The role of Ontology Design Patterns in Linked Data projects

Valentina Presutti[1], Giorgia Lodi[1], Andrea Nuzzolese[1], Aldo Gangemi[3,1], Silvio Peroni[2], and Luigi Asprino[2,1]

[1] Institute of Cognitive Sciences and Technologies, National Research Council of Italy
{name.lastname}@istc.cnr.it
[2] DISI, University of Bologna
{name.lastname}@unibo.it
[3] LIPN, Université Paris 13 Sorbonne Cité, CNRS

Abstract. The contribution of this paper is twofold: (i) a UML stereotype for component diagrams that allows for representing ontologies as a set of interconnected Ontology Design Patterns, aimed at supporting the communication between domain experts and ontology engineers; (ii) an analysis of possible approaches to ontology reuse and the definition of four methods according to their impact on the sustainability and stability of the resulting ontologies and knowledge bases. To conceptually prove the effectiveness of our proposals, we present two real LOD projects.

Keywords: ontology, ontology design patterns, linked data, ontology reuse, eXtreme Design

1 Introduction

Linked Data (LD) is rapidly increasing, especially in the public sector where opening data is becoming a consolidated institutional activity. However, the importance of providing LD with a high quality ontology modelling is still far from being fully perceived. The result is that LD are mostly modelled by direct reuse of individual classes and properties defined in external ontologies, overlooking the possible risks caused by such a practice. We claim that this practice may compromise the level of semantic interoperability that can be achieved. Therefore, the need of clear practices for motivated guidelines for ontology reuse arise. Ontology Design Patterns (ODPs) proved to be an effective means for improving the quality of ontologies. Another neglected aspect in ontology projects is the need of proper tools for sharing ontology details with domain experts, without requiring training sessions in knowledge representation.

Starting from the eXtreme Design (XD) methodology, the contribution of this paper is twofold. Firstly, we introduce a new task in XD concerning the communication with domain experts. We define a UML stereotype that allows representing ontologies as a web of ODPs modelled as UML components. This notation hides the complexity of OWL representations while still conveying the main semantics to domain experts. Secondly, we provide motivated guidelines for

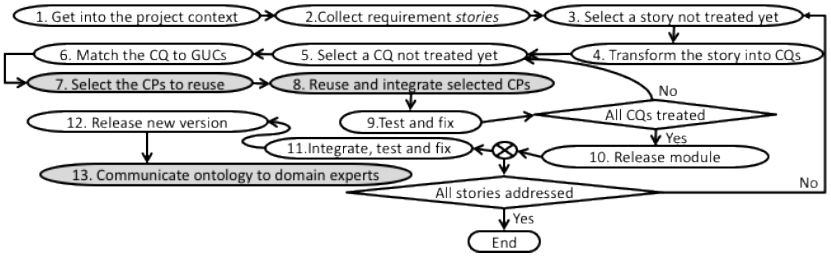

Fig. 1. Extended XD workflow

reusing external ontologies and ODPs in ontology design projects. To prove the effectiveness of our contributions we discuss two real examples of LOD projects.

Background. Originally ontologies were seen mainly as *portable* components [6], while nowadays one of the most challenging areas of ontology design is *reusability* [5]. Although ontology reuse is a recommended practice in most ontology design methodologies [11], a standardisation of ontology reuse practices is still missing. Most literature on ontology reuse is focused on the challenging issue of ontology selection, while our perspective is on how to implement reuse once the selection finalised. We contribute to this issue with an analysis of possible reuse approaches, emphasising the role of Ontology Design Patterns (ODPs) [4, 1] in this process. Design patterns and ontology reuse have been investigated in ontology engineering since early stages and they became hot topics in the context of the Semantic Web. ODPs enabled *pattern-based methodologies* in ontology engineering. These methodologies formalise approaches and provide facilities for re-using ODPs, however they do not provide motivated alternative guidelines on how to implement such a task.

The eXtreme Design (XD) [9, 2] is an agile design methodology providing guidelines for performing ontology design through an incremental and iterative process based on the reuse of ontology design patterns (ODP) [8]. It is inspired by the eXtreme Programming (XP) and it recommends pair and test driven development, refactoring, and a divide-and-conquer approach to problem-solving [10].

Paper structure. Section 2 introduces the main contributions; namely, the new UML stereotype for ontology and ODPs representation and different approaches to ontology reuse while Section 3 provides real examples where the main contribution of the paper are applied. Section 4 concludes the paper.

2 Extending eXtreme Design

Figure 1 shows the *eXtreme Design* (XD) methodology highlighting in grey the contributions of this paper. The first contribution (involving tasks 7 and 8) regards the need of a *model that describes possible approaches to ODP reuse*, allowing ontology engineers to choose the most appropriate model for their project according to its specific characteristics. The second contribution (task 13 in Figure 1) regards the need of describing ontologies to domain experts. It is important to provide them with enough insights about the ontology structure, its main

concepts and usage, without exposing them to the burden of learning logics and knowledge representation languages.

2.1 Approaches to semantic web ontology reuse

Ontology reuse models can be classified based on (i) the type of reused ontology (e.g. foundational, top-level, ontology design patterns, domain ontologies), (ii) the type of reused ontology fragment (e.g. individual entities, modules, ontology design patterns, arbitrary fragments), (iii) the amount of reused axioms (e.g. import of all axioms, of only axioms in a given neighbourhood of an entity, of no axioms), (iv) and the alignment policy (e.g. direct reuse of entities, reuse via equivalent relations such as `rdfs:subClassOf` and `owl:equivalentClass`). The only characteristic that all these models share is to reuse entities with the same logical type as they were defined (e.g. an entity defined as `owl:Class` in an ontology is commonly reused as such).

A certain choice of reuse practice impacts significantly on the semantics of an ontology, its sustainability, and its interoperability.

2.2 Guidelines for Ontology reuse

In this paper we provide guidelines for ontology reuse in the context of ontology projects that exhibit these characteristics: (i) there is no ontology that addresses all or most of the requirements of the local ontology project; (ii) the ontology under development is meant to be used as a reference ontology for a certain domain, and (iii) there is the willingness to comply with existing standards. We identify the following possible approaches to ontology reuse.

Direct reuse of individual entities. This approach consists on directly introducing individual entities of external ontologies in local axioms. This practice is very common in the LD community, however it is a routine, not a good practice, at all. It is essentially driven by the intuition of the semantics of concepts based on their names, instead of their axioms. In this case, the risk that the formal semantics of the reused entities is incompatible with the intended semantics to be represented is rather high. Moreover, with this practice a strong dependency of the local ontology with all the reused ontologies is created. This dependency may put at risk the sustainability and stability of the local ontology and its associated knowledge bases: if a change in the external ontology introduces incoherences in the local one, they must be dealt with a redesign process and consequential change in the ontology signature.

Indirect reuse of ontology modules and alignments. With this approach, the modelling of some concepts and relations, which are relevant for the domain but applicable to more general scopes, is delegated to external ontologies by means of ontology module reuse. An ontology module is a fragment that may be identified as providing a solution to one or more specific requirements of the local ontology. For example, let us consider an external ontology modelling the participation of an individual (e.g. through a property `ex:isInvolvedIn`) to an event (e.g. a class `ex:Event`). If the local ontology needs to specify a particular involvement in an event (e.g. `lo:hosted`) it should specialize (it indirectly

Table 1. Pros and cons of different approaches to ontology reuse.

Reuse method	**Fragment**	**Pros**	**Cons**
Direct reuse	individual entity	linked data practise	Semantic ambiguity, difficulty in verifying the consistency among the diverse reused concepts, dependency on external ontologies, instability and unsustainability
Direct reuse	ontology module	Stability and sustainability of domain relations and concepts, modularity, interoperability	Possible heterogeneity in module usage, dependency on external modules, instability and unsustainability limited to external modules
Direct reuse	ODP	Stability and sustainability of domain relations and concepts, modularity, interoperability, easier redesign in case of external changes	Dependency on external modules, mitigated risk of instability and unsustainability limited to external ODPs
Indirect reuse	ODP	Stability and sustainability of domain relations and concepts, modularity, interoperability, dependency on external modules limited to alignment axioms	Slightly increased design effort for moulding ODPs.

reuses) the relation of the external one (i.e. `ex:isInvolvedIn`). The fragment of the external ontology identified as relevant for the local ontology may be communicated in some usage documentation provided with the ontology. Nevertheless, it is difficult to provide third parties with a formal indication of the fragment that was meant to be relevant. This may lead to high heterogeneity in the usage of external fragments in data modelled through the local ontology. As for ontology sustainability, when a change in the external ontology provokes possible incoherences, the redesign process would be easier dealt with as compared to the previous approach.

Direct reuse of ontology design patterns and alignments. If the fragment is clearly and formally identified, since it is embedded in a dedicated ontology, some of the previous remarked issues can be mitigated. Let us consider that the earlier example class `ex:Event` is defined in an external ontology that implements a specific ODP. In this case, a scenario in which a redesign process must be undertaken may be less frequent. In fact, ODPs are developed for reuse purposes and thus they are unlikely to change. In the light of these observations, it is recommended to reuse ODPs in contrast to individual entities.

Indirect reuse of ontology design patterns and alignments. ODPs are used as templates. This approach is an extension of the previous one. At the same time, the ontology guarantees interoperability by keeping the appropriate alignments with the external ODPs, and provides extensions that satisfy more specific requirements. The alignment axioms may be published separately from the core of the ontology. With this type of reuse, the potential impact of possible changes in the external ODP is minimised. In fact, should incoherences show after a change in the external ODP (which is rather unlikely to happen) the redesign process would be very simple. The ontology signature and axioms would remain unchanged, as incoherences would be resolved by simply removing or revising the alignment axioms.

Table 1 summarises the advantages and disadvantages of the discussed four approaches. In general, among all of them, the recommended one is the fourth

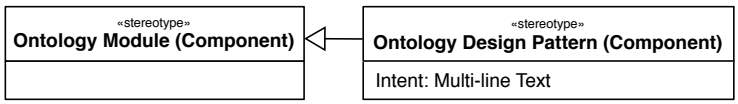

Fig. 2. Ontology Module and ODP stereotypes for UML Component Diagrams

approach: in the situation of incoherence raised by a change in an external reused ontology, it guarantees the easiest maintenance.

2.3 UML profile for representing ODP-based ontologies

Ontology development processes are all supported by languages, notations and tools for producing inputs and outputs of the various phases. XD provides detailed guidelines on how to perform some of its process tasks. For example, user stories are collected by means of *story cards* with a template, OWL is used as ontology modelling language and its related UML profile defined in the Ontology Definition Metamodel (ODM) [7] is used as a graphical notation for representing the ontology in its documentation. ODM provides stereotypes for both class diagrams and packages. The package profile is the only one addressing the representation of whole ontologies or ontology modules and their inter-relations. The relation between packages (i.e., among ontology modules) can only be an *import* relation referring to `owl:import` for its semantics.

This package- and OWL-based notation may be inadequate in some context, in particular when the target user is a domain expert without an expertise in knowledge representation. Our proposal extends the ODM OWL profile by introducing a stereotype for component diagrams that enables the representation of ODPs as components that implement and/or reuse certain interfaces.

The concept of ontology interface has been investigated in the literature related to ontology modularisation and knowledge encapsulation [3]. An ontology module defines its content (e.g. classes and relations) by means of *interfaces* that constitute the access point to its model.

Our main focus is to provide a notation that can be used for sketching the design of an ODP-based ontology and for communicating (sharing) the ontology model and discussing it with domain experts by hiding implementation details. The proposed ODM profile extension is depicted in Figure 2. We define two stereotypes that can be used with UML components diagrams: *Ontology Module* (OM) and *Ontology Design Pattern* (ODP). The latter inherits from the former and has a tagged value *intent*, i.e., a multi-line text that describes the modelling problem addressed by the ODP.

Each component defines two types of interfaces (compliant with standard UML notation): (i) the interfaces that the ODP *implements* (i.e., realises) denoted by *lollipops*; (ii) the interfaces that a ODP *uses* denoted by *sockets*. An ontology engineer may exploit this profile to sketch an abstract view of the adopted design choices when she has not decided yet what specific implementation of an ODP will be reused. The abstract view can be shared with domain experts in a phase between the collection of requirements (i.e., user stories) and the implementation, in order to share design choices and tune them before the

actual reuse happens, if needed. Referring to Figure 1, this would be between Task 7 and Task 8.

3 Applying the XD extensions in Linked Data projects

We applied the described contributions in two real Linked Open Data projects of the e-government sector. The first project was developed in the context of cultural heritage, in collaboration with the Italian Ministry of Cultural Heritage and Activities and Tourism; the second was carried out within the agriculture domain, in collaboration with the Italian Ministry of Agriculture.

3.1 Cultural-ON: Cultural ONtologies

Cultural-ON[4] is a suite of ontology modules for modelling knowledge in the cultural heritage domain. In Cultural-ON we applied the pattern-based ontology engineering approach, extensively reusing ODPs. The class *Cultural Institute or Site* (shortly, *CIS*) is used to model the different types of cultural heritage institutes or sites (e.g., museums, libraries, monumental areas). A number of organizations or juridicial entities (i.e., agents), playing specific roles on CISs, are represented in the ontology by `cis:Agent`. *CISs* are located in specific physical places that are precisely identified by geographical coordinates and/or addresses. CISs host collections (`cis:Collection` using the ODP *Collection*) and/or cultural heritage objects (`cis:CulturalHeritageObject`). Finally, cultural events (i.e., the class `cis:Event` that reuses the ODP *TimeIndexedSituation*) can be hosted in a *CIS*.

ODPs for domain experts communication. In the project we wanted domain experts to be focussed on the requirements. However, it was important to let them understand the main concepts of the ontology in order to facilitate reuse, and favour technological transfer to them who are ultimately responsible for ontology maintenance. In doing so, we faced the same issues as those earlier discussed: several times we were more focussed on explaining details of logics and ontology design best practices, and on convincing them not to concentrate on mere terms, as ontologys classes and properties were mostly seen. In the light of this, we elaborated the UML notation of section 2.3.

Figure 3 illustrates the UML component diagram that describes Cultural-ON as a set of interconnected ODPs. For example, the component CulturalInstituteOrSite depicts the class CIS and some of its main characterisations such as the composition (i.e., the partOf relation). The component RoleInTime, an application of the TimeIndexed ODP, exposes three main concepts; namely, Role, Time and Agent. It is then linked to a CIS by means of the concept roleAt. Note that this notation allows us to hide the OWL-specific modelling of an n-ary relation, requiring reification, while still conveying its semantics to domain experts.

External ontologies reuse. We adopted the earlier fourth model and we identified most relevant ODPs of external ontologies that were selected during

[4] `http://stlab.istc.cnr.it/documents/mibact/cultural-ON_xml.owl`

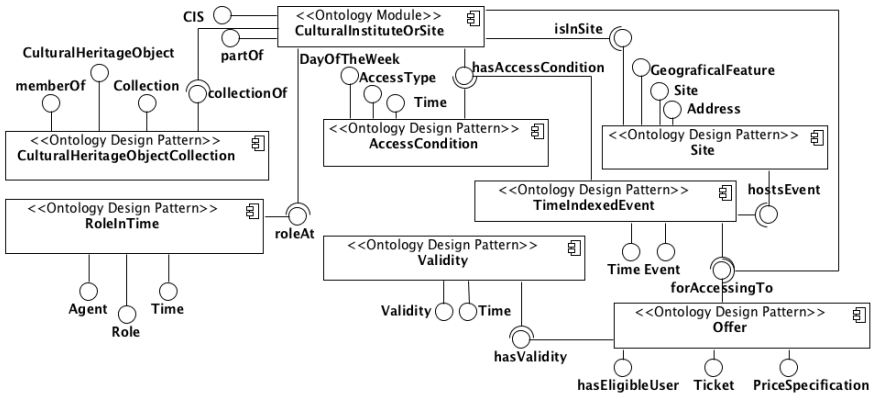

Fig. 3. Cultural-ON: UML Component Diagram for ODPs representation

an alignment process. We reproduced those ODPs in Cultural-ON so that to use ODPs as templates.

3.2 FOOD: FOod in linked Open Data

FOOD[5] aims at publishing LOD data of EU quality schemes, known as PDO and PGI. Each PDO and PGI agriculture product is described, in its characteristics, by a policy document.

The data contained in these documents were modelled as OWL ontologies, reusing ODPs. Specifically, we produced an upper ontology that represents general elements that contribute to form the content of the documents. The upper ontology represents the product name to be protected, its different types, the geographical area where it is produced and its characteristics[6] (including raw materials, and principal physical, chemical, microbiological or organoleptic characteristics). Specific ontologies per single product category were also produced in order to specialize the elements modelled in the upper ontology.

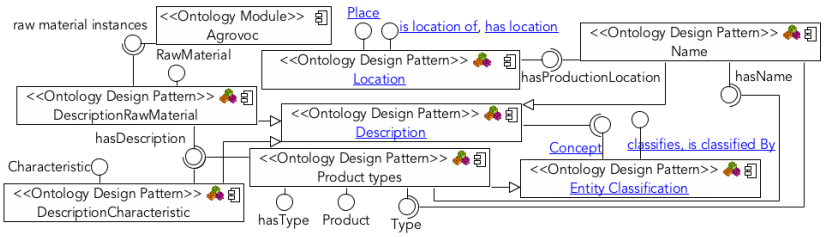

Fig. 4. FOOD: UML Component Diagram for the upper ontology

ODPs reuse in domain experts communication. Figure 4 shows the resulting UML component diagram of the upper ontology. Underlined components

[5] `http://w3id.org/food/`

[6] Reusing `http://ontologydesignpatterns.org/wiki/Submissions:Description`

represent more general ODPs. The Description ODP applied to raw materials of a product, the component DescriptionRawMaterial exposes the concept RawMaterial and is connected through hasDescription with the component Product Type hiding the OWL representation details.

External ontologies reuse. In FOOD we applied both indirect and direct reuse (Agrovoc has been directly reused for representing raw materials). The direct use of domain dependent controlled vocabularies (coverage-oriented ontologies) such as Agrovoc can be recommended in order to maintain the produced ontologies fully aligned with possible evolutions of those vocabularies, which can be viewed as domain reference standards usually developed by reference bodies in stable processes.

4 Conclusions

In this paper we discussed the role of ODPs in the design of ontologies within Linked Data projects. In particular, we extended the eXtreme Design methodology in order to address two issues we faced in practice: communication with domain experts and approaches to ontology reuse. Two real e-government Linked Data projects are described in order to prove the applicability of the introduced XD extensions. Future works focus on performing user-based surveys for a larger scale evaluation of our proposals by both ontology and domain experts.

References

[1] Eva Blomqvist and Kurt Sandkuhl. "Patterns in Ontology Engineering: Classification of Ontology Patterns". In: *Proc. of ICEIS*. (Miami, Florida, USA). CEUR-WS, 2005, pp. 413–416.

[2] Eva Blomqvist et al. "Experimenting with eXtreme Design". In: *Proc. of EKAW 2010*. (Lisbon, Portugal). Springer, 2010, pp. 120–134.

[3] F. Ensan and W. Du. "A knowledge encapsulation approach to ontology modularization". In: *Knowledge and Information Systems* 26.2 (2010), pp. 249–283.

[4] Aldo Gangemi. "Ontology Design Patterns for Semantic Web Content". In: *Proc. of ISWC*. (Galway, Ireland). Springer, 2005, pp. 262–276.

[5] Aldo Gangemi and Valentina Presutti. "Ontology Design Patterns". In: *Handbook on Ontologies, 2nd Ed.* Berlin, Germany: Springer, 2009, pp. 221–243.

[6] Thomas R. Gruber. "A translation approach to portable ontology specifications". In: *Knowledge acquisition* 5.2 (1993), pp. 199–220.

[7] *ODM*. Version 1.1. OMG, Sept. 2014.

[8] Valentina Presutti and Aldo Gangemi. "Content Ontology Design Patterns as practical building blocks for web ontologies". In: Springer, 2088, pp. 128–141.

[9] Valentina Presutti et al. "eXtreme Design with Content Ontology Design Patterns". In: *Proc. of WOP*. (Washington, DC, USA). CEUR-WS.org, 2009.

[10] Valentina Presutti et al. "Pattern-Based Ontology Design". In: *Ontology Engineering in a Networked World*. Berlin, Germany: Springer, 2012, pp. 35–64.

[11] Elena Paslaru Bontas Simperl, Malgorzata Mochol, and Tobias Bürger. "Achieving Maturity: the State of Practice in Ontology Engineering in 2009". In: *IJCSA* 7.1 (2010), pp. 45–65.