Experiences on applying SPL Engineering Techniques to Design a (Re) usable Ontology in the Energy Domain

Javier Cuenca, Felix Larrinaga
Electronics and Computing Department
Mondragon University/Faculty of Engineering
Mondragon, Spain
jcuenca@mondragon.edu, flarrinaga@mondragon.edu

Edward Curry
Insight Centre For Data Analytics
National University of Ireland
Galway, Ireland
edward.curry@insight-centre.org

***Abstract*—Global ontologies must provide a balance of reusability-usability to minimize the ontology reuse effort in different applications. To achieve this balance, ontology design methods focus on designing layered ontologies that classify into abstraction layers the common domain knowledge (reused by most applications) and the variant domain knowledge (reused by specific application types). This classification is performed from scratch by domain experts and ontology engineers. Hence, the design of reusable and usable ontologies that represent complex domains takes a lot of effort. Considering how common and variant software features are classified when designing Software Product Lines (SPLs), we argue that SPL engineering techniques can facilitate the domain knowledge classification taking as reference existing ontologies. In this paper, we show the experiences of applying SPL and ontology design techniques in combination to design a reusable and usable global ontology for the energy domain. Domain experts and ontology engineers evaluated the proposed method. The results show that SPL engineering techniques enable a systematic and accurate domain knowledge classification, thus saving ontology design effort.**

Keywords-ontology design; ontology reusability; ontology usability; Software Product Line; energy domain.

I. INTRODUCTION

Ontologies are formal vocabularies that represent a data domain as a set of concepts and relations. The main ontology elements are classes (to represent entities, i.e., *device*, *appliance)*, instances (individuals that belong to a certain class, i.e., *MU_fridge*, *MU_building_11)* and properties (relations that relate classes and individuals, i.e., *isA*, *isIn*) (Fig. 1). With these elements, ontologies enable to create a generic knowledge that can be shared across different software applications [1].

Among the different ontology types, *global ontologies* include common vocabularies to provide a common domain knowledge representation (i.e., Soupa [2]). The knowledge of global ontologies is a reference to develop ontologies for specific applications (*application ontologies*) [3]. This common knowledge representation overcomes the vocabulary differences and the heterogeneity of ontologies in the domain concerned to enable interoperability between ontology-based applications [3].

A global ontology must represent abstract knowledge to support different applications: it must be *reusable* [4]. However, if the ontology is too abstract, the effort of adapting it to satisfy specific knowledge requirements would be high. Therefore, a global ontology must be as specific as possible to minimize the ontology reuse effort when it is reused to develop application ontologies: it must be *usable* [4]. Both reusability and usability are objectives in "*in natural conflict*" [4]. Hence, a global ontology must achieve a balance of reusability-usability so that it is reused in different applications with moderate effort.

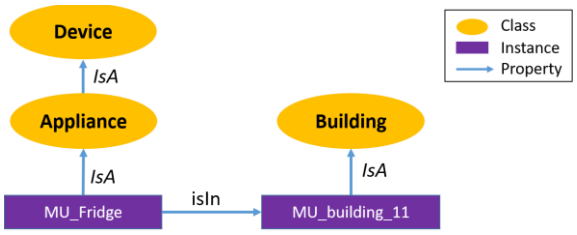

Figure 1: Ontology example

To date, *layered ontologies* have been applied to achieve a balance of reusability-usability (i.e., OntoCape ontology [4]). They separate and classify into 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. In addition, the knowledge of each layer is divided into ontology modules, which represent the knowledge of a particular topic of the represented domain (each module imports the modules whose knowledge it requires or extends) [5]. Layered ontologies enable ontology developers to reuse only the necessary knowledge at the proper level of abstraction to develop application ontologies, thus reducing the ontology reuse effort in different applications [6].

Current methods applied to design reusable and usable ontologies [4], [7], [8] provide guidelines to define the ontology layers and the knowledge they include. However, they do not provide systematic guidelines to decide whether the domain knowledge is common or variant and in which layer it is placed. Domain experts and ontology engineers define and classify from scratch the common and variant domain knowledge. Ontologies are usually developed in complex domains (i.e., energy) [7]. Hence, a significant effort is required to classify the ontology knowledge from scratch by applying existing methodologies when designing reusable and usable ontologies.

Layered ontologies are quite similar in concept to Software

DOI reference number: 10.18293/SEKE2019-111

Product Lines (SPLs), software families that contain common reusable parts and variable parts that depend on specific customer needs [9]. To design SPLs, software features for a set of applications are analyzed and classified into *common features* (common to most applications) and *variant features* (implemented by specific applications) [9]. This process is known as *commonality and variability analysis* (CVA) [8]. This analysis is usually performed systematically taking as reference the software feature similarities and differences of legacy applications to complement domain experts' and software engineers' expertise [8]. This approach avoids classifying the software features from scratch, thus reducing the SPL design effort [10]. In addition, the software features reused by few applications are identified and the accuracy of the software feature classification is maximized [10].

After several decades of ontology development, many ontologies are available and developed to support certain application types. Hence, the CVA applied to design SPLs can be replicated in the ontology engineering field to design reusable and usable ontologies. The similarities and differences of the knowledge represented by existing ontologies can be analyzed. This analysis would complement domain experts and ontology engineers' expertise and prevent them from classifying the domain knowledge from scratch. In addition, the variant domain knowledge reused by specific applications could be identified, thus leading to an accurate domain knowledge classification.

In this paper, we show the experiences of applying SPL engineering techniques and ontology design principles in combination to design a reusable and usable global ontology for the energy domain. We discuss how applicable are SPL engineering techniques to design reusable and usable ontologies and the benefits they bring to the ontology design process.

The paper is structured as follows. Section 2 motivates the application of SPL engineering techniques to design a reusable and usable global energy ontology. Section 3 positions the proposed method with current ontology and SPL design methodologies. Section 4 describes the process we followed to design a global energy ontology by applying SPL engineering and ontology design techniques in combination. In Section 5, domain experts and ontology engineers evaluate the proposed method and Section 6 summarizes the learnt lessons. Section 7 summarizes the conclusions of the study and the future work.

II. MOTIVATIONAL SCENARIO

From the beginning of the current decade, ontologies that represent the knowledge from different energy domains have been developed. These ontologies support energy management applications focused on improving the current grid sustainability to make the Smart Grid vision a reality [11]. These applications can be classified into different types according to the Smart Grid scenario/infrastructure where they are deployed, i.e., Smart Home or building energy management applications. We define these application types as *Smart Grid scenarios*. Each Smart Grid scenario encompasses more specific application types. For instance, within Smart Home energy management applications, there are applications focused on home energy assessment or appliance Demand Response (DR) management. To see in more detail this classification we refer the reader to [12].

Energy ontologies are heterogeneous, since they represent the same energy domains (i.e., energy equipment data) with different vocabularies [12]. The energy management in real scenarios will require the knowledge exchange among applications that operate in different scenarios. This knowledge exchange is hampered by the heterogeneity of energy ontologies. Hence, there is the need to create a global ontology that provides a common energy domain representation [12]. It should support different energy management applications and provide balance of reusability-usability to minimize the ontology reuse effort in each application. Since the energy domains are complex, the application of existing reusable and usable ontology design methodologies would require a great effort. Since there are many developed energy ontologies, their knowledge similarities/differences can be analyzed to save ontology design effort.

III. RELATED WORK

The first knowledge classification proposals correspond to frameworks that classify ontologies according to their generality/specificity level. Guarino [13] presented the first ontology classification framework, which was refined by Gomez-Perez [14]. Layered ontologies (introduced in Section 1) are based on the aforementioned frameworks and they are the main approach to design ontologies that provide a balance of reusability-usability. In the last decade, several reusable and usable ontology design methodologies (based on the layered ontology approach) have been proposed. Spyns et al. [8] presented the DOGMA methodology, which specifies how to represent and separate the common and variant domain knowledge to design ontologies that provide a balance of reusability-usability. Thakker et al. [7] set out a methodology to develop reusable and usable ontologies for complex domains. Morbach et al. [4] developed the OntoCape ontology, a reusable and usable ontology for the chemical process engineering domain. In these methodologies, the classification of the domain knowledge is performed from scratch based on domain experts' and ontology engineers' expertise. They analyze the knowledge requirements of the application types that will be supported by the ontology (in collaboration with stakeholders). In contrast, in the method presented in this paper the common and variant domain knowledge is identified and classified through a CVA of existing ontologies conducted by applying SPL engineering techniques. Apart from this differential aspect, the proposed method applies the ontology design principles applied by current methodologies.

Regarding SPL design approaches, Pohl et al. [9] provide guidelines and enumerate the techniques to conduct a CVA. The proposed method follows these guidelines. In addition, several works have combined techniques from ontology and SPL engineering. Most of these works consist on the use of ontologies to improve the representation of common and variant software features of SPLs [15], [16]. In other works, ontologies and SPLs have been applied in combination, i.e., to manage cloud service configurations [17]. The proposed method also combines ontology and SPL techniques. In this case, SPL engineering techniques are applied to improve the ontology design process.

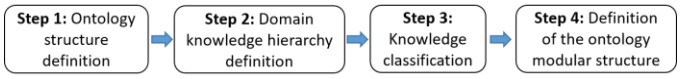

Figure 2: DABGEO design steps

IV. PROPOSED METHOD

This section explains the process we followed to design a global energy ontology: the DABGEO ontology. DABGEO provides a common energy domain representation and classifies the domain knowledge into abstraction layers to provide a balance of reusability-usability. DABGEO and its documentation are published online [1], so that ontology developers can understand its structure and reuse it. The DABGEO design and development team included energy domain experts and ontology engineers. DABGEO was designed by applying SPL engineering and ontology design techniques in combination. The design process followed four steps (Fig. 2), described in the following subsections.

A. Step 1: Ontology Structure Definition

In this step, the DABGEO structure was defined by the domain experts based on the layers proposed by the methods reviewed in Section 3. DABGEO includes three layers (Fig. 3). The *common-domain layer* represents the knowledge common to the 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 and is divided into the *Smart Grid scenario* and the *application type* sublayers. The former represents the knowledge reused by a certain Smart Grid scenario and the later represents the knowledge reused by a certain application type of a Smart Grid scenario. The lower the layer, the more specific the knowledge it represents. Hence, the modules from low-level layers will import the modules from upper layers. For more information about the ontology structure, we refer the reader to the ontology publication page[1].

B. Step 2: Domain Knowledge Hierarchy Definition

In this step, domain experts and ontology engineers defined and structured DABGEO knowledge. The knowledge was defined as a knowledge hierarchy where the represented domains were divided into specific knowledge pieces. This knowledge hierarchy enabled (1) to separate the abstract knowledge that is likely to be reused in most of applications from the specific knowledge and (2) to classify of the defined knowledge pieces into the layers of the ontology structure (performed in Step 3). Fig. 4 shows part of DABGEO knowledge hierarchy, which includes three elements:

- *Domains:* the domains represented by the ontology are located in the first level of the hierarchy. For instance, the *energy equipment domain* encompasses the knowledge about energy devices and their operation.

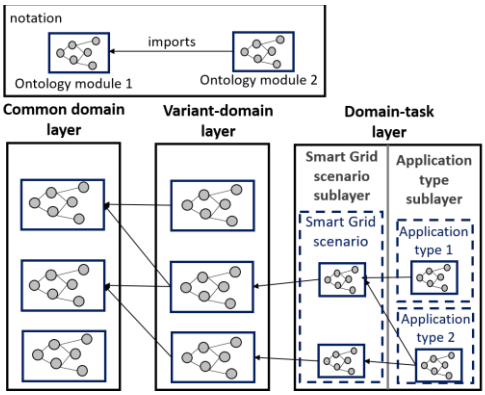

Figure 3: DABGEO ontology structure

- *Subdomains:* they cover the knowledge of an important part of the domain and are located in the second level of the hierarchy. For instance, the *energy equipment domain* encompasses the *energy consumption systems* and *device operation* subdomains, which represent the knowledge about energy consumption devices and device functional features respectively.
- *Knowledge Areas (KAs):* in the third level of the hierarchy, consider a KA as a potential module of the designed ontology that encompasses the knowledge of a specific topic of a subdomain. For instance, within the *energy consumption systems subdomain* the *appliances KA* represents the knowledge about appliance types. KA can be divided into "child" sub-KAs that represent more specific knowledge. For example, the *appliances KA* includes the *white goods* and *brown goods* KAs, which represent the knowledge about white and brown goods types respectively. Hence, a sub-KA extends the knowledge of a "parent" KA. Finally, some KAs may require the knowledge from other KAs to represent the knowledge they encompass. For instance, the *energy consumption systems operation KA* describes the states and functionalities of energy consumption systems. It requires the knowledge of *device state* and *device functionality* KAs, which represent the knowledge about device states and functionalities respectively.

The proposed method classifies the ontology domain knowledge based on a CVA of existing energy ontologies. Thus, the knowledge hierarchy includes the knowledge represented by existing ontologies. The domain experts and ontology engineers collaborated to perform a manual analysis of the elements of existing ontologies in the Protégé ontology editor[2] to identify the domains they represent and to divide them into KAs. The identified domains were divided into subdomains, which were divided into KAs taking as reference the Competency Questions (CQs) answered by existing ontologies. CQs are the queries that ontologies should answer to ontology-based applications, and they are used to define the ontology functional

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

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

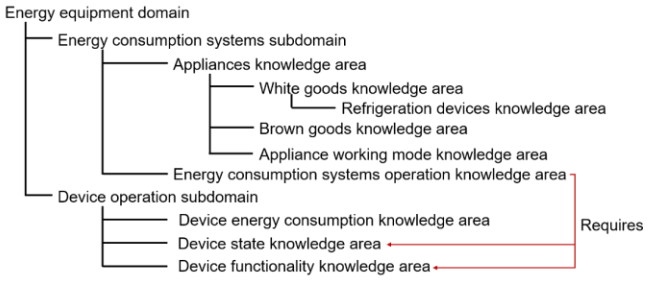

Figure 4: Part of the knowledge hierarchy of DABGEO

requirements [18]. To answer each CQ the ontology must include a specific part of the represented knowledge [18]. Thus, CQs are a natural guide for splitting the ontology knowledge into KAs [19]. CQs offer an abstract method to divide the knowledge represented by existing ontologies regardless of their heterogeneity. However, the CQs defined to develop ontologies are not always available [19].

Therefore, ontology engineers manually analyzed the elements of existing energy ontologies (classes and properties) to identify and extract the CQs they answer. This strategy is also followed when designing SPL taking as reference existing applications [20]. For instance, the exiting energy ontologies include the *consumesEnergy*, *actuallyConsumesEnergy* and *maxConsumesEnergy* properties to answer the *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?* CQs respectively. To avoid an unmanageable number of KAs, the CQs covering similar topics were grouped by domain experts to define a KA that encompasses all the knowledge required to answer grouped CQs. For instance, the aforementioned CQs describe knowledge about device energy consumption. They were grouped into the *device energy consumption KA* (it also includes CQs answered by other energy ontologies), which encompasses the knowledge that answers these CQs. The defined KAs were classified into domains and subdomains according to the knowledge they represent and into a hierarchy level according to the knowledge they require or extend.

Finally, the domain experts provided a complete description of each KA and the knowledge it encompasses.

C. Step 3: Knowledge Classification

In this step, the ontology engineers classified each defined KA into one layer by applying SPL engineering techniques. First, the existing energy ontologies were analysed manually with Protégé to determine whether they represent each defined KA. If the ontology contained classes or properties related with the knowledge encompassed by the KA, the KA was considered as represented. The domain experts collaborated with ontology engineers to give additional explanations about the knowledge encompassed by KAs. It is worth mentioning that if a "child" KA was represented by the ontology, the "parent" KA that represents more abstract knowledge was also considered as represented.

Second, a CVA of existing ontologies was conducted to determine whether the KAs were common to Smart Grid scenarios. In particular, the *application-requirements matrix* technique proposed by Pohl et al. [9] was applied (taking as reference application-requirements matrix applied by Moon et al. [21]) to determine whether the KAs are common to Smart Grid scenarios depending on how many ontologies represent them. One application-requirements matrix was created to classify the KAs of each subdomain. As an example, Table 1 shows the application-requirements matrix of a set of KAs of the *energy consumption systems* and *device operation subdomains* (34 KAs were defined in total for these subdomains). The left column contains the KAs of the subdomain. The top rows list the Smart Grid scenarios and the energy ontologies classified by the Smart Grid scenarios they support (this classification can be consulted at [12]). To simplify the matrix, we have omitted a couple of ontologies. The matrix indicates if an ontology represents a KA ('X') or not ('-'). With this information, we could deduce which Smart Grid scenarios reuse each KA. We considered that a Smart Grid scenario reuses a KA if the KA is represented by at least one ontology developed to support the Smart Grid scenario. KAs were classified into common and variant according to their *commonality ratio* (CV ratio) (right column in Table 1): the ratio of the number of Smart Grid scenarios that reuse the KA to the total number of Smart Grid scenarios. In particular, 75% was used as the threshold value of the CV ratio to classify the KAs. The KAs equal or above the threshold were considered as

TABLE 1: APPLICATION-REQUIREMENTS MATRIX

	Smart Grid scenarios								
	Smart Home energy management			**Building/district/city energy management**		**Organization energy management**		**Smart Grid Demand Response management**	
Ontologies / Knowledge areas	ThinkHome ontology	EnergyUse ontology	SAREF4EE ontology	SEMANCO ontology	BOnSAI ontology	DEFRAM project ontology	DERI Linked dataspace	ProSGV3 ontology	**Commonality ratio**
Appliances	X	X	X	X	-	X	-	X	100%
Brown goods	X	X	-	-	X	-	X	X	100%
White goods	X	X	X	X	-	-	-	X	75%
Refrigeration devices	X	X	-	-	-	-	-	X	50%
Device energy consumption	X	X	X	-	X	-	X	X	100%
Energy consumption systems operation	X	X	-	-	-	-	-	-	25%
Appliance working mode	-	-	X	-	-	-	-	-	25%

TABLE 2: CVA AT APPLICATION TYPE LEVEL

	Smart Home energy management		
	Home energy assessment	**Home energy saving advice**	**Home appliances Demand Response management**
Ontologies / Knowledge areas	ThinkHome ontology	EnergyUse ontology	SAREF4EE ontology
Energy consumptions systems operation	X	X	-
Appliance working mode	-	-	X

common, while the rest were considered as variant.

Third, each KA was classified into one layer according to the CVA results. The common KAs were placed in the *common-domain layer*. Variant KAs reused in more than one Smart Grid scenario were assigned to the *variant-domain layer*. The KAs reused only in one Smart Grid scenario were assigned to one of the sublayers of the *domain-task layer* according to a CVA at the application type level. The KAs reused by more than one application type of a Smart Grid scenario are likely to be reused in more application types of that scenario and were placed in the *Smart Grid scenario sublayer*. The KAs reused only by one application type were assigned to the *application type sublayer*. Following the sample CVA of Table 1, the *energy consumption systems operation* and the *appliance working mode* KAs were reused only by Smart Home energy management applications. Thus, they were included in the CVA at application type level (Table 2). The *energy consumption systems operation KA* was reused by more than one Smart Home energy management application type. Hence, it was placed in the *Smart Grid scenario sublayer*. The *appliance working mode KA* was reused only by one Smart Home energy management applications and placed in the *application type sublayer*.

D. Step 4: Definition of the Ontology Modular Structure

In this step, the ontology engineers structured the knowledge of each layer into ontology modules to complete the ontology design. This step was performed taking as reference the ontology modularization principles applied by the main reusable and usable ontology design methods: loosely coupling and self-containment [5]. One module was defined for each KA and placed in one ontology layer/sublayer according to the CVA results. The modules were related according to the knowledge dependencies defined in Step 2. The modules of the *Smart Grid scenario* and *application type* sublayers were classified into the Smart Grid scenario/application types where the KAs they represent are reused.

V. EVALUATION

The ontology design method presented in Section 4 was evaluated by domain experts and ontology engineers. A group of domain experts and ontology engineers conducted Steps 1 and 2 and different ontology engineers (eight in total) conducted Steps 3 and 4 to design parts of DABGEO. Each ontology engineer performed Steps 3 and 4 individually in a blind process. A survey was performed to capture the knowledge classification obtained by each ontology engineer. The survey also included a questionnaire to identify the main advantages and improvement aspects of the proposed method. This questionnaire can be found online in the appendix: `https://innoweb.mondragon.edu/innoweb/questionnaire_SPL_ontology_method.pdf`.

Fig. 5 shows the number of modules defined by each engineer for each ontology layer. It 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 designed ontology parts were limited to support three application types (shown in Fig. 5). In general terms, 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 defined KAs into layers. We understand by degree of consensus of a KA the percentage of ontology engineers that classified the KA into the same layer. The average degree of consensus of the KAs was 76%. Therefore, there was a high consensus when classifying the common and variant domain knowledge. In addition, 81% of the KAs that were classified into one application type within the *domain-task* layer by most of ontology engineers, were not classified into other application types by other ontology engineers. Hence, the knowledge reused only by specific application types was identified.

Regarding the questionnaire, we received eight responses from participants of the proposed method evaluation. 100% respondents considered that the application of SPL engineering techniques was useful and 80% would recommend the proposed method to design reusable and usable ontologies in other domains apart from the Energy. According to the respondents, the main benefits of the proposed method are the following: (1) it provides clear and mechanical steps to classify the ontology domain knowledge taking as reference existing ontologies and (2) the CVA of existing ontologies provides a detailed classification of the knowledge reused by specific application types, while keeping separate the knowledge reused by most applications. On the other hand, the main improvement aspect deals with the required manual effort. Although the proposed method prevented from classifying the domain knowledge from scratch, it required a significant manual analysis effort to check whether each KA is represented by existing ontologies.

VI. LESSONS LEARNT

Considering the similar knowledge classifications obtained in Section 5, domain experts and ontology engineers could apply the steps of the proposed method to (1) perform a CVA of existing ontologies and (2) classify the domain knowledge into different layers. The evaluation participants did not need to perform an analysis of the requirements of each application type to classify the common and variant domain knowledge of DABGEO from scratch. In addition, they considered the proposed method useful and easy to follow. Hence, we can state that the CVA of existing ontologies complements domain experts and ontology engineers' expertise when designing

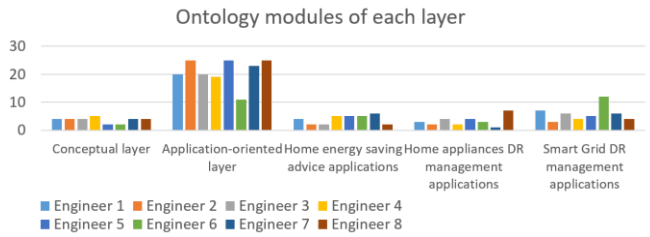

Figure 5: Ontology modules of each layer

reusable and usable ontologies in complex domains, thus saving ontology design effort. In addition, the ontology engineers identified the variant domain knowledge reused in specific applications. Thus, we can state that the CVA of existing ontologies enables an accurate domain knowledge classification. 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 identified and their knowledge should be divided and classified into different abstraction levels. Then, the ontologies should be analyzed to classify the domain knowledge based on their knowledge similarities and differences through a CVA.

Despite of these promising results, the CVA should be conducted with tool support to automate the process of checking whether certain KAs are represented by existing ontologies. In particular, tools that check (semi)automatically if a set of CQs are answered by ontologies should be developed. These tools can take as input the CQs encompassed each KA to check whether existing ontologies represent the KAs [18].

VII. Conclusions and Future Work

In this paper, we have shown the experiences of applying SPL engineering and ontology design techniques in combination to design DABGEO, a reusable and usable global energy ontology. The proposed method analyses the knowledge similarities and differences of existing ontologies to classify the common and variant domain knowledge into different layers. The method was applied by domain experts and ontology engineers to design part of DABGEO. The results show that that the application of SPL engineering techniques enables a systematic and accurate domain knowledge classification that complements domain experts and ontology engineers' expertise. Hence, ontology design effort of reusable and usable ontologies is saved. Bearing in mind these benefits, the proposed approach should be applied to design ontologies that will be reused in different applications and represent complex domains.

Our current work is focused on defining a methodology based on the proposed method, so that it can be applied and replicated in other complex domains apart from the Energy. We are working on describing in detail and generalizing each step. The medium-term work will focus on incorporating tool support to reduce the manual ontology analysis effort.

References

[1] T. Gruber, *Ontology*. Springer, 2009.

[2] H. Chen, F. Perich, T. Finin, and A. 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*, 2004, pp. 258–267.

[3] H. Wache, T. Voegele, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann, and S. Hübner, "Ontology-based integration of information-a survey of existing approaches," in *IJCAI-01 workshop: ontologies and information sharing*, 2001, vol. 2001, pp. 108–117.

[4] J. Morbach, A. Wiesner, and W. Marquardt, "OntoCAPE—A (re) usable ontology for computer-aided process engineering," *Computers & Chemical Engineering*, vol. 33, no. 10, pp. 1546–1556, 2009.

[5] M. d Aquin, "Modularizing ontologies," in *Ontology Engineering in a Networked World*, Springer, 2012, pp. 213–233.

[6] J. Morbach, A. Yang, and W. Marquardt, "OntoCAPE:A large-scale ontology for chemical process engineering," *Engineering applications of artificial intelligence*, vol. 20, no. 2, pp. 147–161, 2007.

[7] D. Thakker, V. Dimitrova, L. Lau, R. Denaux, S. Karanasios, and F. Yang-Turner, "A priori ontology modularisation in ill-defined domains," in *Proceedings of the 7th International Conference on Semantic Systems*, 2011, pp. 167–170.

[8] P. Spyns, Y. Tang, and R. Meersman, "An ontology engineering methodology for DOGMA," *Applied Ontology*, vol. 3, no. 1–2, pp. 13–39, 2008.

[9] K. Pohl, G. Böckle, and F. J. van Der Linden, *Software product line engineering: foundations, principles and techniques*. Springer Science & Business Media, 2005.

[10] A. Fantechi, S. Gnesi, I. John, G. Lami, and J. Dörr, "Elicitation of use cases for product lines," in *International Workshop on Software Product-Family Engineering*, 2003, pp. 152–167.

[11] E. Curry, W. Derguech, S. Hasan, C. Kouroupetroglou, and U. 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*, vol. 90, pp. 405–422, 2019.

[12] J. Cuenca, F. Larrinaga, L. Eciolaza, and E. Curry, "Towards Cognitive Cities in the Energy Domain," in *Designing Cognitive Cities*, Springer, 2019, pp. 155–183.

[13] N. Guarino, "Semantic matching: Formal ontological distinctions for information organization, extraction, and integration," in *Information Extraction A Multidisciplinary Approach to an Emerging Information Technology*, Springer, 1997, pp. 139–170.

[14] A. Gomez-Perez, M. Fernández-López, and O. Corcho, *Ontological Engineering: with examples from the areas of Knowledge Management, e-Commerce and the Semantic Web*. Springer Science & Business Media, 2006.

[15] K. Czarnecki, C. Hwan, P. Kim, and K. Kalleberg, "Feature models are views on ontologies," in *10th International Software Product Line Conference (SPLC'06)*, 2006, pp. 41–51.

[16] T. Asikainen, T. Männistö, and T. Soininen, "Kumbang: A domain ontology for modelling variability in software product families," *Advanced Engineering Informatics*, vol. 21, no. 1, pp. 23–40, 2007.

[17] C. Quinton, N. Haderer, R. Rouvoy, and L. Duchien, "Towards multi-cloud configurations using feature models and ontologies," in *Proceedings of the 2013 international workshop on Multi-cloud applications and federated clouds*, 2013, pp. 21–26.

[18] M. C. Suárez-Figueroa, "NeOn Methodology for building ontology networks: specification, scheduling and reuse," Informatica, 2010.

[19] F. B. Ruy, G. Guizzardi, R. A. Falbo, C. C. Reginato, and V. A. Santos, "From reference ontologies to ontology patterns and back," *Data & Knowledge Engineering*, 2017.

[20] A. Harhurin and J. Hartmann, "Service-oriented commonality analysis across existing systems," in *2008 12th International Software Product Line Conference*, 2008, pp. 255–264.

[21] M. Moon, K. Yeom, and H. S. 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*, vol. 31, no. 7, pp. 551–569, 2005.