Ontology Analysis Based on Ontology Design Patterns

María Poveda, Mari Carmen Suárez-Figueroa, Asunción Gómez-Pérez
Ontology Engineering Group. Departamento de Inteligencia Artificial.
Facultad de Informática, Universidad Politécnica de Madrid.
Campus de Montegancedo s/n.
28660 Boadilla del Monte. Madrid. Spain
mpoveda@delicias.fi.upm.es, {mcsuarez, asun}@fi.upm.es

Abstract. The so-called Ontology Design Patterns (ODPs), which have been defined as solutions to ontological design problems, are of great help to developers when modelling ontologies since these patterns provide a development guide and improve the quality of the resulting ontologies. However, it has been demonstrated that, in many cases, developers encounter difficulties when they have to reuse ontology design patterns and include errors in the modelling. Thus, to avoid errors in ontology modelling, this paper proposes classifying errors into two types: (1) errors related to existing ODPs, called *anti-patterns*, and (2) errors not related to existing ODPs, called *worst practices.* This classification is the result of analysing a set of ontologies which come from an academic experiment. In addition, the paper presents a general classification of the worst practices found and a set of worst practice examples. Finally, the paper shows an example of how the aforementioned worst practices could be related among them.

Keywords: patterns, anti-patterns, worst practices, ontology.

1 Introduction

Modelling ontologies has become one of the main topics of research within ontological engineering because of the difficulties it involves. In recent years, the emergence of Ontology Design Patterns (ODPs), which are defined as solutions to design problems [3], has supposed a great help to ontology developers.

Some experiments [2] carried out in ontology engineering have demonstrated that design patterns are perceived as an aid to modelling ontologies, a development guide, and a way to improve the quality of the resulting ontologies. However, it is well known that [2, 1], in some cases, ontology developers experience difficulties when reusing the patterns during modelling, and include errors in the modelling. Therefore, in order to understand and use correctly ODPs, we need a better support that prevents the emergence of modelling errors.

Thus, to avoid the appearance of errors in ontology development, we are working on the creation of a new set of methodological guides. These guides, based on the identification and classification of modelling errors, classify errors into two types: (1) errors related to ODPs, called *anti-patterns*; and (2) errors not related to ODPs, called *worst practices*. This classification of errors arises from performing an analysis, based

on ODPs, of a set of ontologies. In this paper, such a classification is presented. In addition, we include a classification of the worst practices identified and a set of examples of such worst practices.

The remainder of the article is structured as follows: Section 2 presents the state of the art of patterns and anti-patterns in ontological engineering. Section 3 describes the experiment carried out in order to obtain the set of ontologies analyzed in this paper. Section 4 describes the analysis carried out in the aforementioned ontologies, which shows the presence of ODPs and describes anti-patterns and worst practices. Section 5 includes the classification of the worst practices found, a set of examples of worst practices, and an example of how the worst practices could be related among them. Finally, Section 6 includes the conclusions drawn and future lines of work.

2 State of the Art: Patterns and Anti-Patterns

In ontology engineering, the ontology design patterns can be considered as modelling solutions to problems widely known in the area. These solutions are based on good practices and solve modelling problems.

In the ODPs field, we can distinguish between logical patterns and conceptual patterns. With regard to logical patterns, the W3C work team, known as "*Semantic Web Best Practices and Deployment* (SWBPD)[1], has established that in order to provide support to developers and users of the Semantic Web, a set of good practices is required. To that purpose, this group proposes patterns that solve design problems in the OWL language, independently of the particular specification, which solve logical problems. Regarding conceptual patterns [3], the author proposes patterns (in OWL or any other logical language) that solve design problems for specific domains, which solve content problems.

In addition to the distinction mentioned above, in [5, 4] the authors classify the ODPs into content, structural, lexico-syntactic, reasoning, presentation and correspondence, and their subtypes.

The work described in [5, 4] is focused on content patterns and provides guidelines on how to apply content ODPs using import, specialization, composition and expansion functions. In the same work, the content ontology anti-pattern concept is defined as a design that is different from a content pattern in that the former codes the solution to a problem in a wrong way.

However, we have observed that none of the papers analyzed have carried out a thorough study on the use of any type of ODPs and their corresponding anti-patterns. We have also observed that there is no previous work focused on identifying and preventing to model errors not related to any existing ODP.

ODPs can be found in on-line libraries that include both the description and the OWL code associated to the patterns as, for example, "the *Ontology Design Pattern* Portal"[2], or they can be obtained from the work team "*Semantic Web Best Practices and Deployment".* Some other libraries [5, 6] do not provide the pattern code, but

[1] http://www.w3.org/2001/sw/BestPractices/

[2] http://ontologydesignpatterns.org/

they store descriptions of a great number of ODPs. These libraries, which follow a software engineering approach, use a template for describing the patterns.

3 Experiment Description

With the aim of identifying the types of errors normally made when developing ontologies, we have analyzed 11 ontologies that tackle different aspects (art, architecture, geography, tastes and likings, and community services) of the domain of Saint James's Way. These ontologies were developed by Master students as a practical assignment on the "Ontologies and the Semantic Web" subject at the *Universidad Politécnica de Madrid*, during 2007-2008.

Before performing the practical assignment, theoretical issues related to ontology development were explained to the students. Such theoretical lessons were combined with hands-on activities that allowed students to get the basis for building ontologies.

To develop the ontologies in the practical assignment, students worked in pairs bearing in mind the particular aspect within the Saint James's Way domain that they choose. Student groups carried out the following steps to develop their ontologies:

1. To find knowledge resources suitable for reuse in the ontology development.
2. To identify the life cycle model and the life cycle for the ontology to be developed, and to schedule the ontology development project.
3. To identify the requirements that the ontology to be developed should fulfil.
4. To model the knowledge to be represented in the ontology, integrating (if needed) the knowledge from the resources found in step 1.
5. To implement the ontology in OWL.

Students were free to choose as the ontology language among Spanish, English, or both, although most of them developed the ontologies in Spanish. Finally, we obtained 2 ontologies about art, both in Spanish; 2 ontologies about architecture, 1 in Spanish and 1 in English; 2 ontologies about geography, 1 in Spanish and 1 in English; 1 ontology about tastes and likings in Spanish; and 4 ontologies about community services, 3 in Spanish and 1 in English.

It is worth mentioning that ODPs issues were not explained to students during the master course. Thus, students did not have any notion about ODPs to be applied during the ontology development; therefore, it did not make sense to focus our study on the correctly or incorrectly reuse of ODPs. Instead, we focused our study on examining whether ODPs could be detected in the resulting ontologies. The approach adopted in the analysis was based on the manual search of ODPs in ontologies and on the manual identification of two types of modelling errors: (1) those related to ODPs, called *anti-patterns;* and (2) those not related to ODPs, called *worst practices*.

4 Analysis based on Ontology Design Patterns

As mentioned in Section 3, we focused our study on analysing the 11 ontologies in the Saint James' Way with the aim of searching whether ODPs could be detected in the resulting ontologies. The approach adopted in the analysis was based on the

manual search of three possible scenarios in the 11 ontologies: (1) ODPs appearing in ontologies; (2) modelling errors related to ODPs; and (3) modelling errors not related to ODPs.

As a result of the analysis carried out, we have identified the possibilities shown in the tree of Fig. 1. In the first place, we can distinguish two main branches in the tree: the left one that represents whether a design equivalent to an ODP has been observed; and the right one that represents the case when no design equivalent to an ODP has been observed. Second, and finally, the tree classifies each of the scenarios found in some of the four leaf nodes and determines if such scenario matches a pattern identification, an anti-pattern identification, or a worst practice identification in the following way:

1. *Identification of an ontology design pattern.* (1st leaf from the left) We have identified some cases that have a correct design for the modelling problem; such a design matches an ODP of those in the available libraries.
2. *Identification of an anti-pattern*. (2nd and 3rd leaf) In some cases we have identified a design that matches an ODP but this design is not a suitable solution to the modelling problem. In some other cases we have not identified a design that could match an ODP in a suitable solution to the modelling problem; however, there is a suitable pattern that could have been applied if we had known the ODPs.
3. *Identification of a worst practice (WP)*. (4th leaf) In some other cases we have observed that there is an unsuitable solution to the modelling problem whereas there is not an ODP suitable for the modeling problem. Most of these cases have been treated as WPs; however, a few set of cases could have been solved by means of adapting or combining existing ODPs.

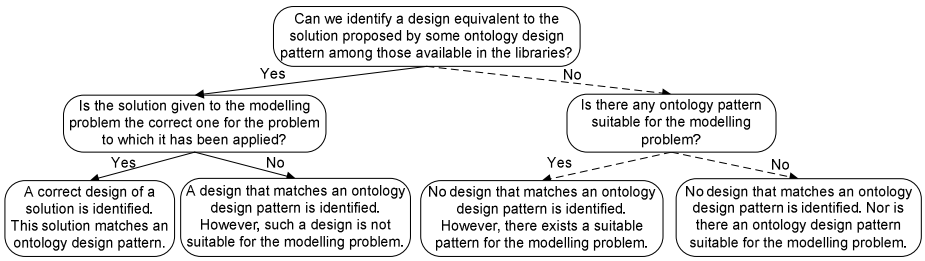

Fig. 1. Decision tree for classifying patterns, anti-patterns and worst practices.

During the analysis performed with the aforementioned 11 ontologies, we have found 208 cases in which we have identified a correct design of a solution that matches an ODP. We have also found 117 cases of anti-patterns, of which 83 correspond to situations in which the solution matches the modelling proposed in some ODP. However, such a solution is not suitable for the design problem intended to solve. Besides, we have found 34 cases in which we were unable to identify a design solution matching an ODP though a suitable ODP exists. Finally, we have come across a set of 231 WPs, that is, solutions unsuitable for the design problem in the domain of the ontologies studied for such design problem; such WPs do not have an ODP associated. These WPs are described in Section 5.

5 Classification and Examples of Worst Practices

As can be seen in Section 4, we have identified designs unsuitable for solving a modelling problem for which there is no an available ODP. In this paper, this type of design solutions is named worst practices (WPs).

We have classified the WPs identified in the analyzed ontologies bearing in mind the types of ODPs proposed in [5, 4]. In Fig. 2, each WP is associated to the type(s) of ODP to which a pattern created to avoid such a WP could belong. In Table 1 we provide a brief description of each WP classified.

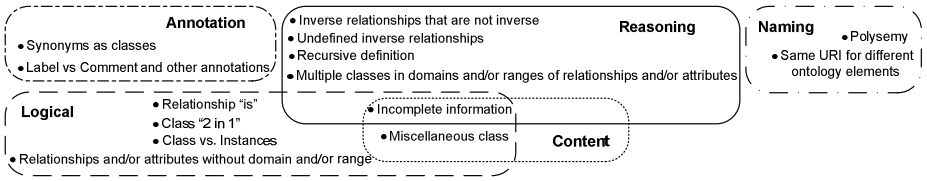

Fig. 2. Classification of the worst practices identified.

As can be observed in the WPs classification appearing in Fig. 2, the types of WPs identified are the following: 1) **Annotation WP** that refers to the ontology usability from the user's point of view by including additional information in the form of annotations in the ontology; 2) **Reasoning WP** that refers to the implicit knowledge derived from the ontology when reasoning procedures are applied to such an ontology; 3) **Naming WP** that refers to the ontology usability from the user's point of view and, specifically, to the naming of the ontology elements; 4) **Logical WP** that refers to the solution to design problems in which the primitives of the representation language used do not provide support; and 5) **Content WP** that refers to the solution to design problems related to the ontology domain.

Table 1. WPs descriptions.

WP Name	WP Description
Synonyms as classes	This WP consists both in creating several classes whose identifiers are synonyms and in defining them as equivalent.
Label vs. Comment and other annotations	This WP consists in interchanging the contents of the annotations of the types "label" and "comment" and in not including any annotation of the type "label" and "comment".
Inverse relationships that are not inverse	This WP consists in defining two relationships as inverse when, in fact, they are not.
Undefined inverse relationships	This WP consists in having inverse relationships in the ontology, but they are not defined as such.
Recursive definition	This WP entails using an ontology element in its own definition.
Multiple classes in domains and/or ranges of relationships and/or attributes	This WP consists in defining the domains and/or ranges of the relationships and/or attributes by intersecting several classes in cases in which they should be the union of such classes.
Polisemy	This WP entails using an ontology element to represent concepts different from the domain under consideration.
Same URI for different ontology elements	This WP entails assigning the same URI to two different ontology elements.

WP Name	WP Description
Relationship *"is"*	This WP entails confusing the subclass relationship (*subclassOf*), the membership to a class (*instanceOf*), or the equality between instances *(sameIndividual*) with an ad hoc relation called "is".
Class 2 in 1	This WP entails creating a class whose name is "Class1AndClass2".
Classes vs. Instances	This WP consists in deepening into a hierarchy so that the more specific classes do not have instances since such classes become class instances of the upper level of the hierarchy.
Relationship and/or attributes without domain and/or range	This WP consists in not specifying the domain or range in the relationships/attributes.
Incomplete information	This WP entails not representing all the knowledge that could be included in the ontology.
Miscellaneous class	This WP consists in creating an artificial miscellaneous class to classify in a certai n level the instances not belonging to any of the sibling classes of this level.

In Section 5.1 we present a set of WPs that have appeared very frequently in the ontologies analyzed.These WPs are related among them in Section 5.2.

5.1 Worst practices in relationship modelling

This section presents three examples of WPs, identified in the ontologies we have analyzed, that would affect the relationship modelling. The first one is related to logical patterns, and the other two, to reasoning patterns.

1. **Relationships and/or attributes without domain and/or range.** This WP entails not specifying the domain and/or range in the relationships/attributes. In the ontology analyzed we found that the relationship "itIsIn" has been defined without specifying nor its domain nor range. That relationship should be defined including the domain and the range involved. For example, the student should define the relationship "itIsIn" that have "Construction" as domain and "Place" as range.
2. **Undefined inverse relationships.** The WP consists in having inverse relationships in the ontology but not defining them as such. This implies that when reasoning we obtain less information than that we could infer. For example, within an ontology on architecture, we have observed how the relationships "itIsIn" and "isBuilt" have been created without defining them as inverse. With this modelling, if we have that `"building1 itIsIn site3"`, then, when we activate a reasoner, we will not obtain that in `"site3 isBuilt building1"`. If relationships were defined as inverses then the reasoner could infer the expected knowledge.
3. **Inverse relationships that are not inverse.** This WP consists in defining two relationships as inverse when, in fact, they are not. As a consequence of this WP, undesired knowledge is probably obtained when a reasoner is applied to the ontology. For example, within an ontology on art, we have seen that the relationships "isSoldIn" and "isBoughtIn" are defined as inverse. This fact could cause an error at the semantic level, since if we have that `"object1 isSoldIn place3"`, then the reasoner will infer that `"place3 isBoughtIn object1"`. An appropriate solution would be to replace the relationship "isBoughtIn" by "isPointOfSale". In this way, the semantic error above

commented disappears, since if "`object1 isSoldIn place3`", then the reasoner will infer that "`place3 isPointOfSale object1`".

5.2 Relationships among worst practices

During the analysis of the ontologies we have observed that the WPs can be related through different types of relationships. For instance, some WPs can be specific cases of a more general WP, or a WP can occur as a consequence of other WP.

With regard to the WPs presented in Section 5.1, it should be noted that the "Inverse relationships that are not inverse" and "Not defined inverse relationships" WPs are inverse. That is, in the first case, not valid knowledge is represented, whereas in the second case knowledge is omitted although is valid. In addition, these two WPs can be a consequence of the "Relationships and/or attributes without domain and/or range" WP, since in the case of "Not defined inverse relationships" WP, two inverse relationships may not be defined as such because their ranges and domains have not been defined. In other words, if there are two relationships in which the domain of one is the range of the other and vice versa, we may hint that both are inverse; however, since both domains and ranges are not defined, the hint is not at all clear. We would try to clarify these notions through the picture shown in Fig. 3[3].

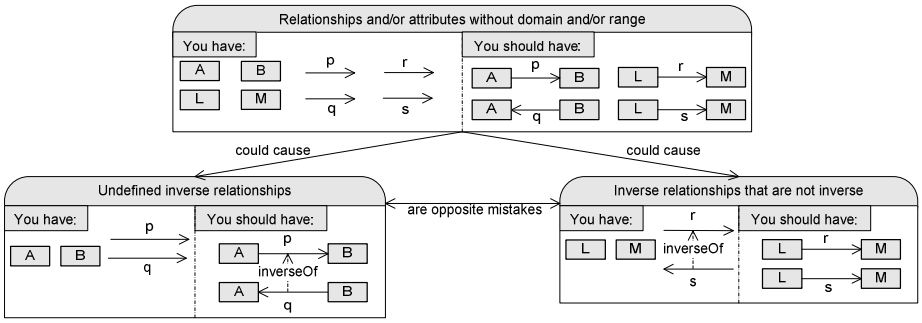

Fig. 3. Some relationships among worst practices

We think that the relationships between the WPs can be very useful during the development of the methodological guides to avoid WPs. Therefore, we are currently analyzing the existence and types of relationships.

6 Conclusions and Future Lines of work

This paper presents the analysis carried out on 11 ontologies, which come from an academic experiment, with the aim of identifying a series of patterns and anti-patterns. During the analysis we have also found a set of WPs that may appear during

[3] The rectangles, the arrows with continuous line and discontinuous line represent the classes, the relationships, and the *inverseOf* properties within an ontology respectively.

the ontology development. The WPs identified have been classified according to a subset of the types of ODPs found in the literature with which they could be related. In addition, a set of examples of these WPs is provided in this paper.

However, even though it is possible to find relationships among the different WPs, as already shown, we have not studied all the possible relationships. Therefore, as a future line of work we propose the identification of the relationships between the different WPs. It would be interesting to identify the groups of WPs that usually appear simultaneously so that, once the concurrence of a WP is identified, the other WPs that could appear could also be identified.

We also propose to analyze the existence of WPs and their relationships in domains other than those represented in the ontologies studied in this paper.

Another interesting work could be to carry out the analysis described in this paper but modifying the ontology development process followed by the students. In this new process we propose to include notions about ODPs in order to compare the resulting ontologies with and without notions about OPDs and analyze to what extent the WPs do not appear.

Finally, as we have already mentioned, the WPs identified are not related to any ODP present in the current libraries; therefore, we are investigating the creation of new design patterns with the objective of avoiding the use of such WPs.

Acknowledgments. This work has been partially supported by the European Commission project NeOn (FP6-027595) and the Spanish project GeoBuddies (TSI2007-65677-C02).

References

1. Aguado De Cea, G., Gómez-Pérez, A., Montiel-Ponsoda, E., and Suárez-Figueroa, M.C. *Natural Language-Based Approach for Helping in the Reuse of Ontology Design Patterns*. In *Proceedings of the 16th International Conference on Knowledge Engineering (EKAW)*, pp. 32-47. (2008)
2. Blomqvist, E.; Gangemi, A.; Presutti, V. *Experiments on Pattern-based Ontology Design*. In *Proceedings of the Fifth International Conference on Knowledge Capture (K-CAP)*, pp. 41-48. (2009)
3. Gangemi, A. *Ontology Design Patterns for Semantic Web Content*. Musen *et al.* (eds.): Proceedings of the Fourth International Semantic Web Conference (ISWC 2005), Galway, Ireland. LNCS, vol. 3729, pp. 262–276. Springer, Heidelberg (2005)
4. Gangemi, A.; Presutti, V. *Ontology Design Patterns*. Handbook on Ontologies (Second Edition). S. Staab and R. Studer Editors. Springer. International Handbooks on Information Systems. (2009)
5. Presutti, V.; Gangemi, A.; David S.; Aguado de Cea, G.; Suárez-Figueroa, M.C.; Montiel-Ponsoda, E.; Poveda, M. *NeOn D2.5.1: A Library of Ontology Design Patterns: reusable solutions for collaborative design of networked ontologies. NeOn project.* http://www.neon-project.org. (2008)
6. Suárez-Figueroa, M.C.; Brockmans, S.; Gangemi, A.; Gómez-Pérez, A.; Lehmann, J.; Lewen, H.; Presutti, V.; Sabou, M. *NeOn D5.1.1: NeOn Modelling Components*. *NeOn project.* http://www.neon-project.org. (2007)