Economic Insights – Trends and Challenges	Vol.IV(LXVII) No. 2/2015	87 - 94

Comparative Analysis between OWL Modelling and UML Modelling

Aurelia Pătraşcu

Faculty of Economic Sciences, Petroleum-Gas University of Ploieşti, Bd. Bucureşti 39, 100680, Ploieşti, Romania
e-mail: patrascuaura@yahoo.com

Abstract

In this paper I propose to realize a comparative analysis of two methods using as tools for modelling, the OWL Protégé editor and the Visual Paradigm for UML tool. In the past years, the business environment has adopted ontologies as a mean to process and reuse the domain knowledge. Ontologies provide the premises of resources conceptual modelling. At the same time, companies use, more and more, the Unified Modeling Language (UML) due to the quick and accurate design of the systems where they are concurrent processes. UML and OWL have different objectives and approaches, but they are similar regarding structure representation, respectively class diagram. Finally, the UML class diagram and the OWL ontology for a document management system will be presented.

Keywords*: UML modelling; OWL ontology; class diagram; document management system*

JEL Classification: *C81; C61*

Introduction

The business processes modelling has become a priority for organizations because it represents an important step in supporting the business by IT methods in order to increase the quality of the offered services and to reduce the order processing time, thus, minimizing costs that will lead to increase the profits. In order to model a business, it can be used a modelling language where a set of syntactic and semantic rules is defined to build a set of diagrams and related objects.

This article presents a contribution to the software and ontologies development activity. Thus, starting from a Unified Modeling Language model for the document management, I will generate an OWL ontology. Modelling techniques offer different views of an application.

Unified Modeling Language (UML) is a unified modeling language which uses graphical notations that can treat many aspects of the software development process, from database designing to code modules interaction. UML modelling techniques model the systems entities in classes with attributes and behaviour and the diagrams are means to represent and view the modelling elements (Udrică, Martinov and Lupoaie, 2009).

Ontologies represent the description of knowledge from an interest field. The main purpose of ontologies is to represent the objects semantics from a given domain. This objective is achieved

by assigning the objects to classes and by describing those, using properties. According to the model of the used ontology, such descriptions are managed by different restrictions.

OWL facilitates the information content interpretation by software and was designed to process web information content. In OWL, an object may belong to any number of classes and can be described by any subset of properties. Therefore, each domain object belonging to a certain class has its own structure.

In contrast to this, a database schema aims to describe similar objects by an identical logical structure, in order to optimize the queries using indexing techniques. In the absence of any special assumption about the representation of objects, the only possible common structure consists in associating each object with: (1) any subset of the set of classes and (2) any subset of the set of properties.

The disadvantages of such structures are obvious:

- requires significant storage cost (by using a systematic representation of all the properties for each instance);
- involves a significant response time because of the need to perform a large number of association operations (if for each instance are represented only their significant properties and classes) (Bellatreche et al 2006; Jean et al, 2007) .

Therefore, in the context of those presented, according to the ontology used different restrictions are established (IBM and Sandpiper Software, 2006).

OWL Ontology versus UML Modelling

Lately, the methodologies used for building a knowledge base have developed greatly, however, regardless of the chosen methodology, the ontology development is very important in the development of a coherent, complete and non-redundant knowledge base.

An ontology consists of a base terms vocabulary and a precise specification of what they mean. However, an ontology, is more than just a vocabulary. It is the starting point for knowledge structures development - not only the classifications of concepts, but, also, complex relationships (Trăuşan – Matu, 2004).

Thus, a common vocabulary for researchers who want to share information from a domain is created by defining an ontology. The vocabulary includes the definitions of some fundamental concepts in the domain and the relationships between them.

Currently, many disciplines develop standardized ontologies that experts can use to share and annotate the information in their fields of activity. For example, medicine has created standardized structured and bulky vocabularies, such as SNOMED, as well as the semantic network for unified medical language system (Unified Medical Language System - UMLS) (Noy, Mc Guiness).

OWL is a product of the Word Wide Web Consortium (W3C) and makes possible the description of concepts, but it has a richer set of operators and is based on a different logical concept that provides the ability to define concepts and described them at the same time. The logical model uses a *reasoner* that checks if all the definitions and declarations are mutually consistent and can recognize what concepts fit and under what definitions.

OWL is part of the semantic Web and can represent a future where Web information has a precise meaning**,** can be processed by the computer and the computers can integrate information across the web.

The semantic Web goal consists in ensuring the communication between computers on the Internet. One of the most important development directions in the semantic web is represented by improving the HTML documents content metadescriptors (Davies, Studer and Warren 2006; Eriksson 2007; Zhou, Ding, Finin 2010). Languages RDF and OWL allow web site designers to define ontologies that add semantic content to HTML pages. Semantic search engines (swoogle) can use this metadata to provide search services based on ontologies.

UML language notoriety has led to the emergence of dedicated program products belonging to the Computer Assisted / Aided Systems / Software Engineering (CASE) type (Popa 2004).The CASE UML products are used for developing of some complex management systems. Subsystems, analysis, design and implementation models, UML specific diagrams etc. are developed using a computer (Cucui 2006).These products offer benefits to all those involved in the project: analysts programmers who can identify the system requirements with a use case diagram, designers who can produce models through which illustrates the interactions between objects and developers who can, easily, convert the model into a functional application.

Both UML modelling and OWL modelling present the strengths and the weaknesses, as well as similarities and differences. The resemblance between OWL and UML consists in the fact that both use the concept of class.

A class in the OWL is interpreted as a set that contains individuals. The individual is defined independently of the class. Classes are organized in a hierarchy of superclasses - subclasses that is known as *taxonomy***.** In order to match a class, an individual must meet several conditions and classes are constructed from descriptions that specify these conditions.

A class in UML presents the system structure, in general terms, and involves specifying the attributes that defines the structure, the operations that define the behaviour and the relationships with other classes.

Both UML and OWL support a model structure, called the package in UML and ontology in OWL.

A key difference is that there is a class named **Thing** in OWL that represents the set of all individuals and all the classes are subclasses of *OWL***: Thing.**

Both allow multiple inheritances and, also, allow that a subclass of a class should be declared disjoined.

OWL properties are always binary and their ends are called *domain* and *range*. In OWL a property when it is applied to a class can be constrained by cardinality restrictions on the domain which gives the minimum (**minCardinality**) and a maximum (**maxCardinality**) number of instances that can participate in a relationship. In addition, an OWL property can be globally declared as functional (**functionalProperty**) or inverse functional (**inverseFunctional**).

In UML an association can have minimum and maximum cardinalities (multiplicity) specified for any of its ends. OWL allows individual-valued properties (**ObjectProperty**) to be declared in pairs, one the inverse of the other (IBM and Sandpiper Software 2006).

In Table 1 are described concepts of UML and OWL.

In Table 2 are shown the UML concepts that have no equivalent in OWL and OWL features that have no correspondent in UML.

Table 1. Similar concepts

UML Elements	OWL Elements	Comment
Class	Class	
Instance	Individual	Unrelated individuals in OWL class
Attribute, Binary association	Property	Property OWL can be globally
Subclass	Subclass	
Enumeration	OneOff	
Multiplicity	minCardinality maxCardinality	In OWL cardinality is declared only for the range
Package	Ontology	

Source: IBM and Sandpiper Software 2006

Table 2. Differences UML and OWL

UML Elements	OWL Elements
navigable, non-navigable	-
derived	-
abstract classifier	-
Classes as instances	-
-	Thing, global properties, autonomous individual
-	allValuesFrom, someValuesFrom
-	SymmetricProperty, TransitiveProperty
-	Classes as instances
-	disjointWith, complementOf

Source: IBM and Sandpiper Software 2006

Transforming a UML Model in OWL Ontology

Starting from a UML model for document management I will generate an OWL ontology. The transformation is based on transformation rules that allow connecting the UML and OWL concepts.

A UML class diagram for document management in an organization is presented in Figure 1. This diagram is designed in Visual Paradigm for UML and represents a simplified form of some previous versions. (Pătraşcu and Tănăsescu 2010; Pătraşcu 2014). As it is shown in the Figure 1, this diagram classes are: existing documents (**Document),** document pages (**Pages**), folders in which documents are included (**Folder**), registers which included folders (**Register**), the type of documents (**Document Type**), the departments who have issued documents (**Departments**), users (**User**) and documents archive (**Document archive)**.

A diagram of classes can contain both the relationships between classes and relationships between instances of classes. *Relationships between classes* are represented by generalization, dependency and realization. *Relations between a class instances* are represented by the association relationships and aggregation/composition relationships.

I have defined the following relationships types for the document management application:

- *3 composition relationships between classes* **Folder** and **Register**, **User** and **Departments, Pages** and **Document** (for example, an object of the folder class is part of an instance of the class register and cannot be part of another instance);

- *2 one to many unidirectional associations between classes* **Document** and **User,** as well as **Document** and **Document Type**
- *1 one to one bidirectional association between classes* **Document and DocumentArhive**;
- *1 many to many bidirectional association between classes* **Folder and Document.**

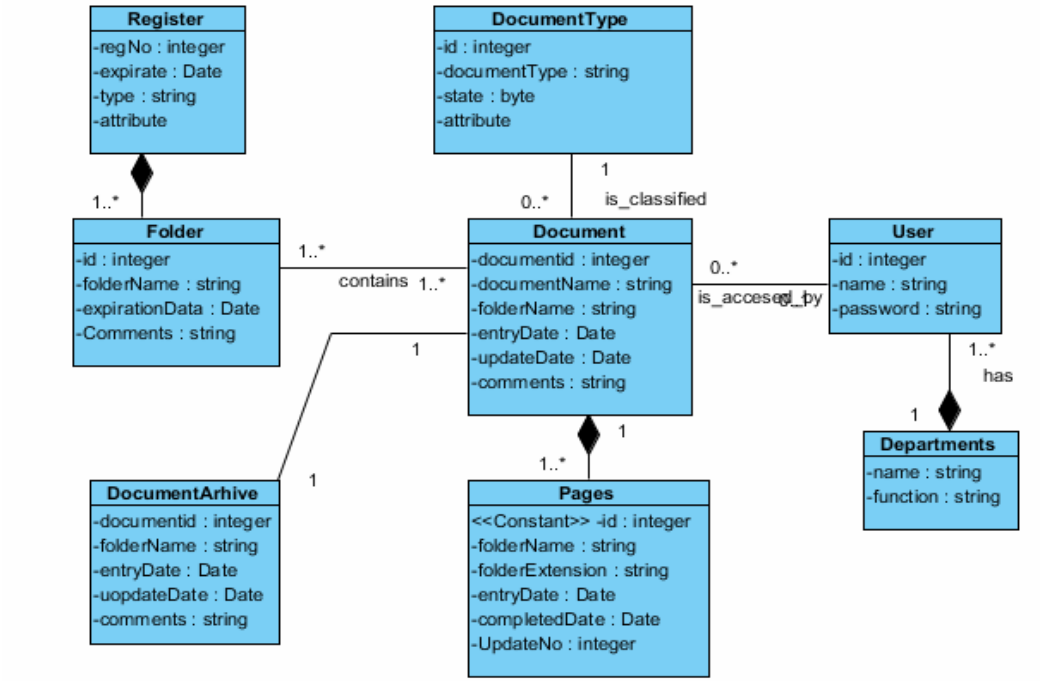

Fig. 1. Class diagram of the classes.

Source: adaptation after (Pătraşcu and Tănăsescu 2010; Pătraşcu 2014).

Building an ontology is a process that represents the exact description of things and their relations. For web, the ontology represents the exact description of web information and relationship about web information. OWL was designed to provide a way to process the information content on the web.

A tool to create ontologies is represented by Protégé Editor. One advantage of this editor is that the plugins can be added which allow it to be expanded in order to satisfy any necessity. The most important plugin is the one that allows access to the mechanisms of the OWL language.

UML and OWL are designed with differences but both support a modular structure called package in UML and ontology in OWL. Transforming a package into an ontology is a simple process, the package being mapped directly in ontology.

An OWL ontology is constructed from individuals, properties and classes that have slots and classes as correspondent in Protégé Frame.

Starting from UML class diagram (Figure 1), I have mapped in Protégé an ontology that includes 9 classes: **Departments, Users, DocumentArhive, Folder, Document_Folder, Document, Register, Pages, DocumentType**.

The relations established between the 9 classes of the ontology are shown in Figure 2.

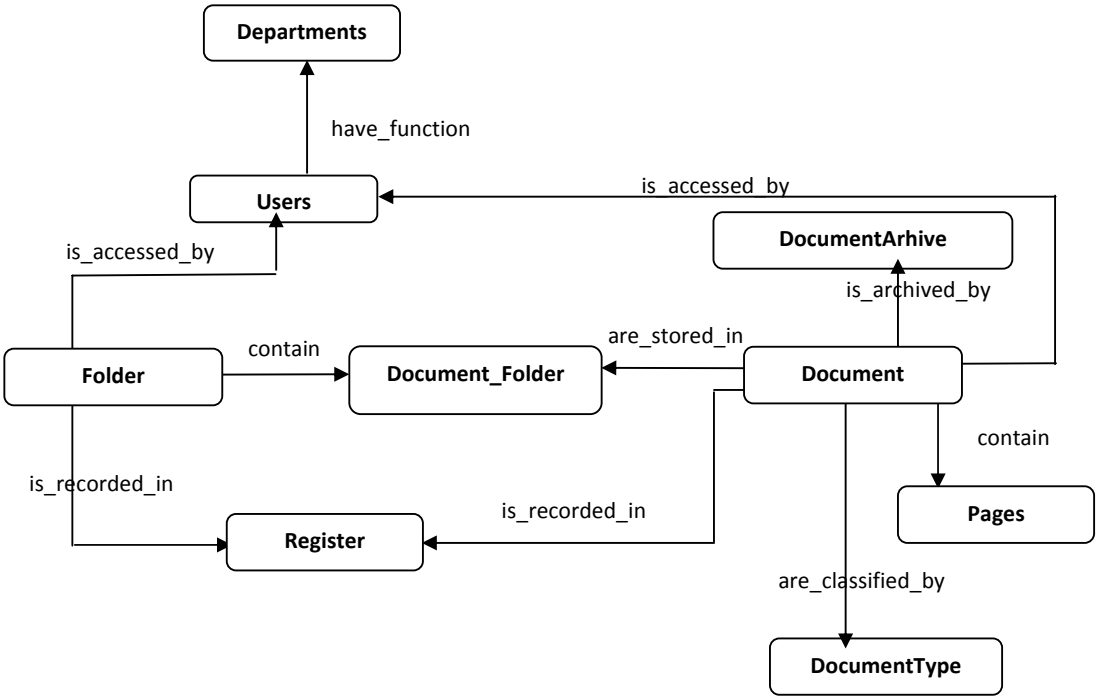

Fig. 2. The existing relationships between ontology classes.

Source: made by the author.

In Figure 3 is represented the class hierarchy created in Protégé OWL.

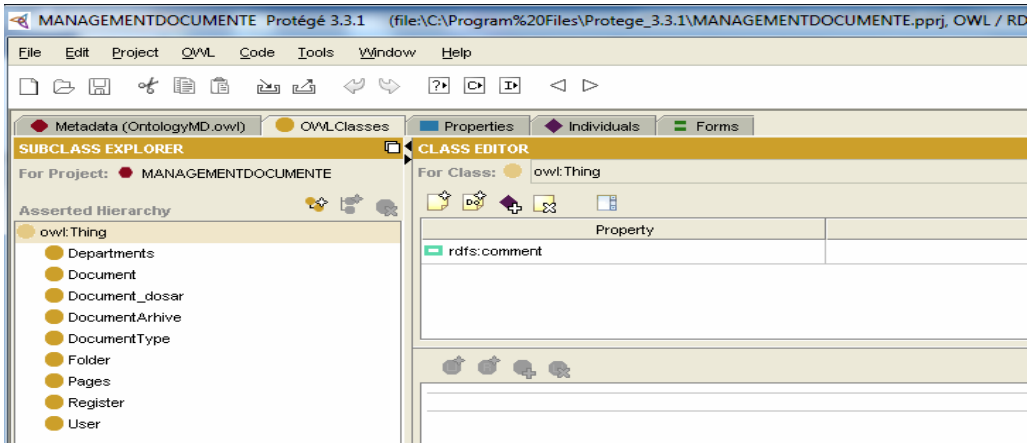

Fig . 3. Class hierarchy in Protégé OWL.

Source: made by the author.

The **Document** class (fig. 4) contains the following properties:

- **documentid**, an integer type property;
- **documentName, folderName** and **comments**, string type properties;
- **entryDate** and **updateDate**, data type properties;
- **hasDosar and hasUser**, object properties.

OWL classes are interpreted as sets containing individuals (sets of objects). Owl: Thing class is the class that represents the set of all individuals because all classes are subclasses of owl: Thing.

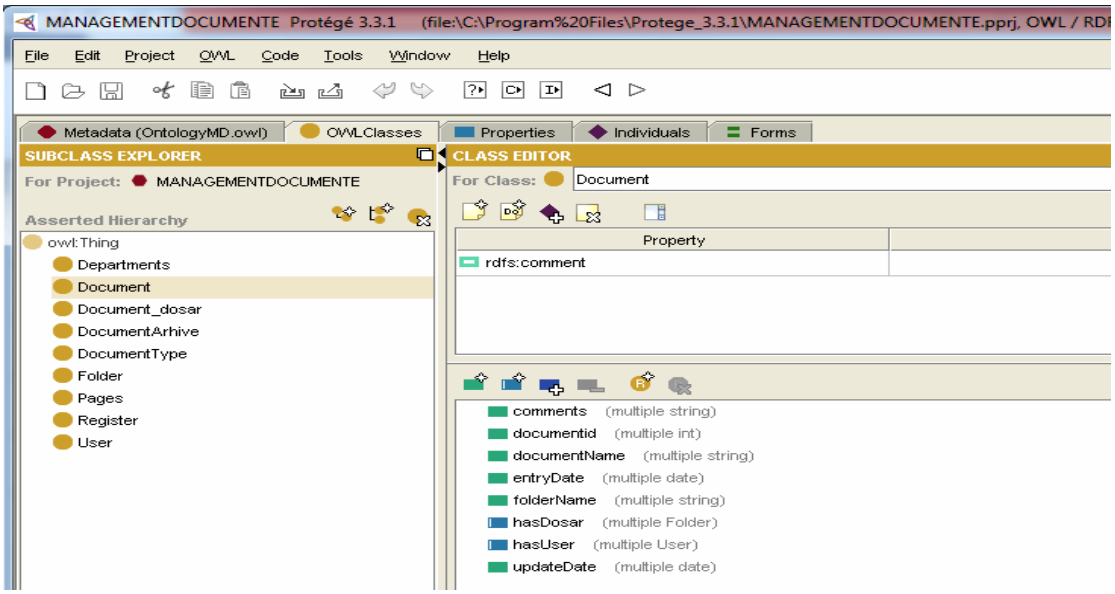

Fig. 4. Document class properties.

Source: made by the author.

As it is shown in Figure 4, there are two types of properties: object properties and data type properties.

OWL is produced by Word Wide Web and makes possible the description of concepts, but also has a richer set of operators and is based on a different logical concept that allows the concepts to be defined and described in the same time. OWL is part of the semantic Web and can represent a future where Web information have an exact sense, can be processed by the computer and the computers can integrate information across the web.

Conclusions

In this article the author offers two solutions for the modelling of the document management in an organization.

The *first method* is used in order to:

- provide different perspectives that guide planners in using object concepts;
- offer a graphical notation, behind which lies a proper semantics and support building and documenting the components of the document management software system;
- allow specifying the systems by precise, non-ambiguous and complete methods at all levels of detail: analysis, design and implementation.

The *second method* presents the following advantages:

- ensures the unitary using of domain concepts reuniting the view points of domain experts;

- can train the employees of an organization, unfamiliar with the type of documents that are issued and processed in it;
- can be easily integrated in other document management system to support the decision of top-managers from an organization;
- achieves an analysis of the document management domain knowledge;
- allows sharing the knowledge between the employees of an organization.

References

1. Bellatreche, L. et al, Contribution of ontology-based data modeling to automatic integration of electronic catalogues within engineering databases, *Computers in Industry*, vol. 57, nr. 8-9, 2006, pp. 711-724.
2. Cucui, G., Consideraţii privind ingineria software şi managementul proiectelor TIC, *Annales Universitatis Apulensis Series Oeconomica*, vol. 2, nr. 8, 2006, ISSN 1454-9409, http://www.oeconomica.uab.ro/upload/lucrari/820062/18.pdf.
3. Davies, J., Studer, R., Warren P., *Semantic Web Technologies: Trends and Research in Ontology-based Systems*, John Wiley & Sons Ltd, 2006.
4. Eriksson, H., The semantic-document approach to combining documents and ontologies, *International Journal of Human-Computer Studies*, vol. 65, nr. 7, 2007, pP. 624-639.
5. IBM and Sandpiper Software. Ontology DeØnition Metamodel. Sixth Revised Submission, Object Management Group, June 2006.http://www.omg.org/cgi-bin/doc?ad/2006-05-01
6. Jean, S. et al, OntoDB: It Is Time to Embed Your Domain Ontology in Your Database, in *Advances in Databases: Concepts, Systems and Applications, Lecture Notes in Computer Science*, Springer Verlag, Berlin, vol. 4443, pg. 1119-1122, 2007, Proceedings of 12th International Conference on Database Systems for Advanced Applications, editors: Kotagiri R., Krishna R., Mohania M., Nantajeewarawat E., DASFAA 2007, Bangkok, Thailand, April 9-12, 2007.
7. Noy, N.F., Mc Guiness, D.L., Ontology *Development 101: A Guide to Creating Your First Ontology*,http://protege.stanford.edu/publications/ontology_development/ontology101.pdf.
8. Pătraşcu, A., Tănăsescu A., *Document management static modelling*, Proceedings of the 14th International Business Information Management Association Conference, 23-24 June, Istanbul, Turkey, 2010.
9. Pătraşcu, A., *Document Management Processes and Use Case Scenarios Elaboration,* Annals of the „Constantin Brâncuşi" University of Târgu Jiu, Economy Series, Special Issue/2014, Information society ad sustainable development.
10. Popa, L., Proiectarea obiectuală UML a unui sistem de gestiune a costurilor unei organizaţii prin metoda ABC. Diagramele cazurilor de utilizare şi ale claselor, *Revista Română de Informatică şi Automatică*, ICI Publishing House, Buchaest, vol. 14, nr. 3, 2004.
11. Trăuşan – Matu, Ş. - *Programare în Lisp. Inteligenţă artificială şi web semantic*, POLIROM Publishing House, Iaşi, 2004.
12. Udrică, M., Martinov, M.D., Lupoaie, A.E., *UML prin aplicaţii. Studii de caz privind dezvoltarea sistemelor informatice*, Renaissance Publishing House, Bucharest, 2009.
13. Zhou, L., Ding, L., Finin, T., How is the Semantic Web evolving? A dynamic social network perspective, *Computers in Human Behavior*, in press, corrected proof, available online 21 August 2010.