REUSING UML CLASS MODELS TO GENERATE OWL ONTOLOGIES

A Use Case in the Pharmacotherapeutic Domain

Jesús M. Hermida[1], M. Teresa Romá-Ferri[2], Andrés Montoyo[1] and Manuel Palomar[1]
[1]*Department of Software and Computing Systems, University of Alicante, Spain*
[2]*Department of Nursing, University of Alicante, Spain*

Keywords: Ontologies, Knowledge reuse, UML class diagrams, Pharmacotherapeutic domain, OntoFIS.

Abstract: This paper presents a method for the reuse of existing knowledge in UML software models. Our purpose is being able to adapt fragments of existing UML class diagrams in order to build domain ontologies, represented in OWL-DL, reducing the required amount of time and resources to create one from scratch. Our method is supported by a CASE tool, VisualWADE, and a developed plug-in, used for the management of ontologies and the generation of semantically tagged Web applications. In order to analyse the designed transformations between knowledge representation formalisms, UML and OWL, we have chosen a use case in the pharmacotherapeutic domain. Then, we discuss some of the most relevant aspects of the proposal and, finally, conclusions are obtained and future work briefly described.

1 INTRODUCTION AND MOTIVATION

The pharmacological therapeutic process requires reliable and up-to-date information to know the properties of drugs and their suitability to treat a certain health problem according to some individual characteristics. The information must cover the needs of the diverse actors who take part into the pharmacotherapeutic process: physicians, pharmacists, nurses (profesional backgrounds) and citizen/patient (lay profile). However, retrieving the required information during the health care process in the shortest time is still an obstacle for the daily practice.

This obstacle is explained by the number of available drugs and information sources. On the one hand, we must consider the number of marketed drugs. Although each drug might not be available in all contries, when it is, its name might change from one country to another, which is a drawback for the free movement of citizens and patients in the European Union. On the other hand, information sources are diverse in their structure, their contents and their language. There are sources of narrative form or textual and resources that provide structured information. In adition, some information sources are focused on the chemical composition of a drug, but in the health process and , at least, in Spain, identifying the trade name of a certain drug is essential to prescribe it.

The reuse of professional vocabularies for knowledge representation of limited scope or database schemas or class diagrams opens the door to the generation of new semantic resources with less waste of resources and time. This has been an important line of work. Among the existing types of representations we have focused on reusing those UML class diagrams that are used for defining the static structure of a large number of information systems. Specifically, our objective has been the reuse of UML class diagrams that have been design with the VisualWADE CASE tool (Gómez, 2004), based on OO-H Web design method. From a collection of software models, this tool is able to automatically generate Web applications. Obtaining a semantic resource from these diagrams will allow us to develop other tasks, such as, semantic annotation of the content of Web applications.

From the existing knowledge representation formalisms, we chose ontologies because *(i)* they are the current formalism for knowledge representation in the Semantic Web and *(ii)* its wide use in projects in which it is necessary to manage some type of knowledge to fulfill its goals.

UML is also used for the representation of ontologies, because of its simplicity and its wide use, for

M. Hermida J., Teresa 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*, pages 281-286
DOI: 10.5220/0002307802810286

instance, in the development of data-intensive Web applications (Cranefield and Purvis, 1999). However, UML has some lacks when representing general knowledge that limit the complexity of the ontologies: *(i)* small number of representation mechanisms; *(ii)* local knowledge, hard to share even with XMI; and *(iii)* no semantic relation between models. Therefore, it is not a suitable candidate as knowledge representation language to be used in semantic tagging tasks. The main advantage of OWL-DL regarding UML is that it was design to be used by machines and its expressivity level, which can be adapted to the different types of existing applications.

Limitations of UML diagrams and CASE tools in conjunction with the necessity of generating semantically annotated Web applications made necessary formalise domain models in OWL-DL by reusing UML models. The goal of this paper is to describe our transformation process and the functionalities of the tool we have developed for supporting this process. In addition, the whole analysis is carried out with a use case in the pharmacotherapeutic domain. This paper is organised as follows: next section presents the case study where the anylisis will be carried out; Section 3 is the main part of the paper, where the reuse method and the developed plugin are introduced; and, finally, last section discusses some aspects of the paper obtaining some conclusions, and it explains our lines of future work.

2 USE CASE

Considering the initial requirements of information, we designed a new ontology called OntoFIS (in Spanish, "Ontología Farmacoterapéutica e Información para el Seguimiento"), which is aimed at representing knowledge from the clinical practice, focused on the rational use of medicines. Based on ontology principles from (Gruber, 1993; Gomez-Perez et al., 2004), OntoFIS is a conceptualization of common and shared knowledge of the pharmacotherapeutics domain that has been created as a potential multilingual resource to help in semantic annotation of Web resources and tasks related with natural language processing: automatic retrieval and classification of documents as well as information extraction from narrative texts related with this domain.

The chosen methodology for the design of OntoFIS was based on common aspects of current proposals (Uschold and Jasper, 1999; Fernández et al., 1999; Noy and McGuiness, 2001; Gomez-Perez et al., 2004). The process is divided into three stages: *(i)* systematic capture of domain knowledge using an informal formalisation; *(ii)* description of captured knowledge by means of a conceptual modeling language, in this case, UML; and *(iii)* formalisation of knowledge by using a knowledge representation language, OWL.

The next subsections describe the first two stages of the development of OntoFIS.

2.1 Capture of the Domain Knowledge

The characteristics of the information of this domain dificults the use of non-supervised or semi-supervised methods for creating ontologies, because this information might be critical in many processes. Systems based on our ontology will be used for managing health processes in people. An error caused by this system would be hard to solve. Considering the current performance of these type of learning ontologies from text, we decided to invest the available time in carrying out this process manually. In this first stage, we carried out a set of tasks with the purpose of localize information sources tailored to the domain of interest: *(i)* search in bibliographic databases; and *(ii)* analysis of different specific information sources: manuals about Internal Medicine, Pharmacology and Nursing cares and inserts of different types of drugs.

In a first step, from these information sources we extracted the most significative terms and definitions, and a collection of questions usually asked by users with different profiles, expressing their need of information, which would be the basic knowledge core of our ontology. These terms are related to: medicine indication (pathophysiological process, signs, symptoms); prescription (drug, period, frequency of shots, quantity, pharmaceutical form); medicine (name, composition, classification); administration (route, preparation, mode of supply); and, security (effect, time, adverse reactions, precautions). In the end, 144 representative terms were collected and, simultaneously, each term was defined or described with the available diccionaries and manuals, creating a gloss of terms. In a second step, the selected terms were classified in different groups: essential terms, secondary terms (high specifity or attributes) and synonyms (also translations).

In the next stage of the design process, we used the mechanisms of knowledge representation provided by UML class diagrams in order to describe the conceptual model of OntoFIS.

2.2 Knowledge Description using UML

This second stage began with the analysis of the gloss of essential terms in order to establish the type of con-

cept they represent, according to our criteria of functionality and level of specification of our ontology. From this collection of terms we have extracted 23 concepts of medium granularity. These concepts include intrinsic properties (attributes, an average of 4, in a range from 3 to 6) and extrinsic properties (terminological labels, synonyms and translations).

Each association between two concepts has been described by at least one of the predefined relations of an existing taxonomy of 59 relations (Slaughter et al., 2006). Further on, several relations with different semantics can exist between each pair of concepts, capturing our reality from the different points of view of the care process.

Our selection process has reduced the initial taxonomy to 36 types of relation. We have identified 708 relations between our 23 concepts. The relation triples (subject, relation, object) were built manually from the gloss of terms and the collection of user questions.

Once we had analysed the captured information of our domain, we designed the UML class diagram of OntoFIS. Fig. 1 shows a fragment of this class diagram. The picture shows three of the main classes and a subset of the existing relations. This small example provides a vision of the complexity of the scenario where this work is developed. The next section describes the designed process of reusing the conceptual UML model of OntoFIS to generate an OWL ontology.

3 REUSING UML ELEMENTS IN AN OWL ONTOLOGY

Representation languages determine the type of formalisation of the captured knowledge. The final choice of this language has to be focused on the type of representation (formalisation) and the inference mechanisms which will be used according to the purpose of the ontology.

OWL-based knowledge representations contain knowledge which is represented with a higher detail level, easily shareable on the Web and ready to be used to semantically tag texts, among other tasks.

Due to the wide use of the UML class diagram in software and databases designs and the diverse domains that can be represented, UML models are a valuable resource to be able to obtain new knowledge and, thus, reuse the effort that was carried out in the conceptualization phase.

This section is focused on describing the transformation process that has been designed to turn the different elements of a UML class diagram into elements

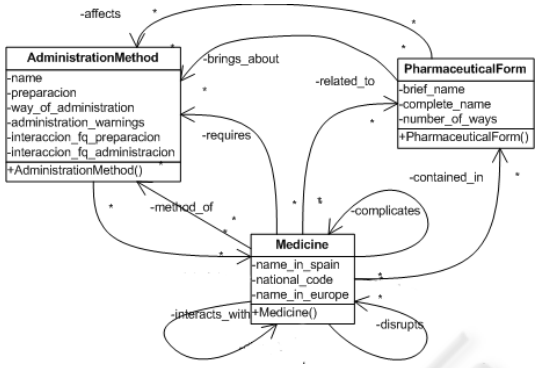

Figure 1: Fragment of OntoFIS UML Class Diagram.

of an OWL ontology. This process is divided in two phases: content transformation and name transformation. Each of these phases with different associated problems we have tried to solve optimally according to our initial purposes: implementing this process as an extension in our Web design tool VisualWADE.

3.1 Proposal of Transformations

The first task to carry out in the design of the process of reuse was the definition of a collection of transformations able to transform a target UML class diagram into an OWL representation of our ontology from a semantic point of view. From this, the purpose is to obtain elements or mechanisms in both diagrams capable of describing equivalent knowledge. Transformations would be applied between two representations in the same level of abstraction (class-class).

A simpler approach would consist in using an ontology to describe the elements of a class diagram (class-instance transformation). This approach may be useful when our purpose is interchanging information between tools. However if we want to use the contained knowledge of a class diagram it is not an optimal approach, since knowledge would be described in instances but it would not be suitable to use it in more complex tasks, in which a reasoner may be involved.

Final transformations were designed and applied, sorted by their complexity level, beginning with the simplest mechanisms (clases and hierarchy) and finishing with the most complex ones (aggregations and compositions).

Table 1 briefly summarises our collection of transformations, turning elements of the UML Class diagram into OWL elements.

Table 1: Summary of Transformations UML Class Diagram - OWL.

Elements of UML Class Diagram	Elements of OWL
Class	`owl:Class`
Attribute (Simple datatype)	`owl:DatatypeProperty`
Attribute (Complex datatype)	`owl:ObjectProperty`
Inheritance	`rdfs:subClassOf`
Association Role	`owl:ObjectProperty`
Role Multiplicity	`owl:Restriction` + `owl:cardinality`, ...
Aggregation	`owl:UnionOf`
Composition	Generic properties `partOf` and `hasPart` + subproperties
Class Description	`rdfs:comment`
Representative Terms and Translations	`rdfs:label`

3.1.1 Filling the Gap between Representations

Note that there is a representation gap between UML class diagram and OWL documents. On the one hand, OWL has a higher degree of expresivity and, therefore, there are some characteristics that UML cannot represent. UML has no direct mechanism for the representation of some characteristics of OWL ObjectProperties, such as transitivity or symetry. Instead of using OCL (Object Constraint Language), which would considerably increase the complexity of the transformations, we have included new functionalities in our support tool (described in the following section). On the other hand, UML provide some representation mechanisms which have no trivial or no possible equivalence. Although the concept of association is one of the most utilized elements of class diagrams, it does not have an equivalent representation in OWL. Instead, association roles are used to obtain the different OWL properties of an ontology. Moreover, aggregations and compositions are the elements of a class diagram whose capture and processing is more complex, since their semantics vary according to each use case. Although (Rector and Welty, 2005) and (Veres, 2005) provide generic solutions for these problems, the use of NLP techniques would be necessary in order to process a higher number of elements by relating the element names with their associated semantics.

3.2 Resolving the Naming Gap

Another important aspect in the transformation process was the design of rules for name conversion between both representations to avoid conflicts.

There is also a clear naming gap between UML and OWL identifiers. While an OWL ontology use URIs to identify uniquely each of its elements (global identification), some elements from different class diagram may have the same name (local identification), for instance, attributes or association roles. Table 2 shows a summary of these name conversions. This is the second phase of the transformation process. Once we have obtained an element with equivalent semantics in OWL, our method assigns an URI to it.

Table 2: Summary of URI Generating Rules for Ontology and Knowledge Base Elements.

Target Element	URI Generation Rule
`owl:Class`	URL_ONTODEF[1] + '#' + CLASS_NAME[2]
`owl:DatatypeProperty`	URL_ONTODEF + '#' + CLASS_NAME + { '-', '..', ':', ';' } + ATTR_NAME[3]
`owl:ObjectProperty`	URL_ONTODEF + '#' + ASSOC_NAME[4] + { '-', '..', ':', ';' } + ROLE_NAME[5]
Instance	URL_INSTANCE + '#' + CLASS_NAME + { '-', '..', ':', ';' } + INSTANCE_ID

[1] URL_ONTODEF — URL of the file which contains the definition of the ontology.
[2] CLASS_NAME — Name of the UML class from which comes the element.
[3] ATTR_NAME — Name of the UML attribute from which comes the element.
[4] ASSOC_NAME — Name of the UML association from which comes the element.
[5] ROLE_NAME — Name of the UML role of association from which comes the element.

3.3 Applying Transformations to OntoFIS

In this subsection, we show the results of the algorithm after applying it to the class diagram depicted in Fig. 1. Although it shows a subset of the elements of OntoFIS, we can appreciate the transformation process. The results shown in the next points are sorted by their generation time and the complexity of the transformation.

Classes:

```
<owl:Class
   rdf:ID="http://www.dlsi.ua.es/ontofis.owl#Medicine">
<!-- Disjoined with other classes -->
<!-- Description of the concept in natural language -->
  <rdfs:comment xml:lang="es">A chemical... </rdfs:comment>
<!-- Most representative terms and translations -->
  <rdfs:label xml:lang="es">Medicamento</rdfs:label>
  <rdfs:label xml:lang="en">Drug</rdfs:label>
</owl:Class>
```

Inheritance:

```
<owl:Class rdf:ID="Action">
  <!-- Class Information -->
</owl:Class>
<owl:Class rdf:ID="DesirableEfect">
  <rfds:subClassOf rdf:resource="#Action"/>
  <!-- Class Information -->
</owl:Class>
```

Class Attribute:

```
<owl:DatatypeProperty rdf:ID="Medicine..national_code">
  <!-- Comments and translations -->
  <rdfs:Domain rdf:resource="#Medicine"/>
  <rdfs:Range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>
```

Association Role:

```
<owl:ObjectProperty rdf:ID="AS3..interacts_with">
  <!-- Comments and translations -->
  <rdfs:Domain rdf:resource="#Medicine"/>
  <rdfs:Range rdf:resource="#Medicine"/>
</owl:ObjectProperty>
```

Eventually, as aforementioned, this process was not designed to be carried out manually, since our UML design was supported by the VisualWADE tool. A new plugin for visualWADE has been implemented to automatically generate OWL representations from our UML class diagrams by using our transformation algorithm.

3.4 Tools for Supporting the Process

The main support tool used along this process was VisualWADE (Gómez, 2004), which is a CASE tool for the development of data-intensive web applications. From a collection of designed models, this tool can automatically generate an entire web application, including database, business logic and interface, from different software models. Fig. 2 depicts the interface of VisualWADE while designing a class diagram.

Although it is not a knowledge representation tool, we have chosen VisualWADE because of three aspects: *(i)* it uses UML class diagram for the modeling of the domain of Web applications; *(ii)* it is a well-known tool; and *(iii)* it can be easily extended thanks to its plug-in extension mechanism.

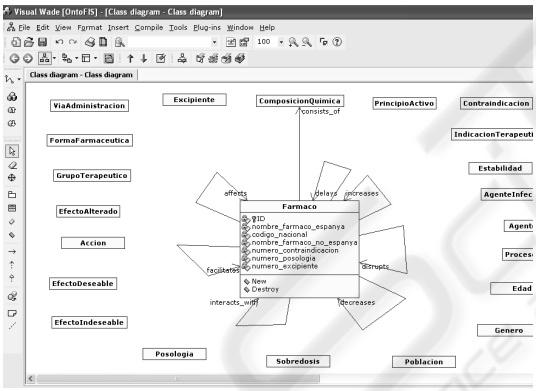

Figure 2: Picture of VisualWADE CASE Tool.

To apply the designed transformations in a easy manner, we extended the tool by developing a plug-in. Specifically, we developed the plug-in *Semantic Web Tools for VisualWADE*, depicted in Fig. 3. The purpose of this plug-in is to become the main ontology management core of the application, provinding functionalities that class diagrams and the own program cannot provide by itself.

The most relevant functionalities of our plug-in are presented in the following list: *(i)* multilinguality and synonimy —inclusion of synonyms and translations of all the used terms—; *(ii)* description of elements —inclusion of the description of an element in multiple languages—; *(iii)* characteristics of association roles: direct and inverse functionality, transitivity

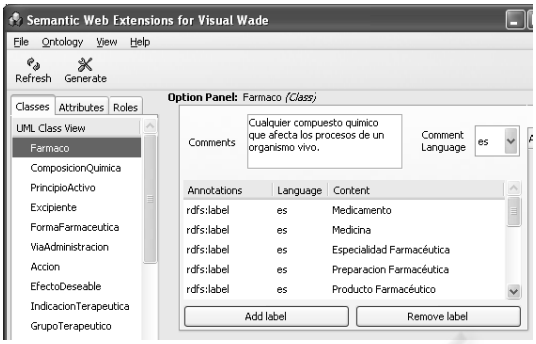

Figure 3: Picture of Semantic Web Tools for VW.

and symetry; and *(iv)* OWL generation —it carries out the whole transformation process already described.

OWL documents generated by our plug-in were verified by Protégé ontology editor (Knublauch et al., 2004) in order to check that the generated ontology was complete according to its contents (concepts, properties and relations), and ensuring that its OWL code had been generated correctly (no syntax errors), which indirectly ensures the exportability of the document and the reusability of OntoFIS.

4 DISCUSSION AND CONCLUSIONS

This paper presents a method for reusing existing UML models to generate OWL ontologies. Once we have explained the main aspects of our methodology and the reuse strategy for UML class diagrams, we would like to describe the main benefits of our proposal, some controversial aspects and improvements to the current transformation process, and our lines of future work.

The main benefits of the method of knowledge reusing are the following. Firstly, it allows us to reuse existing knowledge from UML class diagram, which are very common in different fields. Although knowledge contained in diagrams may not be ontologies, it might facilitate the process of creating new ontologies. Secondly, in our case, it will help us to generate semantically tagged Web applications taking advance of previously done designs in VisualWADE. Therefore, we have developed a plugin to manage the use of ontologies inside the tool that also includes the required transformation algorithm.

An interesting issue to discuss in our methodological approach is the use of UML as ontology representation language. Currently there are more modern approaches which solves some of its derived problems from the viewpoint of modeling ontologies, such as,

ODM (Ontology Definition Metamodel). The main advantages of ODM is that it provides a visual language to represent ontologies with all the mechanisms of OWL and a set of transformations to generate OWL documents. However, this approach was discarded because of two disadvantages: *(i)* it is based on OWL and *(ii)* the difficulty of adapting this solution in our final tool. Instead, although we analysed the disadvantages, we decided to use UML because it is more general and it may facilitate the generation of another type of semantic resource. This would also imply an smaller effort in case we decided to change our tool.

Our strategy of reuse provides us two representations that may be redundant.Why haven't we generated an OWL representation with Protégé in the first stage? Our two representations produce two different types of ontologies with different purposes. While UML produces a task ontology focused on specific problems, such as web generation, OWL can represent domain ontologies with more general knowledge, more easily reused and suitable for future developments in the Semantic Web field.

As can be appreciated, this paper does not assess the content of the ontology of our use case. It simply assesses the correctness of the OWL code generated in the transformation process in order to ensure the compatibility and exportability of our code into other tools. A deeper analysis of the content of OntoFIS can be found in (Romá-Ferri et al., 2009).

An important aspect of the transformation algorithm between models that has to be improved is the processing of associations and aggregations. Other mentioned papers describe various solutions that can be possible depending on the semantics of each association. The main problem is that the transformation mechanism has to vary depending on the meaning of each aggregation or association. We decided to include the solution for the most probable cases in our collection of transformations. However, in order to improve the quality in the transformation process, it is necessary to disambiguate element names in the process of model transformation.

Finally, our future work is focused on four aspects: *(i)* improving the transformation process by including NLP techniques to disambiguate the meaning of aggregations and compositions; *(ii)* analysing the reuse process in other domains to check if domain-dependant problems appear; *(iii)* debugging the developed plug-in and implement NLP processing techniques; and *(iv)* designing a method for the generation of semantically tagged Web applications in VisualWADE.

ACKNOWLEDGEMENTS

This work has been partially funded by the Spanish Government under the CICyT project number TIN2006-1526-C06-01, the FIS program of the Spanish Health Institute "Carlos III" (PI051438) and the FPU program of the Spanish Ministry of Science and Innovation (AP2007-3076).

REFERENCES

Cranefield, S. and Purvis, M. (1999). Uml as an ontology modelling language. In *Proceedings of the Workshop on Intelligent Information Integration, IJCAI'99*, pages 46–53.

Fernández, M., Gómez-Pérez, A., Pazos, J., and Pazos, A. (1999). Building a chemical ontology using methontology and the ontology design environment. *IEEE Intelligent Systems*, 14(1):37–46.

Gómez, J. (2004). Model-driven web development with visualwade. In *ICWE*, volume 3140 of *Lecture Notes in Computer Science*, pages 611–612. Springer.

Gomez-Perez, A., Corcho, O., and Fernandez-Lopez, M. (2004). *Ontological Engineering: with examples from the areas of Knowledge Management, e-Commerce and the Semantic Web. 1st Edition.* Springer.

Gruber, T. R. (1993). Toward principles for the design of ontologies used for knowledge sharing. *International Journal of Human-Computer Studies*, pages 907–928.

Knublauch, H., Fergerson, R. W., Noy, N. F., and Musen, M. A. (2004). The protégé owl plugin: An open development environment for semantic web applications. In *ISWC 2004*, pages 229–243.

Noy, N. F. and McGuiness, D. L. (2001). Ontology development 101: A guide to creating your first ontology. Technical report, Stanford University.

Rector, A. and Welty, C. (2005). Simple part-whole relations in OWL ontologies. Editor's draft, World Wide Web Consortium.

Romá-Ferri, M. T., Cruanes, J., and Palomar, M. (2009). Quality indicators of the 'ontofis' pharmacotherapeutic ontology for semantic interoperatibility. In *Proceedings of the International Conference e-Health 2009, Algarve (Portugal)*, pages 107–114. IADIS.

Slaughter, L. A., Soergel, D., and Rindflesch, T. C. (2006). Semantic representation of consumer questions and physician answers. *Int J Med Inform*, 75(7):513–529.

Uschold, M. and Jasper, R. (1999). A framework for understanding and classifying ontology applications. In *Twelfth Workshop on Knowledge Acquisition Modeling and Management KAW 99*, pages 16–21.

Veres, C. (2005). Aggregation in ontologies: Practical implementations in owl. In Lowe, D. and Gaedke, M., editors, *ICWE*, volume 3579 of *Lecture Notes in Computer Science*, pages 285–295. Springer.