse of the ontology knowledge in the second iteration of the DABGEO development process or in any knowledge-based application. The output of this iteration was the OEMA ontology network (introduced in Section 7.2.2), which provides a common representation of the energy domains represented by existing energy ontologies.

2. *Second iteration*: in this iteration the second requirement of DABGEO (R2) is addressed: it must follow a layered structure. The knowledge represented by OEMA was reused, reengineered and implemented according to the layered ontology structure designed in Step 2. This iteration encompasses the step 6 of DABGEO development process. The output of this iteration is the final version of DABGEO.

In addition, as shown in Figure 7.4, both iterations included some of the key steps of the ontology development process.

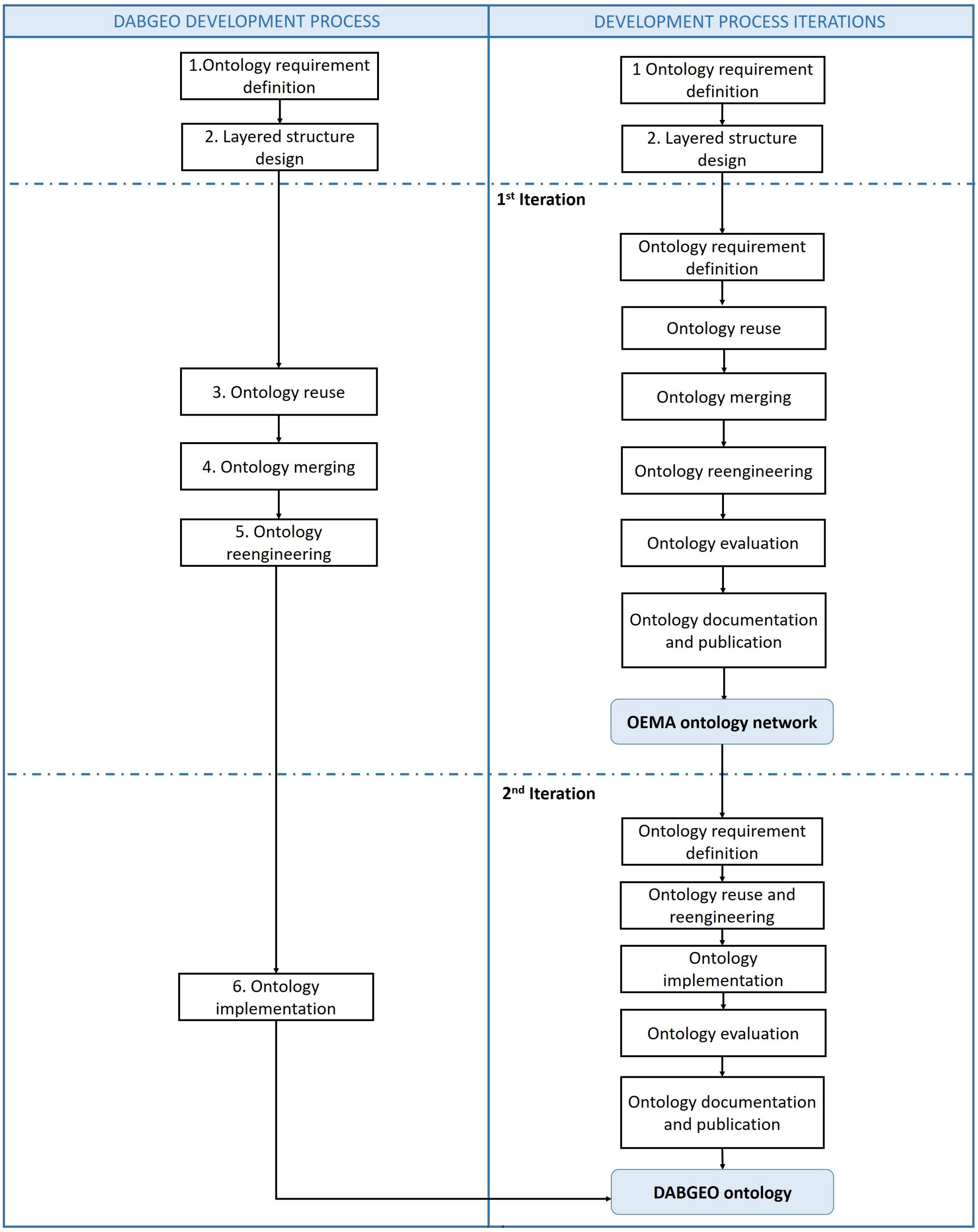

DABGEO DEVELOPMENT PROCESS
DEVELOPMENT PROCESS ITERATIONS
1.Ontology requirement definition
2. Layered structure design
3. Ontology reuse
4. Ontology merging
5. Ontology reengineering
6. Ontology implementation
1 Ontology requirement definition
2. Layered structure design
1st Iteration
Ontology requirement definition
Ontology reuse
Ontology merging
Ontology reengineering
Ontology evaluation
Ontology documentation and publication
OEMA ontology network
2nd Iteration
Ontology requirement definition
Ontology reuse and reengineering
Ontology implementation
Ontology evaluation
Ontology documentation and publication
DABGEO ontology

**Figure 7.4:** DABGEO development process iterations

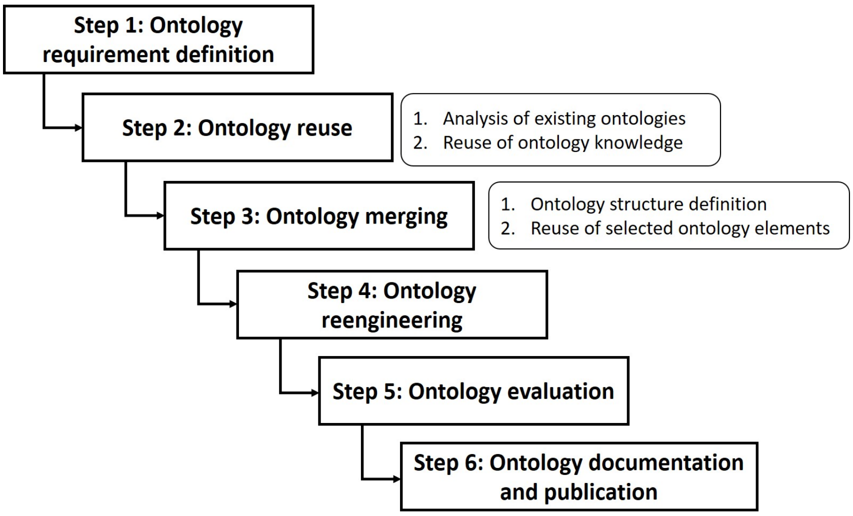

Step 1: Ontology requirement definition
Step 2: Ontology reuse
1. Analysis of existing ontologies
2. Reuse of ontology knowledge
Step 3: Ontology merging
1. Ontology structure definition
2. Reuse of selected ontology elements
Step 4: Ontology reengineering
Step 5: Ontology evaluation
Step 6: Ontology documentation and publication

**Figure 7.5:** OEMA development steps

- *Ontology requirement definition*: the functional and non-functional requirements necessary to address each of the high-level requirements are defined.
- *Ontology evaluation*: the syntax and the logical consistency of the ontology is checked.
- *Ontology documentation and publication*: the developed ontology is documented and published in the web.

The next subsections focus on the steps followed in each iteration of DABGEO development process.

### 7.4.1 First Iteration: OEMA Ontology Network Development

In the first iteration, the knowledge of existing ontologies was merged into a single ontology. The result of this iteration was the OEMA ontology network. OEMA provides a common knowledge representation of the energy domains represented with different vocabularies by heterogeneous energy ontologies. OEMA is freely available at http://www.purl.org/oema. This subsection describes OEMA development steps, as well as OEMA ontologies as a result of this iteration.

Figure 7.5 shows in detail OEMA development steps, as well as the activities they encompass.

#### 7.4.1.1 Step 1: Ontology Requirement Definition

In this step, the requirements of the OEMA ontology network were defined by domain experts and ontology engineers.

Regarding the functional requirements, OEMA must represent all energy data domains (and corresponding subdomains) represented by existing energy ontologies at a high level of detail. These domains and subdomains were defined in Chapter 6 during the DABGEO layered structure design process, and they were described in Section 7.3.1.

On the other hand, OEMA non-functional requirements cover the following aspects:

- The ontology modularization was established as a requisite to facilitate the reuse of the knowledge and its classification into different layers in the second iteration of the DABGEO ontology development. In particular, OEMA must represent the knowledge of each of the aforementioned domains in one ontology.
- The knowledge of each domain must be represented with unique vocabularies and a unique ontology structure. The ontology must provide unique names to the overlapping concepts and relations represented heterogeneously by existing energy ontologies. In addition, the ontology must provide a unique class hierarchy to represent the overlapping concepts represented with different class hierarchies by existing ontologies.
- The overlapping knowledge represented by existing energy ontologies must be linked according to their knowledge correspondences to provide interoperability among legacy applications that reuse these ontologies and new applications.
- The so-called CamelCase notation must be used to name the elements reused from different energy ontologies. This notation is typically suggested for developing ontologies and is adopted by well-known ontologies such as FOAF ontology [14, 109]. Specifically, the upper CamelCase notation must be applied for naming classes and the lower CamelCase for naming properties.

#### 7.4.1.2 Step 2: Ontology Reuse

This step consisted on the selection of the elements of existing energy ontologies to be reused during OEMA implementation. In addition, the knowledge correspondences between energy ontologies were identified to take them as reference to merge and link the ontologies when implementing OEMA. This step was divided into two sequential activities: analysis of existing ontologies and reuse of ontology knowledge.

### 1. Analysis of Existing Ontologies

The existing energy ontologies were analysed by ontology engineers to select the knowledge that was reused to implement OEMA. The analysed ontologies include the ontologies that were selected to design DABGEO structure in Chapter 6 by applying MODDALS. The ontologies were evaluated taking into account OEMA requirements. The authors checked whether previous ontologies represent the knowledge of the energy domains defined in Step 1 and the granularity with which the domains are represented. Table 7.2 shows the level of detail with which energy ontologies represent each energy domain/subdomain.

### 2. Reuse of Ontology Knowledge

The ThinkHome ontology is one of the ontologies that represents most energy domains at a high level of detail. This ontology represents, in a machine-readable way, home energy consumption, production and energy-related contextual data. Although it is designed to be used by home energy management applications, the ontology can also be extended to other Smart Grid scenarios such as Organizations energy management or microgrid energy management. This can be achieved making few changes in ThinkHome ontology structure. In addition, the ThinkHome ontology classifies the energy domain knowledge into different domain ontologies, thus facilitating ontology reuse and maintenance. Due to its completeness and its modular approach, the ThinkHome ontology was selected to be the base of OEMA. Considering this, the structure of the ThinkHome ontology was taken as reference to define a unique ontology structure.

Five independent ontologies or modules form the ThinkHome ontology:

- *Building ontology*: knowledge about building physical elements (i.e., materials, surface, and spaces), building internal and external equipment and building geometrical features. It also represents building internal and external environmental conditions such as temperature or occupancy.
- *User information ontology*: knowledge about home user attributes (i.e., age, gender) comfort preferences and user schedules.

| Ontologies | Energy data domains | ThinkHome ontology | DEFRAM project ontology | SAREF4EE ontology | BOnSAI ontology | EnergyUse ontology | ProSGV3 ontology | LCC ontology | Mirabel ontology | DERI Linked dataspace | SEMANCO ontology |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Energy equipment domain** | Energy consumption systems data | H | L | M | L | H | H | - | L | L | H |
| | DERs data | M | - | L | - | - | H | - | L | - | M |
| | Metering/actuation equipment data | H | - | M | M | M | M | - | - | L | - |
| | Device operation data | H | - | H | M | H | L | L | M | L | L |
| **Infrastructure domain** | Infrastructure structural data | H | - | M | L | L | L | - | - | L | M |
| | Infrastructure operation data | L | - | - | - | - | L | - | - | - | H |
| **Energy performance domain** | Energy usage data | H | - | L | L | L | H | L | - | L | M |
| | Energy key performance indicators | L | - | - | - | - | - | - | - | - | H |
| **Energy external factors domain** | Weather/climate data | H | - | - | L | - | M | - | - | - | L |
| | Geographical data | L | - | - | - | - | - | L | - | - | H |
| | Environmental data | - | - | - | L | - | - | - | - | - | H |
| | Socio-economic data | - | - | - | - | - | - | - | - | - | H |
| **Smart Grid stakeholders domain** | Individual actors data | M | - | - | L | M | - | - | - | - | - |
| | Organisation data | - | L | - | - | - | - | - | - | H | - |
| | Energy market roles data | M | - | - | - | L | L | - | L | L | L |

**Table 7.2:** Level of detail of energy domain representation by energy ontologies (H=High/M=Medium/L=Low)

- *Exterior ontology*: knowledge about weather and climate conditions such as weather phenomenon or weather forecast.
- *Energy and resource ontology*: knowledge about building energy equipment or devices (i.e., home appliances, energy measurement sensors), device operation parameters (i.e., function, state), energy demand, supply and cost data, energy providers and energy tariffs.
- *Processes ontology*: knowledge about home system processes (i.e., cooling process, heating process) and user activities data, i.e., device usage patterns.

In addition, elements from the rest of ontologies of Table 7.2 were selected for reuse. The selected elements represent the following knowledge:

1. The knowledge that extends the knowledge from the energy domains represented by ThinkHome.
2. The knowledge from the domains not represented by the ThinkHome ontology that complements the knowledge represented by this ontology.

This knowledge was later integrated into the knowledge of ThinkHome during the ontology merging step (Step 3). This ontology merging approach consists on populating a target ontology with the knowledge from other ontologies. It avoids redundant and inconsistent knowledge in the merged ontology, thus facilitating the integration of multiple ontologies [133]. To identify and select the knowledge that extends and complements the knowledge represented by ThinkHome, two ontology matching activities were conducted:

1. *Knowledge that extends the ThinkHome ontology knowledge*: the knowledge correspondences between the ThinkHome ontology and the rest of the energy ontologies were analysed to identify the overlapping knowledge of these ontologies. The more/less-general correspondences were identified and analysed. Then, the ontology classes, properties and axioms that extend the knowledge represented by the ThinkHome ontology were selected from the other ontologies for reuse. These elements were added to the ThinkHome ontology elements during OEMA implementation. In addition, the equivalence correspondences identified in this ontology matching activity were taken as reference to link the ontology elements from the ThinkHome ontology with the ontology elements from other ontologies that represent the same knowledge during OEMA implementation.

2. *Knowledge that complements the ThinkHome ontology knowledge*: the knowledge correspondences between the ontologies that represent the knowledge from domains not represented by ThinkHome were identified. These correspondences were taken as reference to define a unique structure to represent the domains not covered by ThinkHome. The equivalence correspondences were analysed to identify the overlapping concepts in these domains. For each identified concept, the elements from a target ontology that describes the knowledge about the concept with more level of detail (taking into account the granularity of the class hierarchies, and the number of properties and axioms used to describe the concept) were selected for reuse. Then, the more/less-general correspondences were analysed to select for reuse the ontology elements that extend the knowledge represented by the target ontology. These elements added to the ThinkHome ontology elements during OEMA implementation. In addition, the identified equivalence correspondences were taken as reference to link the ontology elements that represent the same knowledge during OEMA implementation.

The aforementioned ontology matching activities were performed semi-automatically following the guidelines proposed by Euzenat and Le Duc [51]:

1. Firstly, an ontology matching tool was selected to identify the knowledge correspondences between different ontologies. Since the obtained knowledge correspondences where using to merge the energy ontologies, the ontology matching tool must provide completeness, that is, it must identify as many correspondences as possible [51]. In the last years, the *AgreementMaker* ontology matching tool [29] has offered a good performances over the last years in terms of completeness [142]. Hence, this tool was applied to identify semi-automatically the knowledge correspondences between energy ontologies.

2. Then, the identified knowledge correspondences were manually analysed by domain experts to determine whether they were correct.

3. Finally, the ontology developers analysed manually the energy ontologies to identify the knowledge correspondences not identified by the ontology matching tool. This manual analysis enables to identify additional correspondences between ontologies [133].

Elements from other energy ontologies than the ones shown in Table 7.2 were also selected for reuse in order to complement the knowledge of the ThinkHome ontology. In particular, the elements of DBpedia and FOAF[5] ontologies were selected for reuse. DBpedia *"is a community effort to extract structured information from Wikipedia and to make this information available on the Web"* [5]. This ontology represents with high level of detail geographical, Organizations and people data [173]. FOAF adds more detail to people attributes data.

#### 7.4.1.3 Step 3: Ontology Merging

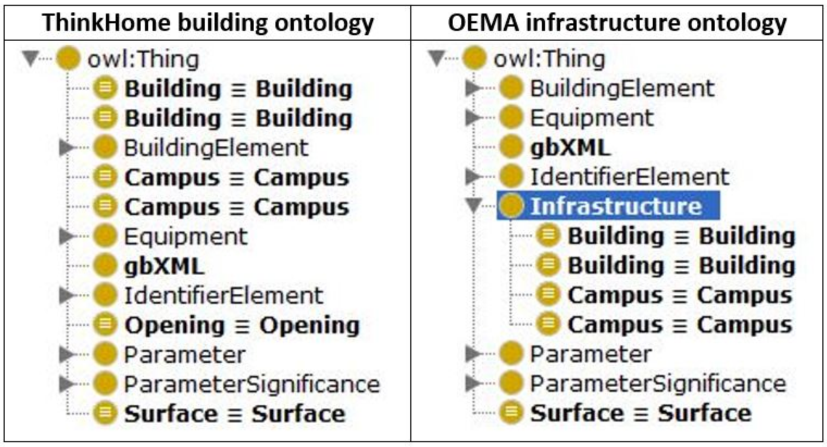
The next step was the implementation of the ontology by merging the knowledge selected in Step 2. OEMA was implemented by ontology engineers with Protégé following two sequential activities: ontology structure definition and reuse of selected ontology elements.

**1. Ontology Structure Definition**

The OEMA base structure was defined. As said before, the ThinkHome ontology is the base of OEMA. OEMA reuses five ThinkHome domain ontologies. However, these ontologies are oriented to smart home energy management. Hence, ThinkHome ontologies were restructured and generalized to adapt them to other Smart Grid scenarios in addition to the smart home energy management scenario. This restructuring process consisted of renaming and including new super-classes for domain ontologies, so that the can represent more knowledge that the one represented by ThinkHome. As a result, the first OEMA domain ontologies were created:

- *OEMA Infrastructure ontology*: this ontology reuses the whole *ThinkHome building ontology*. It adds the *owl:Infrastructure* class to represent more infrastructures (i.e., microgrids, VPPs) apart from homes and buildings. Then, all top-level classes of ThinkHome building ontology that represent buildings were included as sub-classes of *owl:Infrastructure* class. Figure 7.6 shows the difference between ThinkHome building ontology and OEMA infrastructure ontology after performing the restructuring process.
- *OEMA Smart Grid stakeholders ontology*: this ontology reuses the whole *ThinkHome user information ontology*. It adds the *owl:Stakeholder* class to represent more Smart Grid stakeholders (i.e., utilities, Organizations) apart from home users. Then, all top-level classes of ThinkHome user information ontology that represent home users were included as sub-classes of *owl:Stakeholder class*. Figure

[5] http://xmlns.com/foaf/spec/20071002.html



ThinkHome building ontology

- owl:Thing
  - Building ≡ Building
  - Building ≡ Building
  - BuildingElement
  - Campus ≡ Campus
  - Campus ≡ Campus
  - Equipment
  - gbXML
  - IdentifierElement
  - Opening ≡ Opening
  - Parameter
  - ParameterSignificance
  - Surface ≡ Surface

OEMA infrastructure ontology

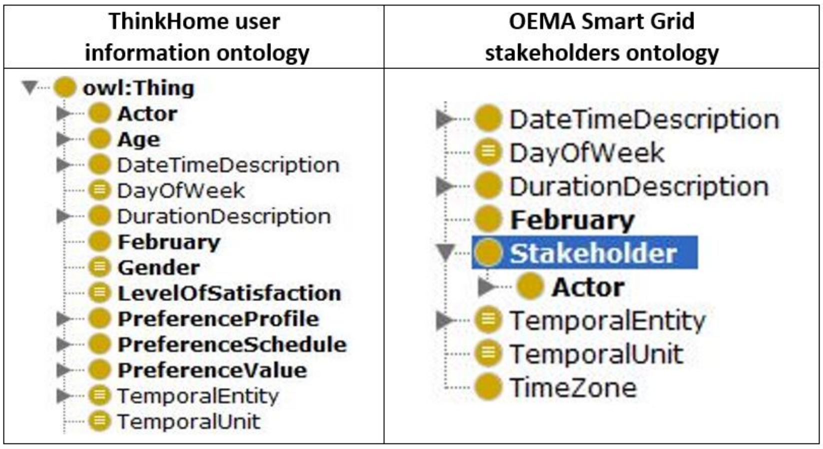
- owl:Thing
  - BuildingElement
  - Equipment
  - gbXML
  - IdentifierElement
  - Infrastructure
    - Building ≡ Building
    - Building ≡ Building
    - Campus ≡ Campus
    - Campus ≡ Campus
  - Parameter
  - ParameterSignificance
  - Surface ≡ Surface

**Figure 7.6:** Difference between ThinkHome building ontology and OEMA infrastructure ontology

7.7 shows the difference between ThinkHome user information ontology and OEMA Smart Grid stakeholders ontology.

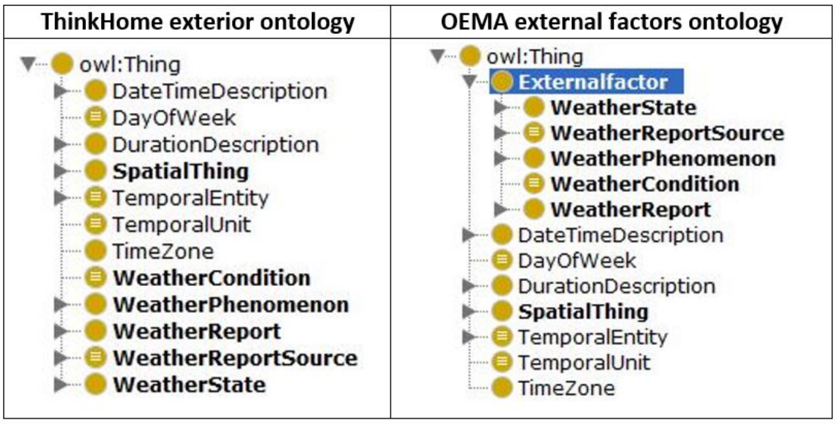
- *OEMA external factors ontology*: it reuses the whole *ThinkHome exterior ontology*. It adds the *owl:ExternalFactor* class to represent more external factors (i.e., environmental conditions, air pollutants), apart from weather data, that should be taken into account to manage the energy performance in wider infrastructures than homes. Then, all top-level classes of ThinkHome exterior ontology that represent weather concepts were included as sub-classes of *owl:ExternalFactor* class. Figure 7.8 shows the difference between ThinkHome user information ontology and OEMA Smart Grid stakeholders' ontology.

- *OEMA energy and equipment ontology*: it reuses the whole *ThinkHome energy and resource ontology* and *ThinkHome processes ontology*. It adds the *owl:EnergyEquipment* and *owl:EnergyParameter* classes to represent any type of energy equipment and energy performance parameters respectively, apart from home and building equipment and the specific energy performance parameters represented by ThinkHome. Then, all top level classes of ThinkHome energy and resource ontology that represent to building equipment or specific energy performance parameters were included as sub-classes of *owl:EnergyEquipment* and *owl:EnergyParameter* classes. Figure 7.9 shows the difference between ThinkHome energy and resource ontology and OEMA energy and equipment ontology.



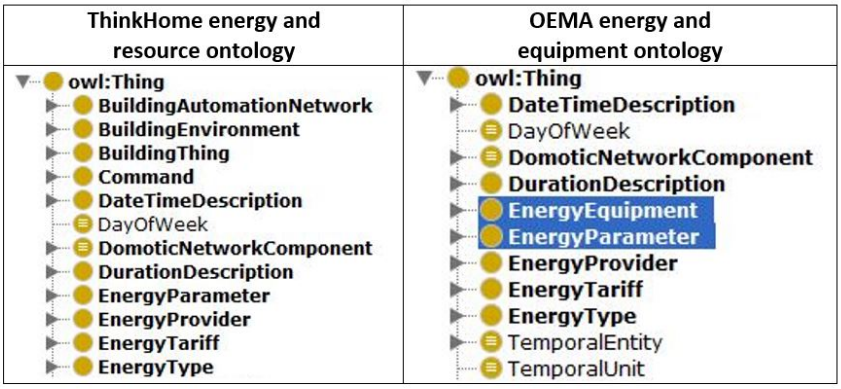
| ThinkHome user information ontology | OEMA Smart Grid stakeholders ontology |
|---|---|
| owl:Thing<br>Actor<br>Age<br>DateTimeDescription<br>DayOfWeek<br>DurationDescription<br>February<br>Gender<br>LevelOfSatisfaction<br>PreferenceProfile<br>PreferenceSchedule<br>PreferenceValue<br>TemporalEntity<br>TemporalUnit | DateTimeDescription<br>DayOfWeek<br>DurationDescription<br>February<br>Stakeholder<br>Actor<br>TemporalEntity<br>TemporalUnit<br>TimeZone |

**Figure 7.7:** Difference between ThinkHome user information ontology and OEMA Smart Grid stakeholders ontology



| ThinkHome exterior ontology | OEMA external factors ontology |
|---|---|
| owl:Thing<br>DateTimeDescription<br>DayOfWeek<br>DurationDescription<br>SpatialThing<br>TemporalEntity<br>TemporalUnit<br>TimeZone<br>WeatherCondition<br>WeatherPhenomenon<br>WeatherReport<br>WeatherReportSource<br>WeatherState | owl:Thing<br>Externalfactor<br>WeatherState<br>WeatherReportSource<br>WeatherPhenomenon<br>WeatherCondition<br>WeatherReport<br>DateTimeDescription<br>DayOfWeek<br>DurationDescription<br>SpatialThing<br>TemporalEntity<br>TemporalUnit<br>TimeZone |

**Figure 7.8:** Difference between ThinkHome exterior ontology and OEMA external factors ontology



| ThinkHome energy and resource ontology | OEMA energy and equipment ontology |
|---|---|
| owl:Thing<br>BuildingAutomationNetwork<br>BuildingEnvironment<br>BuildingThing<br>Command<br>DateTimeDescription<br>DayOfWeek<br>DomoticNetworkComponent<br>DurationDescription<br>EnergyParameter<br>EnergyProvider<br>EnergyTariff<br>EnergyType | owl:Thing<br>DateTimeDescription<br>DayOfWeek<br>DomoticNetworkComponent<br>DurationDescription<br>EnergyEquipment<br>EnergyParameter<br>EnergyProvider<br>EnergyTariff<br>EnergyType<br>TemporalEntity<br>TemporalUnit |

**Figure 7.9:** Difference between ThinkHome energy and resource ontology and OEMA energy and equipment ontology

### 2. Reuse of Selected Ontology Elements

The ontology elements from the rest of energy ontologies (apart from ThinkHome) selected in Step 2 (see Table 7.2) were added to and linked with each OEMA domain ontology according to the identified knowledge correspondences.

Most of the ontology elements were added manually with OEMA ontologies, since part of the knowledge correspondences between ontologies were identified manually in Step 2. In particular, the following techniques were applied.

1. *Specialization*: adding reused classes and properties of one ontology as subclasses and subproperties of another ontology. For example, infrastructure types reused from other ontologies were added as subclasses of the OEMA infrastructure ontology *owl:Infrastructure* class (see Figure 7.10).

2. *Generalization*: adding reused classes and properties of one ontology as superclasses and super-properties of another ontology. For example, classes that represent white goods types (i.e., cleaning devices, cooking devices) reused from other ontologies were added as superclasses of classes that represent specific cleaning and cooking white goods to the OEMA energy and equipment ontology (see Figure 7.11).

3. *Equivalence*: equivalence relations were established between classes and properties of OEMA ontologies and other energy ontologies that represent the same knowledge. All these equivalence relations were defined and stored in an .owl document that was published along with the ontology. These equivalence relations enable the knowledge exchange between legacy applications that use the vocabularies of existing ontologies.

4. *Import*: some of the selected classes of reused ontologies are described with more classes, properties and axioms. These classes and the associated ontology elements that describe them were extracted from the reused ontologies with the *NeOn toolkit ontology module extraction plugin* [45]. This tool enables the semi-automatic extraction of modules from an ontology at different recursion levels, based on different ontology module extraction operators and in an interactive way. By applying this tool, the selected classes and the associated ontology elements were extracted into a ontology modules. In that way, the knowledge was not lost when reusing the selected ontology elements from the reused ontologies. The obtained modules were directly imported into OEMA ontologies.

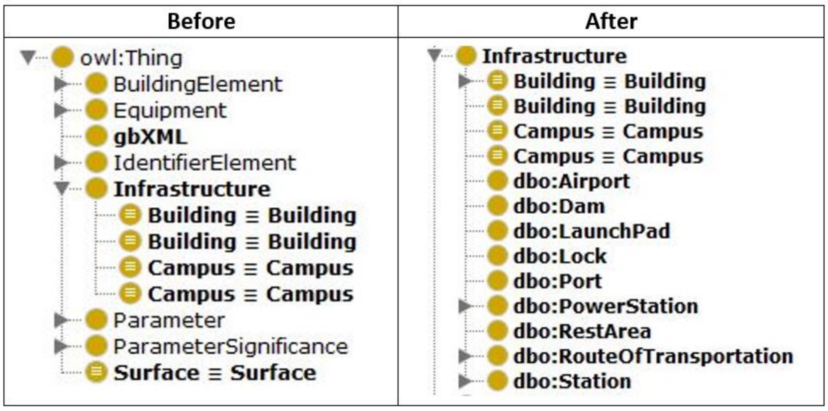


**Figure 7.10:** OEMA specialization

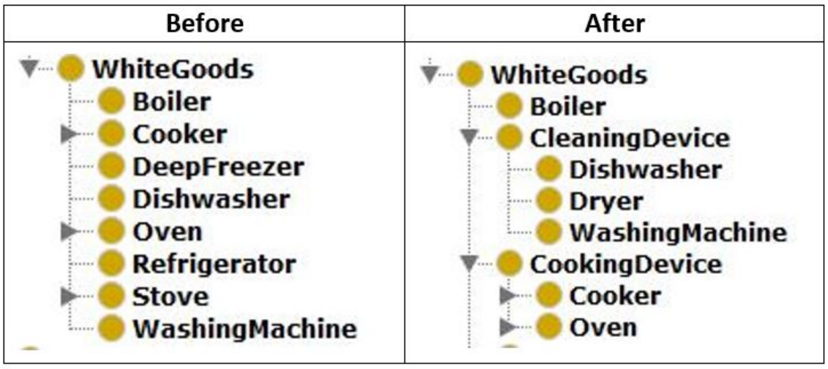


**Figure 7.11:** OEMA generalization

In addition, the DERI Linked dataspace includes small ontologies that cover specific topics. Hence, these ontologies were also imported into OEMA ontologies instead of adding specific ontology elements by applying the aforementioned techniques.

#### 7.4.1.4 Step 4: Ontology Reengineering

In this step, a set reengineering activities were performed (1) to complete the linking of the reused ontology elements (activities A and B), (2) to ensure that the represented knowledge includes unique vocabularies and follows a unique structure (activity C) and (3) to facilitate the reuse of the ontology knowledge in the second iteration of DABGEO development process, as well as the reuse of the knowledge in any knowledge-based application (activity D).

The following ontology reengineering activities were performed in this step:

A. *Knowledge extension*: after implementing OEMA, additional relations between OEMA and reused elements were identified apart from the correspondences identified in Step 2. This and the next activity (activity B) were performed to link the ontology elements according to the identified relations. This activity deals with creating new classes and properties for relating the OEMA elements with elements from reused ontologies. For example, when adding infrastructure premises (i.e., residential premises, business premises) to the OEMA infrastructure ontology, the *owl:hasPremise* property was created. This property relates infrastructures with their premises.

B. *Changing property domains and ranges*: adding new domains and ranges to properties of reused ontologies. For example, the *owl:containsBuilding* property from OEMA infrastructure ontology was changed to have also the *owl:Infrastructure* class as a domain.

C. *Renaming ontology elements*: renaming reused ontology classes or properties by changing their identifiers according to the CamelCase notation. This change was done to preserve a unique naming notation among the reused ontology elements.

D. *New ontologies creation*: the DBpedia ontology includes many elements that represent knowledge about geographical locations, persons and Organizations. Adding all these elements to any of the OEMA ontologies would hinder the ontology reuse. Hence, two new ontologies were created: *OEMA geographical ontology* and *OEMA person and organization ontology*. The first ontology reuses the ontology elements from DBpedia and other ontologies that represent the knowledge about geographical locations. The second ontology reuses the ontology elements from DBpedia and other ontologies that represent the knowledge about persons and Organizations.

   Secondly, some units of measure (i.e., volume, currency) are linked with concepts represented in different OEMA ontologies. Thus, a new ontology was created in order to modularize units of measure: the *OEMA units ontology*. With this modularization, any change in these units of measure data will only affect to the OEMA units ontology. This ontology is imported by other OEMA domain ontologies.

Table 7.3 shows which ontology reuse techniques and which ontology reengineering techniques were applied to restructure the knowledge of each OEMA domain ontology respecting to each energy subdomain.

| Ontology | Energy subdomain | Ontology reengineering activities |
|---|---|---|
| OEMA energy and equipment ontology | Energy consumption systems data | A,B |
| | Distributed Energy Resources data | A |
| | Sensors/actuators data | A |
| | Energy usage data | A,B |
| | Energy Key Performance Indicators | A,B |
| | Device operation data | A,B,C |
| OEMA infrastructure ontology | Infrastructure structural and operation data | A,B,C,D |
| OEMA Smart Grid stakeholders ontology | Individual actor (home user and building occupant) data | F |
| | Energy market roles | A,D |
| OEMA person and organization ontology | Individual actor (Person and employees) data | A |
| | Organization data | A,B |
| OEMA external factors ontology | Weather/climate data | A,B |
| | Environmental data | A,B |
| | Socio-economic data | A,B,D |
| OEMA units ontology | Units of measure | B |

**Table 7.3:** Applied ontology reuse techniques and ontology reengineering activities

Finally, OEMA domain ontologies were imported into a single ontology to enable a joint reuse of all the ontologies and the domains they represent. Top-level relationships were identified between the top-level concepts of OEMA ontologies i.e., infrastructure, place, energy equipment), which were linked through top-level properties according to the identified relationships.

#### 7.4.1.5 Step 5: Ontology Evaluation

OEMA was evaluated following the guidelines proposed by Radulovic et al. [132]. On the one hand, the ontology syntax and structure was evaluated with the OOPS! Pitfall Scanner [129]. The OOPS! Pitfall Scanner detects common pitfalls made during the ontology development process. These pitfalls go from simple mistakes such as missing annotations to important (i.e., lack of disjoint axioms) or critical mistakes, i.e., class cycles. According to OOPS! Pitfall Scanner feedback, OEMA ontologies' pitfalls were corrected. On the other hand, the logical consistency of OEMA domain ontologies was evaluated. According Vrandečić [163], an ontology is logically consistent when it does not contain contradictory knowledge. Specifically, the Pellet ontology reasoner [145] was used to check the consistency of OEMA.

#### 7.4.1.6 Step 6: Ontology Documentation and Publication

The last step of OEMA development process was its documentation and publication. Firstly, the human-oriented documentation of OEMA was created semi-automatically with the Widoco[6] ontology documentation tool. Secondly, the ontology, its documentation and the ontology file that contains the equivalence relations between the elements of existing energy ontologies were published in an Apache Server. The documentation and specification of the OEMA ontology network can be found online[7].

#### 7.4.1.7 Description of the OEMA Ontology Network

In this subsection, the OEMA ontology network is described as a result of the ontology development process described in previous subsection. Each ontology represents one or more energy domains. These ontologies are connected by a core ontology. OEMA top-level structure is shown in Figure 7.12.

- *OEMA infrastructure ontology*: represents knowledge about Infrastructures/buildings. This knowledge includes infrastructure/building types (i.e., household, microgrid, and power station), technical data (i.e., material, surface, and orientation), spaces data (i.e., floors, rooms), geometrical data (i.e., floor area), external and internal equipment (i.e., light control, hydronic loops, furniture) and internal and external environmental conditions, i.e., occupation, internal temperature.

[6] https://github.com/dgarijo/Widoco

[7] http://www.purl.org/oema

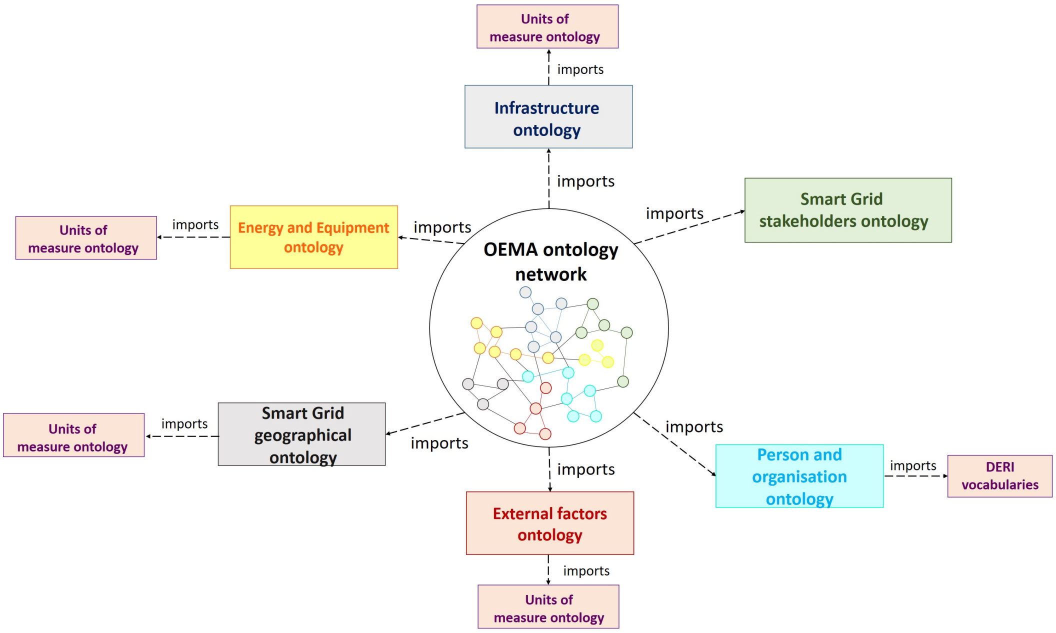

Units of measure ontology
imports
Infrastructure ontology
imports
Smart Grid stakeholders ontology
imports
Units of measure ontology
imports
Energy and Equipment ontology
imports
OEMA ontology network
Units of measure ontology
imports
Smart Grid geographical ontology
imports
imports
imports
Person and organisation ontology
imports
DERI vocabularies
External factors ontology
imports
Units of measure ontology

**Figure 7.12:** Structure of the OEMA ontology network

- *OEMA energy and equipment ontology*: represents energy equipment such as building automation system resources (sensors, actuators/controllers and HVAC systems), industrial equipment (i.e., construction and manufacturing equipment), energy generators (i.e., electric vehicles, Home Power Plants), Loads (white and brown goods), power storage/energy carriers (i.e., gas energy carriers, electrical batteries) and wearable devices. The ontology also represents knowledge about DR events, flex-offers, devices power curve and power profile, device operation category (i.e., on-off device, finite state machine), energy sources (renewable and non-renewable), devices consumption category and device state.

- *OEMA geographical ontology*: represents geographical knowledge about infrastructures and energy equipment locations. This knowledge includes the following: populated places (i.e., country, city, and district), natural places (i.e., mountain, sea), other places (i.e., protected area) and places geographical attributes, i.e., altitude, depth, area.

- *OEMA external factors ontology*: represents external factors that can influence in energy usage. These factors include climate type (i.e., alpine, continental), climatic index (i.e., rain index), environmental conditions (i.e., lighting, noise, air pollutants), pollutant indicators (i.e., pollutant level, pollutant limit value), household socio-economic factors (i.e., household income, housing price), people socio-economic factors (i.e., salary, education level), population socio-economic factors (i.e., density, main origin, mean income), weather phenomenon (i.e., temperature, precipitation), weather reports, and weather state, i.e., rainy, sunny.

- *OEMA person and organization ontology*: represents knowledge about persons and organizations. This knowledge includes person and person attributes (i.e., age, gender), organization, organization internal structure (i.e., departments), organization processes (i.e., business processes, projects), projects economic data, Organizations economic data (i.e., endowment, net income, etc.), business processes energy consumption amount, person roles in Organizations (i.e., role in project, occupation), person travels and means of transport. It imports DERI dataspace vocabularies that represent additional concepts about organizations, i.e., business processes, projects financial data.
- *OEMA Smart Grid stakeholders ontology*: represents Smart Grid stakeholders and roles in the energy market (i.e., energy consumers, energy suppliers, DSOs) and energy flexibility operations, i.e., market processes, flex-offers exchange.
- *OEMA units ontology*: This ontology represents different units of measure used by the OEMA domain ontologies. These units of measure include the following: energy units, area units, capacity units, currency, density units, emission units, lighting units, length units, power units, pressure units, temperature units, volume units and weight units. The OEMA units of measurement ontology is reused by OEMA infrastructure, energy and equipment, geographical and external factors ontologies.

### 7.4.2 Second Iteration : DABGEO Ontology Development

In the second iteration, the knowledge represented by OEMA was reused and implemented according to the layered structure obtained in Chapter 6 by applying the MODDALS methodology. The result of this iteration was the DABGEO ontology. This subsection describes the steps in this iteration (Fig. 7.13).

Below we summarize how these steps were conducted:

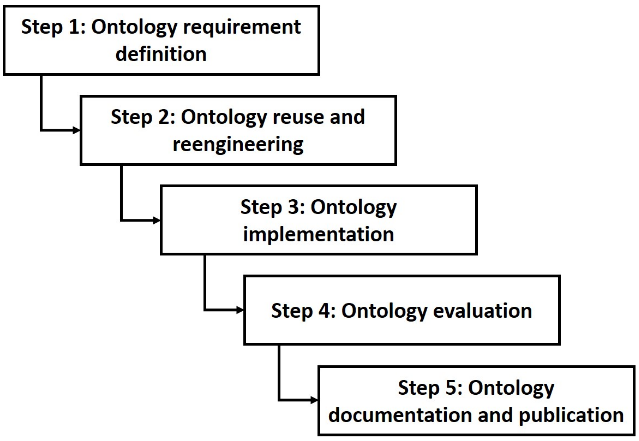

Step 1: Ontology requirement definition
Step 2: Ontology reuse and reengineering
Step 3: Ontology implementation
Step 4: Ontology evaluation
Step 5: Ontology documentation and publication

**Figure 7.13:** DABGEO development process

#### 7.4.2.1 Step 1: Ontology Requirement Definition

The ontology requirements of DABGEO were defined by domain experts and ontology engineers. Regarding the functional requirements, DABGEO represents the energy domains represented by OEMA. The main additional non-functional requirement of DABGEO is that it must follow the structure obtained by applying MODDALS.

#### 7.4.2.2 Step 2: Ontology Reuse and Reengineering

The knowledge of OEMA was reused to develop DABGEO, since it provides a common representation of the energy domains. OEMA knowledge had to be modularized by ontology engineers to adapt it to DABGEO structure. In particular, *NeOn toolkit ontology module extraction plugin* [45] was used to extract from OEMA the modules defined in DABGEO layered structure. Then, the extracted modules were analysed to check whether knowledge was missing or they contained knowledge from other modules. If that was the case, the knowledge was reclassified among the modules to adjust them to DABGEO structure.

#### 7.4.2.3 Step 3: Ontology Implementation

DABGEO was implemented by ontology engineers. In particular, the ontology modules were related so that each module includes one or more modules according to DABGEO structure.

#### 7.4.2.4 Step 4: Ontology Evaluation

The syntax and logical consistency of DABGEO modules were checked with the OOPS pitfall scanner and the Pellet reasoner respectively.

#### 7.4.2.5 Step 5: Ontology Documentation and Publication

The last step of DABGEO development process was its documentation and publication. Firstly, the human-oriented documentation of DABGEO was created semi-automatically with the Widoco ontology documentation tool. Secondly, the ontology and its documentation were published in an Apache Server. In particular, the documentation of DABGEO includes an explanation about DABGEO structure and a description of each ontology layer and the modules it includes.

In addition, the ontology file that links the knowledge between OEMA and existing energy ontologies created during the Step 3 of the OEMA development process was published along with DABGEO.

## 7.5 Evaluation of DABGEO

In this section we report on the evaluation of DABGEO. The choice of a suitable ontology evaluation approach depends on the purpose of validation, as well as the aspect of the ontology we are trying to evaluate [165]. The main objective of DABGEO is to include a common energy domain representation and to provide a balance between reusability and usability, so that its common vocabularies can be reused in different applications with moderate effort.

Hence, the evaluation of DABGEO has focused on determining whether it provides a balance of reusability-usability. The reusability of an ontology is demonstrated by reusing it in different applications that belong to different application types/contexts [112], as was done with well-known reusable ontologies developed in other domains (i.e., [19, 139]). The usability is demonstrated by showing that the ontology reduces the reuse effort in different applications [112]. Therefore, the evaluation of MODDALS has focused on answering the following RQs:

- Can DABGEO be reused in energy management applications that operate in different scenarios?

- Does DABGEO reduce the ontology reuse effort in different energy management applications compared to a global ontology that was not designed to prioritize the balance of reusability-usability?

To answer these questions, we conducted an experiment to measure the DABGEO ontology reuse effort in two knowledge-based energy management applications that operate in different Smart Grid scenarios. Two ontology engineers reused separately the elements of DABGEO to develop application ontologies that satisfy the knowledge requirements of each application. One of the ontology engineers was part of the ontology development team, while the other was not. Regarding their background, both engineers have knowledge about ontology engineering and have previously contributed in the development of ontologies for specific applications. They have also worked in projects related to data representation within the energy domain.

The ontology reuse effort of DABGEO was compared with the reuse effort of a global energy ontology which does not prioritize the balance of reusability-usability. In this experiment, we want to show the benefits that the DABGEO structure brings to the ontology reuse process in comparison to the design of previous global energy ontologies. Therefore, since the OEMA ontology network represents the same knowledge as DABGEO with a similar level of documentation, the effort of reusing DABGEO in the energy management applications was compared with the effort of doing so with OEMA. In this way, we prevented other factors apart from the ontology design that affect the ontology reuse effort (i.e., ontology documentation [12] or represented knowledge) from influencing the result of the experiment.

Finally, the ontology engineers who reused DABGEO and OEMA were provided with a questionnaire to give feedback about DABGEO. The questionnaire was performed to identify the main the main benefits DABGEO brings to the ontology reuse process. These benefits were taken as reference to demonstrate the balance of reusability-usability provided by DABGEO along with the ontology reuse results. In addition, through the questionnaire we identified the main aspects that should be improved in future versions of DABGEO.

The following subsections describe the conducted experiment. They describe (1) the energy management applications where DABGEO and OEMA were reused, (2) how the ontology reuse of both energy ontologies in each application was performed, (3) how the effort of this process was quantified and (4) the results of the reuse process.

It is worth mentioning that the objective of this evaluation is to analyse the ontology reuse effort in both applications; not to show how DABGEO can be reused in different applications. Therefore, we explain and analyse the time and the main activities required to reuse the DABGEO and OEMA ontologies, rather than explaining the ontology reuse process.

### 7.5.1 Energy Management Applications

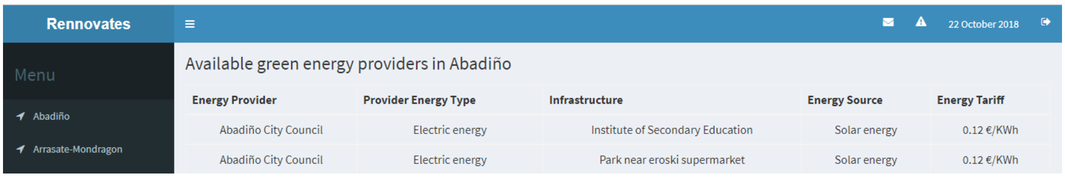
The purpose of the experiment is to analyse the reuse effort of both ontologies in these applications. Therefore, only a high-level description of each application main functions is provided, without going into technical or implementation details. A detailed description of these applications can be found in appendixes A and B respectively.

The first application corresponds to a pilot demonstrator of a Green Energy Provider Selection System (GEPSS) for smart homes. The second application corresponds to a pilot demonstrator of an infrastructure EPAS for green buildings self-sufficient using solar energy. The data used by both systems are represented with the vocabularies of ontologies developed for both applications. The ontology of each application was developed reusing the knowledge of DABGEO. The GEPSS and EPAS are part of a pilot demonstrator of the Rennovates European project[8] deployed by Mondragon University in Abadiño and Oñati municipalities and Urkiola natural park (Spain). Both GEPSS and EPAS systems are briefly described below:

- The *GEPSS system* is a multi-agent system for smart homes that provides a list of the available green energy providers to the home energy consumer in the area where the home is located (the available energy providers are those that have surplus energy). Specifically, the system displays the provided energy type (i.e., electric energy, thermal energy), the energy source (i.e., solar power, wind power), the infrastructure that generates the energy (i.e., a solar panel installation) and the price at which the supplier sells the energy. Figure 7.14 shows an example of how this information is displayed.

- The *EPAS system* is a multi-agent system that provides a holistic view of the energy generation performance of green buildings self-sufficient in solar energy. Specifically, it displays the recent energy consumption, as well as the short-term weather and energy production forecast. It also represents the energy generation of an infrastructure of a certain period and the forecasted energy generation of that period. With this information, the user can take actions to reduce the building energy consumption and adjust it to the infrastructure energy performance forecast. The user can also analyse any gap between the real and forecasted energy generation to detect any problem in the energy generation devices. In addition, if the gap is minimal, the user can take previous predictions as reference to make long-term energy generation forecasts. Figure 7.15 shows an example of how this information is displayed.

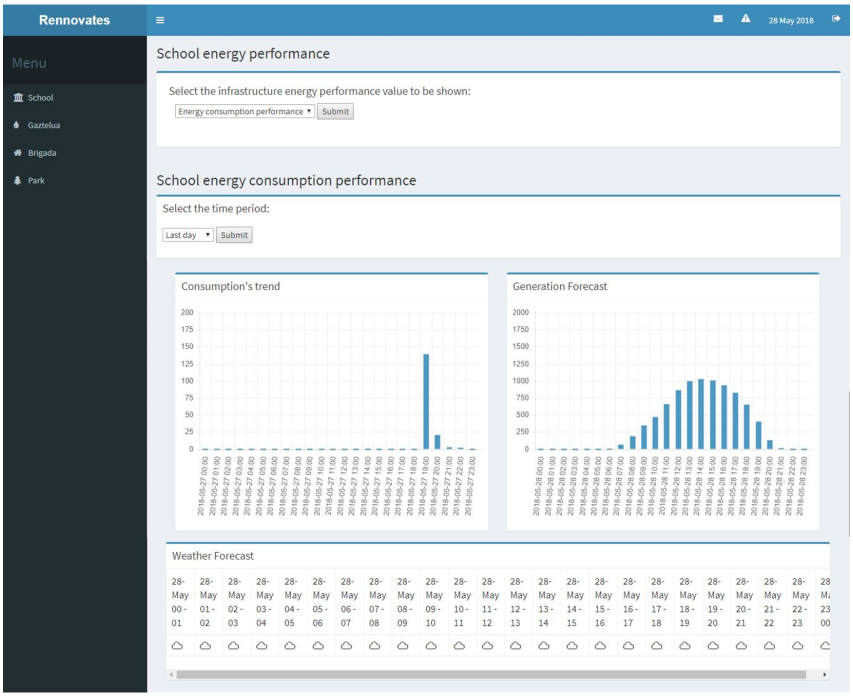
[8] https://rennovates.eu/



Available green energy providers in Abadiño

| Energy Provider | Provider Energy Type | Infrastructure | Energy Source | Energy Tariff |
|---|---|---|---|---|
| Abadiño City Council | Electric energy | Institute of Secondary Education | Solar energy | 0.12 €/KWh |
| Abadiño City Council | Electric energy | Park near eroski supermarket | Solar energy | 0.12 €/KWh |

**Figure 7.14:** Data displayed by the GEPSS



School energy performance

Select the infrastructure energy performance value to be shown:

Energy consumption performance ▾ Submit

School energy consumption performance

Select the time period:

Last day ▾ Submit

Weather Forecast

| 28- May 00 - 01 | 28- May 01 - 02 | 28- May 02 - 03 | 28- May 03 - 04 | 28- May 04 - 05 | 28- May 05 - 06 | 28- May 06 - 07 | 28- May 07 - 08 | 28- May 08 - 09 | 28- May 09 - 10 | 28- May 10 - 11 | 28- May 11 - 12 | 28- May 12 - 13 | 28- May 13 - 14 | 28- May 14 - 15 | 28- May 15 - 16 | 28- May 16 - 17 | 28- May 17 - 18 | 28- May 18 - 19 | 28- May 19 - 20 | 28- May 20 - 21 | 28- May 21 - 22 | 28- May 22 - 23 | 28 Ma 23 00 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ |

**Figure 7.15:** Data displayed by the EPAS

### 7.5.2 Ontology Reuse Process and Quantification

The ontology reuse process was performed and its effort quantified by taking as reference the ONTOCOM ontology engineering cost model [144, 143]. ONTOCOM is applied to estimate the ontology building, reuse and maintenance effort.

Firstly, the ontology engineers were provided with the knowledge requirements of each application ontology in the form of CQs. The list of CQs defined for each application can be found in appendixes A and B respectively.

Then, DABGEO and OEMA were reused to develop GEPSS and EPAS ontologies, which were developed with Protégé. It is important to mention that the ontology engineers did not have the chance to develop the GEPSS and EPAS ontologies from scratch or reusing other ontologies, since the experiment was conducted to compare

DABGEO and OEMA ontology reuse effort. DABGEO and OEMA were reused following the main phases of ontology reuse defined by the ONTOCOM model [12]:

- **Ontology understanding and evaluation:** is the process of getting familiar with an ontology to understand the knowledge it represents and to determine if it satisfies the application ontology requirements. In this phase, the ontology engineers analysed the description and specification of DABGEO ontology modules. Those modules that meet the requirements of GEPSS and EPAS ontologies were selected to be reused.

- **Ontology customization:** the process of adapting an ontology to the application requirements by performing changes in the ontology elements and adding new knowledge. In this phase, the ontology engineers reused the selected ontology modules to develop GEPSS and EPAS ontologies. They had the option to reference the elements of the selected ontology elements or to import the selected ontologies as a whole. During the ontology reuse process, the ontology engineers conducted the necessary ontology reengineering activities to adapt the reused knowledge to the requirements of the developed ontologies. In particular, they conducted the following reengineering activities: ontology module extraction, property domain and ranges modification, class hierarchy restructuring, axiom modification and knowledge addition. For a more detailed definition of each ontology reengineering activity the reader should refer to [150].

The DABGEO and OEMA ontologies were reused in GEPSS and EPAS applications following the aforementioned phases. Ontology development methodologies that guide the reuse of existing ontologies to develop new ones include preliminary phases such as ontology evaluation and selection for reuse [150]. These phases involve the analysis of existing ontologies to select for reuse the ones closer to the requirements of the application ontology to be developed. However, in the DABGEO evaluation experiment both DABGEO and OEMA were already selected for reuse to analyse their ontology reuse effort. Therefore, both ontologies come from selected energy ontologies and the preliminary ontology reuse phases were already completed.

Each ontology engineer conducted the ontology reuse phases to develop the ontologies of the GEPSS and EPAS systems twice: reusing OEMA and reusing DABGEO. Once DABGEO or OEMA have been reused in one application, the ontology reuse process of the other ontology in the same application will be simplified because the second time the application ontology is developed the ontology requirements are known. Hence, it should be noted that OEMA and DABGEO were reused in different order in each application to minimise the impact of this aspect in the experiment. In particular, the ontology understanding and evaluation phase was firstly conducted with OEMA in the GEPSS ontology reuse process, while this phase was firstly conducted with DABGEO in the EPAS ontology reuse process. The ontology customization phase was conducted in the opposite order.

Finally, it is also worth mentioning that the ontologies developed by the ontology engineers were evaluated to check whether they met the GEPSS and EPAS requirements. The developed ontologies were loaded into the semantic repository. Then, SPARQL queries were executed against the semantic repository to check whether the obtained results were correct and the ones required by the GEPSS and EPAS systems.

### 7.5.3 Ontology Reuse Effort Quantification

The effort of performing any ontology activity is the time required to complete the activity [11]. Considering this, the time required to perform the each ontology reuse phase was measured to quantify the ontology reuse effort.

The main cost drivers of ontology reuse process affected by the reused ontology design were also measured to analyse the impact of the ontology design of DABGEO and OEMA on the ontology reuse effort. The purpose was to determine whether DABGEO is likely to keep moderate the ontology reuse effort in other applications apart from the GEPSS and EPAS. According to the ONTOCOM model, the main reuse cost drivers affected by the ontology design are *ontology complexity* and *performed ontology reengineering activities* [12].

The ontology complexity is quantified with a set of metrics that measure ontology design features that hamper the ontology reuse process. The following are the main ontology complexity metrics [12]:

- **Size of vocabulary (SOV):** the number of elements (classes, properties, instances and axioms) that the ontology contains.
- **Vocabulary expressiveness (VE):** the VE is simple if it contains class hierarchies, moderate if it adds property types or complex if it adds restrictions and axioms.
- **Ontology graph structure:** measures how far the ontology class hierarchy is from a tree structure. The tree structure of the ontology can be measured with the Tree Impurity (TIP) metric proposed by [171].

In addition, Zhang et al. [171] proposed the following complexity metric:

- **Ontology edge node ratio (ENR)**: measures the ontology density taking as reference the percentage of relations between the ontology elements.

Considering this, the ontology complexity was quantified by applying the aforementioned metrics. A high value in these metrics means that the reused ontology is complex. It also means that the ontology design is likely to (1) hamper the ontology understandability, (2) to increase the number of changes required to adapt the reused ontology and (3) to increase the effort required to analyse the impact of the changes performed in the ontology during the ontology customization phase [171].

There are complexity metrics that measure more complex aspects of the ontology design (i.e., the complexity of each ontology class) [171]. In the DABGEO evaluation experiment, the purpose of measuring the ontology complexity is not to perform an exhaustive analysis of the complexity of both ontologies but to check whether the ontology complexity of DABGEO and OEMA influenced in the ontology reuse effort. Therefore, in the experiment we focus only on the main ontology complexity metrics.

Regarding the performed ontology reengineering activities, we analysed the reengineering activities applied by the ontology engineers to adapt each reused ontology to the application ontologies of the GEPSS and EPAS. We also analysed the number of ontology elements affected by the ontology reengineering activities.

### 7.5.4 Results

The following subsections show the results of the ontology reuse process as well as ontology engineers' feedback about DABGEO.

#### 7.5.4.1 Ontology Reuse Effort

Table 7.4 shows the average time that each ontology reuse phase took the ontology engineers when reusing each ontology in both energy management applications. The results show that the ontology reuse phases took less time with DABGEO than OEMA in both energy management applications. In particular, DABGEO reduced the ontology reuse time in 0.4 and 1.1 person-hours in the GEPSS and EPAS respectively.

As for ontology reuse cost drivers, Table 7.5 compares the average values of ontology complexity metrics of the OEMA domain ontologies and DABGEO modules. In addition, it shows the average complexity of the modules of each DABGEO layer.

| | OEMA ontology network | | DABGEO ontology | |
|---|---|---|---|---|
| | GEPSS | EPAS | GEPSS | EPAS |
| Ontology understanding and evaluation phase | 1.7 person-hours | 1.3 person-hours | 1.6 person-hours | 1.6 person-hours |
| Ontology customization phase | 2.9 person-hours | 2.8 person-hours | 2.6 person-hours | 1.4 person-hours |
| **Total time** | **4.6 person-hours** | **4.1 person-hours** | **4.2 person-hours** | **3 person-hours** |

**Table 7.4:** Average ontology reuse effort

| Ontology / Complexity metric | OEMA ontology network | DABGEO ontology | Common-domain layer | Variant-domain layer | Domain-task layer |
|---|---|---|---|---|---|
| **SOV** | 1012.00 | 175.83 | 104.76 | 133.06 | 248.98 |
| **ENR** | 1.58 | 1.35 | 1.37 | 1.33 | 1.41 |
| **TIP** | 549.00 | 34.34 | 21.88 | 42.31 | 48.8 |
| **VE** | Complex | Moderate | Moderate | Moderate | Complex |

**Table 7.5:** Ontology complexity values

It is worth mentioning that Protégé was used to calculate SOV and VE metrics, since it provides the values of these metrics. The calculation of the rest of the metrics was performed with the Jena API [106]. As can be seen in Table 7.5, DABGEO reduced all the complexity metrics with respect to OEMA. In addition, the results evidence that the ontology complexity increase in the domain-task layer, which includes the knowledge reused in specific Smart Grid scenarios.

Figures 7.16 and 7.17 show the ontology reengineering activities needed by the ontology engineers to adapt each reused ontology to the GEPSS and EPAS ontology requirements respectively. Figures 7.16 and 7.17 also show the average number of ontology elements affected by the ontology reengineering activities. These elements can be ontology modules (in the case of the ontology module extraction activity), classes, properties and axioms.

In general terms, the reuse of both ontologies required similar ontology reengineering activities and these activities affected to a similar number of ontology elements. The case of ontology module extraction and ontology pruning activities was different. *Ontology module extraction* is the task of extracting small modules from the reused ontology to obtain only the knowledge necessary to satisfy the application ontology requirements [45]. *Ontology pruning* deals with deleting from

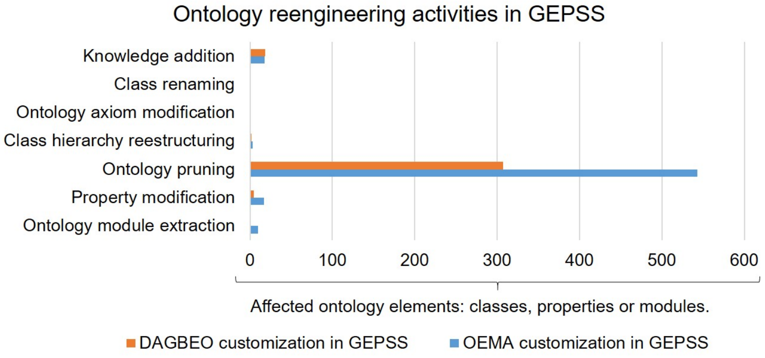


**Figure 7.16:** Ontology reengineering activities in GEPSS

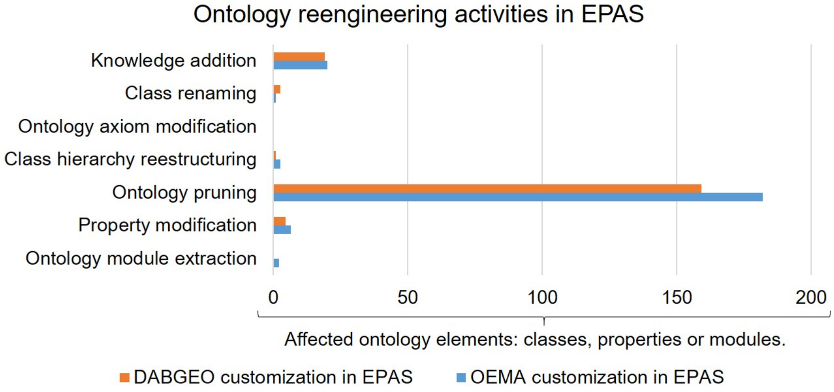


**Figure 7.17:** Ontology reengineering activities in EPAS

the reused ontology the elements unnecessary in the application ontology to be developed [150]. The customization of OEMA required to extract ontology modules and to prune more ontology elements. Both activities are usually performed when reusing large ontologies as they deal with discarding the unnecessary knowledge to improve computation performance. Thus, depending on the size of the reused ontology, both activities may require a significant effort.

#### 7.5.4.2 DABGEO Ontology Feedback

According to the ontology engineers, the benefits of DABGEO structure are the following:

- *Selection of knowledge at the proper level of abstraction*: the abstraction layering and the specificity of DABGEO modules enables to select at the proper level of abstraction the necessary knowledge to develop the application ontologies. Thus, ontology engineers do not need to remove too specific or too abstract knowledge when discarding the knowledge unnecessary for their applications.

- *Simplification of the analysis of the impact of ontology customization changes*: DABGEO modules are not very complex and are linked with other ontology modules only when necessary. Thus, the analysis of the impact of ontology customization changes in the rest of the ontology is simpler than in OEMA.
- *Easy adaptation to changes in application requirements*: if the developed application requires new knowledge (i.e., due to new functionalities it must provide or due to changes in requirements), the ontology engineer can select and reuse at the proper level of abstraction only the modules that satisfy the new knowledge requirements. Thus, DABGEO not only reduces the ontology reuse effort during application ontology development, but also during the application ontology maintenance.
- *High control of ontology changes*: the low complexity of DABGEO modules, along with abstraction layering, provides more control when performing the ontology pruning process. Hence, the possibility of removing the knowledge useful for the application ontology is reduced.

On the other hand, the following are the aspects of DABGEO to be improved:

- In DABGEO, the ontology modules are classified into domain/subdomains and abstraction layers. Hence, when reusing DABGEO, ontology engineers do not need to analyse all the modules. They can focus on analysing only the modules of the subdomains related with the requirements of the application ontology. Then, ontology developers can analyse the ontology modules that may meet the application ontology requirements. They can start analysing the modules of the domain-task layer that belong to similar applications to the one to be developed and continue analysing the modules from upper layers, while discarding the rest of modules. However, DABGEO encompasses many modules. Ontology developers may have to download and analyse multiple ontology modules to understand and analyse in detail the knowledge that can be reused to develop the application ontology. This process involves an extra effort to understand the DABGEO knowledge when it is reused for the first time.

## 7.6 Discussion

In this section, we discuss whether DABGEO provides a balance of reusability-usability based on the results of the experiment conducted in Section 7.5.

On the one hand, **DABGEO could be adapted to fit the requirements of various energy management applications deployed in different Smart Grid scenarios.** Hence, **we can state that the ontology has proven to be reusable for the energy domain.**

As shown in 7.5.4.1, the DABGEO ontology reduced the values of the main ontology complexity metrics with respect to the OEMA. In addition, the complexity of DABGEO ontology modules increases in low level layers. This shows that DABGEO separates the knowledge common to the Smart Grid scenarios, while preserving knowledge the reused by specific energy management applications, which is more complex. In other words, DABGEO reduces the overall complexity with respect to OEMA while preserving the same knowledge as OEMA. The low complexity of DABGEO eases the ontology understanding, reuse and maintenance with respect to OEMA [171, 12]. Apart from the complexity, DABGEO also reduced the ontology reengineering activities required to adapt the DABGEO ontology to GEPSS and EPAS knowledge requirements.

As stated in 7.5.4.2, the first time DABGEO is reused it requires an additional effort to analyse the knowledge of each module. Hence, as shown in 7.5.4.1, the ontology understanding and evaluation phase required similar effort in the reuse process of DABGEO and OEMA. Once the initial understanding process of DABGEO is completed, it simplifies the reuse ontology modules at the proper level of abstraction and their customization to meet the requirements of each energy management applications. Therefore, the reduction of ontology reuse cost drivers was reflected in DABGEO reuse effort, which was reduced with respect to OEMA. In addition, if the developed ontology requires new knowledge, ontology engineers can reuse only the necessary knowledge to satisfy the new requirements. They can also reuse only the necessary knowledge to develop new application ontologies. Hence this, the effort reduction will be more remarkable when reusing DABGEO to maintain the application ontology or to develop new ontologies, since in these cases the ontology engineer is almost familiarized with the knowledge represented by each module.

Considering this, **DABGEO reduced the ontology reuse effort in the GEPSS and EPAS, and it is likely to keep moderate the ontology reuse effort in new applications.** Thus, **DABGEO has proven to be usable for specific energy management applications.**

In conclusion, **the experiment shows that DABGEO could be reused in two**

**energy management applications reducing the ontology reuse effort, thus providing a balance between reusability and usability.**

## 7.7 Conclusions

In this chapter we have presented DABGEO, a reusable and usable global ontology for the energy domain. DABGEO can be reused by ontology engineers to develop ontologies for specific energy management applications. It classifies the domain knowledge into different abstraction layers. With this structure, DABGEO aims to provide a balance between reusability-usability to reduce the ontology reuse effort in different energy management applications. The development process of DABGEO encompassed two iterations, each of which produced as output an implemented version of DABGEO:

1. Unification of the knowledge of existing heterogeneous energy ontologies into a global energy ontology that provides a common representation of energy domains.

2. Classification of the common and variant domain knowledge into different abstraction layers and ontology modules according to the layered structure obtained by applying MODDALS.

Different ontology engineers reused DABGEO in two energy management applications to demonstrate its balance of reusability-usability. DABGEO reuse effort was compared with the effort of reusing another global energy ontology which does not prioritize the balance of reusability-usability: the OEMA ontology network. The results show that DABGEO could be adapted to both applications. In addition, DABGEO reduced with respect to OEMA the ontology reuse time (0.4 and 1.1 person-hours in each application respectively) and main cost drivers that depend on the ontology design, thus providing the following benefits to the ontology reuse process: (1) selection of knowledge at the proper level of abstraction, (2) simplification of the analysis of the impact of ontology customization changes, and (3) high control of ontology changes. These benefits reduced the overall ontology reuse effort. Therefore, we can state that DABGEO provides a balance of reusability-usability in the energy domain. On contrast, the main improvement aspect of DABGEO is the reduction of the effort needed to understand the ontology knowledge when it is reused for the first time.

Taking into account DABGEO evaluation results, we have validated the ontology and demonstrated second hypothesis (H2) of the thesis: *a layered ontology that provides* *a common energy domain representation reduces the ontology reuse time and cost drivers in comparison to a global ontology without this structure.*

# Part IV

# Final Remarks

Chapter 8

# Conclusions

This chapter summarizes the main contributions and conclusions of this dissertation and outlines the research directions that can be explored in future work. Specifically, Section 8.1 summarizes the contributions and discusses the validation of the hypotheses. Section 8.2 discusses a set of lessons learned we extracted from the thesis. Section 8.3 highlights the main limitations of the proposed solutions. Finally, the future work is exposed in Section 8.4. .

## 8.1 Summary of Contributions

This dissertation presents advances in the state of the art of the design of reusable and usable ontologies. By addressing the challenges in this field, the thesis has also contributed to the advancement in the state of the art of the semantic representation of energy knowledge. In the following subsections we summarize the main solutions developed to address the main challenges in each contribution area.

### 8.1.1 Design of Reusable and Usable Ontologies

An ontology that supports different applications must achieve a balance of reusability-usability so that it can be reused in different applications with moderate effort [112, 83]. This challenge is known as the ontology *reusability-usability tradeoff problem* [90]. Achieving a balance of reusability-usability is particularly important in complex domains.

In Chapter 3, we analysed previously defined reusable and usable ontology design methodologies. Reusable and usable ontology design methodologies provide guidelines to design layered ontologies that provide a balance of reusability-usability. Layered ontologies classify into different abstraction layers the common domain knowledge (reused by most applications) and the variant domain knowledge (reused by specific application types). Such a classification enables ontology developers to reuse only the necessary knowledge at the proper level of abstraction to develop ontologies that satisfy specific application requirements. Hence, the ontology reuse effort in different applications is reduced. During the definition of the layered ontology structure, the classification of the domain knowledge into different layers is performed from scratch by domain experts and ontology engineers in collaboration with application stakeholders. In particular, they analyse in collaboration with stakeholders the theoretical framework and the knowledge requirements of the application types that will be supported by the layered ontology. Hence, the application of these methodologies is time consuming, which hinders the development of reusable and usable ontologies in complex domains.

Layered ontologies are similar in structure to Software Product Lines (SPLs), software families that contain common reusable parts and variable parts that depend on specific customer needs to support mass customization. In both approaches, software features are classified into common and variant to support large-scale software reuse. SPLs are usually designed taking as reference existing applications and legacy systems to classify systematically the software features into common and variant. Specifically, a domain analysis of existing applications is conducted to identify the similarities and differences of their software features. Depending on how many applications implement them, the software features are classified into those common to a set of applications and those only implemented by specific applications. This approach complements the expertise of domain experts and software engineers, thus minimising their involvement and effort when designing the SPL. Given that currently many developed ontologies are available, a domain analysis of existing applications applied to design SPLs can be adapted in the ontology engineering field. This approach would enable a systematic classification of the domain knowledge taking as reference existing ontologies, preventing domain experts and ontology engineers from classifying the domain knowledge from scratch. As far as we know, previous reusable and usable ontology design methodologies do not take advantage of existing ontologies to save effort when designing the layered ontology structure (as SPL design approaches do).

The **MODDALS methodology** (presented in Chapter 4) is the first contribution of the thesis and was defined to address the aforementioned challenge. MODDALS is the result of combining the best practices of the ontology engineering and SPL engineering fields. MODDALS adopts the main activities and ontology design principles applied by previous reusable and usable methodologies to define the layered ontology structure. In contrast to these methodologies, SPL engineering techniques are applied to classify the domain knowledge into common and variant and into the defined layers according to a domain analysis of existing ontologies. This approach complements domain experts' and ontology engineers' expertise and prevents them from classifying the domain knowledge from scratch, facilitating the design of the layered ontology structure. This contribution was published in the *Applied Ontology Journal* [34] and at the *30th International Conference on Software Engineering and Knowledge Engineering* [31].

MODDALS was evaluated in Chapter 6 to determine whether it enables to classify the domain knowledge by taking as reference existing ontologies. As explained in Chapter 5, the Energy is one of the domains where MODDALS is needed. Therefore, domain experts and ontology engineers applied the methodology to design parts of the layered structure of a global ontology for the energy domain called DABGEO. As shown in Section 6.3, they were able to follow MODDALS steps to obtain similar knowledge classifications (the degree of consensus when classifying the domain knowledge into different layers was 76%). Hence, we can conclude that MODDALS enables to classify the domain knowledge by taking as reference existing ontologies.

Bearing in mind these results, we have addressed the main challenge of the thesis identified in Chapter 1: *define a methodology to design layered ontology structures of reusable and usable ontologies from an existing set of ontologies*.

In addition, we proved the first hypothesis of the thesis:

**H1.** Ontology and SPL design techniques applied in combination enable to classify the domain knowledge into different abstraction layers with high level of consensus taking as reference existing ontologies.

### 8.1.2 Semantic Representation of Energy Knowledge

From the beginning of the current decade, Semantic Web technologies have been applied for developing energy ontologies that represent the knowledge from different energy domains. This knowledge is used as knowledge base by specific systems focused on improving current grid sustainability and resilience [32].

In Chapter 5, we analysed previously developed energy ontologies and knowledge-based energy management applications. This analysis was published in a chapter of the book titled *Designing Cognitive Cities* (Springer) [32]. The analysis has revealed that existing energy ontologies apply heterogeneous vocabularies to represent the same energy domains, which hampers the interoperability between energy management applications that operate in different scenarios. This problem hinders the knowledge exchange of knowledge-based sustainability and resilience solutions in real scenarios. Therefore, there is the need to create a global ontology that provides a common energy domain representation. In that way, the knowledge of a global energy ontology could be reused as a base to develop interoperable ontologies that represent the knowledge required by different energy management applications. Although these ontologies may not represent the same knowledge, the knowledge they have in common would be represented with the same vocabularies. Energy domains are complex and a global energy ontology should provide support to a wide variety of energy management applications. Thus, a global ontology should provide a balance of reusability-usability to reduce the ontology reuse effort in different applications.

The **DABGEO ontology** (presented in Chapter 7) is the second contribution of the thesis and was developed to address the aforementioned challenge. DABGEO provides a common representation of the energy domains represented by previously developed heterogeneous energy ontologies. In addition, it includes links between the vocabularies of the existing energy ontologies that represent heterogeneously the overlapping knowledge to enable interoperability between new and legacy knowledge-based energy management applications. DABGEO classifies the energy domain knowledge it represents into different abstraction layers that distinguish between the common and variant domain knowledge to provide a balance of reusability-usability. This contribution was published in the Journal of Web Semantics [33] and at the *2nd International Workshop on Ontology Modularity, Contextuality, and Evolution* (co-located with *International Semantic Web Conference*) [30].

In Chapter 6, DABGEO layered structure was designed by applying MODDALS. In Chapter 7, DABGEO was developed in two iterations. In the first iteration, the knowledge of existing energy ontologies was unified into a single ontology. The result of this iteration was the **OEMA ontology network**. OEMA provides a common knowledge representation of the energy domains represented heterogeneously by existing energy ontologies. In the second iteration, the knowledge of the OEMA was classified into different ontology modules and layers according to the design obtained by applying MODDALS, thus completing the DABGEO development process.

To evaluate the reusability-usability balance of DABGEO, different ontology engineers reused the ontology in two pilot demonstrators of two energy management applications (see Section 7.5). DABGEO reuse effort was compared with the effort of reusing OEMA. The results show that DABGEO could be adapted to each application. In addition, DABGEO reduced with respect to OEMA the average ontology reuse time by 0.4 and 1.1 person-hours in each application respectively.

DABGEO also reduced the main cost drivers for ontology reuse that depend on the ontology design: ontology complexity and the number of performed ontology reengineering activities. Hence, the ontology reuse process was simplified and we can conclude that DABGEO provides a balance of reusability-usability.

Bearing in mind these results, we have proved the second hypothesis of the thesis:

**H2.** A layered ontology that provides a common energy domain representation reduces the ontology reuse time and cost drivers in comparison to a global ontology without this structure.

## 8.2 Lessons Learned

This section summarizes lessons learned from the research carried out during this Ph.D. thesis. These lessons can be employed as a guidelines either, by researchers or industrial practitioners.

1. *The application of SPL and ontology design techniques in combination is a useful and practical method to design reusable and usable ontologies in complex domains:* the application of MODDALS to design DABGEO layered structure enabled domain experts and ontology engineers to exploit the knowledge from existing ontologies when designing the layered structure. They did not need to analyse the knowledge requirements of different applications to classify the ontology domain knowledge from scratch, facilitating the design of the layered ontology structure. Bearing in mind these benefits, ontology developers should consider applying SPL engineering techniques when designing ontologies that (1) will be reused in different applications and (2) represent complex domains. In particular, existing ontologies should be analysed and their knowledge should be divided and classified into different abstraction levels. Then, the ontologies should be analysed to classify the domain knowledge based on their knowledge similarities and differences.

2. *The layered ontology design approach facilitates the ontology reuse in different applications, especially when the ontology represents the knowledge from complex domains:* in the last decade, several research publications have proposed layered ontologies as an ontology design approach for ontologies that are reused in different applications [146, 112, 154]. Layered ontologies provide a balance between reusability-usability, thus facilitating the ontology reuse in different applications. In this thesis we have demonstrated the usefulness of

this design approach in large-scale ontologies developed in complex domains. In particular, we developed the DABGEO ontology and we evaluated it by comparing its reuse effort with the OEMA ontology network. The evaluation of DABGEO shows that through its layered structure it facilitates the ontology reuse in different applications. Thus, we highly recommend ontology developers to apply the layered ontology design approach in ontologies that will be reused in different applications (i.e., global ontologies) and that represent complex domains.

## 8.3 Limitations of the Proposed Solutions

In this section we discuss the limitations of the solutions proposed in the thesis.

### 8.3.1 Limitations of MODDALS Methodology

The following are the main limitations of MODDALS methodology:

- As stated in Section 6.3.2, MODDALS guidelines are limited to design the layered structure of the first version of the ontology. As more ontologies are developed for new applications, the layered structure of the ontology may change. MODDALS does not include guidelines to manage these changes.

- In the preliminary step of MODDALS (see Section 4.3.1), the available ontologies developed for specific applications in the domain concerned are selected. The knowledge from these ontologies is taken as reference to classify the knowledge of the layered ontology. When selecting the existing ontologies, their quality is not evaluated, we assume that they are developed in collaboration with domain experts.

- In the Step 2 of MODDALS (see Section 4.3.3), the Competency Questions (CQs) answered by existing ontologies are taken as reference to divide their knowledge, which is later classified into different layers. As explained in Section 2.5, the CQs are the queries that the ontologies should be able to answer when applications access to the knowledge they represent. Ideally, the CQs answered by each ontology should be available and open. However, in MODDALS we assume that the CQs answered by available ontologies are not always available [136]. Therefore, domain experts and ontology engineers analyse the elements of existing ontologies to identify the CQs they answered.

- In step 3 of MODDALS (see Section 4.3.4), the knowledge classification into different layers is performed through the application-matrix technique [127]. By applying this technique the knowledge is classified into common or variant (and by extension into different layers) depending on how many ontologies reuse it. As explained in Section 4.3.4, to the best of our knowledge, there is no systematic method to determine the exact threshold value to distinguish the common and variant software features in SPL design. The software features are considered as common if they are present in most of applications [110, 13, 115]. In MODDALS this limitation is also present and the knowledge is classified as common if it is reused by most of application types supported by the ontology. Hence, the number of application types supported by the ontology has influence in the classification.

- MODDALS was evaluated only in the energy domain and the evaluation was limited to its application to design DABGEO. The application of MODDALS in other domains was out of the scope of this work.

- The evaluation of MODDALS was limited to determine whether it enables to classify the domain knowledge by taking as reference existing ontologies. We consider that this approach will require less time and effort than designing the layered ontology structure from scratch, as previous reusable and usable ontology design methodologies do. To demonstrate that MODDALS reduces the effort of designing the layered ontology structure, the time required to apply MODDALS should have been compared with the time required by applying previous reusable and usable ontology design methodologies. However, it is unlikely that anyone will be willing to pay twice for building or designing the same extended ontology using different approaches. In addition, the application of a methodology is a complex process where too many conditions that may affect the evaluation results cannot be controlled [48]. Hence, we assumed this limitation when evaluating the MODDALS methodology.

### 8.3.2 DABGEO Ontology

The following are the main limitations of DABGEO ontology:

- DABGEO includes only the ontology elements from the existing energy ontologies reused in specific applications, since it provides a common representation of these ontologies.

- As stated in Chapter 5, the main objectives of a global energy ontology are to include a common energy domain representation and to provide a balance of reusability-usability. Hence, in this thesis we have focused on developing and evaluating DABGEO to ensure that it meets these objectives. DABGEO also enables interoperability between new and legacy energy management applications supported by heterogeneous ontologies. However, the evaluation and demonstration of DABGEO interoperability was out of the scope of this work.

## 8.4 Future Work

Assuming that our study has some limitations, in this section we outline research directions that can be explored in future work. The following subsections summarize the possible future research directions to improve the main solutions developed in the thesis. Currently, we are working in some of them, specifically in those related with MODDALS methodology.

### 8.4.1 Future Work in MODDALS Methodology

Considering MODDALS evaluation results (see Section 6.3), MODDALS is still a first step towards a widely accepted methodology to design layered ontology structures for reusable and usable ontologies in complex domains. Bearing this in mind, we have defined the possible future research directions to improve MODDALS.

#### 8.4.1.1 Tool Support to Automate the Knowledge Classification Process

As stated by MODDALS evaluation participants (see Section 6.3.2), MODDALS still has room for improvement when it comes to reducing the ontology design effort. Specifically, the analysis conducted in the knowledge classification step to classify the KAs into different layers requires a significant manual effort. The ontology engineers analyse manually each ontology to check whether it answers the CQs encompassed by each KA, and by extension, whether the KA is represented by the ontology. Ontology validation tools [150] that automate this process can help to reduce the effort required to perform the knowledge classification step.

We are exploring the possibility of integrating with MODDALS the Themis ontology validation tool (developed by the Ontology Engineering Group from the Technical University of Madrid[1]) [57]. Themis is used to check (semi) automatically

[1] http://www.oeg-upm.net/

whether a set of CQs are answered by certain ontologies. This tool takes as input the statements (written by the ontology engineer) that answer each CQ and checks automatically whether these statements are represented in the ontology.

Considering these features, Themis can be used to reduce the manual effort required to check the representation of KAs by existing ontologies. In particular, ontology engineers can introduce the statements of the CQs encompassed by each KA to check whether these CQs are answered by each ontology and, by extension, whether the KA is represented.

To check the applicability of Themis in MODDALS, we repeated the experiment of Section 6.3 to design part of DABGEO ontology by applying this tool. Instead of analysing manually whether KAs were represented by energy ontologies, the CQs they encompass were converted into statements to check with Themis whether these CQs were answered by the ontologies. The knowledge classifications results were similar than the ones obtained by the manual analyses conducted by ontology engineers (shown in Section 6.3.1): a similar number of ontology modules were classified into each layer and application type (Fig 8.1) and the degree of consensus when classifying KAs remained stable (76%). In addition, the manual effort was considerably reduced. These results are promising for the automation of MODDALS knowledge classification process.

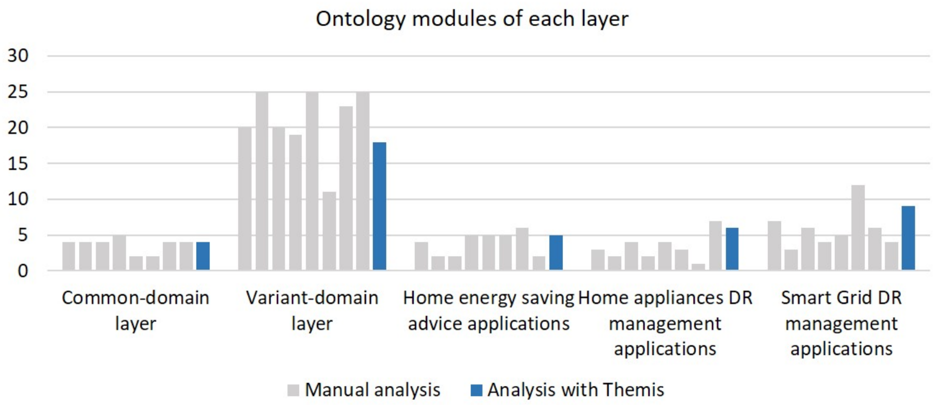


**Figure 8.1:** Knowledge classification with Themis

However, Themis is used to check if an ontology represents a set of statements to verify if the ontology meets the defined functional requirements. Hence, the ontology engineer is expected to know the ontology class and property names when writing the statements. In the context of MODDALS, the application of Themis requires ontology engineers to know how each of the existing ontologies names the concepts and properties expressed in each statement. Then, the statement must be introduced with the vocabularies of each ontology. The engineers who applied Themis were the ones who designed the DABGEO ontology structure in Chapter 6 and they were already familiarized with the existing energy ontologies. However, in a real case the ontology engineers would need to check how they name the concepts and properties of the defined statements. Therefore, the manual ontology analysis effort would not be reduced.

To reduce the manual effort, the Themis tool could be combined with ontology matching algorithms. Ontology engineers could write the statements that answer each CQ using their own concepts. Then, the ontology matching algorithm can identify in existing ontologies equivalent classes/properties to the ones defined in the statements to translate the statement into each ontological representation. Finally, the Themis tool could check whether the statement is represented by each ontology to determine whether the CQ is answered and the corresponding KA is represented.

Considering this, the integration of an ontology verification tool such as Themis with ontology matching algorithms is a future research direction to automate the knowledge classification process in MODDALS. Once this integration is performed, Themis can be combined with user-friendly interfaces and algorithms to automate the whole knowledge classification step (Step 3) and to define the ontology modular structure (Step 4). Ontology engineers can specify in the interface the CQs of each KA in collaboration with domain experts. Themis could check (semi)automatically whether ontologies represent the KAs. Based on these results, an algorithm would automatically (1) apply the application-knowledge matrix technique, (2) classify the KAs into layers and (3) define the ontology modules.

Finally, the guidelines proposed by Themis to define the statements that answer a CQ can be followed to define the ontology statements that answer the CQs of each KA. These statements would lead to a more accurate KA description, avoiding different interpretations of the knowledge encompassed by the KA. Then, the application of the ontology verification tool would enable to classify the domain knowledge in an objective way.

#### 8.4.1.2 Application of MODDALS in other Domains

As stated in Section 8.3.1, MODDALS evaluation was limited to its application to design DABGEO. Hence, MODDALS should be applied in other domains apart from the Energy to demonstrate that it is applicable in other domains. In particular, as stated in Chapter 4, MODDALS is applicable in complex domains where (1) the developed ontology must provide a balance of reusability-usability, since it is developed to be reused by different applications, (2) there are already developed ontologies that support different application types in the domain.

The Industry 4.0 is a domain that meets these conditions and another potential case study where the MODDALS is needed. Industry 4.0 deals with the integration of new Internet technologies (Cyber Physical Systems, Internet of Things, and cognitive computing) in manufacturing processes to enable manufacture data exchange and analysis. The integration of these technologies will enable an efficient and resilient manufacturing process [95]. Semantic interoperability, device integration and data availability are some of the key challenges towards the achievement of the Industry 4.0 vision. Therefore, the Semantic web is a key enabler of this vision [67].

In recent years, there has been a trend towards the development of ontologies based on the standards proposed by Industry 4.0 for the exchange of information between Industry 4.0 systems [67, 66, 93]. To date, only few standards have been semantically represented. Therefore, we can say that the development of a common data representation of manufacturing domains, and by extension the achievement of the Industry 4.0 vision, is still in early stages.

A global ontology that represents semantically Industry 4.0 standards would enable the interoperability between Industry 4.0 systems focused on improving the product manufacturing process [67]. However, there are manufacturing management systems already implemented and under development that rely on already developed manufacturing ontologies. These ontologies are heterogeneous [98] and they represent the manufacturing domains with vocabularies different from those of the Industry 4.0 standards. This heterogeneity hampers the interoperability (1) between manufacturing management systems that use the heterogeneous vocabularies from developed manufacturing ontologies and (2) between the aforementioned systems and new systems that use the vocabularies based on new Industry 4.0 standards.

Therefore, the first step towards a common information model for the Industry 4.0 is a global ontology that provides a common representation of the manufacturing domains represented heterogeneously by well established and new Industry 4.0 standard ontologies [98]. A global Industry 4.0 ontology would represent complex manufacturing domains. In addition, it should provide support to a wide variety of manufacturing management systems and provide that manage different product manufacturing aspects (i.e., energy efficiency, security, and resource optimization). Therefore, the application of current reusable and usable ontology design methodologies (reviewed in Section 3.3.3) would require a high involvement and effort to domain experts and ontology engineers to classify the ontology domain knowledge from scratch.

There are already developed ontologies that represent the knowledge of manufacturing domains to support system that manage the product manufacturing process, i.e., [97, 114, 101]. Thus, the necessary conditions are met to apply MODDALS in the Industry 4.0. Bearing this in mind and the great effort that would be involved in the application of existing methodologies, MODDALS should be applied to design a reusable and usable Industry 4.0 ontology.

#### 8.4.1.3 Definition of Methodological Guidelines for Ontology Maintenance

As stated in Section 8.3.1, MODDALS guidelines are limited to design the layered structure of the first version of the ontology. As more ontologies are developed for new applications, the layered structure of the ontology may change. Some common knowledge may not be reused by new applications, while they may reuse some variant knowledge. Hence, a domain analysis that takes into account new ontologies should be conducted to obtain a more accurate knowledge classification. However, redesigning the layered ontology structure each time a new ontology is developed may lead to an unmanageable ontology maintenance cost and a single ontology may not be sufficiently relevant to reclassify the knowledge of the ontology. In addition, the classification of the application types may change as new and existent applications may be grouped into new application types. Therefore, we consider the definition of methodological guidelines to decide when and how to update the ontology structure as the last research direction to improve MODDALS.

This problem also constitutes a challenge to address in order to improve the maintenance of SPLs. In particular, there is the need of methodological guidelines to update the classification of common and variant domain features when new applications are developed in the domain where the SPL is applied [108].

### 8.4.2 Future Work in DABGEO Ontology

Regarding DABGEO ontology, we would like to present the following lines of research to improve the ontology in future versions.

#### 8.4.2.1 Development of Tools that Facilitate the Ontology Initial Understanding

As stated in Section 7.5.4, DABGEO requires an extra effort to understand its knowledge the first time it is used due to its large number of modules. Considering the features of the Themis tool (introduced in Section 8.4.1), one of its possible uses (apart from ontology validation) is to check whether multiple ontologies meet a set of requirements. Hence, Themis enables ontology developers to select and reuse only the ontologies that meet the requirements of the ontology to be developed [57].

Taking this into account, algorithms that check whether ontologies meet specific requirements can help to reduce the understanding effort of DABGEO. These algorithms could be incorporated to DABGEO ontology publication page through an interface. In this way, ontology developers could check which DABGEO ontology modules meet the requirements of their ontology without analysing other ontology modules.

#### 8.4.2.2 Representation of Ontology Domain Independent Knowledge

As stated in Section 8.3.2, DABGEO includes only the knowledge from the existing energy ontologies reused in specific applications. Hence, it does not include domain independent knowledge that can be reused in different domains. The knowledge it includes is subject to the domain of energy. The top-level layers of DABGEO may be modified when it is extended to support new energy management applications that require the knowledge from energy domains other than the ones represented. Therefore, a top layer that represents domain independent knowledge should be added to DABGEO, so that the ontology structure remains consistent when the knowledge of new domains is added. In this way, the maintainability of DABGEO would be facilitated.

#### 8.4.2.3 Evaluation of DABGEO Interoperability

As stated in Section 8.3.2, DABGEO was evaluated to demonstrate its balance between reusability-usability. The future work should focus on ensuring that DABGEO provides interoperability between new and legacy energy management applications.

To achieve this interoperability, DABGEO links the equivalent knowledge represented heterogeneously by existing energy ontologies. Most of these equivalence relations were identified semi-automatically through the *Agreementmaker* ontology matching tool, as explained in Section 7.4.1.2. However, as stated in Section 2.6.2, the ontology matching tools do not identify all the semantic correspondences between ontologies, and not all the correspondences they identify are correct [104]. To find additional equivalence relations between existing energy ontologies we should take into account new ontology matching tools based on semantic-similarity algorithms.

In addition, DABGEO includes only the equivalence relations between classes and properties of energy ontologies that represent the same knowledge. Hence, interoperability between different applications is only possible when ontologies differ only on the vocabularies used to represent the same properties and classes. If the ontologies represent the same knowledge differing on more complex aspects (i.e., operators to relate an entity in one ontology to a combination of entities in the other [50, 52]), interoperability between different applications is reduced. Therefore, complex correspondences between DABGEO and existing energy ontologies should be defined. Based on these correspondences, a query mediator could convert queries from one ontology representation to another to enable the data exchange between any application supported by heterogeneous ontologies [135].

Once identified the aforementioned equivalence relations, the ontology interoperability should be evaluated to demonstrate that it provides interoperability between new and legacy energy management applications. The evaluation should focus on determining if datasets represented with the vocabularies from existing ontologies can be queried taking as reference the links and knowledge correspondences defined by DABGEO. In that way, we would demonstrate that DABGEO enables the knowledge exchange between energy management applications that rely on heterogeneous ontologies.

#### 8.4.2.4 Alignment of DABGEO with Standard Ontologies

As stated in Section 8.3.2, DABGEO includes the knowledge from the ontologies reused in specific energy management applications. Therefore, we plan to align DABGEO ontology with standard ontologies that represent the energy domains covered by DABGEO. For instance, the standard SOSA/SSN ontology [82] is an standard ontology that represents the knowledge about sensors/actuators and measurements. This knowledge is included by many modules of DABGEO, so the SOSA/SSN ontology should be aligned with DABGEO. The alignment of DABGEO with existing standards would bring it closer to standardization.

# Bibliography

[1] Prototype of the semanco integrated platform. Available: http://www.semanco-project.eu/index_htm_files/SEMANCO_D5.4_20131028.pdf, 2013. Accessed: 2016.

[2] Ejaz Ahmed, Ibrar Yaqoob, Abdullah Gani, Muhammad Imran, and Mohsen Guizani. Internet-of-things-based smart environments: state of the art, taxonomy, and open research challenges. *IEEE Wireless Communications*, 23(5):10–16, 2016.

[3] Ahmed Alasoud, Volker Haarslev, and Nematollaah Shiri. An effective ontology matching technique. In *An A., Matwin S., Raś Z.W., Ślęzak D. (eds) Foundations of Intelligent Systems. ISMIS 2008. Lecture Notes in Computer Science*, volume 4994, pages 585–590. Springer, Berlin, Heidelberg, 2008.

[4] Sven Apel, Don Batory, Christian Kästner, and Gunter Saake. *Feature-oriented software product lines*. Springer-Verlag, 2016.

[5] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. In *Aberer K. et al. (eds) The Semantic Web. ISWC 2007, ASWC 2007. Lecture Notes in Computer Science*, volume 4825, pages 722–735. Springer, Berlin, Heidelberg, 2007.

[6] Payam Barnaghi, Wei Wang, Cory Henson, and Kerry Taylor. Semantics for the internet of things: early progress and back to the future. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 8(1):1–21, 2012.

[7] Tim Berners-Lee, James Hendler, Ora Lassila, et al. The semantic web. *Scientific american*, 284(5):34–43, 2001.

[8] Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data: The story so far. In *Semantic services, interoperability and web applications: emerging concepts*, pages 205–227. IGI Global, 2011.

[9] Eva Blomqvist and Patrik Thollander. An integrated dataset of energy efficiency measures published as linked open data. *Energy Efficiency*, 8(6): 1125–1147, 2015.

[10] Eva Blomqvist, Karl Hammar, and Valentina Presutti. Engineering ontologies with patterns-the extreme design methodology. In *Hitzler, P., Gangemi, A., Janowicz, K., Krisnadhi, A., Presutti, V. (eds.) Ontology Engineering with Ontology Design Patterns – Foundations and Applications, Studies on the Semantic Web*, volume 25, pages 23–50, 2016.

[11] E Paslaru Bontas, Malgorzata Mochol, and Robert Tolksdorf. Case studies on ontology reuse. In *Proceedings of the IKNOW05 International Conference on Knowledge Management*, volume 24, pages 345 –353, 2005.

[12] Elena Paslaru Bontas and Malgorzata Mochol. A cost model for ontology engineering. *Technical Report TR-B-05-03, FU Berlin, ftp://ftp.inf.fu-berlin.de/pub/reports/tr-b-05-03.pdf*, 2005.

[13] H. P. Breivold, S. Larsson, and R. Land. Migrating industrial systems towards software product lines: Experiences and observations through case studies. In *2008 34th Euromicro Conference Software Engineering and Advanced Applications*, pages 232–239, Sep. 2008. doi: 10.1109/SEAA.2008.13.

[14] Dan Brickley and Libby Miller. Foaf vocabulary specification 0.99. http://xmlns.com/foaf/spec/, January 2014.

[15] Agustina Buccella, Alejandra Cechich, Domenico Gendarmi, Filippo Lanubile, Giovanni Semeraro, and Attilio Colagrossi. Building a global normalized ontology for integrating geographic data sources. *Computers & Geosciences*, 37 (7):893–916, 2011.

[16] Alexander Budanitsky and Graeme Hirst. Evaluating wordnet-based measures of lexical semantic relatedness. *Computational Linguistics*, 32(1):13–47, 2006.

[17] Grégoire Burel, Lara SG Piccolo, and Harith Alani. Energyuse-a collective semantic platform for monitoring and discussing energy consumption. In

*Groth P. et al. (eds) The Semantic Web – ISWC 2016. ISWC 2016. Lecture Notes in Computer Science*, volume 9982, pages 257–272. Springer, Cham, 2016.

[18] Tony Buzan. *Mind mapping*. Pearson Education, 2006.

[19] Harry Chen, Filip Perich, Tim Finin, and Anupam Joshi. Soupa: Standard ontology for ubiquitous and pervasive applications. In *Mobile and Ubiquitous Systems: Networking and Services, 2004. MOBIQUITOUS 2004. The First Annual International Conference on*, pages 258–267. IEEE, 2004.

[20] Lianping Chen, Muhammad Ali Babar, and Nour Ali. Variability management in software product lines: a systematic review. In *Proceedings of the 13th International Software Product Line Conference*, pages 81–90. Carnegie Mellon University, 2009.

[21] Namyoun Choi, Il-Yeol Song, and Hyoil Han. A survey on ontology mapping. *ACM Sigmod Record*, 35(3):34–41, 2006.

[22] Rudi L Cilibrasi and Paul MB Vitanyi. The google similarity distance. *IEEE Transactions on knowledge and data engineering*, 19(3):370–383, 2007.

[23] Philipp Cimiano, Siegfried Handschuh, and Steffen Staab. Towards the self-annotating web. In *Proceedings of the 13th International Conference on World Wide Web*, WWW '04, page 462–471, New York, NY, USA, 2004. Association for Computing Machinery. ISBN 158113844X.

[24] Paul Clements and Linda Northrop. *Software product lines: practices and patterns*, volume 3. Addison-Wesley Reading, 2002.

[25] IEEE Computer Society Software Engineering Standards Committee. Ieee standard glossary of software engineering terminology. 1990.

[26] Vincenzo Corrado, Ilaria Ballarini, Leandro Madrazo, and German Nemirovskij. Data structuring for the ontological modelling of urban energy systems: The experience of the semanco project. *Sustainable Cities and Society*, 14(1):223–235, 2015.

[27] Edward Corry, Pieter Pauwels, Shushan Hu, Marcus Keane, and James O'Donnell. A performance assessment ontology for the environmental and energy management of buildings. *Automation in Construction*, 57:249–259, 2015.

[28] Isabel F Cruz, William Sunna, Nalin Makar, and Sujan Bathala. A visual tool for ontology alignment to enable geospatial interoperability. *Journal of Visual Languages & Computing*, 18(3):230–254, 2007.

[29] Isabel F Cruz, Flavio Palandri Antonelli, and Cosmin Stroe. Agreementmaker: efficient matching for large real-world schemas and ontologies. *Proceedings of the VLDB Endowment*, 2(2):1586–1589, 2009.

[30] Javier Cuenca, Felix Larrinaga, and Edward Curry. A unified semantic ontology for energy management applications. In *Bozzato L. et al. (eds) Joint Proceedings of the Web Stream Processing workshop (WSP 2017) and the 2nd International Workshop on Ontology Modularity, Contextuality, and Evolution (WOMoCoE 2017) co-located with 16th International Semantic Web Conference*, volume 1936, pages 86–97. CEUR-WS, 2017.

[31] Javier Cuenca, Felix Larrinaga, and Edward Curry. Experiences on applying SPL engineering techniques to design a (re) usable ontology in the energy domain. In *Proceedings of the International Conference on Software Engineering and Knowledge Engineering, SEKE, 2019*, volume 2019-July, pages 606–611. KSI Research Inc. and Knowledge Systems I., 2019. doi: 10.18293/SEKE2019-111.

[32] Javier Cuenca, Felix Larrinaga, Luka Eciolaza, and Edward Curry. Towards cognitive cities in the energy domain. In *Portmann E., Tabacchi M., Seising R., Habenstein A. (eds) Designing Cognitive Cities. Studies in Systems, Decision and Control*, volume 176, pages 155–183. Springer, Cham, 2019.

[33] Javier Cuenca, Felix Larrinaga, and Edward Curry. Dabgeo: A reusable and usable global energy ontology for the energy domain. *Journal of Web Semantics*, In Press, February 2020. doi: https://doi.org/10.1016/j.websem.2020.100550.

[34] Javier Cuenca, Félix Larrinaga, and Edward Curry. Moddals methodology for designing layered ontology structures. *Applied Ontology*, (Preprint):1–33, January 2020.

[35] C. Curino, G. Orsi, and L. Tanca. X-som: A flexible ontology mapper. In *18th International Workshop on Database and Expert Systems Applications (DEXA 2007)*, pages 424–428, Sep. 2007.

[36] E. Curry. System of systems information interoperability using a linked dataspace. In *2012 7th International Conference on System of Systems Engineering (SoSE)*, pages 101–106, July 2012.

[37] E. Curry, S. Hasan, and S. O'Riain. Enterprise energy management using a linked dataspace for energy intelligence. In *2012 Sustainable Internet and ICT for Sustainability (SustainIT)*, pages 1–6, Oct 2012.

[38] Edward Curry. The big data value chain: definitions, concepts, and theoretical approaches. In *Cavanillas J., Curry E., Wahlster W. (eds) New Horizons for a Data-Driven Economy*, pages 29–37. Springer, Cham, 2016.

[39] Edward Curry, James O'Donnell, Edward Corry, Souleiman Hasan, Marcus Keane, and Seán O'Riain. Linking building data in the cloud: Integrating cross-domain building data using linked data. *Advanced Engineering Informatics*, 27 (2):206–219, 2013.

[40] Edward Curry, Schahram Dustdar, Quan Z Sheng, and Amit Sheth. Smart cities–enabling services and applications. *Journal of Internet Services and Applications*, 7(1):6, 2016.

[41] Edward Curry, Wassim Derguech, Souleiman Hasan, Christos Kouroupetroglou, and Umair ul Hassan. A real-time linked dataspace for the internet of things: Enabling "pay-as-you-go" data management in smart environments. *Future Generation Computer Systems*, 90:405–422, 2019.

[42] Wasila M Dahdul, James P Balhoff, David C Blackburn, Alexander D Diehl, Melissa A Haendel, Brian K Hall, Hilmar Lapp, John G Lundberg, Christopher J Mungall, Martin Ringwald, et al. A unified anatomy ontology of the vertebrate skeletal system. *PloS one*, 7(12), 2012.

[43] Laura Daniele, Frank den Hartog, and Jasper Roes. Created in close interaction with the industry: the smart appliances reference (saref) ontology. In *Cuel R., Young R. (eds) Formal Ontologies Meet Industry. FOMI 2015, Lecture Notes in Business Information Processing*, number 225, pages 100–112. Springer, Cham, 2015.

[44] Laura Daniele, Monika Solanki, Frank den Hartog, and Jasper Roes. Interoperability for smart appliances in the iot world. In *Groth P. et al. (eds) The Semantic Web – ISWC 2016. ISWC 2016. Lecture Notes in Computer Science*, number 9982, pages 21–29. Springer, Cham, 2016.

[45] Mathieu d'Aquin. Modularizing ontologies. In *Suárez-Figueroa M., Gómez-Pérez A., Motta E., Gangemi A. (eds) Ontology Engineering in a Networked World.*, pages 213–233. Springer, Berlin, Heidelberg, 2012.

[46] Ricardo de Almeida Falbo. Sabio: Systematic approach for building ontologies. In *Guizzardi et al. (eds) Proceedings of the 1st Joint Workshop ONTO.COM / ODISE on Ontologies in Conceptual Modeling and Information Systems Engineering co-located with 8th International Conference on Formal Ontology in Information Systems (FOIS 2014), CEUR Workshop Proceedings*, volume 1301. CEUR-WS, 2014.

[47] Jos De Bruijn, Marc Ehrig, Cristina Feier, Francisco Martín-Recuerda, François Scharffe, and Moritz Weiten. Ontology mediation, merging and aligning. *Semantic Web Technologies: Trends and Research in Ontology-based Systems*, pages 95–113, 2006.

[48] R De Hoog. Methodologies for building knowledge based systems: Achievements and prospects. *The Handbook of Applied Expert Systems*, 1(11), 1998.

[49] Lisa Ehrlinger and Wolfram Wöß. Towards a definition of knowledge graphs. *SEMANTiCS (Posters, Demos, SuCCESS)*, 48, 2016.

[50] J. Euzenat, A. Polleres, and F. Scharffe. Processing ontology alignments with sparql. In *2008 International Conference on Complex, Intelligent and Software Intensive Systems*, pages 913–917, March 2008.

[51] Jérôme Euzenat and Chan Le Duc. Methodological guidelines for matching ontologies. In *Suárez-Figueroa M., Gómez-Pérez A., Motta E., Gangemi A. (eds) Ontology Engineering in a Networked World*, pages 257–278. Springer, Berlin, Heidelberg, 2012.

[52] Jérôme Euzenat, Pavel Shvaiko, et al. *Ontology matching*, volume 18. Springer, 2007.

[53] N F. Noy and Deborah Mcguinness. Ontology development 101: A guide to creating your first ontology. *Technical report, Stanford Knowledge Systems Laboratory*, 2001.

[54] Xi Fang, Satyajayant Misra, Guoliang Xue, and Dejun Yang. Smart grid: The new and improved power grid: A survey. *IEEE communications surveys & tutorials*, 14(4):944–980, 2012.

[55] Alessandro Fantechi, Stefania Gnesi, Isabel John, Giuseppe Lami, and Jörg Dörr. Elicitation of use cases for product lines. In *van der Linden F.J. (eds) Software Product-Family Engineering. PFE 2003. Lecture Notes in Computer Science*, volume 3014, pages 152–167. Springer, Berlin, Heidelberg, 2003.

[56] Anna Fensel, Vikash Kumar, and Slobodanka Dana Kathrin Tomic. End-user interfaces for energy-efficient semantically enabled smart homes. *Energy Efficiency*, 7(4):655–675, 2014.

[57] Alba Fernández-Izquierdo and Raúl García-Castro. Themis: a tool for validating ontologies through requirements. In *Proceedings of the International Conference on Software Engineering and Knowledge Engineering, SEKE, 2019*, volume 2019-July, pages 573–753. KSI Research Inc. and Knowledge Systems I., 2019.

[58] Mariano Fernández-López, Asunción Gómez-Pérez, and Natalia Juristo. Methontology: from ontological art towards ontological engineering. In *Proceedings of the AAAI97 Spring Symposium Series on Ontological Engineering*. Stanford, 1997.

[59] A. Fernbach, I. Pelesic, and W. Kastner. Linked data for building management. In *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*, pages 6943–6945, Oct 2016.

[60] Matthias Finger and Edy Portmann. What are cognitive cities? In *Portmann E., Finger M. (eds) Towards Cognitive Cities. Studies in Systems, Decision and Control*, volume 63, pages 1–11. Springer, Cham, 2016.

[61] Aldo Gangemi and Valentina Presutti. Ontology design patterns. In *Staab S., Studer R. (eds) Handbook on Ontologies. International Handbooks on Information Systems*, pages 221–243. Springer, Berlin, Heidelberg, 2009.

[62] Syed Gillani, Frederique Laforest, and Gauthier Picard. A generic ontology for prosumer-oriented smart grid. In *Selçuk Candan K. et al. (eds) Proceedings of the Workshops of the EDBT/ICDT 2014 Joint Conference (EDBT/ICDT 2014). CEUR Workshop Proceedings*, volume 1133, pages 134–139. CEUR-WS, 2014.

[63] Syed Gillani, Gauthier Picard, and Frédérique Laforest. Intelscep: Towards an intelligent semantic complex event processing framework for prosumer-oriented smartgrid. In *Proceedings of the 2014 International Workshop on Web Intelligence and Smart Sensing, ACM International Conference Proceeding Series*, pages 1–2. ACM, 2014.

[64] Fausto Giunchiglia, Pavel Shvaiko, and Mikalai Yatskevich. S-match: an algorithm and an implementation of semantic matching. In *Bussler C.J., Davies J., Fensel D., Studer R. (eds) The Semantic Web: Research and Applications. ESWS*

*2004. Lecture Notes in Computer Science*, volume 3053, pages 61–75. Springer, Berlin, Heidelberg, 2004.

[65] Asunción Gómez-Pérez, Mariano Fernández-López, and Óscar Corcho. *Ontological Engineering: with examples from the areas of Knowledge Management, e-Commerce and the Semantic Web*. Springer Science & Business Media, 2006.

[66] I. Grangel-González, L. Halilaj, S. Auer, S. Lohmann, C. Lange, and D. Collarana. An rdf-based approach for implementing industry 4.0 components with administration shells. In *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–8, Sep. 2016.

[67] I. Grangel-González, L. Halilaj, G. Coskun, S. Auer, D. Collarana, and M. Hoffmeister. Towards a semantic administrative shell for industry 4.0 components. In *2016 IEEE Tenth International Conference on Semantic Computing (ICSC)*, pages 230–237, Feb 2016.

[68] B Cuenca Grau, Ian Horrocks, Yevgeny Kazakov, and Ulrike Sattler. Modular reuse of ontologies: Theory and practice. *Journal of Artificial Intelligence Research*, 31:273–318, 2008.

[69] Tom Gruber. *Ontology*. LIU L., ÖZSU M.T. (eds) Encyclopedia of Database Systems, Springer, Boston, MA, 2009.

[70] Michael Grüninger and Mark S Fox. Methodology for the design and evaluation of ontologies. In *International Joint Conference on Artificial Inteligence (IJCAI95), Workshop on Basic Ontological Issues in Knowledge Sharing*, pages 1–10, 1995.

[71] Nicola Guarino. Semantic matching: Formal ontological distinctions for information organization, extraction, and integration. In *Pazienza M.T. (eds) Information Extraction A Multidisciplinary Approach to an Emerging Information Technology. SCIE 1997. Lecture Notes in Computer Science (Lecture Notes in Artificial Intelligence)*, volume 1299, pages 139–170. Springer, Berlin, Heidelberg, 1997.

[72] V Cagri Gungor, Dilan Sahin, Taskin Kocak, Salih Ergut, Concettina Buccella, Carlo Cecati, and Gerhard P Hancke. A survey on smart grid potential applications and communication requirements. *IEEE Transactions on industrial informatics*, 9(1):28–42, 2013.

[73] A. Harhurin and J. Hartmann. Service-oriented commonality analysis across existing systems. In *2008 12th International Software Product Line Conference*, pages 255–264, Sep. 2008. doi: 10.1109/SPLC.2008.19.

[74] Tom Heath and Christian Bizer. Linked data: Evolving the web into a global data space. *Synthesis lectures on the semantic web: theory and technology*, 1(1): 1–136, 2011.

[75] John Hebeler, Matthew Fisher, Ryan Blace, and Andrew Perez-Lopez. *Semantic web programming*. John Wiley & Sons, 2011.

[76] J. L. Hippolyte, S. Howell, B. Yuce, M. Mourshed, H. A. Sleiman, M. Vinyals, and L. Vanhee. Ontology-based demand-side flexibility management in smart grids using a multi-agent system. In *2016 IEEE International Smart Cities Conference (ISC2)*, pages 1–7, Sep. 2016.

[77] J-L Hippolyte, Yacine Rezgui, Haijiang Li, Bejay Jayan, and Shaun Howell. Ontology-driven development of web services to support district energy applications. *Automation in Construction*, 86:210–225, 2018.

[78] Pascal Hitzler, Aldo Gangemi, and Krzysztof Janowicz. *Ontology Engineering with Ontology Design Patterns: Foundations and Applications*, volume 25 of *Studies on the Semantic Web*. IOS Press, 2016.

[79] Shushan Hu, Edward Corry, Edward Curry, William JN Turner, and James O'Donnell. Building performance optimisation: A hybrid architecture for the integration of contextual information and time-series data. *Automation in Construction*, 70:51–61, 2016.

[80] Stuart Hutchesson and John McDermid. Trusted product lines. *Information and Software Technology*, 55(3):525–540, 2013.

[81] M Iqbal, M Azam, M Naeem, AS Khwaja, and A Anpalagan. Optimization classification, algorithms and tools for renewable energy: A review. *Renewable and Sustainable Energy Reviews*, 39:640–654, 2014.

[82] Krzysztof Janowicz, Armin Haller, Simon JD Cox, Danh Le Phuoc, and Maxime Lefrançois. Sosa: A lightweight ontology for sensors, observations, samples, and actuators. *Journal of Web Semantics*, 56:1–10, 2019.

[83] Mustafa Jarrar and Robert Meersman. Ontology engineering–the dogma approach. In *Dillon T.S., Chang E., Meersman R., Sycara K. (eds) Advances in*

*Web Semantics I. Lecture Notes in Computer Science*, volume 4891, pages 7–34. Springer, Berlin, Heidelberg, 2008.

[84] B. Johnson and B. Shneiderman. Tree-maps: a space-filling approach to the visualization of hierarchical information structures. In *Proceeding Visualization '91*, pages 284–291, Oct 1991.

[85] Yannis Kalfoglou and Marco Schorlemmer. If-map: An ontology-mapping method based on information-flow theory. In *Spaccapietra S., March S., Aberer K. (eds) Journal on Data Semantics I. Lecture Notes in Computer Science*, volume 2800, pages 98–127. Springer, Berlin, Heidelberg, 2003.

[86] Kyo C Kang, Sholom G Cohen, James A Hess, William E Novak, and A Spencer Peterson. Feature-oriented domain analysis (foda) feasibility study. Technical report, Technical Report CMU/SEI-90-TR-21, Software Engineering Institute, Carnegie Mellon University, 1990.

[87] Wolfgang Kastner, Georg Neugschwandtner, Stefan Soucek, and H Michael Newman. Communication systems for building automation and control. *Proceedings of the IEEE*, 93(6):1178–1203, 2005.

[88] Zubeida Casmod Khan and C Maria Keet. An empirically-based framework for ontology modularisation. *Applied Ontology*, 10(3-4):171–195, 2015.

[89] Haklae Kim. Towards a sales assistant using a product knowledge graph. *Journal of Web Semantics*, 46:14–19, 2017.

[90] Georg Klinker, Carlos Bhola, Geoffroy Dallemagne, David Marques, and John McDermott. Usable and reusable programming constructs. *Knowledge Acquisition*, 3(2):117–135, 1991.

[91] Graham Klyne and Jeremy J Carroll. Resource description framework (rdf): Concepts and abstract syntax. 2006.

[92] Mario J Kofler, Christian Reinisch, and Wolfgang Kastner. A semantic representation of energy-related information in future smart homes. *Energy and Buildings*, 47:169–179, 2012.

[93] O. Kovalenko, M. Wimmer, M. Sabou, A. Lüder, F. J. Ekaputra, and S. Biffl. Modeling automationml: Semantic web technologies vs. model-driven engineering. In *2015 IEEE 20th Conference on Emerging Technologies Factory Automation (ETFA)*, pages 1–4, Sep. 2015. doi: 10.1109/ETFA.2015.7301643.

[94] Markus Krötzsch and Veronika Thost. Ontologies for knowledge graphs: Breaking the rules. In *Groth P. et al. (eds) The Semantic Web – ISWC 2016. ISWC 2016. Lecture Notes in Computer Science*, volume 9981, pages 376–392. Springer, Cham, 2016.

[95] Heiner Lasi, Peter Fettke, Hans-Georg Kemper, Thomas Feld, and Michael Hoffmann. Industry 4.0. *Business & Information Systems Engineering*, 6(4): 239–242, 2014.

[96] Maxime Lefrançois. Planned etsi saref extensions based on the w3c&ogc sosa/ssn-compatible seas ontology paaerns. In *Fensel A. and Daniele L. (eds) Joint Proceedings of SEMANTiCS 2017 Workshops co-located with the 13th International Conference on Semantic Systems (SEMANTiCS 2017). CEUR Workshop Proceedings*, volume 2063, page 11p. CEUR-WS, 2017.

[97] S. Lemaignan, A. Siadat, J. . Dantan, and A. Semenenko. Mason: A proposal for an ontology of manufacturing domain. In *IEEE Workshop on Distributed Intelligent Systems: Collective Intelligence and Its Applications (DIS'06)*, pages 195–200, June 2006. doi: 10.1109/DIS.2006.48.

[98] Wilfried Lepuschitz, Alvaro Lobato-Jimenez, Emilian Axinia, and Munir Merdan. A survey on standards and ontologies for process automation. In *Mařík V., Schirrmann A., Trentesaux D., Vrba P. (eds) Industrial Applications of Holonic and Multi-Agent Systems. HoloMAS 2015. Lecture Notes in Computer Science*, volume 9266, pages 22–32. Springer, Cham, 2015.

[99] Juanzi Li, Jie Tang, Yi Li, and Qiong Luo. Rimom: A dynamic multistrategy ontology alignment framework. *IEEE Transactions on Knowledge and data Engineering*, 21(8):1218–1232, 2009.

[100] Steffen Lohmann, Paloma Díaz, and Ignacio Aedo. Muto: The modular unified tagging ontology. In *Proceedings of the 7th International Conference on Semantic Systems*, I-Semantics '11, page 95–104, New York, NY, USA, 2011. Association for Computing Machinery. ISBN 9781450306218.

[101] Yuqian Lu and Xun Xu. A semantic web-based framework for service composition in a cloud manufacturing environment. *Journal of manufacturing systems*, 42:69–81, 2017.

[102] Alessio Maffei, Seshadhri Srinivasan, Pedro Castillejo, José Fernán Martínez, Luigi Iannelli, Eilert Bjerkan, and Luigi Glielmo. A semantic-middleware-

supported receding horizon optimal power flow in energy grids. *IEEE Transactions on Industrial Informatics*, 14(1):35–46, 2018.

[103] Anzar Mahmood, Nadeem Javaid, and Sohail Razzaq. A review of wireless communications for smart grid. *Renewable and Sustainable Energy Reviews*, 41: 248–260, 2015.

[104] Mohammed Maree and Mohammed Belkhatir. Addressing semantic heterogeneity through multiple knowledge base assisted merging of domain-specific ontologies. *Knowledge-Based Systems*, 73:199–211, 2015.

[105] Wolfgang Marquardt, Jan Morbach, Andreas Wiesner, and Aidong Yang. *OntoCAPE: A re-usable ontology for chemical process engineering*. Springer Science & Business Media, 2009.

[106] Brian McBride. Jena: Implementing the rdf model and syntax specification. In *Proceedings of the Second International Conference on Semantic Web-Volume 40*, pages 23–28. CEUR-WS. org, 2001.

[107] Deborah L McGuinness, Frank Van Harmelen, et al. Owl web ontology language overview. *W3C recommendation*, 10(10):2004, 2004.

[108] Andreas Metzger and Klaus Pohl. Software product line engineering and variability management: Achievements and challenges. In *Proceedings of the on Future of Software Engineering*, FOSE 2014, page 70–84, New York, NY, USA, 2014. Association for Computing Machinery. ISBN 9781450328654.

[109] Elena Montiel-Ponsoda, Daniel Vila Suero, Boris Villazón-Terrazas, Gordon Dunsire, Elena Escolano Rodríguez, and Asunción Gómez-Pérez. Style guidelines for naming and labeling ontologies in the multilingual web. In *Proceedings of the 2011 International Conference on Dublin Core and Metadata Applications, DC 2011, The Hague, The Netherlands, September 21-23*, pages 105–115, 2011.

[110] Mikyeong Moon, Keunhyuk Yeom, and Heung Seok Chae. An approach to developing domain requirements as a core asset based on commonality and variability analysis in a product line. *IEEE transactions on software engineering*, 31(7):551–569, 2005.

[111] Jan Morbach, Aidong Yang, and Wolfgang Marquardt. Ontocape:a large-scale ontology for chemical process engineering. *Engineering applications of artificial intelligence*, 20(2):147–161, 2007.

[112] Jan Morbach, Andreas Wiesner, and Wolfgang Marquardt. Ontocape—a (re) usable ontology for computer-aided process engineering. *Computers & Chemical Engineering*, 33(10):1546–1556, 2009.

[113] Robert Moyser and Sabina Uffer. From smart to cognitive: A roadmap for the adoption of technology in cities. In *Portmann E., Finger M. (eds) Towards Cognitive Cities. Studies in Systems, Decision and Control*, volume 63, pages 13–35. Springer, Cham, 2016.

[114] Edrisi Muñoz, A Espuña, and Luis Puigjaner. Towards an ontological infrastructure for chemical batch process management. *Computers & chemical engineering*, 34(5):668–682, 2010.

[115] Daren Nestor, Luke O'Malley, Aaron Quigley, Ernst Sikora, and Steffen Thiel. Visualisation of variability in software product line engineering. In *K. Pohl, P. Heymans, K.-C. Kang, and A. Metzger (eds), First International Workshop on Variability Modelling of Software-intensive Systems Proceedings. Lero - Technical Report 2007-01, Limerick, Ireland*, pages 71–78, 2007.

[116] Mehrdad Niknam and Saeed Karshenas. Sustainable design of buildings using semantic bim and semantic web services. *Procedia Engineering*, 118:909–917, 2015.

[117] Mehrdad Niknam and Saeed Karshenas. A shared ontology approach to semantic representation of bim data. *Automation in Construction*, 80:22–36, 2017.

[118] Natalya Fridman Noy and Mark A Musen. Smart: Automated support for ontology merging and alignment. In *Proc. of the 12th Workshop on Knowledge Acquisition, Modelling, and Management (KAW'99), Banf, Canada*, 1999.

[119] Natalya Fridman Noy, Mark A Musen, et al. Algorithm and tool for automated ontology merging and alignment. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on on Innovative Applications of Artificial Intelligence, July 30 - August 3, 2000, Austin, Texas, USA. Available as SMI technical report SMI-2000-0831*, volume 115. AAAI Press / The MIT Press, 2000.

[120] Office of the National Coordinator for Smart Grid Interoperability and Office of the National Coordinator for Smart Grid Interoperability. *NIST framework and roadmap for smart grid interoperability standards, release 1.0.* US Department of Commerce, National Institute of Standards and Technology, 2018.

[121] Lorena Otero-Cerdeira, Francisco J Rodríguez-Martínez, and Alma Gómez-Rodríguez. Ontology matching: A literature review. *Expert Systems with Applications*, 42(2):949–971, 2015.

[122] Aris M. Ouksel and Amit Sheth. Semantic interoperability in global information systems. *ACM Sigmod Record*, 28(1):5–12, 1999.

[123] Prashant Palvia and John T Nosek. An empirical evaluation of system development methodologies. *Information Resources Management Journal (IRMJ)*, 3(3): 23–33, 1990.

[124] Elena Pâslaru-Bontaş. *A contextual approach to ontology reuse*. PhD thesis, Freie Universität Berlin, Germany, 2007.

[125] Heiko Paulheim. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic web*, 8(3):489–508, 2017.

[126] H Sofia Pinto, Steffen Staab, and Christoph Tempich. Diligent: Towards a fine-grained methodology for distributed, loosely-controlled and evolving. In *Proceedings of the 16th European Conference on Artificial Intelligence (ECAI 2004)*, volume 110, page 393, 2004.

[127] Klaus Pohl, Günter Böckle, and Frank J van Der Linden. *Software product line engineering: foundations, principles and techniques*. Springer Science & Business Media, 2005.

[128] Ulrich J Pont, Neda Ghiassi, Stefan Fenz, Johannes Heurix, and Ardeshir Mahdavi. Semergy: Application of semantic web technologies in performance-guided building design optimization. *Journal of Information Technology in Construction*, 20:107–120, 2015.

[129] María Poveda-Villalón, Asunción Gómez-Pérez, and Mari Carmen Suárez-Figueroa. Oops!(ontology pitfall scanner!): An on-line tool for ontology evaluation. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 10(2):7–34, 2014.

[130] Valentina Presutti and Aldo Gangemi. Content ontology design patterns as practical building blocks for web ontologies. In *Li Q., Spaccapietra S., Yu E., Olivé A. (eds) Conceptual Modeling - ER 2008. ER 2008. Lecture Notes in Computer Science*, volume 5231, pages 128–141. Springer, Berlin, Heidelberg, 2008.

[131] Eric Prud, Andy Seaborne, et al. Sparql query language for rdf. *W3C Recommendation*, http://www.w3.org/TR/rdf-sparql-query/, 2008.

[132] Filip Radulovic, María Poveda-Villalón, Daniel Vila-Suero, Víctor Rodríguez-Doncel, Raúl García-Castro, and Asunción Gómez-Pérez. Guidelines for linked data generation and publication: An example in building energy consumption. *Automation in Construction*, 57:178–187, 2015.

[133] Salvatore Raunich and Erhard Rahm. Towards a benchmark for ontology merging. In *Herrero P., Panetto H., Meersman R., Dillon T. (eds) On the Move to Meaningful Internet Systems: OTM 2012 Workshops. OTM 2012. Lecture Notes in Computer Science*, volume 7567, pages 124–133. Springer, Berlin, Heidelberg, 2012.

[134] Ana Armas Romero, Mark Kaminski, Bernardo Cuenca Grau, and Ian Horrocks. Ontology module extraction via datalog reasoning. In *Proceedings of the National Conference on Artificial Intelligence*, volume 2, pages 1410–1416. AI Access Foundation, 2015.

[135] Catherine Roussey, François Pinet, Myoung-Ah Kang, Oscar Corcho, Gilles Falquet, Claudine Métral, Jacques Teller, and Christopher Tweed. Ontologies for interoperability. In *Ontologies in Urban Development Projects. Advanced Information and Knowledge Processing*, volume 1, pages 39–53. Springer, London, 2011. doi: 10.1007/978-0-85729-724-2_3.

[136] Fabiano B Ruy, Giancarlo Guizzardi, Ricardo A Falbo, Cássio C Reginato, and Victor A Santos. From reference ontologies to ontology patterns and back. *Data & Knowledge Engineering*, 109:41–69, 2017.

[137] Khouloud Salameh, Richard Chbeir, Haritza Camblong, Gilbert Tekli, and Ionel Vechiu. A generic ontology-based information model for better management of microgrids. In *Chbeir R., Manolopoulos Y., Maglogiannis I., Alhajj R. (eds) Artificial Intelligence Applications and Innovations. AIAI 2015. IFIP Advances in Information and Communication Technology*, volume 458, pages 451–466. Springer, Cham, 2015.

[138] Ansgar Scherp, Carsten Saathoff, Thomas Franz, and Steffen Staab. Designing core ontologies. *Applied Ontology*, 6(3):177–221, 2011.

[139] C. Schlenoff, E. Prestes, R. Madhavan, P. Goncalves, H. Li, S. Balakirsky, T. Kramer, and E. Migueláñez. An ieee standard ontology for robotics and automation. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1337–1342, Oct 2012.

[140] Nigel Shadbolt, Tim Berners-Lee, and Wendy Hall. The semantic web revisited. *IEEE intelligent systems*, 21(3):96–101, 2006.

[141] W. Shi, E. Lee, D. Yao, R. Huang, C. Chu, and R. Gadh. Evaluating microgrid management and control with an implementable energy management system. In *2014 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, pages 272–277, Nov 2014.

[142] Pavel Shvaiko and Jérôme Euzenat. Ontology matching: state of the art and future challenges. *IEEE Transactions on knowledge and data engineering*, 25(1): 158–176, 2013.

[143] Elena Simperl, Tobias Bürger, Simon Hangl, Stephan Wörgl, and Igor Popov. Ontocom: A reliable cost estimation method for ontology development projects. *Web Semantics: Science, Services and Agents on the World Wide Web*, 16: 1–16, 2012.

[144] Elena Paslaru Bontas Simperl, Christoph Tempich, and York Sure. Ontocom: A cost estimation model for ontology engineering. In *Cruz I. et al. (eds) The Semantic Web - ISWC 2006. ISWC 2006. Lecture Notes in Computer Science*, volume 4273, pages 625–639. Springer, Berlin, Heidelberg, 2006.

[145] Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur, and Yarden Katz. Pellet: A practical owl-dl reasoner. *Web Semantics: science, services and agents on the World Wide Web*, 5(2):51–53, 2007.

[146] Peter Spyns, Yan Tang, and Robert Meersman. An ontology engineering methodology for dogma. *Applied Ontology*, 3(1-2):13–39, 2008.

[147] Thanos G. Stavropoulos, Dimitris Vrakas, Danai Vlachava, and Nick Bassiliades. Bonsai: A smart building ontology for ambient intelligence. In *Proceedings of the 2nd International Conference on Web Intelligence, Mining and Semantics*, WIMS '12, New York, NY, USA, 2012. Association for Computing Machinery. ISBN 9781450309158.

[148] Thanos G Stavropoulos, George Koutitas, Dimitris Vrakas, Efstratios Kontopoulos, and Ioannis Vlahavas. A smart university platform for building energy monitoring and savings. *Journal of Ambient Intelligence and Smart Environments*, 8(3):301–323, 2016.

[149] Heiner Stuckenschmidt and Michel Klein. Integrity and change in modular ontologies. In *IJCAI*, pages 900–908, 2003.

[150] Mari Carmen Suárez-Figueroa. *NeOn Methodology for building ontology networks: specification, scheduling and reuse*. PhD thesis, Universidad Politécnica de Madrid, Spain, 2010.

[151] Mari Carmen Suárez-Figueroa, Asunción Gómez-Pérez, and Mariano Fernández-López. The neon methodology for ontology engineering. In *Suárez-Figueroa M., Gómez-Pérez A., Motta E., Gangemi A. (eds) Ontology Engineering in a Networked World*, pages 9–34. Springer, Berlin, Heidelberg, 2012.

[152] Mari Carmen Suárez-Figueroa, Asunción Gómez-Pérez, and Mariano Fernandez-Lopez. The neon methodology framework: A scenario-based methodology for ontology development. *Applied ontology*, 10(2):107–145, 2015.

[153] York Sure, Steffen Staab, and Rudi Studer. On-to-knowledge methodology (otkm). In *Staab S., Studer R. (eds) Handbook on Ontologies. International Handbooks on Information Systems*, pages 117–132. Springer, Berlin, Heidelberg, 2004.

[154] Dhavalkumar Thakker, Vania Dimitrova, Lydia Lau, Ronald Denaux, Stan Karanasios, and Fan Yang-Turner. A priori ontology modularisation in ill-defined domains. In *Proceedings of the 7th International Conference on Semantic Systems*, I-Semantics '11, page 167–170, New York, NY, USA, 2011. Association for Computing Machinery. ISBN 9781450306218.

[155] Nikola M Tomašević, Marko Č Batić, Luis M Blanes, Marcus M Keane, and Sanja Vraneš. Ontology-based facility data model for energy management. *Advanced Engineering Informatics*, 29(4):971–984, 2015.

[156] Cássia Trojahn, Márcia Moraes, Paulo Quaresma, and Renata Vieira. A co-operative approach for composite ontology mapping. In *Spaccapietra S. (eds) Journal on Data Semantics X. Lecture Notes in Computer Science*, volume 4900, pages 237–263. Springer, Berlin, Heidelberg, 2008.

[157] Eneko Unamuno and Jon Andoni Barrena. Hybrid ac/dc microgrids—part i: Review and classification of topologies. *Renewable and Sustainable Energy Reviews*, 52:1251–1259, 2015.

[158] Hans van der Veer and Anthony Wiles. Achieving technical interoperability. *European Telecommunications Standards Institute*, (3):p.29, 2008.

[159] Willem Robert Van Hage, Sophia Katrenko, and Guus Schreiber. A method to combine linguistic ontology-mapping techniques. In *Gil Y., Motta E., Benjamins V.R., Musen M.A. (eds) The Semantic Web – ISWC 2005. ISWC 2005. Lecture Notes*

*in Computer Science*, volume 3729, pages 732–744. Springer, Berlin, Heidelberg, 2005.

[160] Pierre-Yves Vandenbussche, Ghislain A Atemezing, María Poveda-Villalón, and Bernard Vatant. Linked open vocabularies (lov): a gateway to reusable semantic vocabularies on the web. *Semantic Web*, 8(3):437–452, 2017.

[161] JPC Verhoosel, DCF Rothengatter, FJ Rumph, and MJ Konsman. An ontology for modeling flexibility in smart grid energy management. In *eWork and eBusiness in Architecture, Engineering and Construction - Proceedings of the European Conference on Product and Process Modelling 2012, ECPPM 2012*, pages 931–938. CRC Press, 2012.

[162] Claudia Villalonga, Martin Strohbach, Niels Snoeck, Michael Sutterer, Mariano Belaunde, Ernö Kovacs, Anna V Zhdanova, Laurent Walter Goix, and Olaf Droegehorn. Mobile ontology: Towards a standardized semantic model for the mobile domain. In *Di Nitto E., Ripeanu M. (eds) Service-Oriented Computing - ICSOC 2007 Workshops. ICSOC 2007. Lecture Notes in Computer Science*, volume 4907, pages 248–257. Springer, Berlin, Heidelberg, 2007.

[163] Denny Vrandečić. Ontology evaluation. In *Staab S., Studer R. (eds) Handbook on Ontologies. International Handbooks on Information Systems*, pages 293–313. Springer, Berlin, Heidelberg, 2009.

[164] Holger Wache, Thomas Voegele, Ubbo Visser, Heiner Stuckenschmidt, Gerhard Schuster, Holger Neumann, and Sebastian Hübner. Ontology-based integration of information-a survey of existing approaches. In *roceedings of the IJCAI Workshop on Ontologies andInformation Sharing (IJCAI 2001), International Joint Conferences onArtificial Intelligence, Seattle, WA, US*, pages 108–117. Citeseer, 2001.

[165] Anusha Indika Walisadeera, Athula Ginige, and Gihan Nilendra Wikramanayake. Ontology evaluation approaches: a case study from agriculture domain. In *Gervasi O. et al. (eds) Computational Science and Its Applications – ICCSA 2016. ICCSA 2016. Lecture Notes in Computer Science*, volume 9789, pages 318–333. Springer, Cham, 2016.

[166] Mark Weiser, Rich Gold, and John Seely Brown. The origins of ubiquitous computing research at parc in the late 1980s. *IBM systems journal*, 38(4):693–696, 1999.

[167] Mark D Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E Bourne, et al. The fair guiding principles for scientific data management and stewardship. *Scientific data*, 3, 2016.

[168] Ye Yan, Yi Qian, Hamid Sharif, and David Tipper. A survey on smart grid communication infrastructures: Motivations, requirements and challenges. *IEEE communications surveys & tutorials*, 15(1):5–20, 2012.

[169] Kentaro Yoshimura, Dharmalingam Ganesan, and Dirk Muthig. Defining a strategy to introduce a software product line using existing embedded systems. In *Proceedings of the 6th ACM & IEEE International Conference on Embedded Software*, EMSOFT '06, page 63–72, New York, NY, USA, 2006. ACM. ISBN 1595935428.

[170] Baris Yuce and Yacine Rezgui. An ann-ga semantic rule-based system to reduce the gap between predicted and actual energy consumption in buildings. *IEEE Transactions on Automation Science and Engineering*, 14(3):1351–1363, 2015.

[171] Hongyu Zhang, Yuan-Fang Li, and Hee Beng Kuan Tan. Measuring design complexity of semantic web ontologies. *Journal of Systems and Software*, 83(5): 803–814, 2010.

[172] Jianchao Zhang, Boon-Chong Seet, and Tek Tjing Lie. An event-based resource management framework for distributed decision-making in decentralized virtual power plants. *Energies*, 9(8):595, 2016.

[173] Q. Zhou, S. Natarajan, Y. Simmhan, and V. Prasanna. Semantic information modeling for emerging applications in smart grid. In *2012 Ninth International Conference on Information Technology - New Generations*, pages 775–782, April 2012.

[174] Qunzhi Zhou, Yogesh Simmhan, and Viktor Prasanna. Incorporating semantic knowledge into dynamic data processing for smart power grids. In *Cudré-Mauroux P. et al. (eds) The Semantic Web – ISWC 2012. ISWC 2012. Lecture Notes in Computer Science*, volume 7650, pages 257–273. Springer, Berlin, Heidelberg, 2012.

[175] Qunzhi Zhou, Yogesh Simmhan, and Viktor Prasanna. Scepter: Semantic complex event processing over end-to-end data flows. Technical report, Computer Science Department, University of Southern California, 2012.

[176] Amal Zouaq and Roger Nkambou. A survey of domain ontology engineering: methods and tools. In *Nkambou R., Bourdeau J., Mizoguchi R. (eds) Advances in Intelligent Tutoring Systems. Studies in Computational Intelligence*, volume 308, pages 103–119. Springer, Berlin, Heidelberg, 2010.

# Appendices

Appendix A

# Green Energy Provider Selection System

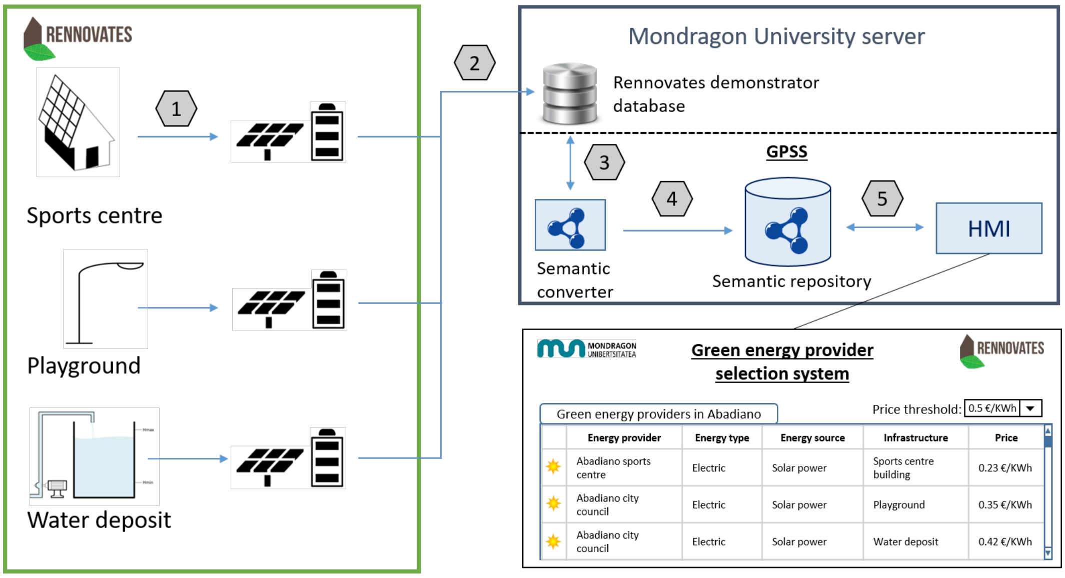
This appendix describes the Green Energy Provider Section System (GEPSS) pilot demonstrator. Specifically, it describes the following aspects: the system architecture, demonstrator technologies and the knowledge requirements of the application ontology that supports the system.

The GEPSS system is a multi-agent system that provides to the home energy consumer a list of the available green energy providers in the area where the home is located (the available energy providers are the ones that have surplus energy). Specifically, the system displays the energy provided energy type (i.e., electric energy, thermal energy), the energy source (i.e., solar power, wind power), the infrastructure that generates the energy (i.e., a solar panel installation) and the price at which the supplier sells the energy. In addition, users can set a price threshold they are willing to pay in order to use green energy.

The GEPSS data is represented through an application ontology and stored in a semantic repository. Specifically, the application ontology of the GEPSS system represents data about energy consumers, energy providers (energy type they provide and energy tariff they apply to each energy source), energy provider infrastructures, infrastructure energy generation and storage devices, and storage device current state. The semantic representation of the energy data of the GEPSS provides the following benefits:

- *Consensual representation of heterogeneous energy data*: data from different domains and collected and at different rates is represented and linked thorough Semantic Web standards.

- *Scalable data integration*: data from different domains linked in a simple and scalable manner. Hence, these data can be accessed and combined by the



**Green energy provider selection system**

Green energy providers in Abadiano

Price threshold: 0.5 €/KWh ▼

| | Energy provider | Energy type | Energy source | Infrastructure | Price |
|---|---|---|---|---|---|
| ☀ | Abadiano sports centre | Electric | Solar power | Sports centre building | 0.23 €/KWh |
| ☀ | Abadiano city council | Electric | Solar power | Playground | 0.35 €/KWh |
| ☀ | Abadiano city council | Electric | Solar power | Water deposit | 0.42 €/KWh |

**Figure A.1:** GEPSS pilot demonstrator architecture

GEPSS with simple queries.

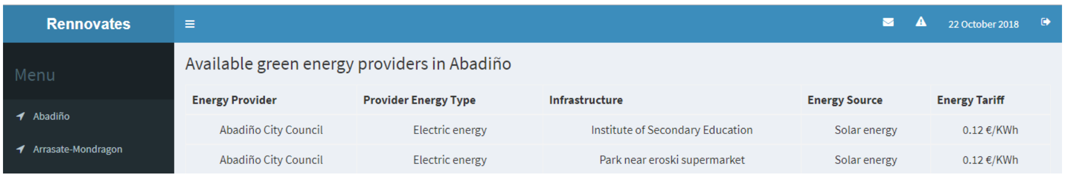
- *Energy data inference*: the system can infer which are the available green energy providers and from explicit facts annotated in a knowledge base. The application ontology represents the available green energy providers as the ones that own at least one infrastructure that produces renewable energy and has currently surplus energy. Hence, by only annotating the infrastructure features and current state of energy storage systems in the semantic repository, the GEPSS can infer whether an energy provider is green and available.

## System Architecture

The GEPSS was integrated into a pilot demonstrator of the Rennovates project[1] deployed in Mondragon University and Abadiano municipality (Spain). Figure A.1 shows the GEPSS architecture and operation, as well as its implementation within the pilot demonstrator.

Within the demonstrator, solar panels and batteries were installed in three infrastructures: a sports centre building, a playground and a water deposit. Hence, we consider that the owners of these infrastructures (the Abadiano sports centre

[1] https://rennovates.eu/



Available green energy providers in Abadiño

| Energy Provider | Provider Energy Type | Infrastructure | Energy Source | Energy Tariff |
|---|---|---|---|---|
| Abadiño City Council | Electric energy | Institute of Secondary Education | Solar energy | 0.12 €/KWh |
| Abadiño City Council | Electric energy | Park near eroski supermarket | Solar energy | 0.12 €/KWh |

**Figure A.2:** Data displayed by the GEPSS

and the Abadiano city council) are the energy providers. The photovoltaic energy production and battery charging state of each infrastructure is measured by electric inverters. An embedded system installed in each infrastructure dumps these data in real time into the Rennovates project database, deployed in Mondragon University servers. Finally, an HMI deployed in this server displays the energy performance data of the infrastructures.

The GEPSS is also deployed in these servers and includes three main elements: the semantic converter, semantic repository and the HMI. The semantic converter reads the battery charging state from the Rennovates database and converts it into semantically represented data, according to the GEPSS application ontology vocabulary. Once the data is converted, the semantic converter stores it in the semantic repository. The repository knowledge base also includes static data about the energy providers, infrastructures, the energy source/type they produce and the energy tariff that energy providers assign to each energy source. Finally, the HMI queries and make inferences about the energy data stored in the repository and displays the information about the home user the available green energy providers (see Figure A.2). In this demonstrator, we consider that energy providers are available when the battery attached to their infrastructures is at its maximum capacity.

## Demonstrator Technologies

This section describes the technologies applied to develop and deploy the elements of GEPSS.

- The semantic converter is a Java application that reuses the libraries of the RDF4J[2] semantic framework to convert the semantic data and store it in the semantic repository. RDF4J is an open source semantic framework written in Java that enables the querying, adding or modifying data of a RDF repository.

[2] http://rdf4j.org/

- The semantic repository corresponds to an Ontotext Graph DB[3] multiplatform semantic repository.
- Finally, the HMI was implemented though DJANGO[4] framework. To query data from the semantic repository, it uses the SPARQLwrapper [5] library. This library enables to application written in DJANGO the access and management of semantic data stored in a semantic repository.

## GEPSS Ontology Requirements

The following are the ontology requirements of the GEPSS system ontology in this experiment. They are expressed as a set of CQs that the ontology must answer. We classified the CQs according to the topics they cover.

Energy provider CQs:

- CQ1: What are the energy providers of a certain municipality?
- CQ2: What type of energy provides an energy provider?
- CQ3: From which energy sources produces energy an energy provider?
- CQ4: What is the energy tariff applied by an energy provider to a specific energy source?
- CQ5: What is the monetary cost of an energy tariff?
- CQ6: What infrastructures owns an energy provider?

Energy provider Infrastructure CQs:

- CQ7: Which generation systems are installed in an infrastructure?
- CQ8: What is the energy source of an energy generation system?
- CQ9: Which batteries are installed in an infrastructure?
- CQ10: What infrastructures have surplus energy?

[3] http://graphdb.ontotext.com/

[4] https://www.djangoproject.com/

[5] https://github.com/RDFLib/sparqlwrapper

- CQ11: What infrastructures are renewable?
- CQ12: What is the name of an infrastructure?
- CQ13: What is the description of an infrastructure?

Energy generation/storage system CQS:

- CQ14: Is a generation system renewable?
- CQ15: Is a battery full or not?
- CQ16: To which energy storage device is connected an energy generation system?

The remaining ontology requirements are expressed as natural language sentences:

1. Green energy providers are the ones that provide energy obtained from a renewable energy source.
2. Available energy providers are the ones that own at least one infrastructure that have surplus energy.
3. Infrastructures that have surplus energy are the ones that have at least one of the installed batteries at the maximum level of their capacity.
4. Renewable infrastructures are the ones that have at least one renewable energy generation system.
5. Renewable energy generation systems are the ones have one renewable energy source.
6. A device is in one infrastructure.
7. An energy generation system is connected to one energy storage system.
8. An energy generation system has one energy source.
9. An energy source tariff has one energy source.
10. An energy source tariff has one energy tariff.
11. The monetary cost of an energy tariff has one double value.
12. The monetary cost of an energy tariff has at least one currency unit.

Appendix B

# Infrastructure Energy Performance Assessment System

This appendix describes the Infrastructure Energy Performance Assessment Systems (EPAS) pilot demonstrator. Specifically, it describes the following aspects: the system architecture, demonstrator technologies and the knowledge requirements of the application ontology that supports the system.

The EPAS is a multi-agent system that provides a holistic view of the energy generation performance of green infrastructures that are self-sufficient in solar energy. Specifically, it displays the recent energy consumption, as well as the short-term weather and energy production forecast. It also represents the energy generation of an infrastructure of a certain period of time and the forecasted energy generation of that period of time. With this information, the user can take actions to reduce the building energy consumption and adjust it to the infrastructure energy performance forecast. The user can also analyse any gap between the real and forecasted energy generation to detect any problem in the energy generation devices. In addition, if the gap is minimal, the user can take previous predictions as reference to make long-term energy generation forecasts.

The EPAS data is represented through an application ontology and stored in a semantic repository. Specifically, the EPAS application ontology represents data about infrastructure stakeholders, the energy consumption and generation values of infrastructure devices, the infrastructure energy generation forecast, weather forecast and the data used to calculate the energy this forecast such as photovoltaic system features (i.e., tilt, capacity) and infrastructure location. The semantic representation of the energy data of the EPAS provides the following benefits:

- *Consensual representation of heterogeneous energy data*: through data from different domains and collected at different rates is represented and linked

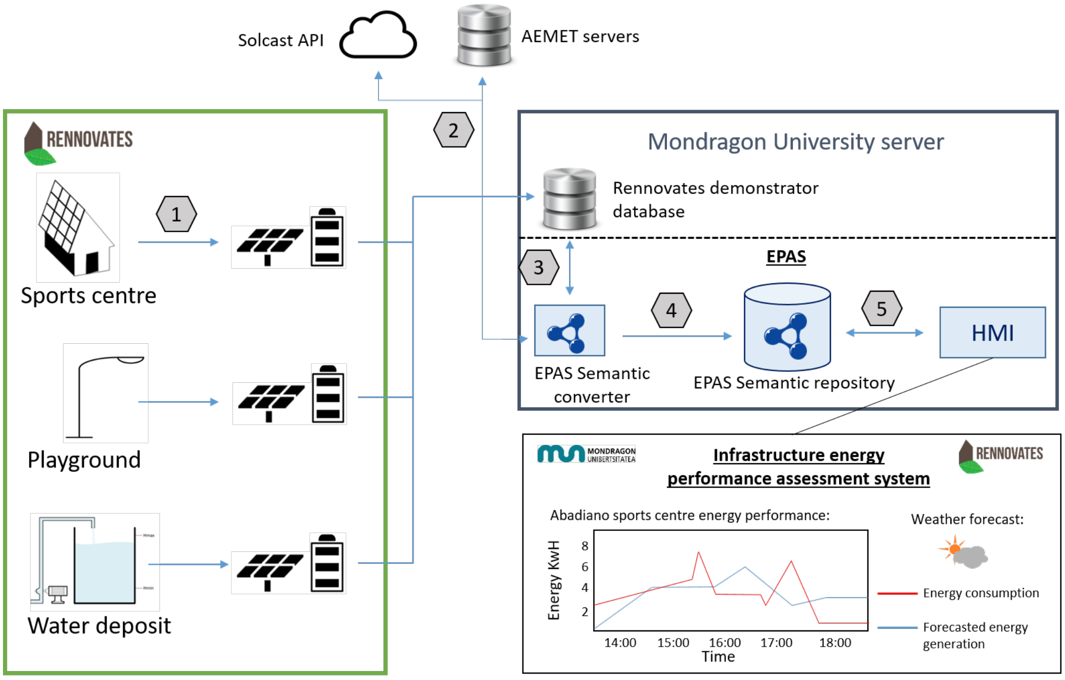


**Figure B.1:** Infrastructure EPAS pilot demonstrator architecture

thorough Semantic Web standards.

- *Scalable data integration*: data from different domains linked in a simple and scalable manner. Hence, these data can be accessed and combined by the EPAS with simple queries.

## System Architecture

The EPAS was integrated into a pilot demonstrator of the Rennovates project[1] deployed in Mondragon University and Abadiano municipality (Spain). Figure B.1 shows the EPAS architecture and operation, as well as its implementation within the pilot demonstrator.

Within the demonstrator, solar panels and batteries were installed in three infrastructures: a sports centre building, a playground and a water deposit. Hence, we consider that the owners of these infrastructures (the Abadiano sports centre and the Abadiano city council) are the energy providers. The energy consumption and generation values of each infrastructure is measured by electric inverters. An embedded system installed in each infrastructure dumps these data in real time

[1] https://rennovates.eu/

into the Rennovates project database, deployed in Mondragon University servers. Finally, an HMI deployed in this server displays the energy performance data of the infrastructures.

The EPAS is also deployed in these servers and includes three main elements: the semantic converter, semantic repository and the HMI. The semantic converter obtains the infrastructure energy consumption and generation from the Rennovates database and the weather forecast from the Spanish Meteorological Agency servers. Then, the semantic converter the converts these data into semantically represented data, according to the EPAS application ontology vocabulary, and stores it in the semantic repository. The semantic converter also obtains the short-term energy generation forecast of each infrastructure from the Solcast API[2]. This API provides the weekly energy generation forecast of a PV system through a web service. The semantic converter uses the static data about the infrastructures location and the features of the PV systems they include to query the energy consumption in the Solcast API. This static data is also stored in the semantic repository. Finally, the HMI queries the energy data stored in the repository and displays the information about the infrastructure energy consumption, production forecast and weather forecast (see Figure B.2).

## Demonstrator Technologies

This section describes the technologies applied to develop and deploy the elements of EPAS.

- The semantic converter is a Java application that reuses the libraries of the RDF4J [3] semantic framework to convert the semantic data and store it in the semantic repository. RDF4J is an open source semantic framework written in Java that enables the querying, adding or modifying data of a RDF repository.

- The semantic repository corresponds to an Ontotext Graph DB[4] multiplatform semantic repository.

- Finally, the HMI was implemented though DJANGO[5] framework. To query data from the semantic repository, it uses the SPARQLwrapper[6] library. This library enables to application written in DJANGO the access and management of semantic data stored in a semantic repository.

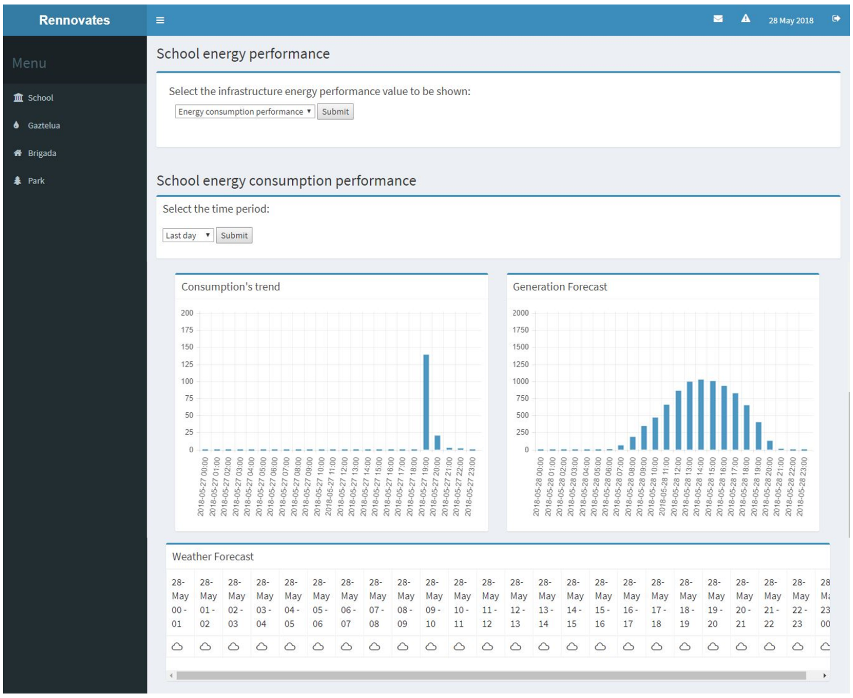
[2] https://solcast.com.au/solar-data-api/api/

[3] http://rdf4j.org/

[4] http://graphdb.ontotext.com/

[5] https://www.djangoproject.com/

[6] https://github.com/RDFLib/sparqlwrapper



Rennovates

28 May 2018

Menu

- School
- Gaztelua
- Brigada
- Park

School energy performance

Select the infrastructure energy performance value to be shown:

Energy consumption performance ▾ Submit

School energy consumption performance

Select the time period:

Last day ▾ Submit

Consumption's trend

Generation Forecast

Weather Forecast

| 28-May 00 - 01 | 28-May 01 - 02 | 28-May 02 - 03 | 28-May 03 - 04 | 28-May 04 - 05 | 28-May 05 - 06 | 28-May 06 - 07 | 28-May 07 - 08 | 28-May 08 - 09 | 28-May 09 - 10 | 28-May 10 - 11 | 28-May 11 - 12 | 28-May 12 - 13 | 28-May 13 - 14 | 28-May 14 - 15 | 28-May 15 - 16 | 28-May 16 - 17 | 28-May 17 - 18 | 28-May 18 - 19 | 28-May 19 - 20 | 28-May 20 - 21 | 28-May 21 - 22 | 28-May 22 - 23 | 28 Ma 23 00 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ | ☁ |

**Figure B.2:** Data displayed by the EPAS

## EPAS Ontology Requirements

The following are the ontology requirements of the EPAS system ontology in this experiment. They are expressed as a set of CQs that the ontology must answer. The CQs are classified according to the topic they cover.

### Infrastructure CQs:

- CQ1: What are the infrastructures of a certain municipality (i.e., city, town, village)?
- CQ2: What is the energy consumption of an infrastructure in a certain period?
- CQ3: What is the energy generation of an infrastructure in a certain period?
- CQ4: What is the energy generation forecast of an infrastructure in a certain period?
- CQ5: What is the name of an infrastructure?
- CQ6: What is the latitude and longitude of an infrastructure?

Device CQs:

- CQ7: Which PV devices are in an infrastructure?

Energy usage parameters CQs:

- CQ8: What is the value of an energy usage parameter (energy consumption and energy ration) measure?
- CQ9: What is the unit of the measured energy usage parameter (energy consumption and energy ration)?
- CQ10: What is the time interval of an energy usage parameter (energy consumption and energy ration) measure?

PVs CQs:

- CQ11: What is the inclination/tilt of a PV?
- CQ12: What is the azimuth angle of a PV?
- CQ13: What is the capacity of a PV?
- CQ14: What is the install date of a PV?
- CQ15: What is the loss factor of a PV?

The remaining ontology requirements are expressed as natural language sentences:

1. The energy consumption devices are HVAC systems, lighting systems and appliances.
2. An energy consumption device is anything that consumes some energy.
3. An energy consumption/generation measure has one value.
4. An energy consumption/generation measure has one energy unit of measure.
5. An energy consumption/generation measure has one time interval.
6. The inclination of a PV has a double value

7. The inclination of a PV has an angle unit of measure.

8. The azimuth of a PV has a double value.

9. The azimuth of a PV has an angle unit of measure.