Dynamic OWL Ontology Design Using UML and BPMN

J. I. Olszewska[1], R. Simpson[2] and T. L. McCluskey[2]
[1]*School of Computing and Technology, University of Gloucestershire, The Park, Cheltenham, GL50 2RH, U.K.*
[2]*School of Computing and Engineering, University of Huddersfield, Queensgate, Huddersfield, HD1 3DH, U.K.*

Keywords: Ontology, OWL, Dynamic Design, Knowledge Engineering, Software Engineering, UML, BPMN.

Abstract: Ontology design is a crucial task for the Semantic Web. In the literature, methodologies have been proposed to develop ontologies, however the phase between knowledge gathering and knowledge coding remains challenging. In this paper, we propose a dynamic ontology design based on dynamic design notations for a systematic identification of the relations between domain concepts. For this purpose, we propose the use of the Unified Modeling Language (UML) and the Business Process Modeling Notation (BPMN), and the mapping of the related dynamic notations to the ontology domain. Our approach has been successfully validated in a study case of an ontology with a publication repository domain.

1 INTRODUCTION

Ontologies are considered to be one of the pillars of the Semantic Web. More specifically, an ontology is a notion defined by Gruber as an explicit specification of a conceptualization (Gruber, 1995). The term (from the Greek, *ontos*: of being and *logia*: study) is borrowed from Philosophy and it refers to the subject of existence. In Artificial Intelligence (AI), an ontology is constituted by a specific vocabulary used to describe a certain reality, plus a set of explicit assumptions regarding the intended meaning of the vocabulary (Green and Rosemann, 2005). Thus, the ontology describes a formal specification of a certain domain, i.e. a shared understanding of a domain of interest as well as a formal and machine understandable model of this domain. Ontologies have been widely used for capturing, sharing, and representing knowledge (Fensel, 2001; Auer et al., 2006; Aljandal et al., 2009; Corsar et al., 2009; Olszewska, 2011).

Despite many methodologies proposed to develop ontologies (Fernández-López, 1999; Corcho et al., 2003; Gómez-Pérez et al., 2004; Seremeti and Kameas, 2009), design of large-scale, interoperable ontologies is still a challenge (Jimenez-Ruiz et al., 2012).

Interoperability is the challenge of getting processes to share and exchange information effectively. Service orientation relates to creating self-contained, self-describing, accessible, and open, computer services.

Both these challenges relate to the representation of the data being exchanged/manipulated. In our application of research data management, there are various existing sources of research information in a University, for example, its publications repository 'ePrints' (ePrints, 2014). Research information is complex, structured data, and the future requirements of it are only partially known. If we commit to one encoding, or even one representation language, later it may turn out to be inadequate or obsolete. Current work (Jain and Pareek, 2010) on these issues points to representing the data in an ontology.

In the e-Business context (Gessa et al., 2006), a mechanism to improve system usability, maintenance, efficiency, and interoperability could reside in the formal description of the semantic of the document-based framework for business collaborations. The formal descriptions could be provided through the definition of an ontology that represents the implicit concepts and the relationships that underlie the business vocabulary.

Hence, in this work, we propose to develop a large-scale, interoperable ontology for research information systems such as ePrints. For this purpose, we have introduced dynamic design notations to systematically construct concept relations in addition to capture the ontology scope with static design notations. Indeed, Unified Modeling Language (UML) has been extensively applied in Software Engineering for requirements analysis and software design (Marshall, 2000; Lunn, 2003). However, UML use

Olszewska J., Simpson R. and McCluskey T..
Dynamic OWL Ontology Design Using UML and BPMN.
DOI: 10.5220/0005159204360444
In *Proceedings of the International Conference on Knowledge Engineering and Ontology Development* (KEOD-2014), pages 436-444
ISBN: 978-989-758-049-9

in Knowledge Engineering (Baclwaski et al., 2001; Kogut et al., 2002; Hermida et al., 2009) has been limited to static design (De Nicola et al., 2009) for class identification.

Our dynamic ontology design involves Unified Modeling Language (UML) and Business Process Modeling Notation (BPMN) to capture the knowledge and uses OWL-DL language to codify the ontology. Thus, our approach allows not only the identification of the concepts, but also a systematic representation of their relations.

The contributions of this work are:

- to combine BPMN with UML to dynamically capture the scope of the ontology;
- to propose an ontology development scheme appropriate for interoperable research information system such as ePrints;
- to develop representations of knowledge for research information in an ontological form.

The paper is structured as follows. In Section 2, we briefly introduce the state-of-the-art methodologies to build an ontology, while in Section 3, we present our approach to design an ontology from scratch, in particular using dynamic design notations in the phase of ontology domain capture. The proposed method has been successfully tested to develop representations of knowledge for research information system in an ontological form as reported and discussed in Section 4. Conclusions are drawn up in Section 5.

2 METHODOLOGIES

The creation of an ontology requires specialized skills and involves various stakeholders. The ontology development process depends on a variety of factors like the choice of the software tool used to build and edit the ontology, the language in which the ontology is implemented, the methodology which will be followed to develop it, the applications in which it will be used, the type of the ontology under construction, the available formal and informal existing knowledge resources, such as lexicons, existing ontologies, etc, and may include a large number of necessary activities.

There is no established and unique procedure to develop ontologies despite several methodologies that have been proposed over time such as Cyc Methodology (Lenat and Guha, 1990), Enterprise Ontology (EO) Methodology (Uschold and King, 1995), Toronto Virtual Enterprise (TOVE) Modelling Methodology (Gruninger and Fox, 1995), KACTUS Methodology (Bernaras et al., 1996), Skeletal Methodology (Uschold and Gruninger, 1996), METHONTOLOGY (Fernández-López et al., 1997), SENSUS Methodology (Swartout et al., 1997), Enhanced Methodology (Ohgren and Sandkuhl, 2005), or Integrated Ontology Development Methodology (Chaware and Rao, 2010).

However, four general tasks to build ontologies have been identified:

- selection (includes selection of the available resources such as related literature, existing ontologies, group of experts in the domain under description, selection of the appropriate tool and language);
- analysis (includes analysis of selected resources, of the classes and the properties of the selected ontology);
- definition (includes definition of what is important for the description of a specific domain through the competency questions, definition of the purpose and the domain of the ontology, the definition of the classes, the class hierarchy, the properties and the instances of the ontology);
- evaluation (includes evaluation of the selected resources, evaluation of the technical quality of the ontology and evaluation of the overall quality of the obtained results).

These tasks are distributed into different phases, as follows:

- the specification phase (answers why the ontology is being built, what its intended uses are, who the end-users are);
- the conceptualization phase (conceptualizes the domain knowledge);
- the implementation phase (transforms the conceptual model into a formal computable model);
- the evaluation phase (assesses the resulting ontology).

These phases correspond roughly to the main steps of software engineering methodologies like presented in the IEEE Standard 1074-1995 for Developing Software Life Cycle Processes (Fernández-López, 1999).

Moreover, the ontology development methodologies could be classified into two categories:

- methodologies focused on building a single ontology for a specific ontology for a specific domain of interest;
- methodologies focused on the construction of ontology networks.

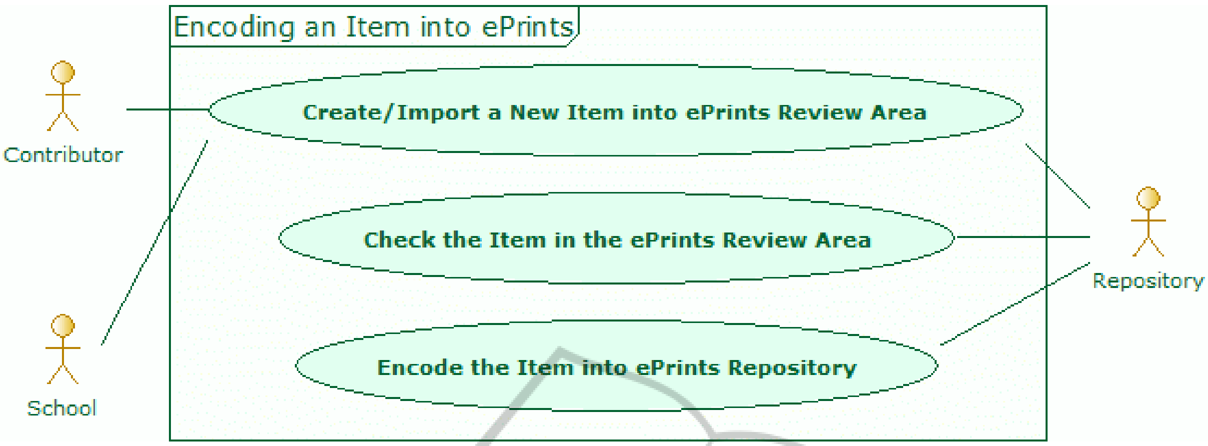

Figure 1: UML Use Case Diagram describing the action of encoding an Item into our ontology.

The single ontologies could be further distinguished among those aiming at building ontologies:

- from scratch;
- by reusing pre-existing ontologies;
- by using non-ontological resources.

These single ontologies are also divided in collaborative and non-collaborative, according to the degree of participation of the involved ontology engineers, users, knowledge engineers and domain experts in the ontology engineering process.

They are also described as application dependent, semi-application dependent and application independent, according to the degree of dependency of the developed ontology on its final application.

The single-ontology capture approach could vary according to the adopted strategy for identifying concepts, and could be bottom-up (from the most concrete to the most abstract), top-down (from the most abstract to the most concrete), or middle-out (from the most relevant to the most abstract and most concrete).

We can further distinguish manual, semi-automatic and automatic ontology construction, according to the degree of human involvement in the building process (Seremeti and Kameas, 2009).

However, the above mentioned criteria are not standards. Furthermore, none of the methodologies proposed in the literature (Fernández-López, 1999) are fully mature and they need then to be adapted to the project needs. Hence, we have followed our original methodology based on the criteria previously enumerated, while developing the new ontology, and we have carried out the actions as described below.

3 PROPOSED APPROACH

As noted above, we decided to investigate the application of ontology using the publications repository. Capturing this information within a standard ontology language would make it universally accessible throughout the Web, and allow it to be analysed, queried and compared using powerful, open tools.

In the first stage, we interviewed the Head of Computing Library Services of the University of Huddersfield, UK, to understand the mechanism of ePrints publication repository in order to capture ePrints knowledge to build system interoperability services. We thus identified the actors interacting with the ePrints system as well as the procedures by which the actors interact with this system. The resulting UseCase diagram (Fig.1) uses the standard Unified Modeling Language (UML) (Ambler, 2005) and shows the functionality of the system as well as its dependencies at a high level viewpoint.

Next, we have modelled the business process using Business Process Modeling Language (BPML) (Dietz and Mallens, 2001) and generated the flowchart shown in Fig.2. The UseCase diagram as well as the Business Process Modeling Notation (BPMN) (Allweyer, 2005) have been designed with Modelio Free Edition v1.2 (Modelio, 2014), mainly because this software supports both UML and business modelling while being a user-friendly and free tool.

These steps have helped us not only in answering to the competency questions to determine the domain and the scope (“what we do with it”) of the ontology, but also to systematically define the relations between the ontology concepts.

In our approach, we have thus designed an on-

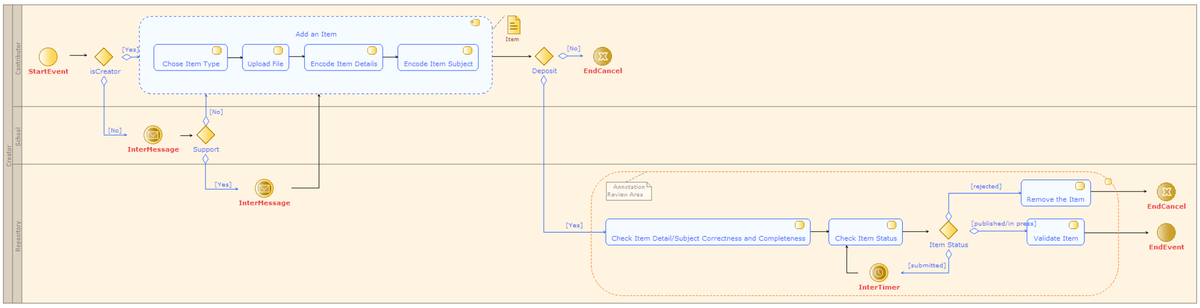

Figure 2: BPMN Diagram describing the action of adding an Item in ePrints.

Table 1: Comparison between UML, BPMN, and OWL.

Modeling Categories	UML UseCase	BPMN	OWL Ontology
Actor	Actor	Pool, Lane	Class
Behaviour	Use Case	Process/Sub-process, Task	Property, Sub-property
Decision		Gateway	(DL) Query
Event		Event (start, intermediate, end)	Open/Run/Close Ontology
Transition	Relationship	Sequence flow	Characteristic
Coaction		Message Event	Annotation
Object		Data Object	Individual

tology resulting from multi-dimensional information combining data and processes. For this purpose, we have proposed the translation of UML and BPMN into OWL, and we have introduced the mapping as shown in Table 1.

For a pilot implementation, we developed the ontology whose covered domain is ePrints (Univerity of Huddersfield ePrints, 2014) and its scope is to enable interoperability/sharing knowledge, efficient maintenance, and also question queries. Moreover, our ontology is application independent, that means that it is the same ontology, e.g. for maintenance as well as for the query purpose.

Then, in order to build the domain ontology, we have selected and analyzed the ePrints vocabulary. The key related words have been identified when logging into ePrints system and doing the task of adding an Item to the repository.

To capture the ontology, we have chosen Protégé v4.3 (Protégé, 2014), running on a Windows platform. This choice is motivated by the fact that this tool (Horridge, 2009) facilities the interoperability with other knowledge-representation systems and has a user-friendly, configurable interface.

The adopted language to express the ontology is the Web Ontology Language (OWL) (Suwanmanee et al., 2005), according to the World Wide Web Consortium (W3C) recommendation (Bechhofer et al., 2014). In particular, we have adopted OWL-DL specie because, on one hand, it is more expressive than OWL-Lite which is an OWL sub-language only adapted for simple situations. On the other hand, OWL-DL is based on Description Logics (DL) such as in (Grimm et al., 2004). Thus, it is possible to perform automated reasoning on OWL-DL-based ontology like in (Peim et al., 2002). Hence, OWL-DL enables the use of a reasoner to compute the inferred ontology class hierarchy and to perform the consistency check. Moreover, OWL-DL's reasoners are tractable, i.e. work in polynomial time, whereas in the case of OWL-Full, which is the union of OWL syntax and Resource Description Framework (RDF)s data representation, automated reasoning is not tractable.

Table 2: Definition equivalence between Protégé and OWL.

Protégé	OWL
Instances	Individuals
Slots	Properties
Classes	Classes

A Protégé ontology consists of classes, slots, facets, and axioms as mentioned in Table 2.

Classes are concepts in the domain of discourse and constitute a taxonomic hierarchy. Slots describe properties or attributes of classes and instances. Facets describe properties of slots. Axioms specify additional constraints. A Protégé knowledge base includes the ontology and individual instances of classes with specific values for slots.

OWL ontology has similar components to Protégé-based one. However, the terminology used to describe these components is slightly different (see

Table 2).

To develop our ontology, we have identified specific basic concepts used as cornerstones for the ontology design.

Next, we have mapped these concepts to a set of OWL main classes, which represent the roots of a set of corresponding subclasses together with their relationships with other classes. Each of these identified concepts represents also a starting point for the browsing of the ontology.

As expressed in Table 1, actors of UseCase diagrams (Fig. 1) could be modeled through pool and lanes in BPMN (De Nicola et al., 2007) as presented in Fig. 2, and could be mapped into class concepts of the ontology as shown in Fig. 3.

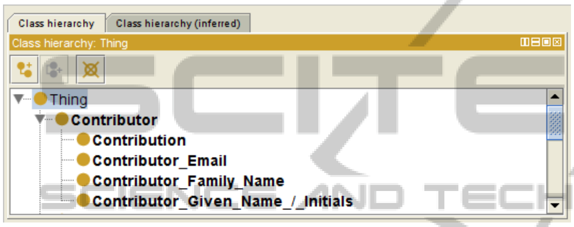

Figure 3: Example of Class (*Contributor*) implemented into our ontology.

An example of our ontology class is presented in Fig. 3. The displayed class is called *Contributor* and contains four subclasses, namely *Contribution, Contributor_Email, Contributor_Family_Name, Contributor_Given_Name/_Initials*).

OWL classes are interpreted as sets that contain individuals. Individuals represent objects in the domain that we are interested in. Instances can be referred to as being "instances of classes". In our example, the subclass *ItemID* has individuals as partially shown in Fig. 4.

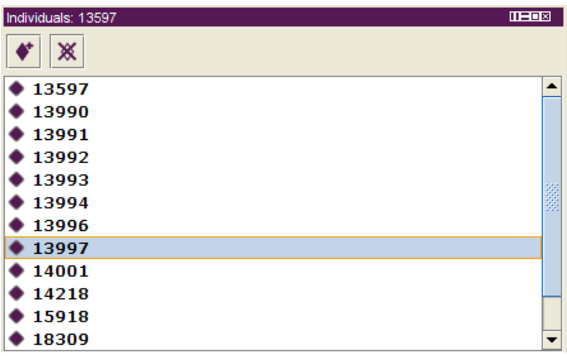

Figure 4: Example of Individuals of the subclass *ItemID* of the class *Item* implemented into our ontology.

Properties are binary relations on individuals, i.e. properties that link two individuals together. They can have inverses. Properties can be limited to having a single value, i.e. to being functional. They can also be either transitive or symmetric.

Properties are also used to create restrictions in OWL. The latter ones could be of three categories, namely, quantifier restrictions, cardinality restrictions, and *hasValue* restrictions.

These quantifier restrictions are composed of a quantifier, a property, and a filler. The two quantifiers that may be used are the existential quantifier, read as *at least* or *some* in OWL speak, and the universal quantifier, read as *only* in OWL speak.

For a set of individuals, an existential restriction ($\exists$) specifies the *existence of a* (i.e. at least one) relationship along a given property to an individual that is a member of a specific class. For example, $\exists$ *hasContributor Contributor* describes all of the individuals that have at least one (some) relationship along the *hasContributor* property to an individual that is member of the class *Contributor* as in Fig. 5.

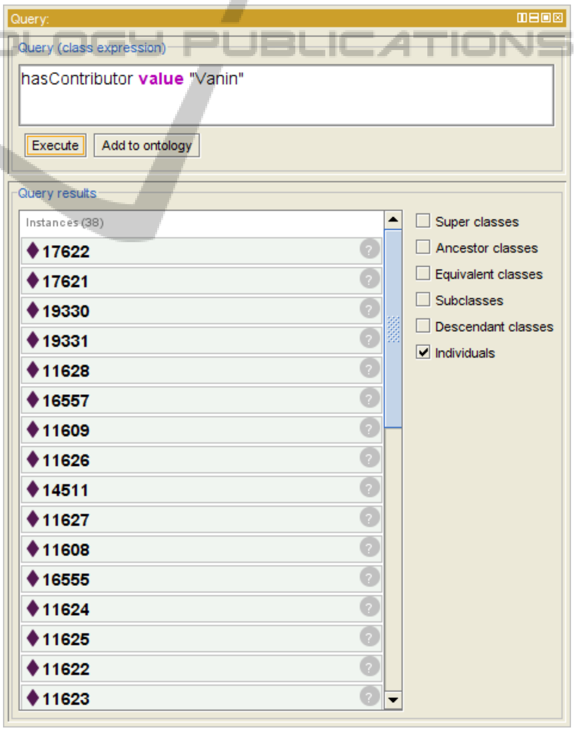

Figure 5: Example resulting from *Check Item Details* activity.

In fact, processes and sub-processes identified in BPMN diagrams could be mapped into concepts properties, as per our Table 1.

As an example, the activity group *Add an Item* in the BPMN diagram (Fig. 2) could be mapped into the data property *hasAddedItem* (Fig. 6), while the related BPMN tasks could be translated into OWL sub-

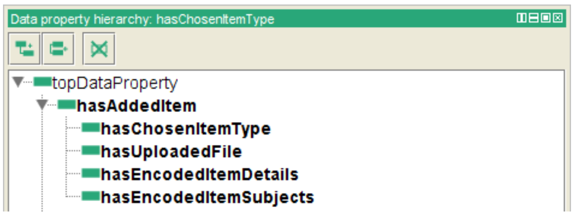

Figure 6: Example of relations resulting from *Add an Item* activity.

properties.

On the other hand, according to our Table 1, gateways present in BPMN diagrams result in ontology queries such as shown in Fig. 7.

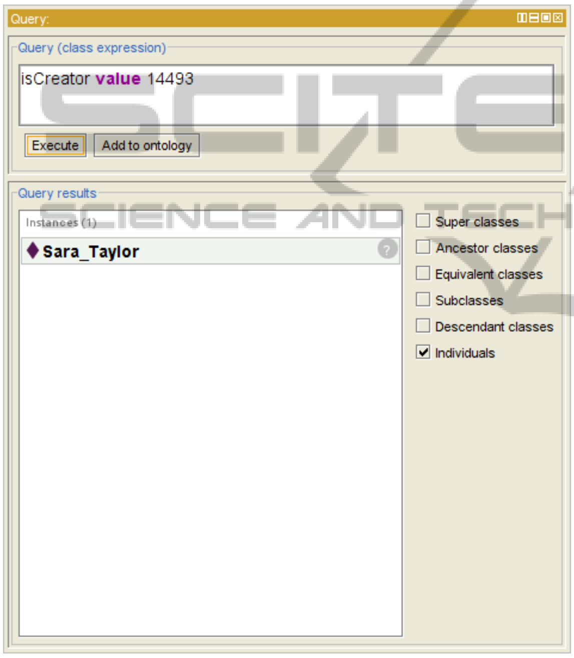

Figure 7: Example of DL query resulting from "is Creator" BPMN gateway.

For example, Fig. 7 illustrates 'isCreator' BPMN gateway (Fig. 2) transformed into a *isCreator* data property resulting in a DL query.

4 RESULTS AND DISCUSSION

The ontology for ePrints Information Management we called ePrOnto could be characterized with regards to the classification presented in the methodology section (Section 2).

Hence, our ontology, developed with Protégé OWL, could be considered as a semi-automatic single ontology with multiple layers (different levels of

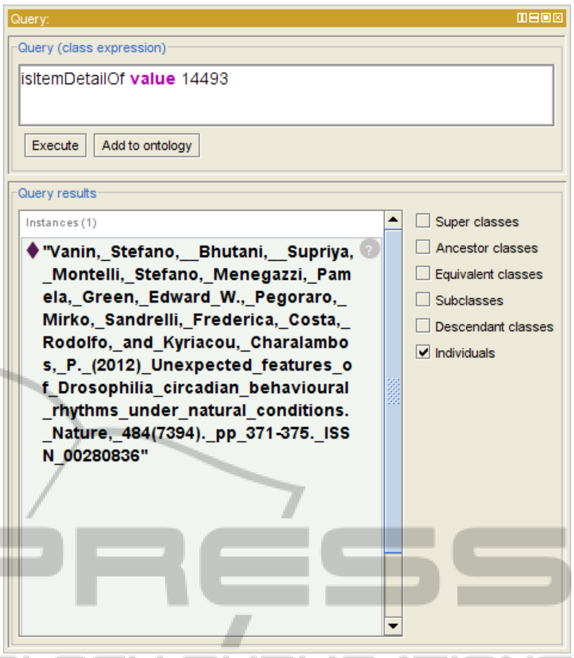

Figure 8: Example of DL query about data relation.

hierarchy). Some of the layers could be seen in left part of Fig. 9 as well as the automatically generated OWL code. An overview of the whole of our ontology structure is demonstrated in the right part of Fig. 9.

Moreover, our ontology was designed in a one-step collaborative way from scratch and is application independent. The adopted approach for the knowledge capture is a middle-out strategy (Olszewska et al., 2010).

As none of the ontology development methodologies described in the literature (Gruber, 1995; Dahlem et al., 2009) were directly suitable, our ontology was developed according to a unique scheme involving dynamic design notations (see Section 3).

Some of the Items (publications) from ePrints (Univerity of Huddersfield ePrints, 2014) have been encoded into our ePrOnto ontology to validate the interoperable services of our designed and implemented ontology. Hence, the proposed ontology is a first ontology developed for ePrints repository management, and which could be compatible with query process such as illustrated in Figs. 5,7,8.

Recent works such as (Wei et al., 2010) have proposed to automatically extract topics from text corpus. Following this direction, our future work will be the development of an innovative method to automatically update the developed ontology in order to provide a fully automatic, interoperable service.

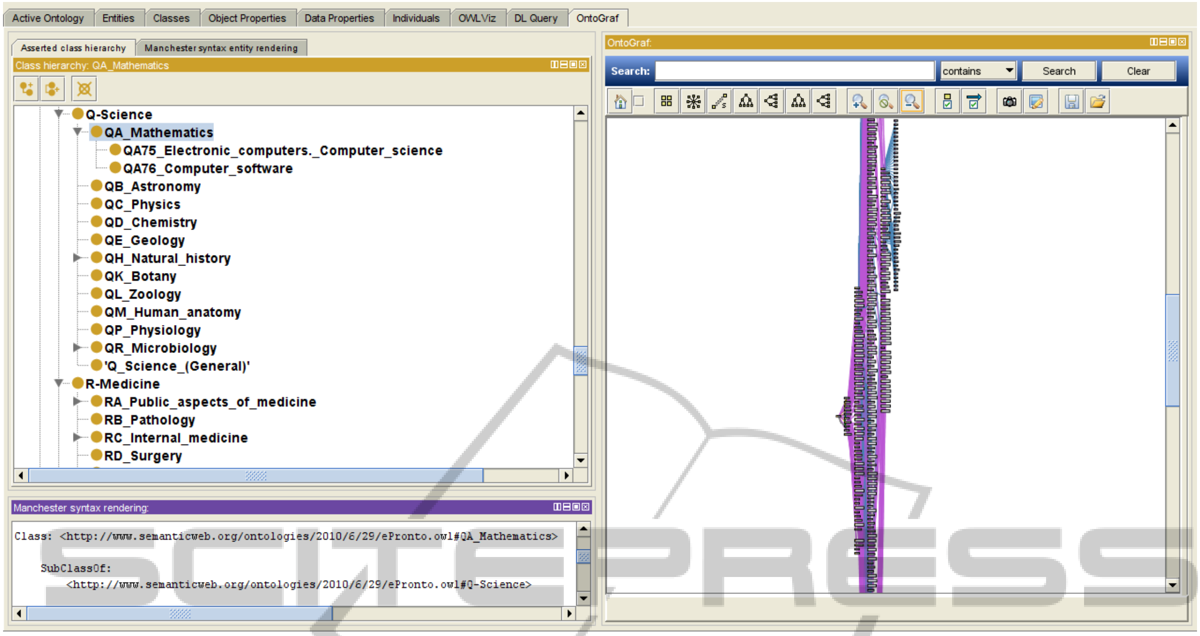

Figure 9: Illustration of our ePronto ontology.

5 CONCLUSIONS

This paper is focused on the dynamic design of ontologies which could be of a large scale. Dynamic design notations such as UML and BPMN have been translated into OWL in order to systematically model ontological concepts and their relations. The proposed approach has led to the modelling and the efficient management of the University publication repository (ePrints) system in an interoperable way. Hence, the related complex data and information have been transformed into structured knowledge through the use of the ontological approach and dynamic design notations.

ACKNOWLEDGEMENTS

This work has been partly supported by the JISC 11/09 grant *BRIM* - Research Information Management.

REFERENCES

Aljandal, W., Bahirwani, V., Caragea, D., and Hsu, H. (2009). Ontology-aware classification and association rule mining for interest and link prediction in social networks. In *Proceedings of the AAAI 2009 Spring Symposium on Social Semantic Web: Where Web 2.0 Meets Web 3.0*, pages 736–749.

Allweyer, T. (2005). *BPMN 2.0: Introduction to the standard for Buisness Process Modelling*. Herstellung und Verlag, Germany, 2nd edition.

Ambler, S. (2005). *The Elements of UML 2.0 Style*. Cambridge University Press, New York, USA.

Auer, S., Dietzold, S., and Riechert, T. (2006). OntoWiki - A tool for social, semantic collaboration. In *Proceedings of the International Semantic Web Conference (ISWC'06)*, pages 736–749.

Baclwaski, K., Kokar, M., Kogut, P., Hart, L., Smith, J., Holmes, W., Letkowski, J., and Aronson, M. (2001). Extending UML to support ontology engineering for the Semantic Web. In *Proceedings of the International Conference on UML (UML'01)*.

Bechhofer, S., van Harmelen, F., Hendler, J., Horrocks, I., McGuinness, D., Patel-Schneider, P., and Stein, L. A. (2014). OWL web ontology language reference. W3C recommendation. Available online at: http://www.w3.org/tr/OWL-ref/.

Bernaras, A., Laresgoiti, I., and Corera, J. (1996). Building and reusing ontologies for electrical network applications. In *Proceedings of the European Conference on Artificial Intelligence (ECCAI'96)*, pages 298–302.

Chaware, S. and Rao, S. (2010). Integrated approach to ontology development methodology with case study. *International Journal of Database Management Systems*, 2:13–19.

Corcho, O., Fernández-López, M., and Gómez-Pérez, A. (2003). Methodologies, tools and languages for building ontologies. Where is their meeting point? *Data and Knowledge Engineering*, 46(1):41–64.

Corsar, D., Moss, L., Sleeman, D., and Sim, M. (2009). *Supporting the Development of Medical Ontologies*, pages 114–125. In Frontiers in Artificial Intelligence and Applications: Formal Ontologies Meet Industry, IOS Press.

Dahlem, N., Guo, J., Hahn, A., and Reinelts, M. (2009). Towards an user-friendly ontology design methodology. In *Proceedings of the IEEE International Conference on Interoperability for Enterprise Software and Applications*, pages 180–186.

De Nicola, A., Lezoche, M., and Missikoff, M. (2007). An ontological approach to business process modeling. In *Proceedings of the Indian International Conference on Artificial Intelligence*, pages 1794–1813.

De Nicola, A., Missikoff, M., and Navigli, R. (2009). A software engineering approach to ontology building. *Information Systems*, 34(2):258–275.

Dietz, J. and Mallens, P. (2001). Business process modeling as a starting point for information systems design (part 1) an integrated, business-oriented perspective on facts and rules. *DataToKnowledge Newsletters*, pages 1–9.

ePrints (2014). Institutional Repositories. Available online at: http://www.eprints.org/.

Fensel, D. (2001). Ontologies and electronic commerce. *IEEE Intelligent Systems*, 16(1):8–14.

Fernández-López, M. (1999). Overview of methodologies for building ontologies. In *Proceedings of the IJCAI Workshop on Ontologies and Problem Solving Methods*, pages 4.1–4.13.

Fernández-López, M., Gómez-Pérez, A., and Juristo, N. (1997). METHONTOLOGY: From ontological art towards ontological engineering. In *Proceedings of the AAAI Spring Symposium Series*, pages 33–40.

Gessa, N., Busanelli, M., Sabbata, P. D., and Vitali, F. (2006). Extracting a semantic view from an ebuisness vocabulary. In *Proceedings of the IEEE International Conference on Enterprise Computing, E-Commerce, and E-Services*, pages 57–57.

Gómez-Pérez, A., Fernández-López, M., and Corcho, O. (2004). *Ontological Engineering*. Springer-Verlag, London.

Green, P. and Rosemann, M. (2005). *Buisness Systems Analysis with Ontologies*. Idea Group Publishingy, Hershey, PA, USA.

Grimm, S., Motik, B., and Preist, C. (2004). Variance in e-Business service discovery. In *Proceedings of the International Semantic Web Conference (ISWC'04) Workshop*.

Gruber, T. (1995). Towards principles for the design of ontologies used for knowledge sharing. *International Journal of Human-Computer Studies*, 43(5-6):907–928.

Gruninger, M. and Fox, M. (1995). Methodologies for the design and evaluation of ontologies. In *Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing*, pages 6.1–6.10.

Hermida, J., Romá-Ferri, M., Montoyo, A., and Palomar, M. (2009). Reusing UML class models to generate OWL ontologies. A Use Case in the pharmacotherapeutic domain. In *Proceedings of the International Conference on Knowledge Engineering and Ontology Development (KEOD'09)*.

Horridge, M. (2009). *A Practical Guide to Building OWL Ontologies Using Protégé 4 and CO-ODE Tools*. University of Manchester, Manchester, UK, 1.2 edition.

Jain, S. and Pareek, J. (2010). Automatic topic(s) identification from learning material: An ontological approach. In *Proceedings of the IEEE International Conference on Computer Engineering and Applications*, pages 358–362.

Jimenez-Ruiz, E., Cuenca-Grau, B., Zhou, Y., and Horrocks, I. (2012). Large-scale interactive ontology matching: Algorithms and implementation. In *Proceedings of the European Conference on Artificial Intelligence (ECCAI'12)*, pages 444–449.

Kogut, P., Cranefield, S., Hart, L., Dutra, M., Baclawski, K., Kokar, M., and Smith, J. (2002). UML for ontology development. *The Knowledge Engineering Review*, 17(1):61–64.

Lenat, D. and Guha, R. (1990). *Building Large Knowledge-Based Systems: Representation and Inference in the Cyc Project*. Addison-Wesley, Boston, USA.

Lunn, K. (2003). *Software Development with UML*. Palgrave MacMillan, New York, USA.

Marshall, C. (2000). *Enterprise Modeling with UML: Designing Successful Software Through Business Analysis*. Adison Wesley Longman, Inc., USA.

Modelio (2014). Open-source modeling tool supporting natively UML and BPMN. Software available online at: http://www.modeliosoft.com/.

Ohgren, A. and Sandkuhl, K. (2005). Towards a methodology for ontology development in small and medium-sized enterprises. In *Proceedings of the IADIS International Conference on Applied Computing*, pages 369–376.

Olszewska, J. (2011). Spatio-temporal visual ontology. In *Proceedings of the 1st EPSRC Workshop on Vision and Language (VL'2011)*.

Olszewska, J., Simpson, R., and McCluskey, T. (2010). ePrOnto: OWL-Based Ontology for Research Information Management. In *JISC Technical Report*.

Peim, M., Franconi, E., Paton, N., and Goble, C. (2002). Query processing with description logic ontologies over object-wrapped databases. In *Proceedings of the IEEE International Conference on Scientific and Statistical Database Management*, pages 27–36.

Protégé (2014). Open-source ontology editor. Software available online at: http://protege.stanford.edu/.

Seremeti, L. and Kameas, A. (2009). A task-based ontology engineering approach for novice ontology developers. In *Proceedings of the IEEE Balkan Conference on Informatics*, pages 85–89.

Suwanmanee, S., Benslimane, D., and Thiran, P. (2005). OWL-based approach for semantic interoperability. In *Proceedings of the IEEE International Conference on Advanced Information Networking and Applications*, pages 145–150.

Swartout, B., Ramesh, P., Knight, K., and Russ, T. (1997). Towards distributed use of large-scale ontologies. In *Proceedings of the AAAI Symposium on Ontological on Ontological Engineering*, pages 33–40.

Univerity of Huddersfield ePrints (2014). University of Huddersfield Repository. Available online at: http://eprints.hud.ac.uk/.

Uschold, M. and Gruninger, M. (1996). Ontologies: Principles, methods and applications. *Knowledge Engineering Review*, 11:1–44.

Uschold, M. and King, M. (1995). Towards a methodology for building ontologies. In *Proceedings of the IJCAI Workshop on Basic Ontological Issues in Knowledge Sharing*, pages 2–15.

Wei, W., Barnaghi, P., and Bargiela, A. (2010). Probabilistic topic models for learning terminological ontologies. *IEEE Transactions on Knowledge and Data Engineering*, 22(7):1028–1040.